Database-SQL-RDBMS HOW-TO pour Linux (PostgreSQL Object Relational Database System), version française.

Al Dev (Alavoor Vasudevan) aldev@hotmail.com
Adaptation française Albert-Paul Bouillot
v2.0, 17 November 1997, version française Janvier 1998

Ce document explique COMMENT mettre en place une Base de Données Relationnelle SQL Objet de la nouvelle génération "PostgreSQL" sur votre système unix qui pourra être utilisée comme Base de Données Serveur d'Application ou Serveur Web. PostgreSQL se rapproche chaque mois un peu plus des standards Internationaux ISO et ANSI SQL 1998,92,89. Ce document donne aussi des informations sur les programmes d'interface à la base de données tels que frontaux GUIs, outils RAD (Développement Rapide d'Application), interfaçage des langages de programmation ("C", "C++", Java, Perl), pilotes ODBC, JDBC ainsi que sur les outils et programmes interfaçage d'une Base de Données Web . L'information donnée ici est valable pour toutes autres plates-formes unix et autres Bases de Données. Cette information sera très utile aux nouveaux utilisateurs de PostgreSQL, des Bases de Données et du langage SQL.

Table des matières

1 Introduction

Tout système informatique au monde a besoin d'une base de données pour stocker/retrouver les informations. Sans base de données, un ordinateur devient inutile. La première raison pour laquelle on utilise un ordinateur est de stocker, retrouver et traiter l'information et de faire cela très rapidement, et donc, de faire économiser du temps. En même temps le système doit être simple, robuste, rapide, fiable, économique et d'utilisation aisée. Les systèmes de gestion de base de données les plus courants sont basés sur les spécifications ISO (International Standard Organisation) SQL lesquelles sont également basées sur les standards américains ANSI SQL. Les spécifications courantes généralement utilisées sont l'ANSI SQL 92 et l'ANSI SQL 89. Le prochain standard est le SQL 1998/99 aussi appelé SQL-3. Les systèmes de gestion de bases de données les plus répandus tels que Oracle, Sybase et Informix s'appuient sur ces standards ou essaient de les implanter.

Ainsi qu'il est indiqué dans ce document, il y a plus de 20 variétés, commerciales/internet, de systèmes de gestion de base de données actuellement utilisés dans le monde et beaucoup, beaucoup plus dans un futur proche. En l'absence d'un standard tel que l'ANSI/ISO SQL, il serait très difficile pour l'utilisateur de développer une application une fois pour toutes et de l'utiliser avec tous les systèmes de gestion de bases de données. Aujourd'hui, l'utilisateur veut développer son application une fois pour toutes en utilisant les normes ISO SQL, ODBC, JDBC et la vendre afin qu'elle puisse être utilisée sur une grande variété de systèmes de gestion de base de données a travers le monde.

PostgreSQL est la Base de Donnée LIBRE la plus populaire au monde qui implante la plupart des standards ISO SQL, ANSI SQL/98, SQL/92 et ANSI SQL/89 RDBMS. PostgreSQL est une base de données relationnelle Objet de la nouvelle génération et les futurs standards ANSI SQL tels que le SQL 1998 (SQL-3) et au-delà traiteront de manière croissante de bases de données Objet et de types de données Objet. PostgreSQL est le seul SGBD (RDBMS - Relational Data Base Management System) au monde qui supporte les bases de données Objet et SQL. Ce document vous explique comment installer le système de gestion de base de données ainsi que tous les paquetages concernant les bases de données, comment mettre en place la base de données Web, la base de données application , les frontaux GUIs et les programmes interfaçage. On ne saurait trop RECOMMANDER d'écrire vos applications bases de données 100 % compatibles avec les standards ISO/ANSI SQL, ODBC, JDBC ceci rendant votre application portable sur de multiples SGDBD (Systèmes de gestion de Bases de Données) tels que PostgreSQL, Oracle, Sybase, Informix etc.

La haute qualité, et un large ensemble de possibilités de PostgreSQL viennent du fait que ce système est développé suivant le principe du 'Modèle de développement de Systèmes Ouvert'. Les systèmes ouverts sont ceux ou la totalité du code source est fournie et ou le développement se fait à travers l'internet par une très large communauté de cerveaux humains en réseau. La tendance future des développement logiciels réside dans ce que l'on appelle les "super autoroutes de l'information" qui s'étendent à travers le monde entier. Dans les années à venir, la croissance d'internet va être explosive et de ce fait favorisera l'adoption de PostgreSQL par les utilisateurs.

Avec l'application des principes de la physique (quantique, classique, thermodynamique), des mathématiques et des statistiques à la qualité des logiciels, on obtient la meilleure qualité des logiciels 'Systèmes Ouverts' tel que PostgreSQL en mettant le code source des programmes a la disposition d'un grand nombre de cerveaux humains interconnectés par les super-autoroutes de l'information. Plus le nombre de cerveaux au travail sera grand, meilleure sera la qualité du logiciel produit. Le modèle "Système Ouvert" évite aussi de réinventer la roue tout en étant particulièrement économique, en diminuant les délais de distribution et en suivant les lois économiques modernes d'optimisation des ressources nationales et globales. Dans un futur proche, à l'aube du 21-ième siècle, la manière de se procurer un logiciel va changer. Les utilisateurs accorderont en premier lieu leur préférence aux logiciels ouverts tel PostgreSQL. Acheter un logiciel PEUT devenir une attitude archaïque. On a seulement besoin d'acheter un bon matériel, cela vaut mieux de dépenser de l'argent en matériel et trouver le logiciel sur internet.

Puisqu'une masse importante de travail a été effectuée sur PostgreSQL au cours des 12 dernières années, cela n'aurait aucun sens de recréer ex-nihilo un autre système de gestion de base de données satisfaisant aux normes ANSI/ISO SQL. Il est bien plus intéressant de prendre le code existant de PostgreSQL, de commencer à l'utiliser, de l'améliorer et d'y ajouter les fonctionnalités manquantes.

PostgreSQL n'est pas seulement un SGBD libre mais aussi un "Produit Internet" et à ce titre mérite respect et reconnaissance de tout un chacun.

2 Qu'est-ce que PostgreSQL?

PostgreSQL Version 6.2.1 patch level 3 est un SGBD libre, son code source complet est fourni. De plus, c'est un SGBD Relationnel-Objet pratiquement conforme (de plus en plus conforme) aux normes ANSI SQL1998,92,89. Il fonctionne sur diverses plates-formes matérielles sous différents Systèmes d'Exploitation. PostgreSQL est une amélioration du SGDB POSTGRES, prototype de recherche de SGDB de la prochaine génération. PostgreSQL, tout en conservant le puissant modèle de données et la richesse des types de données de POSTGRES, remplace le langage de requête PostQuel par un sous-ensemble étendu de SQL.

Le développement de PostgreSQL est réalisé par une équipe de développeurs Internet qui sont tous inscrits à la liste de diffusion "PostgreSQL development mailing list". Son coordinateur actuel est Marc G. Fournier scrappy@postgreSQL.org . Cette équipe est maintenant responsable des développements actuels et futurs de PostgreSQL.

Les auteurs de PostgreSQL 1.01 sont Andrew Yu et Jolly Chen. De nombreux autres ont contribué au portage, aux tests, à la mise au point et a l'amélioration du code. Le code original Postgres, duquel PostgreSQL est issu, est le résultat de l'effort de nombreux étudiants de troisième cycle, de deuxième cycle et d'enseignants sous la direction du Professeur Michael Stonebraker de l'université de Californie, Berkeley.

Le nom original du logiciel à Berkeley était Postgres. Lors de l'ajout des fonctionnalités SQL en 1995, il fut renommé Postgres95. Ce nom fut changé à la fin de 1996 en PostgreSQL.

Des millions d'exemplaires du SGBD PostgreSQL sont installés comme serveurs, serveurs Web et serveurs d'application. Ce SGBD est très avancé, c'est un SGBD Relationnel-Objet (ORDBMS).

PostgreSQL peut stocker plus de types de données que les types traditionnels entier, caractères, etc. - L'utilisateur peut créer des types, des fonctions, de l'héritage de type etc. (La Version 7.0 rendra ces fonctions avancées encore plus puissantes). PostgreSQL fonctionne sur Solaris, SunOS, HPUX, AIX, Linux, Irix, Digital Unix, BSDi, NetBSD, FreeBSD, SCO unix, NEXTSTEP, Unixware et toutes sortes d'unix. Un portage pour Windows 95/NT est en cours de réalisation.

- Titre : PostgreSQL SQL RDBMS Database (Système de Gestion de Base de Données Relationnelle Objet)
- Version actuelle : 6.2.1 patch level 3
- Age : PostgreSQL a 12 ans. En développement depuis 1985
- Auteurs : Développé par des millions d'universités/compagnies sur internet au cours des 12 dernières ANNEES

3 Où le trouver?

On peut acheter le CDROM Redhat 4.2 qui contient déjà postgresql en paquetage rpm (a la fois en code source et en binaires) chez :

- Linux System Labs Web site : `(http://www.lsl.com/)` 6 (U.S. dollars)
- Cheap Bytes Inc Web site : `(http://www.cheapbytes.com/)` 6 (U.S. dollars)

Distributions uniquement en binaire de PostgreSQL :

- On peut exécuter PostgreSQL sans compiler le source. Récupérer les binaires pour Intel-Linux de l'url `(http://www.redhat.com/pub/contrib/i386/)` le fichier se nomme postgresql-6.2-3.i386.rpm. Son format est de type paquetage redhat 'rpm'. Il contient à la fois le source et les binaires de PostgreSQL.
- Sites binaires pour Solaris, HPUX, AIX, IRIX, Linux : `(ftp://ftp.postgresql.org/pub/bindist)` Si vous faites une compilation pour n'importe laquelle des plates-formes envoyer le binaire à ce site, cela peut être utile aux autres.
- ftp site : récupérer les binaires pour Intel-Linux de `(ftp://ftp.redhat.com/pub/contrib/i386/)` le fichier se nomme postgresql-6.2-3.i386.rpm. Son format est de type paquetage redhat 'rpm'. Il contient à la fois le source et les binaires de PostgreSQL..

Sites WWW Web :

- Site Web primaire : `(http://www.postgresql.org/)`
- Site Web secondaire : `(http://logical.thought.net/postgres95/)`
- `(http://www.itm.tu-clausthal.de/mirrors/postgres95/)`
- `(http://s2k-ftp.cs.berkeley.edu:8000/postgres95/)`

– (http://xenium.pdi.net/PostgreSQL/)

– (http://s2k-ftp.cs.berkeley.edu:8000/postgres95/)

Les sites ftp sont indiqués ci-dessous :-

– Primaire FTP : (ftp://ftp.postgresql.org/pub)

– Secondaire FTP : (ftp://ftp.chicks.net/pub/postgresql)

– (ftp://ftp.emsi.priv.at/pub/postgres/)

– (ftp://ftp.itm.tu-clausthal.de/pub/mirrors/postgres95)

– (ftp://rocker.sch.bme.hu/pub/mirrors/postgreSQL)

– (ftp://ftp.jaist.ac.jp/pub/dbms/postgres95)

– (ftp://ftp.luga.or.at/pub/postgres95)

– (ftp://postgres95.vnet.net:/pub/postgres95)

– (ftp://ftpza.co.za/mirrors/postgres)

– (ftp://sunsite.auc.dk/pub/databases/postgresql)

– (ftp://ftp.task.gda.pl/pub/software/postgresql)

– (ftp://xenium.pdi.net/pub/PostgreSQL)

Le code source de PostgreSQL est aussi disponible sur tous les sites miroirs de sunsite.unc (soit environ 1000 sites autour du globe). Il se trouve dans la distribution Linux Red Hat dans le fichier /pub/contrib/i386/postgresql.rpm.

– Pour obtenir la liste des sites miroirs aller à l'url (ftp://sunsite.unc.edu)

4 PostgreSQL Supporte des Base de Données extrêmement grandes ¿200 Gigas

Si vous avez besoin d'utiliser des bases de données extrêmement grandes (supérieures a 5 gigaoctets), il est fortement recommandé d'utiliser des machines 64-bit telles que : Digital Alpha cpu, Sun Ultra-sparc 64-bit cpu, Silicon graphics 64-bit cpu, à venir Intel Merced IA-64 cpu, machines HPUX 64bit , IBM 64-bit machines. Si l'on compile PostgreSQL avec un cpu 64-bit il pourra supporter d'énormes bases de Données et de larges requêtes. Les performances de PostgreSQL pour des interrogations sur de grandes tables et de grandes bases de données sera plus rapide de plusieurs ordres de grandeurs que sur des machines à cpu 32-bit . L'avantage des machines 64-bit est qu'elles disposent d'un grand espace d'adressage mémoire et que le système d'exploitation peut gérer de très grands systèmes de fichiers. Cela permet des meilleures performances avec de grandes bases de données, plus de mémoire centrale (RAM), plus de possibilités etc...

5 PostgreSQL est-il fiable?

Le paquetage de "Test de Régression" permet de s'assurer de son bon fonctionnement. Il est inclus (src/test/regress) dans la distribution et permet à l'ordinateur d'effectuer la vérification des opérations SQL standard ainsi que des capacités de PostgreSQL. L'avantage de ce test effectué par l'ordinateur vient du fait que celui-ci peut effectuer plusieurs millions de test SQL très rapidement. La vitesse de l'ordinateur est un milliard de fois plus rapide que celle de l'esprit humain! Le paquetage de tests contient des centaines de programmes de test SQL. Si vous le jugez utile vous pouvez en ajouter beaucoup d'autres. Dans ce cas penser à envoyer ces tests au site PostgreSQL primaire si vous pensez que cela peut être utile aux autres. Le paquetage de Test de Régression permet d'asseoir la confiance des utilisateurs en PostgreSQL et facilite le déploiement rapide de PostgreSQL sur des systèmes en production sans inquiétude majeure.

6 Outil GUI frontal pour PostgreSQL (Interface Utilisateur Graphique)

PostgreSQL est doté d'une librairie d'interface TCL/TK dans la distribution appelée 'pgtcl'. TCL/TK est un outil de développement rapide d'application et un langage de script extrêmement puissant. On développe une fois pour toutes et on utilise sur NT, Win 95, Linux et tous les systèmes Unix! TCL/TK est également largement utilisé comme langage de script sur internet. Ce qui permet de n'avoir qu'un seul langage pour couvrir tous ses besoins - applications et internet. TCL est l'abréviation de 'Tool Command Language' et TK celle de 'Tool Kit'. Il existe un IDE (Environnement de Développement Intégré) pour TCL/TK nommé SpecTCL. Vérifiez la présence de ce paquetage (format rpm) dans la distribution linux Redhat ou dans l'un des sites sunscript indiquées ci-après. TCL/TK est normalement inclus dans tous les cdrom linux. On peut également l'obtenir à partir des sites -

- `(http://sunscript.sun.com/)`
- `(http://sunscript.sun.com/TclTkCore/)`
- `(ftp://ftp.sunlabs.com/pub/tcl/tcl8.0a2.tar.Z)`
- Manuels de référence: de nombreux livres sur TCL/TK sont disponibles sur le marché.
- Visual TCL `(ftp://ftp.redhat.com/pub/contrib/i386/visualtcl*.rpm)`

7 Outils de développement intégrés pour PostgreSQL (Interface Utilisateur Graphique)

Essayez les outils de développement suivants utilisable en conjonction avec les pilotes odbc/jdbc. Ils sont similaires à Borland C++ Builder, Borland JBuilder.

Vibe est un IDE Java et C++ (Environnement de Développement Intégré) a été nommé IDE de l'année par la revue "Unix Review". Ce produit à 800 US dollars est disponible pour linux au prix extrêmement bas de

79 US dollars pour un durée limitée. Essayez le! C'est le rêve du développeur.

- Des informations complémentaire sur Vibe sont disponibles à (http://www.LinuxMall.com/products/00487.html)

Vous pouvez aussi utiliser Borland C++ Builder, JBuilder, PowerBuilder sur Windows95 pour se connecter à PostgreSQL sur une machine unix au travers de pilotes odbc/jdbc.

Outils IDE gratuits -

Regardez le CDROM de la distribution RedHat.

- FreeBuilder (ftp://ftp.redhat.com/pub/contrib/i386/free*.rpm)
- SpecTCL (ftp://ftp.redhat.com/pub/contrib/i386/spec*.rpm)
- JccWarrior (ftp://ftp.redhat.com/pub/contrib/i386/jcc*.rpm)
- Applixware Tool (http://www.redhat.com)
- XWPE X Windows Programming Environment (http://www.rpi.edu/ payned/xwpe/) (ftp://ftp.redhat.com/pub/contrib/i386/xwpe*.rpm)
- XWB X Windows Work Bench (ftp://ftp.redhat.com/pub/contrib/i386/xwb*.rpm)
- NEdit (ftp://ftp.redhat.com/pub/contrib/i386/nedit*.rpm)

8 ODBC Pilotes pour PostgreSQL

ODBC signifie 'Open DataBase Connectivity'. C'est un standard répandu pour accéder aux informations des bases de données de différents vendeurs. Le fonctionnement des applications écrites en utilisant les pilotes ODBC est garanti quelle que soient les bases de données utilisées PostgreSQL, Oracle, Sybase, Informix etc..

- (http://www.openlinksw.com) Open Link Software Corporation distribue des ODBC pour PostgreSQL et d'autres bases de données. Open Link dispose aussi d'ODBC gratuits (quantité limitée) vérifier auprès d'eux.
- (http://stud1.tuwien.ac.at/ e9025461/) C'est le site primaire de PostODBC (PostgreSQL ODBC) . Malheureusement c'est un site particulièrement lent.
- (http://www.MageNet.com/postodbc/DOC) Ce miroir est maintenu par Julia Anne Case qui est aussi un développeur PostODBC majeur.

Il existe aussi un projet intéressant appelé FreeODBC Pack Package . Il n'y a pas de version PostgreSQL, peut-être pouvez vous y participer.

- (http://www.ids.net/ bjepson/freeODBC/) c'est une version gratuite d'ODBC.

9 Pilotes UDBC pour PostgreSQL

UDBC est une version statique de pilote de gestionnaire et de DLL d'ODBC indépendant, pour intégrer le support de la connectivité base de données directement au niveau des applications.

– (http://www.openlinksw.com) Open Link Software Corporation vends UDBC pour PostgreSQL et autres SGBD. Open Link dispose aussi d'UDBC gratuits (quantité limitée) vérifier auprès d'eux.

10 Pilotes JDBC pour PostgreSQL

JDBC signifie 'Java DataBase Connectivity'. Java est un langage de développement indépendant de la plate-forme d'utilisation développé par Sun Microsystems. Les programmeurs Java sont encouragés à écrire leurs applications en utilisant JDBC pour faciliter la portabilité entre différentes plates-formes telles que PostgreSQL, Oracle, informix, etc. Si vous écrivez des applications Java applications vous pouvez obtenir les pilotes JDBC pour PostgreSQL à partir des sites suivants :

Le pilote JDBC est inclus dans la distribution PostgreSQL.

– (http://www.demon.co.uk/finder/postgres/index.html) Sun's Java connectivity to PostgreSQL

– (ftp://ftp.ai.mit.edu/people/rst/rst-jdbc.tar.gz)

– (http://www.openlinksw.com) Open Link Software Corporation vends JDBC pour PostgreSQL et autres SGBD. Open Link dispose aussi d'JDBC gratuits (quantité limitée) vérifier auprès d'eux.

11 Kanchenjunga - Java RAD Tool for PostgreSQL

Kanchenjunga est un Outil de Développement Rapide d'Application Java pour PostgreSQL. On peut utiliser cet outil pour développer rapidement une application JAVA s'interfaçant à PostgreSQL.

– (http://www.man.ac.uk/ whaley/kj/kanch.html)

12 Classes Java pour PostgreSQL

Ces classes pour PostgreSQL seront très utiles au programmeur JAVA.

– (ftp://www.blackdown.org/pub/Java/Java-Postgres95)

– (http://www.blackdown.org)

13 Pilote d'interface de SGBD Perl (Database Interface DBI) pour PostgreSQL

13.1 interface Perl 5 pour PostgreSQL

Cette interface est incluse dans la distribution de PostgreSQL. On la trouve dans le répertoire src/pgsql_perl5.

- Email : E.Mergl@bawue.de
- On la trouve aussi dans -
- Page d'accueil Perl :

13.2 Qu'est-ce que DBI ???

L'interface de SGBD Perl (Perl Database Interface - DBI) est une interface logicielle d'accès à un SGBD (Application Programming Interface - API) pour le langage PERL. Les spécifications de l'API DBI perl définissent un ensemble de fonctions, de variables et de conventions d'accès à un SGDB cohérent et indépendant du SGBD utilisé.

13.3 Annonce du pilote DBI DBD-Pg-0.63 DBI pour PostgreSQL

Sur le site CPAN on trouve DBD-Pg-0.63.tar.gz. Depuis sortie de la dernière version publique, les modifications suivantes ont été effectuées :

- - adaptation à PostgreSQL-6.2 : o $ sth-¿rows de même que $ sth-¿execute et $ sth-¿do renvoient le nombre de lignes affectées même pour les instructions non-Select. o ajout du support d'autorisation par mot de passe, lire la page du manuel concernant pg_passwd.

- - the data_source parameter of the connect method accepts two additional parameters which are treated as host and port: DBI-¿connect("dbi:Pg:dbname:host:port", "uid", "pwd")

- - support de l'instruction AutoCommit. Lire la documentation de ce module pour évaluer son impact sur vos scripts !

- - prise en compte dans un meilleur style perl du type de donnée bool, Lire la documentation de ce module pour évaluer son impact sur vos scripts !

Pour de plus amples informations voir:

13.4 Notes concernant cette version et fichier LISEZ-MOI

```
#---------------------------------------------------------
#
# $Id: README,v 1.10 1997/10/05 18:25:55 mergl Exp $
#
# Portions Copyright (c) 1994,1995,1996,1997 Tim Bunce
```

```
# Portions Copyright (c) 1997                Edmund Mergl
#
#---------------------------------------------------------

**********************************************************
*                                                        *
*          Cette version contient des modifications      *
*                      INCOMPATIBLES                     *
*                      _------------                     *
*               avec les versions precedentes.           *
*                                                        *
*             Lire la documentation du module            *
*                pour l'attribut AutoCommit              *
*                et le type de donnee bool.              *
*                                                        *
**********************************************************
```

DESCRIPTION :

———–

Ceci est la version 0.63 de DBD-Pg. DBD-Pg est une interface PostgreSQL pour Perl 5 utilisant DBI.

Pour des informations complémentaires concernant DBI consulter:

COPYRIGHT :

———

La distribution de ce document est soumise aux conditions de la licence publique générale GNU ou la licence protégeant la création Artistique (Artistic License), ainsi que c'est spécifié dans le fichier LISEZ-MOI perl.

SI VOUS AVEZ DES PROBLEMES :

————————–

Envoyez vos commentaires et vos rapports d'erreur à E.Mergl@bawue.de

Pensez à inclure les messages affichés par perl -v, et perl -V, la version de PostgreSQL, la version de DBD-Pg, et la version de DBI dans votre rapport d'erreur.

CONTRAINTES LOGICIELLES :

———————–

- compiler, tester et installer Perl 5 (au moins 5.002) - compiler, tester et installer le module DBI (au moins 0.89) - compiler, tester et installer PostgreSQL (au moins 6.2)

PLATEFORMES :

———–-

Cette version def DBD-Pg a été développée pour Linux 2.0 avec chargement dynamique des extensions perl. Merci de me faire savoir si vous rencontrez des problèmes sur d'autres plates-formes.

INSTALLATION :

————–

Dans le fichier Makefile il y a un test la variable d'environnement POSTGRES_HOME ainsi que de quelques arborescences standard, pour trouver le répertoire racine de votre installation Postgres. Taper les commandes suivantes:

1. perl Makefile.PL 2. make 3. make test 4. make install

(de 1. à 3. comme utilisateur normal, pas comme root !)

TEST :

——

Lancer 'make test'. Noter que l'utilisateur lançant ce script doit avoir été créé avec des droits d'accès lui permettant de créer des bases de données "ET" des utilisateurs !. Ne pas lancer ce script en tant que root!.

Si le test échoue avec le message 'login failed', vérifier que l'accès à template1 de la base de données ainsi qu'à pgperltest ne sont pas protégés dans pg_hba.conf.

Si vous utilisez la bibliothèque partagée libpq.so vérifiez que votre chargeur dynamique trouve libpq.so. La commande Linux /sbin/ldconfig -v devrait vous renseigner sur l'endroit où se trouve libpq.so. Si ldconfig ne trouve pas libpq.so, soit ajoutez une entrée appropriée dans /etc/ld.so.conf et relancez ldconfig ou ajouter son chemin dans la variable d'environnement LD_LIBRARY_PATH. On aura comme message typique résultant de ce type d'erreur : install_driver(Pg) failed: Can't load './blib/arch/auto/DBD/Pg/Pg.so' for module DBD::Pg: File not found at

Quelques distributions Linux ont une installation incomplète de perl. Si vous avez des erreurs de compilation du style "XS_VERSION_BOOTCHECK undeclared", exécutez un 'find .../lib/perl5 -name XSUB.h -print' Si ce fichier n'est pas présent, il est indispensable de recompiler et de réinstaller perl.

Utilisateurs SGI : si vous avez un défaut de segmentation, assurez-vous que vous utilisez la version de malloc obtenue avec perl lors de la compilation de perl (ce n'est pas le cas par défaut). "David R. Noble" drnoble@engsci.sandia.gov drnoble@engsci.sandia.gov

————————————————————————

E.Mergl@bawue.de October 05, 1997

————————————————————————

13.5 FAQ pour DBI

On trouvera ci-dessous les Questions Fréquemment Posées (FAQ) pour DBI. La page Web d'accueil se trouve à

`(http://www.hermetica.com/technologia/perl/DBI)`

```
                             DBI Foire Aux Questions v.0.35
                           Derniere mise a jour : 20 Juin, 1997


     * NAME
     * SYNOPSIS
```

```
* VERSION
* DESCRIPTION
* Information & Sources d'Informations

   * 1.1 Qu'est-ce que DBI, DBperl, Oraperl and *perl?
   * 1.2. Ou puis-je les trouver?
   * 1.3. Ou puis-je trouver plus d'informations?

* Problemes de compilation

   * 2.1. Problemes de compilation ou "echec aux tests!"

* Questions de Plates-formes et de Pilotes

   * 3.1 Quelle est la difference entre ODBC et DBI?
   * 3.2 DBI est-il supporte sur les plates-formes Windows 95 / NT ?
   * 3.3 Puis-je acceder a des bases de donnees Microsoft Access ou SQL-Server avec DBI?
   * 3.4 Y-a-t'il un DBD pour X?
   * 3.5 Qu'est-ce que DBM? Et pourquoi dois-je utiliser DBI a sa place?
   * 3.6 Quand mSQL-2 sera t'il supporte?
   * 3.7 Quel systeme de gestion de base de donnees me recommandez vous?
   * 3.8 Est-ce que X est supporte dans DBI?

* Questions de programmation

   * 4.1 Est-ce que DBI est d'une quelconque utilite pour la programmation CGI?
   * 4.2 Comment puis-je obtenir des temps de connexion plus rapides avec DBD::Oracle et CGI?
   * 4.3 Comment puis-je obtenir des connexions persistantes avec DBI et CGI?
   * 4.4 ``Quand je lance un script perl a partir de la ligne de commande, ca
         marche, mais, quand je le lance a partir de C, ca echoue!" Pourquoi?
   * 5.1 Puis-je faire de l'execution en parallele avec DBI?
   * 5.2 Comment manipuler des donnees BLOB avec DBI?
   * 5.3 Comment puis-je invoquer des procedures enregistrees avec DBI?
   * 5.4 Comment puis-je recuperer les valeurs de retour des procedures enregistrees avec DBI?
   * 5.5 Comment puis-je creer ou detruire une base de donnees avec DBI?
   * 5.6 Comment puis-je enregistrer ou annuler une instruction avec DBI?
   * 5.7 Comment les valeurs NULL sont-elles prises en compte par DBI?
   * 5.8 Qu'est-ce que c'est que ces histoires de methodes func?

* Support et formation

   * Assistance Commerciale
   * Formation

* Autres References
* AUTEUR
* COPYRIGHT
```

```
---------------------------------------------------------------------------

NAME

DBI::FAQ -- Foire Aux Questions pour l'interface de SGBD Perl5

---------------------------------------------------------------------------

SYNOPSIS

    perldoc DBI::FAQ

---------------------------------------------------------------------------

VERSION

La version actuelle de ce document , du 20 Juin, 1997, porte le numero 0.35.

---------------------------------------------------------------------------

DESCRIPTION

Ce document contient les reponses aux questions les plus frequemment posees
 a la fois sur les Mailing Lists DBI et personnellement aux membres de l'equipe
de developpement DBI.

---------------------------------------------------------------------------

Information de base & Sources d'Information

---------------------------------------------------------------------------

1.1 Qu'est-ce que c'est que DBI, DBperl, Oraperl and *perl?

Pour citer Tim Bunce, l'architecte et l'auteur de DBI :

    ``DBI est une interface logicielle d'acces aux bases de donnees (Application
      Programming Interface  -API) pour le langage Perl. Les specifications DBI
      API definissent un ensemble de fonctions, de variables et de conventions
      coherents d'interfacage a une base de donnees independant de la base de
      donnees utilisee.''

En langage simple, l'interface DBI permet aux utilisateurs d'acceder de maniere transparente
a de multiples base de donnees. Ainsi, Si vous vous connectez a une
base de donnees Oracle, Informix, mSQL, Sybase ou n'importe quelle autre, vous n'avez
pas besoin de connaitre les mecanismes sous-jacents de la couche 3GL. L'API
definie par DBI fonctionnera sur tous ces types de bases de donnees.
```

```
On obtient un benefice du meme ordre en ayant la possibilite de se connecter a deux bases de
donnees de differents fournisseurs a l'aide du meme script perl, i.e., je veux lire des
donnees d'une base de donnees Oracle et les inserer dans une Informix a partir du meme
programme. La couche logicielle DBI permet de le realiser simplement et efficacement.

Voici un diagramme decrivant ce principe :

                            [ Architecture DBI  ]

DBperl est le nom ancien des specifications de l'interface. Il est utilise maintenant
pour designer les modules perl4 d'interfacage des bases de donnees tels que oraperl,
isqlperl, ingperl et autres. Ces interfaces n'ont pas d'API standard et ne sont
generalement pas supportes.

Voici une liste des modules DBperl, de leur equivalent DBI correspondants et du
support d'information. Notez que les auteurs cites ici ne maintiennent generalement
pas le module DBI de la base de donnees. Les adresses E-mail n'ont pas ete verifiees
et ne doivent etre utilisees que pour les questions concernant les modules perl4 listes
ci-dessous. Les questions sur les pilotes DBI doivent etre directement
adressees aux listes de diffusion des utilisateurs DBI.

    Module Name SGBD requis         Auteur          DBI
    ----------- -----------------   ------          ---
    Sybperl     Sybase              Michael Peppler DBD::Sybase
                                    <mpeppler@itf.ch>
    Oraperl     Oracle 6 & 7        Kevin Stock     DBD::Oracle
                                    <dbi-users@fugue.com>
    Ingperl     Ingres              Tim Bunce &     DBD::Ingres
                                    Ted Lemon
                                    <dbi-users@fugue.com>
    Interperl   Interbase           Buzz Moschetti  DBD::Interbase
                                    <buzz@bear.com>
    Uniperl     Unify 5.0           Rick Wargo      None
                                    <rickers@coe.drexel.edu>
    Pgperl      Postgres            Igor Metz       DBD::Pg
                                    <metz@iam.unibe.ch>
    Btreeperl   NDBM                John Conover    SDBM?
                                    <john@johncon.com>
    Ctreeperl   C-Tree              John Conover    None
                                    <john@johncon.com>
    Cisamperl   Informix C-ISAM     Mathias Koerber None
                                    <mathias@unicorn.swi.com.sg>
    Duaperl     X.500 Directory     Eric Douglas    None
                User Agent

Cependant, certains modules DBI possedent des couches logicielles d'emulation. Ainsi
DBD::Oracle est livre avec une couche d'emulation Oraperl, ce qui permet d'executer
d'anciens scripts oraperl sans modification. La couche logicielle d'emulation traduit
```

```
les appels oraperl API en appels DBI et les execute.

Voici une table des  couches d'emulation :

    Module        Couche d'emulation     Etat
    ------          ---------------     ------
    DBD::Oracle     Oraperl             Complete
    DBD::Informix   Isqlperl            En cours de  developpement
    DBD::Sybase     Sybperl             Fonctionnelle? ( Necessite une
                                        verification)
    DBD::mSQL       Msqlperl            En version experimentale avec
                                        DBD::mSQL-0.61

L'emulation Msqlperl est un cas particulier. Msqlperl est un pilote perl5 pour les
bases de donnees mSQL , mais il ne se conforme pas aux specifications DBI. On
desapprouve son utilisation en faveur de  DBD::mSQL. On peut telecharger Msqlperl a
partir du site CPAN via :

    http://www.perl.com/cgi-bin/cpan_mod?module=Msqlperl

----------------------------------------------------------------------------

1.2. Ou puis-je le trouver?

DBI est disponible en premier sur :

    ftp://ftp.demon.co.uk/pub/perl/db

Il faut utiliser le site CPAN (Comprehensive Perl Archive Network) pour recuperer les
versions a jour des pilotes, en general absentes des sites miroirs. On peut acceder
a CPAN grace au splendide programme "CPAN multiplexeur" de Tom Christiansen's situe
a:

    http://www.perl.com/CPAN/

Pour des informations plus specifiques ainsi que pour les URL exactes des pilotes,
veuillez consulter la liste des pilotes DBI et les pages concernant les modules sur:

    http://www.hermetica.com/technologia/perl/DBI

----------------------------------------------------------------------------

1.3. Ou puis-je trouver plus d'informations?

Il existe quelques sources d'information sur  DBI.

Specifications DBI
```

http://www.hermetica.com/technologia/perl/DBI/doc/dbispec

On trouve deux specifications disponibles a cette adresse: la nouvelle specification Draft (edition provisoire) DBI qui est un document en evolution rapide a mesure que l'equipe de developpement s'approche d'une version stable de l'interface, et l'ancienne specification historique DBperl a partir de laquelle l'interface DBI actuelle a evolue.

Il faut considerer ce dernier document comme ne presentant qu'un interet historique et ne pas l'utiliser en tant que manuel de programmation ou document de reference. Il demeure cependant une source d'informations tres utile.

Documentation POD (Plain Old Documentation)

Les PODs sont des morceaux de documentation generalement noyes a l'interieur des programmes perl qui documentent le code "sur place". Ce sont des ressources tres utiles pour les programmeurs et les utilisateurs des modules. Les PODs pour DBI et pour les pilotes deviennent monnaie courante et la documentation pour les modules contenant ces PODs peut etre lue avec les commandes suivantes.

La Specification DBI

Les PODs pour la specification DBI peut etre lue avec la commande :

```
perldoc DBI
```

Oraperl

Les utilisateurs de la couche d'emulation fournie avec DBD::Oracle, peuvent s'informer sur la maniere de programmer en utilisant l'interface Oraperl en tapant:

```
perldoc Oraperl
```

Ce qui permettra d'obtenir une copie a jour de la page de manuel originale ecrite par Kevin Stock pour perl4. L'API oraperl y est entierement listee et decrite.

DBD::mSQL

Les utilisateurs du module DBD::mSQL peuvent lire des informations sur quelques fonctions privees et bizarreries de ce pilote en tapant :

```
perldoc DBD::mSQL
```

Foire Aux Questions (FAQ)

Ce document, la Foire Aux Questions, est aussi disponible en tant que documentation POD! Vous pouvez le lire sur votre propre systeme en tapant :

```
            perldoc DBI::FAQ

        Ceci peut etre plus pratique pour ceux qui ne sont pas connectes a
        l'Internet ou le sont d'une maniere peu pratique.

    Les POD en general
        On peut lire des informations sur la maniere d'ecrire des PODs,
        ainsi que sur la philosophie des PODs en general en tapant :

            perldoc perlpod

        Les utilisateurs ayant le module Tk installe seront peut-etre
        interesses d'apprendre qu'il existe un lecteur de POD base sur Tk
        nomme tkpod. Il formate les POD de maniere pratique et lisible.

Discussions, Cancans et Observations

         http://www.hermetica.com/technologia/perl/DBI/tidbits

     Il y a , de temps en temps, une serie de discussions de la part de
     certaines personnes, dans les listes de diffusion sur DBI,
     qui, pour essayer d'eclaircir un simple point, finissent par
     transformer en brouillon des documents tout a fait complets. Ces
     documents sont souvent de qualite variable, mais donnent un
     apercu du fonctionnement des interfaces.

``DBI -- L'interface de SGBD en perl5''
     C'est un article ecrit par Alligator Descartes et Tim Bunce sur la structure
     de DBI. Il a ete publie dans le numero 5 de ``The Perl Journal''.
     Il est extremement bon. Allez acheter ce magazine. En fait, achetez les tous!
     Le site WWW de ``The Perl Journal'' est :

         http://www.tpj.com

``DBperl''
     Cet article, publie dans l'edition de novembre 1996 du ``Dr. Dobbs
     Journal'' traitait de DBperl. L'auteur de cet article n'a apparemment
     pas contacte un seul membre de l'equipe de developpement DBI pour
     verifier l'information contenue dans son article. Plusieurs critiques
     de cet article dans les listes de diffusion des utilisateurs
     de dbi ont ete peu flatteuses, c'est le moins que l'on puisse dire.
     Le fait que l'article traite de DBperl au lieu de DBI est un indice
     du manque de fraicheur de l'information.

     Cependant, cette reference est donnee par soucis d'exhaustivite.

``The Perl5 Database Interface''
     Cette reference est celle d'un livre a ecrire par Alligator Descartes
```

```
(pour lui, c'est moi) publie par O'Reilly et Associes a paraitre
cet hiver.

La table des matieres de ce livre devrait contenir :

    * Introduction
        + Les Bases de Donnees
        + CGI / WWW
        + perl
    * Concepts de Base des Bases de donnees
        + Types de Bases de Donnees
            o Flat File
            o AnyDBM
            o RDBMS
        + Utiliser Quelle Base de Donnees, pour Quoi faire...
    * SQL
        + Pourquoi SQL?
        + Structuration de l'Information dans les Bases de Donnees
        + Extraction des Donnees d'une Base de Donnees
        + Manipulation des Donnees et des Structures de Donnees
    * Architecture DBI
    * Programmation avec DBI
        + Initialisation DBI
        + Identifiants
            o Identifiants de pilotes
            o Identifiants de bases de donnees
            o Identifiants d'instructions
        + Connexion and Deconnexion
        + Gestion des Erreurs
        + Emission de Requetes Simples
        + Execution d'Instructions Atomiques
        + Instructions MetaDonnees
        + Instruction plus dans le style perl
        + Liaison
        + Gestion des Transactions
        + Methodes utilitaires
        + Gestion des Attributs et des Variables Dynamiques
    * DBI et ODBC
    * Les Pilotes de Bases de Donnees
        + DBD::Oracle et oraperl
        + DBD::Informix et isqlperl
        + DBD::mSQL et Msqlperl
    * Etude de Cas
        + DBI et le WWW
        + Migration des Donnees et Stockage
        + Logiciel d'Administration
    * Appendice: API Reference / Specification
    * Appendice: Ressources
```

```
Fichiers LISEZMOI
     Les fichiers LISEZMOI fournis avec chaque pilote contiennent de temps
     en temps quelques informations utiles (non, vraiment! ) pouvant etre
     pertinentes pour l'utilisateur.
     S'il vous plait, lisez-les. Cela rendra nos pauvres existences plus
     supportables. On peut toutes les trouver depuis la page WWW DBI
     principale a :

         http://www.hermetica.com/technologia/perl/DBI

Listes de diffusion
     Il y a trois listes de diffusion pour DBI gerees par Ted Lemon.
     On peut s'inscrire a toutes et resilier cette inscription a
     travers le World Wide Web a l'URL :

         http://www.fugue.com/dbi

     Les listes ou les utilisateurs peuvent participer sont:

     dbi-announce
          Cette liste de diffusion est reservee uniquement aux annonces.
          Tres peu de trafic. Les annonces sont generalement postees
          sur la page WWW DBI principale.

          Si vous n'arrivez pas a utiliser le formulaire sur la page WWW
          indiquee ci-dessus, inscrivez-vous a cette liste de la maniere
          suivante :

              Email: 'dbi-announce-request@fugue.com' avec le mot 'subscribe'
          dans le corps du message.

     dbi-dev
          Cette liste de diffusion est a l'usage des developpeurs pour discuter
          des idees et des concepts de l'interface DBI, API et des mecanismes
          des pilotes. Seulement utiles pour les developpeurs et les personnes
          interessees. Trafic faible.

          Si vous n'arrivez pas a utiliser le formulaire sur la page WWW
          indiquee ci-dessus, inscrivez-vous a cette liste de la maniere
          suivante :

              Email: 'dbi-dev-request@fugue.com' with a message body of
              'subscribe'

     dbi-users
          Cette liste de diffusion est un lieu de discussion generale utilisee
          pour les rapports d'erreurs, la discussion sur differents problemes
```

```
          et des demandes de renseignement d'interet general. Trafic moyen.

          Si vous n'arrivez pas a utiliser le formulaire sur la page WWW
          indiquee ci-dessus, inscrivez-vous a cette liste de la maniere
          suivante :

              Email: 'dbi-users-request@fugue.com' with a message body of
              'subscribe'

Archives des Listes de Diffusion
     Archives des Listes de Diffusion US

              http://outside.organic.com/mail-archives/dbi-users/

         Cette archive, avec recherches par hyper-liens, de ces trois listes de
         diffusion, avec un peu du trafic beaucoup plus ancien a ete mis en place
         pour permettre une recherche par les utilisateurs.

     Archives des Listes de Diffusion Europeennes

              http://www.rosat.mpe-garching.mpg.de/mailing-lists/PerlDB-Interest

          Identique a l'archive US ci-dessus.

----------------------------------------------------------------------------

Problemes de compilation

----------------------------------------------------------------------------

2.1. Problemes de compilation ou "Il echoue aux tests!"

En premier lieu, consulter la documentation en ligne concernant ce module, que ce
soit DBI lui-meme ou un module DBD, et verifier que ce n'est pas un probleme de
compilation connu pour votre architecture. On peut trouver ces documents a :

    http://www.hermetica.com/technologia/perl/DBI

Si c'est un probleme connu, vous devrez probablement attendre qu'il ait ete corrige.
Si vraiment vous avez besoin d'une solution, essayez l'une des solutions suivantes :

Essayez de le corriger vous meme
     Cette technique n'est generalement par recommandee aux craintifs. Si vous pensez
     que vous y etes arrive, alors, envoyer un fichier patch (context diff ) a
     l'auteur en expliquant les points suivants :

        o Quel etait le probleme, et, si possible, des jeux d'essai.
```

```
        o Ce que vous avez du faire pour le corriger. Assurez vous que vous
          n'oubliez rien.

        o Donnez des informations concernant la Plate-forme utilisee, les versions :
          de la Base de Donnees, de Perl, du module et de DBI.

Envoyez un Email a l'auteur SANS RALER!
     S'il vous plait, postez votre email a l'adresse indiquee dans les pages
     WWW du pilote avec lequel vous avez rencontre des problemes. Ne pas poster
     directement a une adresse que vous connaissez a moins qu'elle ne
     corresponde a l'une de celles qui sont indiquees.

     Nous avons un vrai travail a faire, et nous devons consulter les listes de
     diffusion traitant des problemes rencontres. De plus nous ne pouvons pas
     forcement avoir acces a <inserez ici le nom de votre plate-forme favorite,
     celle qui vous cause des lesions cerebrales> et de ce fait vous apporter
     une quelconque assistance!
     Desole de vous paraitre dur, mais c'est comme cela!

     Cependant, vous pouvez tomber sur l'un de ces genies creatifs, qui, a 3
     heures du matin resoudra votre probleme et vous fournira une rustine en 5
     minutes. L'ambiance dans le cercle DBI est que nous apprecions de
     connaitre les problemes des utilisateurs puisque nous travaillons dans
     le meme environnement.

     Si vous envisagez d'envoyer un Email a un auteur, essayez de fournir le
     plus d'informations possible, i.e. :

        o TOUS les renseignements provenant du fichier LISEZMOI du module
          posant probleme.
          Je dis bien TOUS. Nous n'ajoutons pas de lignes dans la documentation
          pour le plaisir, ou pour que les fichiers LISEZMOI se conforment
          a une quelconque norme de taille.

        o Si vous disposez d'un vidage memoire, essayer de generer une trace du
          contenu de la pile (stack) a partir du vidage memoire (core dump) en
          utilisant le module Devel::CoreStack. L'envoyer egalement. On peut
          trouver le module Devel::CoreStack sur CPAN a :

              http://www.perl.com/cgi-bin/cpan_mod?module=Devel::CoreStack

        o Les numeros de version du Module, de perl, des jeux d'essai, du
          systeme d'exploitation ainsi que tout autre information pertinente.

     Souvenez-vous que, plus vous nous enverrez d'informations plus vite
     nous pourrons resoudre le probleme. Si vous n'envoyez rien, n'attendez
     rien en retour.
```

```
Envoyez un Email dans la liste de diffusion des utilisateurs de dbi
     C'est en general une idee astucieuse de rapporter les problemes
     rencontres dans les listes de diffusion. Les auteurs lisent tous
     ces listes d'une part et d'autre part vous ne perdez rien a le faire,
     donc, faites-le.

----------------------------------------------------------------------------

Questions de Plates-formes et de Pilotes

----------------------------------------------------------------------------

3.1 Quelle est la difference entre ODBC et DBI?

Ca, c'est une bonne question! Reponse a rediger de maniere detaillee!

----------------------------------------------------------------------------

3.2 DBI est-il supporte sur les plates-formes Windows 95 / NT ?

Finalement, oui! Jeff Urlwin s'est employe avec constance a porter DBI et DBD::Oracle
sur ces plates-formes, et, depuis la disponibilite d'un perl plus stable et d'un
portage de MakeMaker, le projet a progresse a pas de geant.

Les portages de DBI et de DBD::Oracle pour Win32 ports font maintenant partie
integrante de DBI, donc, la recuperation d'une version de DBI superieure a 0.81 doit
donner satisfaction. Pour ce qui est des rustines necessaires pour DBD::Oracle,
veuillez lire la page d'information de portage pour Win32 a :

    http://www.hermetica.com/technologia/perl/DBI/win32

----------------------------------------------------------------------------

3.3 Puis-je acceder aux bases de donnees Microsoft Access ou SQL-Server avec DBI?

    Contribution de Tim Bunce et Jeff Urlwin

Il existe une couche logicielle d'emulation experimentale pour le module Win32::ODBC
fournie avec DBI-0.79 ( et suivants ). Elle s'appelle DBI::W32ODBC et est, pour le
moment, minimale. Vous aurez besoin du module Win32::ODBC disponible a :

    http://www.roth.net

Etant donne son etat, les rapports de problemes, sans correction, ont de bonnes
chances d'etre ignores. Vous aurez egalement besoin du patch kit Win32 DBI tel qu'il
est fourni par Jeff Urlwin. Pour trouver l'endroit ou il est disponible, lire la
reponse a la question precedente.
```

Jeff Urlwin fournit actuellement un gros travail sur la couche logicielle ODBC.
Pour en revenir a la question initiale, theoriquement, oui, on peut acceder aux bases de donnees Microsoft Access ou SQL-Server avec DBI via ODBC!

3.4 Existe-t-il in DBD pour <inserez le nom de votre SGBD favori ici>?

Est-il cite dans la liste des pilotes DBI ?

    http://www.hermetica.com/technologia/perl/DBI/DBD

Sinon, non. L'absence complete d'un pilote pour un SGBD donne dans cette page signifie que personne n'a manifeste l'intention d'y travailler.

Comme corollaire a l'affirmation ci-dessus, cela signifie que si vous voyez une annonce pour un pilote qui n'est pas dans cette page, il y a une bonne chance que ce soit reellement un pilote DBI, et qu'il puisse ne pas respecter les specifications. Par consequent, les questions pour des problemes concernant ce code ne doivent pas etre posees dans les listes de diffusion DBI.

3.5 Qu'est-ce que DBM? Et pourquoi dois-je utiliser DBI a sa place?

Extrait de ``DBI - L'Interface de Base de Donnees pour Perl 5'' :

``A l'origine UNIX etait bienheureux avec sa "Base de Donnees" rustique reposant sur des fichiers, nommee systeme dbm. Avec dbm vous enregistrez les donnees dans des fichiers et les retrouvez rapidement. Cependant, il souffre de serieux inconvenients.

Verrouillage des fichiers

Les systemes dbm ne permettent par un verrouillage particulierement robuste des fichiers, de meme qu'il n'y a pas de possibilite de corriger les problemes survenants lors d'ecritures [dans la base de donnees] simultanees.

Structures de Donnees Arbitraires

Les systemes dbm permettent seulement une simple structure de donnees fixe: paires cle-valeur. Cette valeur peut etre un objet complexe, tel qu'une structure [C], mais la cle doit etre unique. Ce fut une grande limitation dans l'utilite des systemes dbm.

Cependant, les systemes dbm continuent a offrir des fonctions utiles pour les utilisateurs ayant des ensembles de donnees simples et des ressources limitees,

```
    puisqu'ils sont rapides, robustes et extremement bien testes. Les modules Perl
    pour acceder aux systemes dbm font maintenant partie integrante de la
    distribution Perl via le module AnyDBM_File.''

Pour resumer, DBM est une solution parfaitement satisfaisante pour les bases de
donnees essentiellement en lecture seule, ou pour des ensembles de donnees simples
et reduits. Toutefois, pour des ensembles de donnees plus importants, sans
mentionner un verrouillage des transactions robuste, on recommandera aux utilisateurs
de preferer DBI.

----------------------------------------------------------------------------

3.6 Quand mSQL-2 sera t'il supporte?

De meme que pour DBD::mSQL-0.61, il y a eu un support pour mSQL-2. Cependant, il
n'y a encore aucun reel support pour les nouvelles methodes func, concernant les
index, ajoutees a la bibliotheque de base mSQL. Celles-ci seront prochainement
disponibles et seront accessibles a DBD::mSQL au travers de methodes func privees.
Vous pouvez obtenir plus d'informations concernant ces methodes func privees dans
le POD DBD::mSQL en tapant :

    perldoc DBD::mSQL

a condition d'avoir une installation correcte de DBD::mSQL.

----------------------------------------------------------------------------

3.7 Quel systeme de gestion de base de donnees me recommandez vous?

C'est un sujet particulierement epineux pour lequel une reponse objective est
delicate dans la mesure ou, chaque ensemble de donnees, chaque usage et chaque
configuration du systeme est different d'un utilisateur a l'autre.

Du point de vue de l'auteur, si l'ensemble de donnees est relativement petit, donnant
des tables de moins de 1 million de lignes, et qu'il y ait moins de 1000 tables dans
une base de donnees, alors mSQL est une solution parfaitement acceptable. Ce SGBD
est extremement bon marche, est merveilleusement robuste et beneficie d'un excellent
support. Des informations complementaires sont disponibles sur le site WWW Hughes
Technology a :

    http://www.hughes.com.au

Si l'ensemble de donnees entraine des tables de plus de  1 million de lignes ou plus
de 1000 tables, ou si vous disposez de, soit plus d'argent, soit de machines plus
puissantes, je vous recommanderai le RDBMS (SGBDR en francais) Oracle7. Le site WWW
Oracle's est une excellente source d'information complementaire.

    http://www.oracle.com
```

```
Informix est un autre RDBMS de haut niveau qu'il faut envisager. Il existe plusieurs
differences entre Oracle et Informix qui sont trop complexes pour etre detaillees
dans ce document. On trouvera l'information sur Informix sur leur site WWW a :

    http://www.informix.com

En cas d'utilisation dans des applications en frontal WWW, mSQL peut etre un
meilleur choix du fait de ses temps de connexion courts entre le script CGI et le
SGBD que Oracle RDBMS qui, a chaque connexion, reclame plus de ressources. mSQL est
moins gourmand en ressources et plus rapide.

Ce point de vue n'est pas forcement partage par tout le monde et n'est pas sponsorise
ou dicte par une quelconque societe. Il est donne tel quel.

----------------------------------------------------------------------------

3.8 Est-ce que <inserez une fonctionnalite ici> est supporte par DBI?

Si l'on suppose que la fonctionnalite en question n'est pas, en standard, specifique
d'un SGBD, alors la reponse sera non.

DBI represente un API qui doit fonctionner avec la plupart des SGBD, et n'a pas de
fonctionnalite specifique a un SGDB particulier.

Cependant, les auteurs d'un pilote peuvent, s'ils le desirent, ajouter une
fonctionnalite specifique a un SGBD a travers les methodes func definies dans l'API
DBI. Les developpeurs de Scripts doivent noter que l'utilisation de cette
fonctionnalite au travers de ces methodes func a de bonnes chances d'en sacrifier la
portabilite entre les differents SGDB.

----------------------------------------------------------------------------

Questions de Programmation

----------------------------------------------------------------------------

4.1 Est-ce que DBI est d'une quelconque utilite pour la programmation CGI?

En un mot, oui! DBI est extremement utile pour la programmation CGI! En fait, je
serais tente de repondre que la programmation CGI est une des deux principales
utilisation de DBI.

DBI confere aux programmeurs CGI la possibilite d'offrir des base de donnees WWW a
leurs utilisateurs, ce qui donne a ces utilisateurs la possibilite d'utiliser de
grandes quantites de donnees bien organisees. DBI donne aussi la possibilite , si un
site recoit un trafic trop important pour les performances du serveur, d'ameliorer
ce serveur de base de donnees de facon transparente, sans modifier les scripts CGI.
```

```
---------------------------------------------------------------------------

4.2 Comment puis-je obtenir un temps de connexion plus rapide avec DBD::Oracle et CGI CGI?

    Contribution de John D. Groenveld

Le serveur httpd Apache maintient un ensemble de processus fils httpd pour servir les
requetes clients.

En utilisant le module mod_perl Apache de Doug MacEachern, l'interpreteur perl est
inclus dans le processus fils httpd. Les modules CGI, DBI, et vos autres modules
favoris peuvent etre charges au lancement de chaque processus fils. Ces modules ne
seront pas recharges a moins d'etre modifies sur disque.

Pour de plus amples informations sur Apache, consultez le site WWW du Projet Apache
a :

    http://www.apache.org

Le module mod_perl peut etre recupere de CPAN via :

    http://www.perl.com/cgi-bin/cpan_mod?module=mod_perl

---------------------------------------------------------------------------

4.3 Comment puis-je obtenir une connexion persistante avec DBI et CGI?

    Contribution de by John D. Groenveld

En utilisant le module Apache::DBI de Edmund Mergl, les connexions a la base de
donnees sont enregistrees dans une table avec chacun des processus httpd fils. Si
votre application utilise une base de donnees simple utilisateur, cette connexion
peut etre lancee avec chaque processus fils. Actuellement, les connexions a la base
de donnees ne peuvent pas etre partagees entre processus httpd fils.

Apache::DBI peut etre telecharge de CPAN via :

    http://www.perl.com/cgi-bin/cpan_mod?module=Apache::DBI

---------------------------------------------------------------------------

4.4 ``Quand je lance un script perl de la ligne de commande, ca marche, mais, quand
je le lance sous httpd, ca echoue!" Pourquoi?

Fondamentalement, il y a une bonne chance que cela provienne du fait que
l'utilisateur a partir duquel vous avez lance la ligne de commande a
un ensemble de variables d'environnement correctement configure, ce sont, dans la
```

```
cas de DBD::Oracle, des variables telles que $ORACLE_HOME, $ORACLE_SID or TWO_TASK.

Le processus httpd s'execute habituellement sous un utilisateur id ne correspondant
pas a un utilisateur, ce qui implique qu'il n'y a pas d'environnement configure.
Tous scripts essayant de s'executer dans ces circonstances echoueront.

Pour resoudre ce probleme, initialisez l'environnement de votre base de donnees dans
un bloc BEGIN ( ) en tete de votre script. Ceci devrait resoudre votre probleme.

De meme, vous devriez regarder votre fichier registre d'erreurs pour y trouver des
indices, ainsi que les guides ``Idiot's Guide To Solving Perl / CGI Problems'' et
``Perl CGIProgramming FAQ'' pour avoir des informations complementaires. Il est peu
probable que ce probleme concerne DBI.

Le guide ``Idiot's Guide To Solving Perl / CGI Problems'' peut etre trouve a :

    http://www.perl.com/perl/faq/index.html

ainsi que ``Perl CGI Programming FAQ''. Lisez ces DEUX documents tres soigneusement!

---------------------------------------------------------------------------

5.1 Puis-je faire de l'execution en parallele avec DBI?

A la date de ce document ( voir en tete ), non. perl ne permet pas l'execution en
parallele. Cependant, l'execution en parallele doit faire partie de la distribution
perl de base a compter de la version 5.005, ce qui sous-entend que le support de
l'execution en parallele pour DBI devrait suivre rapidement.

Pour quelques exemples de code OCI pour Oracle ayant des instructions SELECT avec
execution en parallele, voir :

    http://www.hermetica.com/technologia/oracle/oci/orathreads.tar.gz

---------------------------------------------------------------------------

5.2 Comment manipuler des donnees BLOB avec DBI?

A ecrire.

---------------------------------------------------------------------------

5.3 Comment puis-je invoquer des procedures enregistrees avec DBI?

En supposant que vous avez cree une procedure enregistree a l'interieur de la base
de donnees cible, eg, une base de donnees Oracle, vous pouvez utiliser $dbh->do pour
executer immediatement cette procedure. Par exemple,
```

```
    $dbh->do( "BEGIN someProcedure END" );

---------------------------------------------------------------------------

5.4 Comment puis-je recuperer les valeurs de retour de procedures enregistrees avec DBI?

    Contribution de Jeff Urlwin

    $sth = $dbh->prepare( "BEGIN foo(:1, :2, :3); END;" );
    $sth->bind_param(1, $a);
    $sth->bind_param_inout(2, \$path, 2000);
    $sth->bind_param_inout(3, \$success, 2000);
    $sth->execute;

N'oubliez pas d'effectuer un test d'erreur, strict!

---------------------------------------------------------------------------

5.5 Comment puis-je creer ou supprimer une base de donnees avec DBI?

La creation et la suppression de bases de donnees sont des concepts qui sont
beaucoup trop abstraits pour etre supportes par DBI. Par exemple, Oracle ne supporte
pas le concept de detruire une base de donnees du tout! Ainsi, dans Oracle, le
serveur de base de donnees est essentiellement la base de donnees elle-meme alors
que dans mSQL, le processus serveur s'execute tranquillement sans aucune base de
donnees creee. C'est un probleme trop heterogene pour s'y attaquer.

Quelques pilotes, cependant, supportent la creation et la suppression de bases de
donnees a travers des methodes func privees. Il vous faut regarder dans la
documentation des pilotes que vous utilisez pour verifier s'ils supportent de tels
mecanismes.

---------------------------------------------------------------------------

5.6 Comment puis-je valider ou annuler une instruction avec DBI?

A ecrire.

---------------------------------------------------------------------------

5.7 Comment les valeurs NULL sont-elles prises en compte par DBI?

Les valeurs NULL dans DBI sont traitees comme la valeur undef. Des NULLs peuvent
etre inseres dans les bases de donnees en tant que NULL, par exemple :

    $rv =
        $dbh->do( "INSERT INTO table VALUES( NULL )" );
```

```
mais lors d'une interrogation, les NULLs devront etre testes comme des undef. C'est
un standard pour tous les pilotes.

----------------------------------------------------------------------------

5.8 Qu'est-ce que c'est que ces histoires de methodes func?

Une methode func est definie a l'interieur de DBI comme etant un point d'entree pour
une fonctionnalite d'une base de donnees specifique, eg, la possibilite de creer ou
supprimer des bases de donnees. L'invocation de ces methodes specifiques aux pilotes
est simple. Par exemple, pour invoquer une methode createDatabase qui n'a qu'un seul
argument, on ecrira :

    $rv =
        $dbh->func( 'argument', 'createDatabase' );

Les developpeurs de logiciels doivent cependant noter que ces methodes func ne sont
pas portables entre SGDB.

----------------------------------------------------------------------------

Assistance et formation

L'interface aux SGBD Perl5 est un logiciel LIBRE. IL EST DISTRIBUE SANS GARANTIE
D'AUCUNE SORTE.

Cependant, quelques organisations fournissent soit une assistance technique soit des
programmes de formation pour DBI. L'auteur de ce document n'a aucune information sur
la qualite de ces services (le traducteur non plus!). Les liens ci-dessous ne sont
indiques qu'a titre de reference.

----------------------------------------------------------------------------

Assistance technique

The Perl Clinic
     The Perl Clinic peut mettre en place des contrats commerciaux d'assistance pour
Perl, DBI, DBD::Oracle et Oraperl. Cette assistance est effectuee par la compagnie
pour laquelle Tim Bunce, auteur de DBI, travaille. Pour des informations
complementaires sur leurs services, veuillez voir :

         http://www.perl.co.uk/tpc

----------------------------------------------------------------------------

Formation

Aucun programme de formation n'est connu a ce jour.
```

Autres References

Dans cette section, on trouvera des liens WWW divers pouvant presenter un interet pour les utilisateurs de DBI. Ils n'ont pas ete verifies et peuvent mener a des sites inconnus ou a des documents inexistants.

    http://www-ccs.cs.umass.edu/db.html
    http://www.odmg.org/odmg93/updates_dbarry.html
    http://www.jcc.com/sql_stnd.html

AUTEUR

Alligator Descartes <descarte@hermetica.com>

COPYRIGHT

----

14 PGACCESS - Un GUI pour la gestion dePostgreSQL

Il est déjà inclus dans la distribution de PostgreSQL. Pour une copie plus récente, vous pouvez accéder au site web site

- (http://www.flex.ro/pgaccess)
- Si vous avez des commentaires, des suggestions pour des améliorations, n'hésitez pas à envoyer un courrier électronique à : teo@flex.ro

Voici ce que dit la page d'accueil de pgaccess :

```
PgAccess - une interface Tcl/Tk pour PostgreSQL

La derniere version de PgAccess est 0.62 , du Samedi 25 Octobre 1997

Je pense qu'il y a eu quelques problemes au chargement de la bibliotheque libpgtcl.
Je vous invite a lire la section speciale concernant libpgtcl

Que fait PgAccess maintenant!
Voici quelques images des fenetres de PgAccess : Fenetre principale, createur
de tables, vue de table (interrogation), createur visuel d'interrogation .

Tables
- Ouverture de tables pour la visualisation, 200 enregistrements maximum
  ( modifiable dans le menu des preferences )
- redimensionnement d'une colonne en tirant la ligne verticale de la grille
  ( plus pratique a realiser a l'interieur meme de la table que dans l'en-tete)
- texte encadre dans les cellules - disposition enregistree pour chacune des tables)
- importation/exportation vers des fichiers externes (SDF,CSV)
- possibilite de filtres (filtre de saisie, tel que (prix>3.14)
- possibilites de tri (saisie manuelle de(s) champ(s) a trier)
- edition en ligne
- assistant generateur de table ameliore
- edition de champ amelioree
Requetes
- definition, edition et enregistrement de "requetes definies par l'utilisateur"
- enregistrement de requetes en tant que vues
- execution des requetes
- visualisation des resultats de requetes de type select
- suppression et changement de nom de requete
- NOUVEAU !!! Createur visuel de requetes avec possibilites de "glisser/deposer".
  Pour tous ceux d'entre vous qui ont l'extension pour Netscape Navigator de Tcl/Tk
  installee, vous pouvez le voir au travail en cliquant ici
Sequences
- definition de sequences, ainsi que leurs suppression et inspection
Fonctions
- definition, inspection et suppression de fonctions en langage SQL
```

```
Sur la liste A FAIRE!
- conception des tables (ajout de nouveaux champs, changement de nom, etc.)
- definition de fonction
- generateur de rapport
- langage de script de base

Informations complementaires concernant libgtcl

Vous aurez egalement besoin de la bibliotheque d'interface de PostgreSQL a Tcl,
disponible comme moduleTcl/Tk chargeable. Son nom est libpgtcl et le source est
situe dans le repertoire PostgreSQL /src/interfaces/libpgtcl. Precisement, vous
aurez besoin d'une bibliotheque libpgtcl qui soit "chargeable" a partir de Tcl/Tk.
Ce qui est techniquement different d'un fichier objet chargeable PostgreSQL
ordinaire, car libpgtcl est constituee d'un ensemble de fichiers objets. Sous Linux,
on l'appelle libpgtcl.so. Vous pouvez telecharger a partir d'ici une version deja
compilee pour les systemes Linux i386. La seule chose a faire est de copier
libpgtcl.so dans le repertoire bibliotheque systeme (/usr/lib) et c'est tout. Une
solution possible est de supprimer dans le source la ligne contenant load libpgtcl.so
et de charger pgaccess.tcl non pas avec wish, mais avec pgwish (ou wishpg) le wish
qui a ete lie avec la bibliotheque libpgtcl!

De toute maniere, l'application devrait fonctionner sans problemes
```

15 Outil Windows interactif de génération de requête pour PostgreSQL

("Windows Interactive Query Tool" WISQL ou MPSQL) Il est identique au serveur de requête SQL de Microsoft WISQL! Il dispose d'un GUI agréable et d'un historique des commandes. Vous pouvez utiliser le coupé/collé ainsi que d'autres fonctionnalités sympathiques pour accroître votre productivité.

- (http://www.troubador.com/ keidav/index.html)
- Email : keidav@whidbey.com
- (http://www.ucolick.org/ de/) in file tcl_syb/wisql.html
- (http://www.troubador.com/ keidav/index.html)
- Email : de@ucolick.org

16 Outil interactif de génération de requête pour PostgreSQL - ISQL

ISQL est destiné aux terminaux fonctionnants en mode ligne de commande. C'est inclus dans la distribution et nommé PSQL. Il est très similaire à Sybase ISQL, et à Oracle SQLplus. A l'invite unix tapez la commande 'psql' qui en retour vous affichera le caractère d'attente de commande: psql¿. Tapez /h pour obtenir de l'aide sur les commandes utilisables. C'est très convivial et d'utilisation facile. C'est aussi très utile pour écrire des scripts pour les interpréteurs de commandes Bourne, Korn et C-shells.

17 Mise en place de plusieurs machines PostgreSQL avec un seul moniteur

Vous pouvez empiler plusieurs cpu et les connecter à un unique moniteur en utilisant un commutateur pour choisir la connexion avec une unité centrale particulière. Ce qui économise de l'espace et évite l'utilisation de plusieurs écrans(200 US dollars), claviers(60 dollars) et mulots(50 dollars) et évite aussi pas mal de pagaille.

Veuillez consulter ces sites :

- (http://www.networktechinc.com/servswt.html) (120 dollars/PC 8 ports) qui propose 'Server Switches' and 'Video only switches'
- (http://www.scene.demon.co.uk/qswitch.htm)
- (http://www.cybex.com/products/control/4xp.htm)
- (http://www.betterbox.com/info.html)

Utilisez le moteur de recherche yahoo pour trouver d'autres compagnies en utilisant la clé de recherche 'Server Switches'.

On trouvera ci-dessous un extrait du catalogue de networktechnic Inc

```
Commandez maintenant, appelez le  800-742-8324 (appel gratuit a partir des USA)

Pour recevoir notre catalogue, veuillez envoyer votre adresse par courrier electronique a :
sales@networktechinc.com

PILOTEZ PLUSIEURS PC's avec un seul clavier, ecran et mulot
Ces commutateurs electroniques permettent de piloter jusqu'a 64 PC's avec un unique
clavier, ecran et mulot. Son microprocesseur interne simule la presence constante du
clavier, de l'ecran et du mulot pour chacun des PC's connectes.

Fonctionnalites et Applications

* Clavier, face avant ou commande a distance
* utilisation de connecteurs de mulots 9 broches D Serial ou 6 broches miniDIN
```

```
* utilisation de connecteurs de clavier 5 broches DIN ou 6 broches miniDIN
* resolution video 1600x1200 sans degradation
* Utilisation de cables standards
* Boitier pour bureau ou pour montage en rack

Specifications - Clavier

* Tous les connecteurs dont femelles
* 5 broches DIN ou 6 broches miniDIN
* Permettent un redemarrage (boot) materiel ou logiciel de tous les PC's a chaque instant

Controles - Boutons en face avant

* Le toucher d'un bouton permet la connexion au PC correspondant
* L'appui sur un bouton pendant plus de 0.5 secondes entraine le fonctionnement dans
les modes SCAN, BROADCAST ou COMMAND
* Les LEDs en face avant indiquent le mode d'operation du clavier
* L'appui sur CTRL+* entraine le fonctionnement en mode COMMAND
* Les modes SCAN, BROADCAST ou COMMAND sont disponibles

Mulot

* 9 broches D serie
        o Le commutateur NTI emule le mulot Microsoft serie pour tous les PC's
        o 9 broches D male pour mulot
        o 9 broches D femelle pour PC's
* 6 broches miniDIN
        o Le commutateur NTI emule le mulot IBM PS/2 pour tous les PC's
        o Tous les connecteurs sont femelles

Par cable, a distance

* Optionnel--doit etre acquis separement
* Fonctionnement identique a celui "Boutons en face avant"

Information technique

* Demander le manuel technique MAN025

Moniteur (ecran) - VGA

* La bande passante est de 150 MHz
* Resolution 1600X1200 sans degradation
* Tous les connecteurs sont des connecteurs femelles

Alimentation

* 110 ou 220 VAC a 50 ou 60 Hz
```

18 Outil Système de suivi de Problème/Projet pour PostgreSQL

Se trouve à l'url :

- (http://www.homeport.org/ shevett/pts/)

19 Conversion de fichiers dbase dbf en fichiers PostgreSQL

Le programme dbf2msql fonctionne parfaitement avec mSQL et PostgreSQL. Vous pouvez le trouver à

- (ftp.nerosworld.com/pub/SQL/dbf2sql/)
- (ftp://gopher.library.tudelft.nl/pub/misc/dbf2msql-0.4.tar.gz)

Ce programme a été écrit par Maarten Boekhold, Faculty of Electrical Engineering TU Delft, NL Computer Architecture and Digital Technique section M.Boekhold@et.tudelft.nl

Vous pouvez également utiliser une méthode python pour lire des fichiers dbf et les charger dans une base de données postgres.

- Voir (http://www.python.org)

20 PostgreSQL 4GL pour les applications de bases de données web - Le système de développement AppGEN

AppGEN peut être téléchargé de

(http://www.man.ac.uk/ whaley/ag/appgen.html)

(ftp://ftp.mcc.ac.uk/pub/linux/ALPHA/AppGEN).

On trouvera ci-dessous un extrait de la page d'accueil de AppGEN :-

AppGEN est un langage de haut niveau de la quatrième génération ainsi qu'un générateur d'application pour produire des applications destinées au World Wide Web (WWW). Ces applications sont typiquement utilisées à travers l'internet ou dans le cadre d'un réseau intranet d'entreprise. Les applications AppGEN sont implantées en scripts C respectants le standard Common Gateway Interface (CGI) utilisé par la plupart des serveurs Web.

Pour utiliser AppGEN il vous faudra disposer de :-

PostgresSQL, système de gestion de base de données relationnel

Un serveur web compatible CGI tel que HTTPD du NCSA

D'un compilateur ansi C tel que GCC

AppGEN est constitué des exécutables Unix (Linux) suivants :-

- defgen, qui produit une application cadre, de base, à partir de la structure logique des données. Ces applications sont capables d'ajouter, de mettre à jour, de supprimer et de rechercher les enregistrements d'une base de données tout en conservant automatiquement l'intégrité référentielle de cette base.
- appgen, le compilateur AppGEN qui compile le code source appgen en code source C CGI exécutable et en documents formatés HTML prêts pour leur déploiement sur un serveur web.
- dbf2sql, un utilitaire de conversion de fichiers .dbf compatibles dBase III en scripts SQL exécutables. Ce qui permet de migrer les données stockées dans la plupart des bases de données DOS/Windows vers un serveur SQL tel que PostgresSQL.
- De plus, AppGEN comprend un ensemble de documents HTML , de fichiers GIF et d'applets Java qui sont utilisés, à l'exécution, par le système. Et, naturellement, comme pour tout bon logiciel, la totalité du code source est fournie.

L'auteur, Andrew Whaley, peut être conctacté à andrew@arthur.smuht.nwest.nhs.uk tous les commentaires ou suggestions concernant ce logiciel seront les bienvenus.

21 Outil de Conception/Implémentation pour SGBD Web pour PostgreSQL - EARP

`(http://www.oswego.edu/Earp)`

`(ftp://ftp.oswego.edu)` dans le répertoire 'pub/unix/earp'.

On trouvera ci-dessous un extrait de la page d'accueil de EARP:-

Le "Programme de Réponse Facilement Ajustable" ("Easily Adjustable Response Program" - EARP) créé par David Dougherty

21.1 EARP, c'est quoi?

EARP est un outil de Conception/Implémentation pour Base De Données Web, réalisé par-dessus le système de gestion de base de données Postgres95. Ses fonctionnalités comprennent:

- Un système de conception visuel.
- Une interface d'envoi de courrier électronique. (pouvant prendre en charge le courrier entrant et sortant)
- Un Mécanisme de Sécurité amélioré.
- Un pilote cgi.

21.2 Implantation

L'implantation principale de EARP consiste en un binaire CGI qui s'exécute sous le démon http pour fournir l'accès au serveur de base de données. Tous les outils de conception sont intégrés dans le pilote, aucune conception ne se fait sur le web. Les outils eux-mêmes requièrent un navigateur graphique, la compatibilité des objets conçus avec ces outils est indépendante de l'implantation, elle ne dépend uniquement que des préférences de conception individuelles.

21.3 Ce dont vous avez besoin pour faire fonctionner EARP

EARP est sensé fonctionner sur une grande variété de plates-formes avec peu de modifications sinon pas du tout. Les plates-formes pour lesquelles le fonctionnement est connu sont les suivantes:

– Solaris 2.5
– Linux 1.2.13+
– GNU C++
– Postgres95 (Version 1.01 / 1.02)
– netsite server
– NCSA httpd

– GNU C++
– Postgres95 (Version 1.01 / 1.02)
– NCSA httpd
– Apache httpd

21.4 Nouvelles Flash

La version courante (1.3) de Earp a été conçue au-dessus de la version de libpq livrée avec Postgres95 v1.01/1.02. Si vous utilisez une version plus récente de Postgres, vous devez vous attendre à ce que le programme nécessite quelques modifications pour fonctionner correctement. Dans la version de développement (Earp 2.0), la prise en charge de libpq est en cours de prise en compte en tant que module, et de ce fait pourra supporter autant de versions de postgres que nous aurons de temps pour écrire ces modules. La version de développement est attendue, pour le public, aux alentours du milieu du printemps(97).

21.5 Comment fonctionne-t-il?

Une des fonctionnalités principales de EARP est d'utiliser une approche orientée objet pour produire des pages html qui s'interfacent à la base de données. La plupart des pages sont constituées de plusieurs objets. Chaque objet est produit par une sorte d'outil et reçoit un nom, les objets sont alors liés ensembles et appelés en séquence par l'outil de gestion de pages. Les objets sont également réutilisables à l'intérieur de plusieurs pages. Il existe des outils de base pour HTML, pour les Requêtes, pour la saisie dans des formulaires, le formatage variable des requêtes et des objets en entrée, ainsi que pour lier des objets pour en former de nouveaux. On trouve également des outils plus avancés tels que l'outil de courrier électronique et l'outil de création de requêtes en parallèle.

La gestion perfectionnée de la sécurité est une des autres fonctionnalités de EARP. L'accès aux différentes zones du système EARP peut être limité par une grande variété de moyens. Pour faciliter cette sécurité perfectionnée, EARP effectue des contrôles à chaque connexion au système, et détermine à quels "ids" et "groups" appartient l'agent qui se connecte. Les accès aux zones sont définis séparément, et la combinaison des deux permet de décider si l'accès à une certaine zone de Earp est autorisé. De plus , tout ce qui est requis pour réaliser les fonctionnalités de sécurité se trouve dans un serveur http qui effectue une authentification de l'utilisateur minimale (ou meilleure).

21.6 Quelques exemples en ligne

Partie intégrante de l'aide au SGBD ICC Help Database, la Page du Catalogue de Recherche est un document EARP qui peut exécuter plusieurs requêtes. Les boîtes de sélection sont générées par le programme EARP à partir de listes dans la base de données.

Comme autre exemple de ce que l'on peut faire avec EARP... vous pouvez maintenant voir la Liste des Objets de la Base de Données d'aide.

La création de ces trois interfaces m'a pris moins de quinze minutes.

21.7 Où puis-je le trouver?

EARP est disponible via un ftp anonyme à `(ftp://ftp.oswego.edu)` dans le répertoire 'pub/unix/earp'. La version courante à la date de rédaction de ce texte est 1.3.1

S'il vous plaît, après avoir récupéré et mis en service EARP, envoyez moi un mot pour en décrire le succès ou les échecs.

21.8 Documentation disponible

Toute la documentation a été déplacée à la page index "User Docs and Tutorials" (Documentation Utilisateur et Cours).

21.9 L'Histoire d'EARP

Earp 0.1 a démarré en tant que programme à l'automne 1995. Je travaillais à la réalisation d'un livre d'or accessible par le web, dynamiquement configurable. A ce moment là, il y avait tout un tas de programmes cgi faisant tous quelque chose de différent et d'utile, agrégés par un peu de colle SSI et un peu de programmation c. Je réalisais alors que je faisais beaucoup de travail répétitif, et que la plupart des choses que je faisais devaient exécuter dans de nombreuses fenêtres à la fois (netscape, emacs, shell, mail) alors que la mise au point était rapidement devenue un cauchemar. A cette époque mon ami et patron Don Michaels me contacta. Il était intéressé par l'automatisation d'une grosse partie de notre support utilisateur et par la tenue d'une base de données gérant l'historique des appels et des réponses.

Peu de temps après, j'avais terminé le schéma initial de ce qui est en train de devenir rapidement notre base de données d'aide. Seulement, je rechignais à l'idée de construire une base de données d'aide à partir de ce qui était alors un ensemble très primitif d'utilitaires. Lorsque les cours du printemps 96 débutèrent, je démarrais quand même le projet, principalement sous le coup de l'ennui, mais aussi parce que j'étais dans un cours d'architecture de base de données et que je voulais m'entraîner un peu les méninges. Peu de temps après j'avais un prototype acceptable en état de marche, ce qui rendit Don très heureux car il avait renoncé à l'idée de quelqu'un créant une base de données d'aide pour lui. (Le prototype tourne encore sur l'un de mes serveurs...(juin 96)) Le prototype faisait vraiment quelques choses très intéressantes, mais en avril, j'étais à nouveau découragé... A chaque fois que je désirais modifier quelque chose, j'entrais dans un lent processus de recompilation, ou je tombais sur un fichier texte particulièrement distrayant. Je n'avais aussi aucun moyen d'utiliser le concept de building block si utile dans EARP... J'ai réalisé pas mal d'énormes coupés-collés. Vers la fin de l'année scolaire je fus encore une fois découragé sur le déroulement des choses et décidais que ce dont j'avais besoin, c'était d'un ensemble d'outils plus adaptés à ce que je faisais. De plus, je voulais aussi que mon prototype fonctionne sur un e VRAIE base de données relationnelle et je ne chérissais pas l'idée de recommencer tout ce travail d'accès codés en dur, de liens, et de méthodes de sortie.

J'eus un bref moment de répit, si l'on peut dire. Nous soutenions la Conférence SUNY CIT et j'ai tellement été occupé pendant une semaine et demi que je pris du recul par rapport à tout ce que j'avais écrit sur le prototype de base de donnée d'aide excepté sur les idées que j'avais eu quand j'avais écrit la série d'utilitaires initiale et que ce qui m'irritait le plus était l'état actuel des choses.

Peu de temps après la conférence, je commençais le prototype de la version actuelle de EARP(may96), en utilisant le SGBD relationnel postgres95 comme support. A la mi-juin, le prototype avait évolué en une suite intégrée de prototypes d'outils tout à fait séduisante, avec comme premier avantage de tous tourner en html, et de stocker leurs données d'initialisation dans la base de donnée. La plus grande partie de la deuxième moitié de juin fut consacrée à la mise au point du code et à l'utilisation de l'interface.(Pendant ces deux semaines j'ai accédé plus de 5000 fois à notre serveur web.) A la fin juin, la plupart des bogues importants étaient chassés de EARP, et il y avait un nombre suffisant d'objets dans la "nouvelle" base de données d'aide que l'on pouvant en faire l'annonce officielle à notre équipe de support.

A propos, j'ai également tenu un journal pendant le développement de EARP et Don Michaels et moi présentons un papier décrivant la Base de Données d'Aide à la conférence ACM/SIGUCCS à Chicago en Septembre de cette année.

22 WWW Web interface pour PostgresSQL - dbengine

(`http://www.cis-computer.com/dbengine/`)

Un extrait de la page d'accueil de dbengine est donné ci-dessous:-

dbengine est une interface Web plug 'n play pour Postgres95 créée par Ingo Ciechowski

Version 0.82 alpha Documentation au 11/23/96

A propos de dbengine

dbengine est une interface entre le WWW et Postgres95 qui permet un accès simple à n'importe quelle base de données existante en seulement quelques minutes.

Ce petit programme Perl program est né après avoir essayé un tas de paquetages disponibles tels que AppGen, PHP-FI et d'autres. Alors, pourquoi ai-je, en quelque sorte, réinventé la roue?

Eh bien, PHP-FI offre une sorte de langage Perl dans vos documents, mais pas le vrai Perl alors que AppGen et wdb-p95 nécessite la création d'un fichier de configuration pour chacune de vos bases de données – c'est un peu comme si vous deviez apprendre un nouveau métalangage avant de pouvoir commencer à travailler.

C'est à ce moment que je commençais à me sentir plus à l'aise avec une petite applet Perl de mon cru... et maintenant mon dbengine est devenu un outil qui, je pense, est prêt à être partagé avec les autres.

A la différence des autres outils, vous n'avez pas à apprendre un langage de programmation particulier ou un langage de script pour commencer à utiliser dbengine. De plus, il n'y a pas à créer de fichier de configuration pour chaque base de données, et vous n'avez donc pas besoin de vous familiariser avec sa structure. Cependant - au cas où vous voudriez profiter de toutes les possibilités de dbengine, ce serait une bonne idée de connaître le langage Perl.

Le système tout entier peut être configuré à l'aide de simples manipulations d'une base de données complémentaire qui contient les informations de détail sur la manière de visualiser vos accès à la base de données. Vous pouvez même spécifier des Champs Virtuels qui sont calculés en temps réel avant d'être affichés à l'écran.

Licence

dbengine est un logiciel libre selon les mêmes conditions que Perl. Lisez sa licence si vous n'êtes pas sûr de ce que vous pouvez ou ne pouvez pas faire. La dernière ligne indique que c'est une version plus gentille et plus modérée que celle de la licence GNU – une de celle qui n'affecte pas votre travail si vous extrayez des parties de dbengine ou du paquetage pour l'inclure dans un produit commercial!

23 Module Apache Webserver pour PostgreSQL - NeoSoft NeoWebScript

Apache est un serveur Web bien connu. On trouvera un module d'interface de PostgreSQL pour le serveur Web Apache à - `(http://www.neosoft.com/neowebscript/)`

Voici, ci-dessous, un extrait de la page d'accueil de NeoWebScript:-

NeoWebScript est un langage de programmation qui permet d'inclure des programmes simples ou compliqués dans des fichiers HTML.

Quand une page HTML contenant un script NeoWebScript est appelée, le serveur Web , ayant le NeoWebScript activé, exécute le(s) script(s) inclus, produisant une page Web dont le contenu, personnalisé, a été créé par le programme.

NeoWebScript est un moyen rapide, sûre, facile à apprendre de créer des programmes interactifs puissants directement en code HTML dans des pages Web. Avec NeoWebScript, les compteurs, les formulaires de courrier électronique, "graffiti walls", livres d'or et suivi des visiteurs sont aisés, même pour un programmeur débutant. Regardez comment NeoWebScript se défend bien par rapport à PERL et JavaScript.

NeoWebScript 2.2 vient de sortir! Le 24 juin 1997, NeoSoft rend disponible la version 2.2 de NeoWebScript 2.2, en l'intégrant avec le nouveau serveur Apache 1.2.0 .

Si vous utilisez un serveur web ayant NeoWebScript installé et que vous vouliez commencer à l'utiliser, il y a un tout un tas d'Informations Utilisateur disponibles. Les réponses de bases pour démarrer se trouvent dans la "FAQ Nouvel Utilisateur". Des didacticiels vous guident dans l'apprentissage du langage, alors que des Démonstrations vous apportent des applications toutes faites que vous pouvez télécharger et adapter. Le document "Commandes et les Variables" constitue la référence complète du langage, et celui de Maintenance contient des conseils et des astuces pour vous aider à résoudre tous les problèmes.

Au cas où vous souhaiteriez installer NeoWebScript sur votre serveur web, votre webmestre doit lire notre "FAQ Syso" pour se lancer. Le document "Théorie de fonctionnement" explique comment NeoWebScript fonctionne, celui d'Installation est un guide pas à pas des opérations à effectuer. Le guide de Gestion traite de la manière de configurer et de faire fonctionner le serveur, celui de Tests permet de vérifier le fonctionnement correct de NeoWebScript, enfin, celui de Maintenance traite des problèmes de serveur.

Hé, attendez une minute direz-vous, combien voulez-vous en contrepartie d'un tel logiciel, hein? Il n'y a aucuns frais pour l'utilisation de NeoWebScript-2.2 que ce soit pour votre ISP, votre intranet, ou votre extranet. Vous pourrez voir un licence complète quand vous vous enregistrerez pour télécharger le logiciel, mais l'essentiel est que nous aimerions recevoir la somme monstrueuse de $ 99 si vous désirez l'inclure dans votre propre produit ou l'utiliser dans un serveur commercial (eg. SSL).

NeoWebScript est un module pour le serveur web Apache qui vous permet d'inclure , en tant que langage de script, pour vos pages web, le langage de programmation Tcl/Tk . Il a été inventé par Karl Lehenbauer, Directeur Technique chez NeoSoft, et documenté, renforcé et étendu par les programmeurs de NeoSoft et par des rédacteurs techniques.

Le serveur Apache est le serveur web le plus populaire au monde, totalisant 42 %des 1 044 163 sites interrogés lors de l'étude "Netcraft Web Server" de Mai 1997. La deuxième plus grande valeur enregistrée concerne les différents serveurs Microsoft, totalisant légèrement plus de 16 %, soit une différence d'environ 270,000 serveurs.

Tcl/Tk est un langage de script puissant, libre et multi-plateformes développé par le Dr. John Ousterhout, maintenant "Sun Distinguished Engineer" (n.d.t. : distinction honorifique de la Cie Sun). Selon ses propres termes :

"Tcl/Tk permet aux développeurs de logiciels de réaliser un travail dix fois plus rapidement qu'avec des outils basés sur C ou C++. C'est aussi un grand langage de synthèse permettant de faire travailler ensemble des applications existantes en les rendant plus graphiques et orientées Internet."

Avec une communauté de développement de plus de 500 000 personnes de par le monde et des milliers d'applications commerciales, Sun vient juste d'annoncer la création d'un nouveau groupe de travail nommé SunScript, pour soutenir cette communauté avec un environnement de développement intégré et pour développer une suite de produits pour établir un lien entre Tcl le Web et Java.

Karl Lehenbauer, Fondateur et Directeur Technique de NeoSoft, a participé au développement de Tcl/Tk dès le tout début. Avec Mark Diehkans, ils sont les auteurs de Tcl étendu (Extended Tcl), connu également sous le nom de TclX ou NeoSoft Tcl, qui constitue un ensemble puissant d'extensions au langage. Beaucoup des commandes courantes essentielles de Tcl proviennent de Tcl étendu, et furent introduites dans le langage par le Dr. Ousterhout.

NeoSoft Inc., 1770 St. James Place, Suite 500, Houston, TX 77056 USA

24 HEITML, extension, côté serveur de HTML et langage 4GL pour PostgreSQL

Heitml est un autre outil d'interfaçage de postgres avec le monde du world wide web. Pour plus de détails contacter

```
Helmut Emmelmann H.E.I. Informationssyteme GmbH
Wimpfenerstrasse 23 Tel. 49-621-795141
68259 Mannheim Germany Fax. 49-621-795161
```

– E-mail Mr.Helmut Emmelmann at emmel@h-e-i.de

– Heitml site web principal `(http://www.heitml.com)`

– Heitml site web secondaire `(http://www.h-e-i.deom)`

Heitml est à la fois une extension, côté serveur, de HTML et un langage de la quatrième génération (4GL). Avec lui, on peut écrire des applications web dans un style HTML et en utilisant de nouvelles balises de style HTML.

heitml (prononcé "H-ail"-TML) est une extension de HTML et un langage de quatrième génération complet permettant aux Applications utilisant le web d'interagir avec des données stockées dans des bases de données SQL, sans mériter l'écriture de scripts CGI complexes.

heitml étend HTML côté serveur, convertissant de manière dynamique les fichiers ".hei" au format HTML et ainsi, les rendant compatibles avec n'importe quel butineur web. Il englobe la syntaxe simple et familière de HTML et apporte un large assortiment de Balises et de Bibliothèques prédéveloppées pour prendre en charge les tâches qui auparavant nécessitaient CGI. De même que XML, heitml permet l'utilisation de balises définies par l'utilisateur. Avec heitml les marqueurs définis par l'utilisateur peuvent être traduits en HTML et envoyés à un butineur.

heitml est destiné à la fois aux concepteurs HTML et aux programmeurs professionnels. Les concepteurs HTML peuvent utiliser les Balises heitml pour fabriquer des pages web dynamiques, des accès aux bases de données SQL ou créer des applications web complètes. On peut créer des Compteurs, des bases de données d'inscriptions, des formulaires de recherche, des formulaires de courrier électronique ou des menus hiérarchisés en utilisant simplement des Balises de style HTML prédéveloppées que l'on trouve dans les nombreuses Bibliothèques de Composants.

Pour les programmeurs, heitml inclut un langage de quatrième génération complet en HTML

```
(e.g. <if>, <while>, et <let> Balises),
```

plus un évaluateur d'expression puissant pour les types de données entiers, réels, booléens chaîne de caractères et tuple. Les tuples ont une référence sémantique comme dans les langages orientés objets modernes et sont stockés sur un tas. Les variables heitml y compris toutes les structures de données complexes stockées sur le tas conservent leur valeur de page en page par l'utilisation du Mode Session. Il vous est possible de définir vos propres balises ou vos balises d'environnement et même de redéfinir les balises HTML.

Avec heitml il devient possible de

- - - développer des Sites Web de manière structurée et modulaire, tout en réduisant de façon drastique la surcharge due à la maintenance.

- - - développer des Sites Web intelligents et interactifs, dont le contenu s'adapte dynamiquement aux besoins de l'utilisateur.

- - - de visualiser le contenu de bases de données SQL sans autre programmation que l'utilisation de notre bibliothèque de Balises prédéfinies "dba".

- - - de développer des applications de bases de données complexes et de Catalogue d'Achat en utilisant les Variables Session

heitml tourne sous Linux avec n'importe quel Serveur Web utilisant l'interface CGI, et il est particulièrement rapide (mis à part la surcharge due à CGI) avec le Serveur Web APACHE (version 1.1.1, 1.1.3, ou 1.2.4) en utilisant l'API apache. Actuellement MSQL (Version 1 et 2), PostgreSQL (Version 6), mysql, et les bases de données yard sont supportés. heitml tourne sous Linux, BSDi, Solaris et SunOS, de même que sous Windows NT avec CGI, ISAPI et ODBC et Windows 95.

heitml (sous linux) est libre pour la recherche, et pour une utilisation privée et non commerciale. Les Sites Web commerciaux doivent payer une licence. La version totalement opérationnelle de heitml est disponible pour une période d'essai et peut être téléchargée librement. (Notez, cependant, que chaque page Web ".hei" que vous développerez affichera un message indiquant qu'il s'agit d'une version à usage non commerciale. Après enregistrement, vous recevrez une clé pour effacer ce message sans avoir à réinstaller le programme.)

Les nouvelles fonctionnalités de la version 1.2 sont

- - - Bibliothèque de Composants pour des Formulaires de Recherche dans un Base de Données, menus hiérarchisés d'ouverture et de fermeture, formulaires de courrier électronique

- - - Le Mode Session a été reconçu et amélioré pour conserver toutes les variables (y compris le tas tout entier) au travers des différentes pages. Cela signifie que les données, de n'importe quelle taille sont conservées à l'intérieur d'une session. Ceci offre l'opportunité d'un champ plus large pour créer de nouvelles applications, e.g. stockage complet des résultats d'une interrogation dans la mémoire de la session.

- - - Les instructions d'installation, la documentation et les bibliothèques d'exemples ont été augmentées de manière significative, un cours d'autoformation a été ajouté

- - - Des balises pour exécuter des commandes shell et pour envoyer des courriers électronique ont été ajoutées

- - - Le support du formatage d'impression et de la mise au point. heitml affiche votre code source en couleurs dans votre butineur et signale les erreurs de façon intuitive. En cas d'erreur à l'exécution, toutes les structures de données internes sont affichées dans le butineur en indiquant et préservant leurs positions dans le code source original.

- - - Diverses nouvelles variables serveur et fonctions intégrées ont été ajoutées

- - - En mode production, heitml collecte maintenant des informations complètes sur les erreurs survenues lors de l'accès, par des utilisateurs, à votre site.

Voici ce que dit la page d'accueil de heitml -

heitml (prononcer "H-ail"-TML) étend et augmente de façon significative les fonctionnalités de HTML grâce aux balises définissables et aux possibilités complètes de programmation. Ce qui permet de créer simplement des applications au contenu dynamique et orientées bases de données dans le monde HTML, sans CGI et sans scripts externes ou langages de programmation. Cela signifie que vous, en tant qu'auteur HTML, vous pouvez inclure des applications dans vos pages web, simplement, en utilisant quelques nouvelles balises sans CGI et sans programmation. D'un autre côté, comme utilisateur avancé ou comme programmeur vous pouvez créer et programmer de puissantes bibliothèques de balises. Cette approche rend heitml souhaitable à la fois pour les utilisateurs nouveaux de HTML et pour les programmeurs professionnels. heitml tourne sur le serveur web et génère dynamiquement du HTML, aussi heitml est compatible avec les standards internet et avec n'importe quel butineur web. Il permet un accès total aux bases de données tout en évitant à l'utilisateur toute la complexité inutile de CGI. heitml a été développé selon les plus récents critères en matière de construction de compilateurs et de systèmes transactionnels.

Les pages heitml sont développées exactement de la même façon que les pages HTML, à l'aide d'un éditeur de texte ou d'un éditeur HTML, et placées comme d'habitude sur le serveur web. Cependant, maintenant, les pages peuvent contenir des balises heitml dynamiques et des accès à des bibliothèques de balises. Vous pouvez utiliser ces balises pour accéder à une base de données, pour créer un contenu dynamique, pour envoyer des courriers électronique, et même pour créer de puissantes applications telles que les bases de données d'inscriptions ou de systèmes d'achats.

Les nouveaux venus à HTML et les programmeurs professionnels seront stupéfaits de la vitesse et de la facilités avec lesquelles il peuvent concevoir des applications passionnantes telle que notre Livre d'Or Interactif sans nécessiter la complexité et l'apprentissage difficiles des scripts CGI, simplement en utilisant les outils fournis dans notre bibliothèque dba.

heitml est accompagné d'un large éventail de bibliothèques de balises, pour créer des livres d'or, des applications de maintenance de bases de données, des formulaires puissants de courrier électronique ou de structuration de votre site web à l'aide de menus hiérarchiques. Ces outils sont prêts à être utilisés, il suffit simplement d'ajouter les balises correspondantes à votre site web.

En tant que programmeur expérimenté, vous pouvez pleinement utiliser l'architecture dynamique persistante de tuple d'heitml : heitml n'est pas simplement un langage de script à typage dynamique, avec évaluateur d'expression, procédures récursives et capacités de passage de paramètres étendues, mais il apporte aussi des possibilités de tuples persistants dynamiques pour conserver automatiquement des données de session de n'importe quelle taille.

heitml a tellement de nouvelles possibilités et d'usages que c'est impossible de les décrire tous dans une simple page web. Pour cette raison, nous avons conçu ce site Web de manière à fournir soit une simple vue d'ensemble du produit, soit à fouiller aussi profondément que vous le désirez dans les "tenants et aboutissants" du langage. Quelle que soit l'approche que vous choisirez, nous pensons que vous trouverez que heitml a beaucoup à offrir, et nous espérons que vous serez d'accord pour trouver que réellement "il porte le monde du World Wide Web à un niveau plus élevé!"

Le menu, sur le côté gauche de l'écran vous aidera à naviguer dans notre Site Web Site de manière organisée et méthodique, mais vous pouvez aussi utiliser notre Menu Rapide (Quick Menu) pour sauter directement à n'importe quelle page avec un seul clic sur le mulot.

Si vous visitez ce Site pour la première fois, nous vous suggérons d'essayer le Livre d'Or Interactif d'heitml pour avoir une démonstration vraiment impressionnante de la façon dont heitml peut rendre vos Pages Web plus interactives. Si vous signez notre Livre d'Or, vous pouvez nous laissez vos commentaires ou nous parler de votre Site Web favori en utilisant les commandes de formatage de HTML. C'est comme si vous créiez votre propre page Web privée et que vous la voyiez publiée immédiatement!

La section traitant des Fonctionnalités de heitml Features fourni un Sommaire Rapide de ses Fonctionnalités et de ses Avantages, de même que des Pages destinées à répondre aux besoins spécifiques et aux questions des Concepteurs et des Programmeurs. Notre Guide du Langage offre un didacticiel, en ligne, qui montre, de manière pratique, quelques unes des façons dont vous pouvez utiliser heitml pour développer, améliorer et simplifier vos Pages Web et vos Applications.

Le document Référence du Langage est destiné à ceux qui cherchent une information spécifique sur la syntaxe, la structure, et l'utilisation des différents éléments du langage. Cependant, ce pourrait être une bonne idée pour tout le monde de lire la Page concernant la Conception Générale qui offre une vue d'ensemble sur les types de problèmes spécifiques pour lesquels heitml a été conçu afin d'y apporter une solution.

Naturellement, vous voudrez savoir si heitml tourne sur le Système d'Exploitation spécifique, le Serveur Web, ou le SGBD SQL que vous utilisez, vous trouverez toutes ces informations dans notre section des Plates-formes Supportées.

Enfin, nous vous invitons à Télécharger une copie d'essai du programme pour l'utiliser sur votre propre système. Nous sommes sûrs que vous apprécierez cette "méthode d'essai avant achat", et nous sommes à l'écoute de tout retour d'informations que vous voudrez bien nous donner.

25 PHP/FI Langage de script côté serveur, avec HTML intégré, pour PostgreSQL

Outil d'Interfaçage WWW `(http://www.vex.net/php)`

Pour les questions envoyer un courrier électronique à : rasmus@lerdorf.on.ca

On trouvera ci-dessous un extrait de la page d'accueil de PHP/FI :-

PHP/FI est un langage de script côté serveur, avec langage HTML intégré. Il vous permet d'écrire des scripts simples directement dans vos fichiers .HTML à la manière JavaScript, sauf que, à la différence de JavaScript, PHP/FI n'est pas dépendant du butineur utilisé. JavaScript est un langage, côté client, avec HTML intégré alors que PHP/FI est un langage côté serveur. PHP/FI est similaire, dans son concept, au produit LiveWire Pro pour Netscape. Si vous en avez les moyens, vous utilisez le serveur du commerce Netscape et l'un des systèmes d'exploitation supportés, et vous jetterez probablement un oeil sur LiveWire Pro. Si vous préférez un logiciel libre, évoluant rapidement, qui est disponible avec la totalité de son code source, vous apprécierez probablement PHP/FI.

25.1 Fonctionnalités principales

Support de CGI Standard, FastCGI et du module Apache. Comme programme CGI standard, PHP/FI peut être installé sur n'importe quelle machine Unix sur laquelle tourne n'importe quel serveur web Unix. Avec le support du nouveau standard FastCGI, PHP/FI peut trouver avantage des gains de vitesse apportés par ce mécanisme. Comme module Apache, PHP/FI devient une alternative extrêmement puissante et brillante à la programmation CGI.

- Enregistrement des accès Avec les possibilités d'enregistrement des accès de PHP/FI, les utilisateurs peuvent entretenir leur propre compteur d'accès et l'enregistrer. Il n'utilise en aucune façon les fichiers de d'enregistrement des connexions du système central, et il permet un suivi des accès en temps réel. Le Script du Visualisateur de Connexions fournit un résumé rapide des accès à un ensemble de pages possédées par un utilisateur individuel. De plus, le paquetage peut être configuré pour générer un pied de page, sur chaque page, qui montre l'information sur les accès. Regardez au bas de cette page pour en avoir un exemple.

- Contrôle d'accès Un écran de configuration intégré, basé sur le web, permet la configuration du contrôle des accès. Il est possible de créer des règles d'accès pour toutes ou quelques unes des pages web possédées par une certaine personne qui met diverses restrictions sur qui peut voir ces pages et comment elles seront vues. Les pages peuvent être protégées par un mot de passe, complètement interdites, à connexion désactivée et l'accès basé sur le domaine du client, le butineur, l'adresse de courrier électronique ou même le document auquel on se réfère.

- Support de Postgres Postgres est un SGBDR (RDBMS) avancé libre. PHP/FI supporte les requêtes Postgres95 et PostgreSQL SQL incluses directement dans les fichiers .html.

- RFC-1867 Support du téléchargement de fichier Le téléchargement de fichier est une nouvelle fonctionnalité de Netscape 2.0. Il permet aux utilisateurs de télécharger des fichiers vers le serveur web. PHP/FI réalise le décodage Mime réel pour faire ce travail et fournit aussi le cadre additionnel pour faire quelque chose d'utile avec le fichier téléchargé une fois que celui-ci a été reçu.

- Contrôle d'authentification basé sur HTTP PHP/FI peut être utilisé pour créer des mécanismes d'authentification personnalisés basés sur HTTP pour le serveur web Apache.

- Variables, Tableaux, Tableaux associatifs PHP/FI supporte des variables typées, des tableaux et même des tableaux associatifs à la Perl. Ils peuvent être passés d'une page web à l'autre en utilisant les méthodes GET ou POST.

- Conditions, Boucles Tant que (While) PHP/FI possède un langage de script aux fonctionnalités complètes de style C. Vous disposez des instructions de tests conditionnels if/then/elseif/else/endif ainsi que des boucles while et des instructions switch/case pour contrôler l'ordre logique et la manière dont les pages html doivent être affichées.

- Expressions Régulières étendues Les expressions régulières sont très utilisées pour le filtrage, le remplacement de séquences et les manipulations générales de chaînes de caractères. PHP/FI supporte toutes les opérations communes sur les expressions régulières.

- Contrôle de l'En-tête HTTP brute La possibilité d'envoyer à partir de pages web des en-têtes HTTP brutes personnalisées en fonction d'une condition est essentielle pour créer un site web de haut niveau. Un usage fréquent est l'envoi d'un emplacement: en-tête URL pour rediriger le client appelant vers une autre URL. Il peut aussi être utilisé pour stopper le stockage ou manipuler la dernière mise à jour de l'en tête de pages.
- Création d'images GIF à la volée PHP/FI prend en compte la bibliothèque d'image GD de Thomas Boutell ce qui offre la possibilité de générer des images GIF à la volée.
- Prise en charge du mode sécurité ("Safe Mode") ISP PHP/FI prend en charge un "Mode de Sécurité" exceptionnel qui permet d'avoir de multiples utilisateurs exécutant des scripts PHP en toute sécurité sur le même serveur.
- C'est Libre! Pour finir, et c'est un point essentiel. Le paquetage est entièrement libre. Il est sous les conditions de la licence GPL qui vous permet d'utiliser ce logiciel pour n'importe quel objectif, commercial ou autre. Reportez-vous au document de la Licence Publique GNU pour des renseignements détaillés.

25.2 Crédits

* De grandes parties de ce code ont été développées à et pour l'université de Toronto. De grands Mercis à Lee Oattes du Département de Développement des Réseaux à l'université pour ses critiques constructives permanentes.

* Le code de prise en charge de Postgres95 a été écrit par Adam Sussman asussman@vidya.com

* d'autres, innombrables, ont participé aux tests et à la mise au point du paquetage.

PHP/FI Version 2.0

25.3 Bref Historique

PHP a commencé sa vie comme simple petite enveloppe cgi écrite en Perl. Je l'ai écrit en un après-midi, dans une période entre deux contrats, alors que j'avais besoin d'un outil rapide pour avoir une idée de la personne qui était en train de lire mon curriculum vitae en ligne. Je n'avais eu l'intention de le voir utiliser en dehors de mon propre usage. Le serveur web sur lequel j'avais mon curriculum vitae était extrêmement surchargé et avait des problèmes constants de création de processus. Je réécris le petit programme Perl en C pour me débarrasser de la surcharge considérable générée par la création d'un processus d'exécution du programme Perl à chaque fois que mon curriculum vitae était consulté.

Par la suite d'autres personnes, sur le même serveur web, prirent connaissance de mon petit programme et me demandèrent s'ils pouvaient l'utiliser. Alors, comme cela devait arriver, ils commencèrent à me demander plus de fonctionnalités. J'ajoutais plus de fonctionnalités et finalement constituais une semi-distribution en y incluant une documentation, une liste de diffusion et une FAQ. Le nom de ce paquetage était "Outils pour une Page d'Accueil Personnelle" (Personal Home Page Tools), nom qui devint plus tard "Kit de Construction de Page d'Accueil Personnelle" (Personal Home Page Construction Kit).

A la même époque, je commençais à m'amuser avec les bases de données et j'écrivis un outil pour inclure facilement des requêtes SQL dans des pages web. C'était fondamentalement un autre petit programme CGI qui analysait les requêtes SQL et facilitait la création de formulaires et de tables basés sur ces requêtes. Cet outil fut appelé FI "Interpréteur de Formulaire" (Form Interpreter).

La version 2.0 PHP/FI est une réécriture complète de ces deux paquetages combinés pour former un simple programme. il a maintenant évolué au point de devenir un simple langage de programmation intégré dans les fichiers HTML. L'acronyme original, PHP, lui est resté. Il n'est plus réellement approprié. PHP/FI est aujourd'hui plus utilisé pour mettre en place des sites web entiers que pour de petites pages d'accueil personnelles. Quel que soit son nom, il élimine le besoin de recourir à de nombreux petits programmes cgi Perl en permettant d'insérer de simples scripts directement dans vos fichiers HTML. Ceci améliore les performances globales de vos pages web puisque la surcharge due au lancement de processus Perl successifs a été éliminée. La gestion de larges sites web a également été facilitée en plaçant tous les composants d'une page web dans un fichier html unique. En incluant le support de différentes bases de données, il devient trivial de développer des pages web mettant en oeuvre des bases de données. Nombreux sont ceux qui trouvent que cette intégration est plus facile à gérer que d'essayer de créer des fichiers HTML et CGI séparés.

Tout au long de cette documentation, toute référence à PHP, FI ou PHP/FI traite de la même chose. La différence entre PHP et FI est seulement conceptuelle. Les deux sont construits à partir de la même distribution du code source. Lorsque je construit le paquetage sans aucun support de connexion d'accès ou de restriction d'accès, je nomme mon binaire FI. Quand je le construit avec ces options, je l'appelle PHP.

25.4 Bon, que puis-je faire avec PHP/FI?

La première chose que vous allez noter, si vous faites tourner une page par l'intermédiaire de PHP/FI, c'est qu'il ajoute un pied de page vous informant du nombre d'accès à votre page (si vous incluez le support des connexions d'accès dans votre binaire). C'est seulement un tout petit exemple de ce que PHP/FI peut faire pour vous. Il joue également le rôle très important d'interpréteur de formulaire cgi, d'où la partie FI de son nom. Par exemple si vous créez un formulaire dans l'une de vos pages web, vous avez besoin de quelque chose pour traiter l'information contenue dans ce formulaire. Même si vous désirez simplement passer l'information à une autre page web, vous aurez besoin d'un programme cgi pour le faire. PHP/FI facilite grandement le processus de prise en charge des données du formulaire pour en faire quelque chose.

25.5 Un simple exemple

Supposons que vous ayez le formulaire :

```
<FORM ACTION="/cgi-bin/php.cgi/~userid/display.html" METHOD=POST>
<INPUT TYPE="text" name="name">
<INPUT TYPE="text" name="age">
<INPUT TYPE="submit">
<FORM>
```

Votre fichier display.html doit contenir quelque chose du genre :

<?echo "Hi $ name, vous avez $ âge ans!

” >

C'est aussi simple que cela! PHP/FI crée automatiquement une variable pour chaque champ de saisie de votre formulaire. Vous pouvez alors utiliser ces variables dans le fichier ACTION URL.

L'étape suivante, après avoir vu comment utiliser ces variables, est de commencer à jouer avec quelques balises de gestion du déroulement logique de vos pages. Par exemple, si vous voulez afficher différents messages dépendants de des informations entrées par l'utilisateur, vous pouvez utiliser la logique si/alors (if/else). Dans notre exemple ci-dessus, on peut afficher différentes choses dépendant de l'âge entré par l'utilisateur en modifiant notre fichier display.html en :

```
<?
    if($age>50);
        echo "Hi $name, vous etes un fossile!<p>";
    elseif($age>30);
        echo "Hi $name, vous etes tres vieux!<p>";
    else;
        echo "Hi $name.";
    endif;
>
```

PHP/FI fournit un langage de script très puissant qui peut faire beaucoup plus que ce qui est exposé dans l'exemple ci-dessus. Regardez la section sur le langage de Script PHP/FI pour des informations complémentaires.

Vous pouvez également utiliser PHP/FI pour configurer qui a le droit d'accéder à vos pages. Ceci est effectué en utilisant l'écran de configuration inclus. Par son intermédiaire , vous pourrez, par exemple, spécifier que seulement les gens appartenant à un certain domaine seront autorisés à voir vos pages, vous pouvez également créer une règle protégeant certaines pages par un mot de passe. Voir la section Contrôle d'Accès pour plus de détails.

PHP/FI est également capable de recevoir des fichiers téléchargés par n'importe quel butineur conforme à la norme RFC-1867. Cette fonctionnalité permet aux gens de télécharger également des fichiers de texte ou binaire. Avec le contrôle de PHP/FI et les fonctions logiques, vous disposez d'un contrôle total sur qui est autorisé à télécharger un fichier et sur ce que l'on peut faire de ce fichier une fois téléchargé. Voir la section Téléchargement de fichier pour de plus amples détails.

PHP/FI supporte le paquetage Postgres95. Il supporte les requêtes SQL incluses dans vos fichiers .HTML. Consultez la section Support Postgres95 pour plus d'informations.

PHP/FI supporte également le paquetage du SGBD mysql. Il supporte les requêtes SQL incluses dans vos fichiers .HTML. Consultez la section Support mysql pour plus d'informations.

25.6 Redirection CGI

Notes Apache 1.0.x

Une bonne manière de faire tourner PHP/FI est d'utiliser un module de redirection cgi avec le serveur Apache. Veuillez noter que vous n'avez pas à vous soucier des modules de redirection si vous utilisez la version module Apache de PHP/FI. Deux de ces modules de redirection sont disponibles. L'un est développé par Dave Andersen angio@aros.net et est disponible à ftp://ftp.aros.net/pub/util/apache/mod_cgi_redirect.c et l'autre arrive inclus avec Apache et est appelé mod_actions.c. Ces deux modules sont extrêmement similaires. Ils diffèrent très légèrement dans leur utilisation. Les deux ont été testés et fonctionnent avec PHP/FI.

Mise en garde importante: à la date de cette rédaction (Apr.20/96), la version courante officielle (1.0.5) a de sévères limitations qui empêchent les requêtes de redirection cgi d'avoir des données ??? post-method data associées. J'ai étudié ce problèmes et l'ai résolu dans ma version d'Apache. Il y a un colmatage officiel dans les Fichiers Archives de la Page d'Accueil.

Une autre mise en garde particulièrement importante au sujet d'Apache 1.0.x est qu'il n'aligne pas correctement les types doubles sur la plupart des architectures. Vous découvrirez d'étranges erreurs de bus provenant de votre httpd en utilisant mod_php. Comme solution, soit vous passez à la version Apache 1.1, soit vous éditez le fichier source Apache alloc.c. Dans ce fichier vous trouverez le morceau de code suivant :

union align { /* Types which are likely to have the longest RELEVANT alignment * restrictions... we don't do much with doubles. */

char *cp; void (*f)(); long l; FILE *fp; };

Il vous faudra ajouter un type double à ce source et recompiler votre serveur Apache. Le morceau de code correct est :

union align { /* Types which are likely to have the longest RELEVANT alignment * restrictions... we don't do much with doubles. */

char *cp; void (*f)(); long l; FILE *fp; double d; };

Regardez dans la documentation Apache sur la manière d'ajouter un module. En général vous ajoutez le nom de module dans un fichier appelé Configuration. La ligne à ajouter si vous voulez utiliser le module mod_actions est :

Module action_module mod_actions.o

Si vous envisagez d'utiliser le module mod_cgi_redirect.c ajoutez cette ligne :

Module cgi_redirect_module mod_cgi_redirect.o

Puis compilez votre httpd et installez-le. Pour configurer la redirection cgi il vous faudra soit créer un nouveau type mime dans votre fichier mime.types soit utiliser la commande AddType dans votre fichier srm.conf pour ajouter le type mime. Le type mime à ajouter doit être quelque chose comme :

application/x-httpd-php phtml

Si vous vous apprêtez à utiliser le module mod_actions.c il vous faudra ajouter la ligne suivante dans votre fichier srm.conf:

Action application/x-httpd-php /cgi-bin/php.cgi

Si vous vous apprêtez à utiliser mod_cgi_redirect.c vous devrez ajouter cette ligne à srm.conf :

CgiRedirect application/x-httpd-php /cgi-bin/php.cgi

N'essayez pas d'utiliser en même temps mod_actions.c et mod_cgi_redirect.c .

Une fois que vous avez l'un de ces modules de redirection cgi installé et configuré correctement, vous pouvez spécifier que vous voulez qu'un fichier soit filtré par php/fi en mettant simplement l'extension .phtml au fichier. De plus, si vous ajoutez index.phtml à votre ligne de configuration DirectoryIndex dans votre fichier srm.conf alors la page de plus haut niveau d'un répertoire sera automatiquement filtrée par php si vous appelez votre fichier index index.phtml.

HTTPD Netscape

Vous pouvez rediriger automatiquement les requêtes pour des fichiers ayant une extension donnée de façon à ce qu'ils soient pris en compte par PHP/FI en utilisant le module de Redirection du Serveur CGI Netscape. Ce module est disponible dans le Fichier Archives de la Page d'Accueil de the PHP/FI. Le fichier LIZSEZ_MOI (README) dans le paquetage explique clairement comment le configurer pour l'utiliser avec PHP/FI.

HTTPD NCSA

Actuellement NCSA ne supporte pas les modules, donc, pour effectuer une redirection cgi avec ce serveur, il vous faudra modifier le code source de votre serveur. Un colmatage pour faire cela avec NCSA 1.5 est disponible dans le fichier archives de PHP/FI.

25.7 Lancer PHP/FI à partir de la ligne de commande

Si vous fabriquez la version CGI de PHP/FI, vous pouvez l'utiliser simplement à partir de la ligne de commande, en tapant: php.cgi nom_de_fichier où nom_de_fichier est le fichier que vous voulez filtrer. Vous pouvez également créer des scripts PHP/FI autonomes en faisant ressembler la première ligne de votre script à quelque chose comme :

```
#!/usr/local/bin/php.cgi -q
```

L'option "-q" supprime l'impression des en-têtes HTTP. Vous pouvez vous passer de cette option si vous le désirez.

26 Interface Python pour PostgreSQL

PyGres95 est une interface pour PostgreSQL. Il est disponible à `(ftp://ftp.via.ecp.fr/pub/python/contrib/Database/PyGres95-1.0b.tar.gz )`

On trouvera ci-dessous un extrait de la page d'accueil de PyGres95:-

PyGres - v1.0b : module Postgres95 pour Python

PyGres95, version 1.0b Une interface Python pour la base de données Postgres95. Ecrit par Pascal Andre, andre@chimay.via.ecp.fr

Postgres95 est un SGBD dérivé de Postgres4.2. Il est conforme à (pour la plupart) ANSI SQL et offre beaucoup de possibilités intéressantes (Liens dynamiques C pour les fonctions ou les définitions de types, temps de réponse, ...). Ce paquetage est protégé par un copyright de l'Université de Californie (Regents of the University of California), et est diffusable librement.

Python est un langage de programmation interprété. Il est orienté objet et facile d'utilisation (syntaxe légère, simple et instructions simples), et dispose de nombreuses extensions pour construire des GUIs, des interfaces WWW,... Un butineur web "intelligent" (à la HotJava) est actuellement en cours de développement (november 1995), et cela devrait ouvrir de nombreuses portes aux programmeurs. Python est protégé par un copyright de Stichting Mathematisch Centrum, Amsterdam, Pays-Bas, et est diffusable librement.

PyGres95 est un module python d'interface à la base de données Postgres95 . Il comprend une bibliothèque de requêtes Postgres95 permettant une utilisation facile des fonctionnalités puissantes de Postgres95 en coopération avec tous les autres modules python. Il a été développé sur un système Linux 1.3/ELF, mais a été testé sur une plate-forme Solaris 2.4. De toutes façons, il devrait fonctionner sur toute plate-forme où python et postgres95 sont disponibles.

26.1 Où le trouver ...?

Les sites principaux des différents paquetages sont :

- Python : `(ftp.python.org:/pub/python)`
- Postgres95 : `(ftp.s2k-ftp.cs.berkeley.edu:/pub/postgres95)`
- PyGres95 : `(ftp.via.ecp.fr:/pub/python/contrib)`

Vous devriez cependant essayer de trouver un site miroir proche de votre propre site. Consultez les sources d'information pour trouver ces sites. PyGres95 devrait se trouver dans les répertoires contrib des sites Python et Postgres95.

26.2 Information et support

Si vous avez besoin d'informations concernant ces paquetages veuillez consulter leurs sites web:

- Python : `(http://www.python.org/)`
- Postgres95 : `(http://epoch.cs.berkeley.edu:8000/postgres95/index.html)`
- PyGres95 : `(http://www.via.ecp.fr/via/products/pygres.html)`

Pour un support :

- Python : newsgroup comp.lang.python
- Postgres95 : mailing list (see package documentation for information)
- PyGres95 : contact me andre@via.ecp.fr pour des rapports d'erreurs, des idées et des remarques

J'essaierai de répondre dans la mesure où mon temps libre me le permettra.

27 Passerelle entre PostgreSQL et WWW - WDB-P95

WDB-P95 - Une interface Web interface aux bases de données Postgres95. C'est à `(http://www.eol.ists.ca/ dunlop/wdb-p95/)`

Voici un extrait de la page d'accueil de WDB-P95 :-

Version 1.4b2 bêta - Créée par J. Douglas Dunlop

A propos de wdb-p95

C'est une version modifiée de wdb-1.3a2 qui fournit une passerelle WWW à Postgres95. Cette version nécessite également un Butineur capable de prendre en compte les Tables HTML pour avoir une sortie tabulée. Ceci n'est pas requis pour le wdb original et on peut facilement y revenir. (Je voulais seulement les tables parce que je n'étais pas d'accord avec les commandes < et >!)

Vous pouvez essayer ma Bande CASI et ma Requête d'Image. Vous pouvez jeter un coup d'oeil au Fichier de Définition de Formulaire (FDF) que j'ai utilisé pour créer la Bande CASI et également la Requête d'Image, qui comprends une jointure (JOIN) de 2 tables.

Cette version contient tous les fichiers nécessaires pour installer et faire tourner WDB-P95 en tant qu'interface à vos bases de données Postgres95 . Le portage de ce système pour une autre base de données devrait être relativement facile - à condition qu'elle supporte le SQL standard et possède une interface Perl.

27.1 Le serveur Postgres95, pgperl, et httpd doivent-ils résider sur le même hôte?

Non - Le serveur Postgres95 n'a pas à être sur le même hôte. Comme WDB-P95 est appelé par le démon http, ils doivent résider sur le même hôte. - Et comme WDB-P95 a été écrit pour utiliser Pg.pm - pgperl doit être sur le même hôte également. Pgperl a été écrit en utilisant la bibliothèque libpq, donc, il sera capable d'accéder à n'importe quel serveur Postgres95 n'importe où sur le réseau, juste comme n'importe quel autre client Postgres95. Comme illustré ci-dessous

{WWW Client (Netscape)} =¿{HTTP Server (NCSA's http) + WDB-P95 + pgperl + libpq}=¿{Postgres95 server}

Les parenthèses () représentent les machines.

Chaque machine peut être d'un type différent : NT, SUN, HP, ... mais il faut que vous ayez la bibliothèque d'interface libpq pour le type de machine sur lequel vous envisagez d'utiliser WDB-P95, puisqu'il vous faut compiler pgperl. (Le système a été conçu pour utiliser les tables HTML donc un client WWW récent est meilleur)

27.2 Nouvelle Version

Les nouvelles versions du logiciel et des pages ci-dessus sont toujours disponibles à partir de la page d'Accueil de WDB-P95. `(http://www.eol.ists.ca/ dunlop/wdb-p95/)`

Pour les questions et pour s'inscrire aux listes de diffusion contacter dunlop@eol.ists.ca

28 Interface au langage "C" pour PostgreSQL

Elle est incluse dans la distribution et s'appelle 'libpq'. Elle est similaire aux bibliothèques OCI Oracle, DB-lib Sybase, ou CLI Informix.

29 Interface au langage "C++" pour PostgreSQL

Elle est incluse dans la distribution et est nommée 'libpq++'.

30 ESQL/C pour PostgreSQL

C'est un précompilateur C intégré pour PostgreSQL ESQL/C comme Pro*C d'Oracle, et ESQL/C d'Informix :

- `(ftp://ftp.lysator.liu.se/pub/linus)`
- Email : linus@epact.se

ESQL/C pour PostgreSQLest une interface de programmation d'application SQL (API) qui permet au programmeur C de créer des applications personnalisées avec des possibilités de gestion de base de données. ESQL/C pour PostgreSQL vous permet d'utiliser un langage de troisième génération avec lequel vous êtes familiarisé tout en profitant des avantages d'un Langage de Requêtes Structuré (SQL).

ESQL/C est composé des éléments logiciels suivants:

- Les bibliothèques ESQL/C de fonctions C fournissent les accès au serveur de base de données.
- Les fichiers d'en-tête ESQL/C apportent les définitions des structures de données, les constantes et les macros utiles dans un programme ESQL/C.
- Le préprocesseur ESQL/C, qui est un préprocesseur de code source qui transforme un fichier C contenant des instructions SQL en fichier exécutable.

31 Opérateurs Bit-par-Bit pour PostgreSQL

Les opérateurs Bit-par-Bit ont été écrits par Nicolas Moldavsky nico@overnet.com.ar

Ce sont des fonctions "C" qui implantent les opérateurs bit-par-bit (AND, OR, XOR, bit complement) dans pgsql. Si quelqu'un désire les utiliser, il peut les récupérer par ftp anonyme de

- `(ftp://ftp.overnet.com.ar/pub/utils/linux/bitpgsql.tgz)`

De plus, il y a un "Makefile" qui marche bien sous Linux.

32 Les Codes Kanji Japonais pour PostgreSQL

Sont très utiles pour les Japonais. On les trouvera au site suivant : `(ftp://ftp.sra.co.jp/pub/cmd/postgres/)`

33 Portage de PostgreSQL Port pour Windows 95/Windows NT

Un portage pour Windows 95/Windows NT est en cours de réalisation. Ce portage de fait en utilisant gcc, gmake pour Win NT/95. Le programme gnu-win32 est utilisé pour compiler le code source sous win32. GNU gcc est disponible pour win32. Consultez ce site -

- `(http://www.cygnus.com/misc/gnu-win32)`

Récupérez-y le fichier cdk.exe (fichier auto-extractible pour gnu-win32)

Le portage peut également être réalisé en utilisant l'outil "Emulateur Unix sous NT" (Unix-Emulator on NT) suivant de

- `(http://www.softway.com)`

34 Listes de Diffusion et Groupes de Discussion

Il y a un groupe de discussion pour PostgreSQL à news.postgresql.org Veuillez suivre les étapes suivantes pour vous inscrire

```
1. Choisir : Fenetre | Netscape News
2. Choisir : Fichier | Open News Host
3. Taper news.postgresql.org dans la boite de dialogue.
4. Le nom du groupe de discussion est comp.databases.postgresql.questions.
Pour l'ajouter, cliquez avec le bouton de droite sur news.postgresql.org, et
choisir : Add Newsgroup.
Taper alors le nom du groupe de discussion dans la boite de dialogue. Le
groupe de discussion sera ajoute a la liste dans l'ecran de gauche.
5. Les noms des Groupes de Discussion seront listes dans l'ecran de gauche.
Tous les messages, dans les groupes selectionnes, seront affiches dans
l'ecran de droite.
```

Regardez les Titres des Listes de Diffusion sur la page web principale à : `(http://www.postgresql.org/)`

- Envoyez vos questions par courrier électronique à: pgsql-questions@postgresql.org
- Développeurs pgsql-hackers@postgresql.org
- Questions spécifiques à un portage pgsql-ports@postgresql.org
- Questions concernant la documentation pgsql-docs@postgresql.org

Vous recevrez une réponse, par courrier électronique, dans la journée qui suit!!

Vous pouvez également vous inscrire aux listes de diffusion. Pour vous inscrire ou vous radier d'une liste, envoyez un courrier à

- pgsql-questions-request@postgresql.org
- pgsql-hackers-request@postgresql.org
- pgsql-ports-request@postgresql.org
- pgsql-docs-request@postgresql.org

Le corps du message doit uniquement contenir la simple ligne

subscribe

(ou)

unsubscribe

Il y a aussi des listes de diffusion qui sont archivées au format html à l'endroit suivant -

- (`ftp://ftp.postgresql.org`) dans le répertoire /pub/majordomo

35 Livres et Documentations

On trouve dans la distribution

- Le 'Guide Utilisateur' pour PostgreSQL,
- Le 'Guide de Réalisation' détaillant la constitution interne de PostgreSQL.
- Les manuels "en ligne".
- Le manuels en ligne au format HTML.
- Egalement les manuels au format Postscript pour faire des éditions papier.

Documents de Référence: Ouvrages de référence utiles :

- "Understanding the New SQL: A Complete Guide" (Comprendre le Nouveau SQL: Un Guide Complet)- by Jim Melton and Alan R.Simon Morgan Kaufman Publisher. C'est un des meilleurs livres sur SQL.
- "A Guide to THE SQL STANDARD" (Un Guide du STANDARD SQL) - by C.J.Date Addison-Wesley Publishing company. C'est également un bon livre

```
Stephen Cannan and Gerard Otten
SQL - The Standard Handbook ,  Novembre 1992 (SQL - Le Manuel Standard)
McGraw-Hill Book Company Europe , Berkshire, SL6 2QL, England
```

```
Martin Gruber, Technical Editor: Joe Celko
SQL Instant Reference , 1993 (SQL Reference Immediate)
SYBEX Inc.  2021 Challenger Drive Alameda, CA 94501
```


Des centaines d'autres titres concernant SQL sont disponibles! Vérifiez-le dans une librairie.

36 Support Technique pour PostgreSQL

– Vous pouvez envoyer par courrier électronique les questions ou problèmes techniques auxquels vous devez faire face à : pgsql-questions@postgresql.org

et vous recevrez par courrier électronique une réponse dans la journée qui suit.

Dans un futur proche, l'organisation PostgreSQL vendra un support technique aux petites et grandes compagnies, les revenus de cette activité serviront à entretenir plusieurs sites miroirs (web and ftp) partout dans le monde. Ces revenus pourront aussi servir à créer une documentation imprimée, des guides, des livres pour aider les clients. Les profits seront également à organiser des conférences annuelles où les clients et professionnels pourront trouver et échanger des idées et des papiers techniques. Ceci devrait aider l'organisation PostgreSQL à se tenir debout.

37 Aspects Economiques et Commerciaux

Les bases de données commerciales paient de nombreuses taxes telles que des taxes fédérales, d'état, sur les ventes, sur les salariées, la sécurité sociale, les taxes pour les soins médicaux, des Indemnités pour les employés, des coûts de marketing et de publicité. Tous ces coûts ne sont pas destinés directement au développement de la base de donnée. Quand vous achetez une base de données commerciale une partie du montant est destiné aux taxes, aux dépenses de recherche et développement (R&D). Donc la valeur réelle pour la base de données est beaucoup plus faible.

De plus les bases de données commerciales doivent payer pour leurs immeubles/biens et pour l'achat de machines Unix, leur installation et leur maintenance. Tous ces coûts sont répercutés sur les clients.

PostgreSQL possède l'avantage, sur les bases de données commerciales, de ne pas supporter de taxes puisque développée sur l'internet. Un très grand nombre de personnes contribuent à son développement. Par exemple, dans un cas hypothétique, s'il y a un million de compagnies aux U.S.A et que chacune contribue pour environ $ 10 (en valeur de logiciel pour PostgreSQL) alors, chaque compagnie recevra dix millions de dollars!! C'est cela la magie du développement sur internet.

Actuellement, le code source de PostgreSQL est constitué d'environ 200 000 lignes de code "C" et "C++". Si le coût de chaque ligne de code "C" est évalué à $ 10, alors le coût total de PostgreSQL, tel qu'il est aujourd'hui est de $ 2 000 000 (deux millions de dollars!!).

De nombreuses compagnies ont déjà développé de grandes quantités de code "C", "C++" maison. Donc, en prenant le code source de PostgreSQL et en collaborant avec les autres compagnies sur internet, cela leur bénéficierait beaucoup en leur faisant économiser du temps et des efforts.

38 Spécifications ANSI/ISO SQL - SQL 1992, SQL 1998

38.1 Documents du standard National et International ANSI/ISO SQL

Ces documents sont situés à l'endroit indiqué ci-dessous. Il y a deux documents, le premier et le second, veuillez lire les deux.

- Aller à l'endroit indiqué ci-dessous (ftp://ftp.postgresql.org/pub/incoming/ANSI-ISO-SQL-SPECIFICATIONS) Et cliquer sur les deux fichiers nommés CLICK_ME_FIRST.html et CLICK_ME_SECOND.html
- (http://www.naiua.org/std-orgs.html)
- (http://www.ansi.org/docs) cliquer sur le fichier cat_c.html et effectuer une recherche de "Database SQL"
- Standard SQL92 (http://www.jcc.com) et cliquer sur le fichier sql_stnd.html

38.2 Syntaxe de l' ANSI/ISO SQL 1992

```
Ce fichier contient la grammaire en BNF du langage, arborescence
exploree d'abord en profondeur, realisee le 27-AOUT-1992 11:03:41.64.
La version specifique de BNF donnee ici est :
ANSI et SQL2-seulement.


<SQL terminal character> ::=
      <SQL language character>
    | <SQL embedded language character>

<SQL language character> ::=
      <simple Latin letter>
    | <digit>
    | <SQL special character>

<simple Latin letter> ::=
      <simple Latin upper case letter>
    | <simple Latin lower case letter>

<simple Latin upper case letter> ::=
          A | B | C | D | E | F | G | H | I | J | K | L | M | N | O
    | P | Q | R | S | T | U | V | W | X | Y | Z

<simple Latin lower case letter> ::=
          a | b | c | d | e | f | g | h | i | j | k | l | m | n | o
    | p | q | r | s | t | u | v | w | x | y | z

<digit> ::=
```

```
    0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<SQL special character> ::=
      <space>
    | <double quote>
    | <percent>
    | <ampersand>
    | <quote>
    | <left paren>
    | <right paren>
    | <asterisk>
    | <plus sign>
    | <comma>
    | <minus sign>
    | <period>
    | <solidus>
    | <colon>
    | <semicolon>
    | <less than operator>
    | <equals operator>
    | <greater than operator>
    | <question mark>
    | <underscore>
    | <vertical bar>

<space> ::= !! <EMPHASIS>(space character in character set in use)

<double quote> ::= "

<percent> ::= %

<ampersand> ::= &

<quote> ::= '

<left paren> ::= (

<right paren> ::= )

<asterisk> ::= *

<plus sign> ::= +

<comma> ::= ,

<minus sign> ::= -

<period> ::= .
```

```
<solidus> ::= /

<colon> ::= :

<semicolon> ::= ;

<less than operator> ::= <

<equals operator> ::= =

<greater than operator> ::= >

<question mark> ::= ?

<underscore> ::= _

<vertical bar> ::= |

<SQL embedded language character> ::=
      <left bracket>
    | <right bracket>

<left bracket> ::= [

<right bracket> ::= ]

<token> ::=
      <nondelimiter token>
    | <delimiter token>

<nondelimiter token> ::=
      <regular identifier>
    | <key word>
    | <unsigned numeric literal>
    | <national character string literal>
    | <bit string literal>
    | <hex string literal>

<regular identifier> ::= <identifier body>

<identifier body> ::=
    <identifier start> [ ( <underscore> | <identifier part> )... ]

<identifier start> ::= <EMPHASIS>(!! See the Syntax Rules)

<identifier part> ::=
      <identifier start>
```

```
    | <digit>

<key word> ::=
      <reserved word>
    | <non-reserved word>

<reserved word> ::=
      ABSOLUTE | ACTION | ADD | ALL
    | ALLOCATE | ALTER | AND
    | ANY | ARE
    | AS | ASC
    | ASSERTION | AT
    | AUTHORIZATION | AVG
    | BEGIN | BETWEEN | BIT | BIT_LENGTH
    | BOTH | BY
    | CASCADE | CASCADED | CASE | CAST
    | CATALOG
    | CHAR | CHARACTER | CHAR_LENGTH
    | CHARACTER_LENGTH | CHECK | CLOSE | COALESCE
    | COLLATE | COLLATION
    | COLUMN | COMMIT
    | CONNECT
    | CONNECTION | CONSTRAINT
    | CONSTRAINTS | CONTINUE
    | CONVERT | CORRESPONDING | COUNT | CREATE | CROSS
    | CURRENT
    | CURRENT_DATE | CURRENT_TIME
    | CURRENT_TIMESTAMP | CURRENT_USER | CURSOR
    | DATE | DAY | DEALLOCATE | DEC
    | DECIMAL | DECLARE | DEFAULT | DEFERRABLE
    | DEFERRED | DELETE | DESC | DESCRIBE | DESCRIPTOR
    | DIAGNOSTICS
    | DISCONNECT | DISTINCT | DOMAIN | DOUBLE | DROP
    | ELSE | END | END-EXEC | ESCAPE
    | EXCEPT | EXCEPTION
    | EXEC | EXECUTE | EXISTS
    | EXTERNAL | EXTRACT
    | FALSE | FETCH | FIRST | FLOAT | FOR
    | FOREIGN | FOUND | FROM | FULL
    | GET | GLOBAL | GO | GOTO
    | GRANT | GROUP
    | HAVING | HOUR
    | IDENTITY | IMMEDIATE | IN | INDICATOR
    | INITIALLY | INNER | INPUT
    | INSENSITIVE | INSERT | INT | INTEGER | INTERSECT
    | INTERVAL | INTO | IS
    | ISOLATION
    | JOIN
```

```
    | KEY
    | LANGUAGE | LAST | LEADING | LEFT
    | LEVEL | LIKE | LOCAL | LOWER
    | MATCH | MAX | MIN | MINUTE | MODULE
    | MONTH
    | NAMES | NATIONAL | NATURAL | NCHAR | NEXT | NO
    | NOT | NULL
    | NULLIF | NUMERIC
    | OCTET_LENGTH | OF
    | ON | ONLY | OPEN | OPTION | OR
    | ORDER | OUTER
    | OUTPUT | OVERLAPS
    | PAD | PARTIAL | POSITION | PRECISION | PREPARE
    | PRESERVE | PRIMARY
    | PRIOR | PRIVILEGES | PROCEDURE | PUBLIC
    | READ | REAL | REFERENCES | RELATIVE | RESTRICT
    | REVOKE | RIGHT
    | ROLLBACK | ROWS
    | SCHEMA | SCROLL | SECOND | SECTION
    | SELECT
    | SESSION | SESSION_USER | SET
    | SIZE | SMALLINT | SOME | SPACE | SQL | SQLCODE
    | SQLERROR | SQLSTATE
    | SUBSTRING | SUM | SYSTEM_USER
    | TABLE | TEMPORARY
    | THEN | TIME | TIMESTAMP
    | TIMEZONE_HOUR | TIMEZONE_MINUTE
    | TO | TRAILING | TRANSACTION
    | TRANSLATE | TRANSLATION | TRIM | TRUE
    | UNION | UNIQUE | UNKNOWN | UPDATE | UPPER | USAGE
    | USER | USING
    | VALUE | VALUES | VARCHAR | VARYING | VIEW
    | WHEN | WHENEVER | WHERE | WITH | WORK | WRITE
    | YEAR
    | ZONE

<non-reserved word> ::=

      ADA
    | C | CATALOG_NAME
    | CHARACTER_SET_CATALOG | CHARACTER_SET_NAME
    | CHARACTER_SET_SCHEMA | CLASS_ORIGIN | COBOL | COLLATION_CATALOG
    | COLLATION_NAME | COLLATION_SCHEMA | COLUMN_NAME | COMMAND_FUNCTION
    | COMMITTED
    | CONDITION_NUMBER | CONNECTION_NAME | CONSTRAINT_CATALOG | CONSTRAINT_NAME
    | CONSTRAINT_SCHEMA | CURSOR_NAME
    | DATA | DATETIME_INTERVAL_CODE
    | DATETIME_INTERVAL_PRECISION | DYNAMIC_FUNCTION
```

```
    | FORTRAN
    | LENGTH
    | MESSAGE_LENGTH | MESSAGE_OCTET_LENGTH | MESSAGE_TEXT | MORE | MUMPS
    | NAME | NULLABLE | NUMBER
    | PASCAL | PLI
    | REPEATABLE | RETURNED_LENGTH | RETURNED_OCTET_LENGTH | RETURNED_SQLSTATE
    | ROW_COUNT
    | SCALE | SCHEMA_NAME | SERIALIZABLE | SERVER_NAME | SUBCLASS_ORIGIN
    | TABLE_NAME | TYPE
    | UNCOMMITTED | UNNAMED

<unsigned numeric literal> ::=
      <exact numeric literal>
    | <approximate numeric literal>

<exact numeric literal> ::=
      <unsigned integer> [ <period> [ <unsigned integer> ] ]
    | <period> <unsigned integer>

<unsigned integer> ::= <digit>...

<approximate numeric literal> ::= <mantissa> E <exponent>

<mantissa> ::= <exact numeric literal>

<exponent> ::= <signed integer>

<signed integer> ::= [ <sign> ] <unsigned integer>

<sign> ::= <plus sign> | <minus sign>

<national character string literal> ::=
    N <quote> [ <character representation>... ] <quote>
      [ ( <separator>... <quote> [ <character representation>... ] <quote> )... ]

<character representation> ::=
      <nonquote character>
    | <quote symbol>

<nonquote character> ::= !! <EMPHASIS>(See the Syntax Rules.)

<quote symbol> ::= <quote><quote>

<separator> ::= ( <comment> | <space> | <newline> )...

<comment> ::=
    <comment introducer> [ <comment character>... ] <newline>
```

```
<comment introducer> ::= <minus sign><minus sign>[<minus sign>...]

<comment character> ::=
      <nonquote character>
    | <quote>

<newline> ::= !! <EMPHASIS>(implementation-defined end-of-line indicator)

<bit string literal> ::=
    B <quote> [ <bit>... ] <quote>
      [ ( <separator>... <quote> [ <bit>... ] <quote> )... ]

<bit> ::= 0 | 1

<hex string literal> ::=
    X <quote> [ <hexit>... ] <quote>
      [ ( <separator>... <quote> [ <hexit>... ] <quote> )... ]

<hexit> ::= <digit> | A | B | C | D | E | F | a | b | c | d | e | f

<delimiter token> ::=
      <character string literal>
    | <date string>
    | <time string>
    | <timestamp string>
    | <interval string>
    | <delimited identifier>
    | <SQL special character>
    | <not equals operator>
    | <greater than or equals operator>
    | <less than or equals operator>
    | <concatenation operator>
    | <double period>
    | <left bracket>
    | <right bracket>

<character string literal> ::=
    [ <introducer><character set specification> ]
    <quote> [ <character representation>... ] <quote>
      [ ( <separator>... <quote> [ <character representation>... ] <quote> )... ]

<introducer> ::= <underscore>

<character set specification> ::=
      <standard character repertoire name>
    | <implementation-defined character repertoire name>
    | <user-defined character repertoire name>
    | <standard universal character form-of-use name>
```

```
    | <implementation-defined universal character form-of-use name>

<standard character repertoire name> ::= <character set name>

<character set name> ::= [ <schema name> <period> ]
      <SQL language identifier>

<schema name> ::=
    [ <catalog name> <period> ] <unqualified schema name>

<catalog name> ::= <identifier>

<identifier> ::=
    [ <introducer><character set specification> ] <actual identifier>

<actual identifier> ::=
      <regular identifier>
    | <delimited identifier>

<delimited identifier> ::=
    <double quote> <delimited identifier body> <double quote>

<delimited identifier body> ::= <delimited identifier part>...

<delimited identifier part> ::=
      <nondoublequote character>
    | <doublequote symbol>

<nondoublequote character> ::= <EMPHASIS>(!! See the Syntax Rules)

<doublequote symbol> ::= <double quote><double quote>

<unqualified schema name> ::= <identifier>

<SQL language identifier> ::=
    <SQL language identifier start>
       [ ( <underscore> | <SQL language identifier part> )... ]

<SQL language identifier start> ::= <simple Latin letter>

<SQL language identifier part> ::=
      <simple Latin letter>
    | <digit>

<implementation-defined character repertoire name> ::=
    <character set name>

<user-defined character repertoire name> ::= <character set name>
```

```
<standard universal character form-of-use name> ::=
    <character set name>

<implementation-defined universal character form-of-use name> ::=
    <character set name>

<date string> ::=
    <quote> <date value> <quote>

<date value> ::=
    <years value> <minus sign> <months value>
        <minus sign> <days value>

<years value> ::= <datetime value>

<datetime value> ::= <unsigned integer>

<months value> ::= <datetime value>

<days value> ::= <datetime value>

<time string> ::=
    <quote> <time value> [ <time zone interval> ] <quote>

<time value> ::=
    <hours value> <colon> <minutes value> <colon> <seconds value>

<hours value> ::= <datetime value>

<minutes value> ::= <datetime value>

<seconds value> ::=
      <seconds integer value> [ <period> [ <seconds fraction> ] ]

<seconds integer value> ::= <unsigned integer>

<seconds fraction> ::= <unsigned integer>

<time zone interval> ::=
    <sign> <hours value> <colon> <minutes value>

<timestamp string> ::=
    <quote> <date value> <space> <time value>
        [ <time zone interval> ] <quote>

<interval string> ::=
    <quote> ( <year-month literal> | <day-time literal> ) <quote>
```

```
<year-month literal> ::=
      <years value>
    | [ <years value> <minus sign> ] <months value>

<day-time literal> ::=
      <day-time interval>
    | <time interval>

<day-time interval> ::=
    <days value>
      [ <space> <hours value> [ <colon> <minutes value>
        [ <colon> <seconds value> ] ] ]

<time interval> ::=
      <hours value> [ <colon> <minutes value> [ <colon> <seconds value> ] ]
    | <minutes value> [ <colon> <seconds value> ]
    | <seconds value>

<not equals operator> ::= <>

<greater than or equals operator> ::= >=

<less than or equals operator> ::= <=

<concatenation operator> ::= ||

<double period> ::= ..

<module> ::=
    <module name clause>
    <language clause>
    <module authorization clause>
    [ <temporary table declaration>... ]
    <module contents>...

<module name clause> ::=
    MODULE [ <module name> ]
      [ <module character set specification> ]

<module name> ::= <identifier>

<module character set specification> ::=
    NAMES ARE <character set specification>

<language clause> ::=
    LANGUAGE <language name>
```

```
<language name> ::=
    ADA | C | COBOL | FORTRAN | MUMPS | PASCAL | PLI

<module authorization clause> ::=
      SCHEMA <schema name>
    | AUTHORIZATION <module authorization identifier>
    | SCHEMA <schema name>
          AUTHORIZATION <module authorization identifier>

<module authorization identifier> ::=
    <authorization identifier>

<authorization identifier> ::= <identifier>

<temporary table declaration> ::=
    DECLARE LOCAL TEMPORARY TABLE
        <qualified local table name>
      <table element list>
      [ ON COMMIT ( PRESERVE | DELETE ) ROWS ]

<qualified local table name> ::=
    MODULE <period> <local table name>

<local table name> ::= <qualified identifier>

<qualified identifier> ::= <identifier>

<table element list> ::=
      <left paren> <table element> [ ( <comma> <table element> )... ] <right paren>

<table element> ::=
      <column definition>
    | <table constraint definition>

<column definition> ::=
    <column name> ( <data type> | <domain name> )
    [ <default clause> ]
    [ <column constraint definition>... ]
    [ <collate clause> ]

<column name> ::= <identifier>

<data type> ::=
      <character string type>
           [ CHARACTER SET <character set specification> ]
    | <national character string type>
    | <bit string type>
    | <numeric type>
```

```
    | <datetime type>
    | <interval type>

<character string type> ::=
      CHARACTER [ <left paren> <length> <right paren> ]
    | CHAR [ <left paren> <length> <right paren> ]
    | CHARACTER VARYING <left paren> <length> <right paren>
    | CHAR VARYING <left paren> <length> <right paren>
    | VARCHAR <left paren> <length> <right paren>

<length> ::= <unsigned integer>

<national character string type> ::=
      NATIONAL CHARACTER [ <left paren> <length> <right paren> ]
    | NATIONAL CHAR [ <left paren> <length> <right paren> ]
    | NCHAR [ <left paren> <length> <right paren> ]
    | NATIONAL CHARACTER VARYING <left paren> <length> <right paren>
    | NATIONAL CHAR VARYING <left paren> <length> <right paren>
    | NCHAR VARYING <left paren> <length> <right paren>

<bit string type> ::=
      BIT [ <left paren> <length> <right paren> ]
    | BIT VARYING <left paren> <length> <right paren>

<numeric type> ::=
      <exact numeric type>
    | <approximate numeric type>

<exact numeric type> ::=
      NUMERIC [ <left paren> <precision> [ <comma> <scale> ] <right paren> ]
    | DECIMAL [ <left paren> <precision> [ <comma> <scale> ] <right paren> ]
    | DEC [ <left paren> <precision> [ <comma> <scale> ] <right paren> ]
    | INTEGER
    | INT
    | SMALLINT

<precision> ::= <unsigned integer>

<scale> ::= <unsigned integer>

<approximate numeric type> ::=
      FLOAT [ <left paren> <precision> <right paren> ]
    | REAL
    | DOUBLE PRECISION

<datetime type> ::=
      DATE
    | TIME [ <left paren> <time precision> <right paren> ]
```

```
          [ WITH TIME ZONE ]
    | TIMESTAMP [ <left paren> <timestamp precision> <right paren> ]
          [ WITH TIME ZONE ]

<time precision> ::= <time fractional seconds precision>

<time fractional seconds precision> ::= <unsigned integer>

<timestamp precision> ::= <time fractional seconds precision>

<interval type> ::= INTERVAL <interval qualifier>

<interval qualifier> ::=
      <start field> TO <end field>
    | <single datetime field>

<start field> ::=
    <non-second datetime field>
        [ <left paren> <interval leading field precision> <right paren> ]

<non-second datetime field> ::= YEAR | MONTH | DAY | HOUR
    | MINUTE

<interval leading field precision> ::= <unsigned integer>

<end field> ::=
      <non-second datetime field>
    | SECOND [ <left paren> <interval fractional seconds precision> <right paren> ]

<interval fractional seconds precision> ::= <unsigned integer>

<single datetime field> ::=
      <non-second datetime field>
          [ <left paren> <interval leading field precision> <right paren> ]
    | SECOND [ <left paren> <interval leading field precision>
          [ <comma> <interval fractional seconds precision> ] <right paren> ]

<domain name> ::= <qualified name>

<qualified name> ::=
    [ <schema name> <period> ] <qualified identifier>

<default clause> ::=
      DEFAULT <default option>

<default option> ::=
      <literal>
    | <datetime value function>
```

```
    | USER
    | CURRENT_USER
    | SESSION_USER
    | SYSTEM_USER
    | NULL

<literal> ::=
      <signed numeric literal>
    | <general literal>

<signed numeric literal> ::=
    [ <sign> ] <unsigned numeric literal>

<general literal> ::=
      <character string literal>
    | <national character string literal>
    | <bit string literal>
    | <hex string literal>
    | <datetime literal>
    | <interval literal>

<datetime literal> ::=
      <date literal>
    | <time literal>
    | <timestamp literal>

<date literal> ::=
    DATE <date string>

<time literal> ::=
    TIME <time string>

<timestamp literal> ::=
    TIMESTAMP <timestamp string>

<interval literal> ::=
    INTERVAL [ <sign> ] <interval string> <interval qualifier>

<datetime value function> ::=
      <current date value function>
    | <current time value function>
    | <current timestamp value function>

<current date value function> ::= CURRENT_DATE

<current time value function> ::=
      CURRENT_TIME [ <left paren> <time precision> <right paren> ]
```

```
<current timestamp value function> ::=
      CURRENT_TIMESTAMP [ <left paren> <timestamp precision> <right paren> ]

<column constraint definition> ::=
    [ <constraint name definition> ]
    <column constraint>
      [ <constraint attributes> ]

<constraint name definition> ::= CONSTRAINT <constraint name>

<constraint name> ::= <qualified name>

<column constraint> ::=
      NOT NULL
    | <unique specification>
    | <references specification>
    | <check constraint definition>

<unique specification> ::=
    UNIQUE | PRIMARY KEY

<references specification> ::=
    REFERENCES <referenced table and columns>
      [ MATCH <match type> ]
      [ <referential triggered action> ]

<referenced table and columns> ::=
     <table name> [ <left paren> <reference column list> <right paren> ]

<table name> ::=
      <qualified name>
    | <qualified local table name>

<reference column list> ::= <column name list>

<column name list> ::=
    <column name> [ ( <comma> <column name> )... ]

<match type> ::=
      FULL
    | PARTIAL

<referential triggered action> ::=
      <update rule> [ <delete rule> ]
    | <delete rule> [ <update rule> ]

<update rule> ::= ON UPDATE <referential action>
```

```
<referential action> ::=
      CASCADE
    | SET NULL
    | SET DEFAULT
    | NO ACTION

<delete rule> ::= ON DELETE <referential action>

<check constraint definition> ::=
    CHECK
        <left paren> <search condition> <right paren>

<search condition> ::=
      <boolean term>
    | <search condition> OR <boolean term>

<boolean term> ::=
      <boolean factor>
    | <boolean term> AND <boolean factor>

<boolean factor> ::=
    [ NOT ] <boolean test>

<boolean test> ::=
    <boolean primary> [ IS [ NOT ]
          <truth value> ]

<boolean primary> ::=
      <predicate>
    | <left paren> <search condition> <right paren>

<predicate> ::=
      <comparison predicate>
    | <between predicate>
    | <in predicate>
    | <like predicate>
    | <null predicate>
    | <quantified comparison predicate>
    | <exists predicate>
    | <unique predicate>
    | <match predicate>
    | <overlaps predicate>

<comparison predicate> ::=
    <row value constructor> <comp op>
        <row value constructor>

<row value constructor> ::=
```

```
       <row value constructor element>
     | <left paren> <row value constructor list> <right paren>
     | <row subquery>

<row value constructor element> ::=
       <value expression>
     | <null specification>
     | <default specification>

<value expression> ::=
       <numeric value expression>
     | <string value expression>
     | <datetime value expression>
     | <interval value expression>

<numeric value expression> ::=
       <term>
     | <numeric value expression> <plus sign> <term>
     | <numeric value expression> <minus sign> <term>

<term> ::=
       <factor>
     | <term> <asterisk> <factor>
     | <term> <solidus> <factor>

<factor> ::=
     [ <sign> ] <numeric primary>

<numeric primary> ::=
       <value expression primary>
     | <numeric value function>

<value expression primary> ::=
       <unsigned value specification>
     | <column reference>
     | <set function specification>
     | <scalar subquery>
     | <case expression>
     | <left paren> <value expression> <right paren>
     | <cast specification>

<unsigned value specification> ::=
       <unsigned literal>
     | <general value specification>

<unsigned literal> ::=
       <unsigned numeric literal>
     | <general literal>
```

```
<general value specification> ::=
      <parameter specification>
    | <dynamic parameter specification>
    | <variable specification>
    | USER
    | CURRENT_USER
    | SESSION_USER
    | SYSTEM_USER
    | VALUE

<parameter specification> ::=
    <parameter name> [ <indicator parameter> ]

<parameter name> ::= <colon> <identifier>

<indicator parameter> ::=
    [ INDICATOR ] <parameter name>

<dynamic parameter specification> ::= <question mark>

<variable specification> ::=
    <embedded variable name> [ <indicator variable> ]

<embedded variable name> ::=
    <colon><host identifier>

<host identifier> ::=
      <Ada host identifier>
    | <C host identifier>
    | <COBOL host identifier>
    | <Fortran host identifier>
    | <MUMPS host identifier>
    | <Pascal host identifier>
    | <PL/I host identifier>

<Ada host identifier> ::= !! <EMPHASIS>(See the Syntax Rules.)

<C host identifier> ::=
    !! <EMPHASIS>(See the Syntax Rules.)

<COBOL host identifier> ::= !! <EMPHASIS>(See the Syntax Rules.)

<Fortran host identifier> ::= !! <EMPHASIS>(See the Syntax Rules.)

<MUMPS host identifier> ::= !! <EMPHASIS>(See the Syntax Rules.)

<Pascal host identifier> ::= !! <EMPHASIS>(See the Syntax Rules.)
```

```
<PL/I host identifier> ::= !! <EMPHASIS>(See the Syntax Rules.)

<indicator variable> ::=
    [ INDICATOR ] <embedded variable name>

<column reference> ::= [ <qualifier> <period> ] <column name>

<qualifier> ::=
      <table name>
    | <correlation name>

<correlation name> ::= <identifier>

<set function specification> ::=
      COUNT <left paren> <asterisk> <right paren>
    | <general set function>

<general set function> ::=
      <set function type>
          <left paren> [ <set quantifier> ] <value expression> <right paren>

<set function type> ::=
    AVG | MAX | MIN | SUM | COUNT

<set quantifier> ::= DISTINCT | ALL

<scalar subquery> ::= <subquery>

<subquery> ::= <left paren> <query expression> <right paren>

<query expression> ::=
      <non-join query expression>
    | <joined table>

<non-join query expression> ::=
      <non-join query term>
    | <query expression> UNION  [ ALL ]
          [ <corresponding spec> ] <query term>
    | <query expression> EXCEPT [ ALL ]
          [ <corresponding spec> ] <query term>

<non-join query term> ::=
      <non-join query primary>
    | <query term> INTERSECT [ ALL ]
          [ <corresponding spec> ] <query primary>

<non-join query primary> ::=
```

```
      <simple table>
    | <left paren> <non-join query expression> <right paren>

<simple table> ::=
      <query specification>
    | <table value constructor>
    | <explicit table>

<query specification> ::=
    SELECT [ <set quantifier> ] <select list> <table expression>

<select list> ::=
      <asterisk>
    | <select sublist> [ ( <comma> <select sublist> )... ]

<select sublist> ::=
      <derived column>
    | <qualifier> <period> <asterisk>

<derived column> ::= <value expression> [ <as clause> ]

<as clause> ::= [ AS ] <column name>

<table expression> ::=
    <from clause>
    [ <where clause> ]
    [ <group by clause> ]
    [ <having clause> ]

<from clause> ::= FROM <table reference>
    [ ( <comma> <table reference> )... ]

<table reference> ::=
      <table name> [ [ AS ] <correlation name>
          [ <left paren> <derived column list> <right paren> ] ]
    | <derived table> [ AS ] <correlation name>
          [ <left paren> <derived column list> <right paren> ]
    | <joined table>

<derived column list> ::= <column name list>

<derived table> ::= <table subquery>

<table subquery> ::= <subquery>

<joined table> ::=
      <cross join>
    | <qualified join>
```

```
    | <left paren> <joined table> <right paren>

<cross join> ::=
    <table reference> CROSS JOIN <table reference>

<qualified join> ::=
    <table reference> [ NATURAL ] [ <join type> ] JOIN
      <table reference> [ <join specification> ]

<join type> ::=
      INNER
    | <outer join type> [ OUTER ]
    | UNION

<outer join type> ::=
      LEFT
    | RIGHT
    | FULL

<join specification> ::=
      <join condition>
    | <named columns join>

<join condition> ::= ON <search condition>

<named columns join> ::=
    USING <left paren> <join column list> <right paren>

<join column list> ::= <column name list>

<where clause> ::= WHERE <search condition>

<group by clause> ::=
    GROUP BY <grouping column reference list>

<grouping column reference list> ::=
    <grouping column reference>
        [ ( <comma> <grouping column reference> )... ]

<grouping column reference> ::=
    <column reference> [ <collate clause> ]

<collate clause> ::= COLLATE <collation name>

<collation name> ::= <qualified name>

<having clause> ::= HAVING <search condition>
```

```
<table value constructor> ::=
    VALUES <table value constructor list>

<table value constructor list> ::=
    <row value constructor> [ ( <comma> <row value constructor> )... ]

<explicit table> ::= TABLE <table name>

<query term> ::=
      <non-join query term>
    | <joined table>

<corresponding spec> ::=
    CORRESPONDING [ BY <left paren> <corresponding column list> <right paren> ]

<corresponding column list> ::= <column name list>

<query primary> ::=
      <non-join query primary>
    | <joined table>

<case expression> ::=
      <case abbreviation>
    | <case specification>

<case abbreviation> ::=
      NULLIF <left paren> <value expression> <comma>
            <value expression> <right paren>
    | COALESCE <left paren> <value expression>
            ( <comma> <value expression> )... <right paren>

<case specification> ::=
      <simple case>
    | <searched case>

<simple case> ::=
    CASE <case operand>
      <simple when clause>...
      [ <else clause> ]
    END

<case operand> ::= <value expression>

<simple when clause> ::= WHEN <when operand> THEN <result>

<when operand> ::= <value expression>

<result> ::= <result expression> | NULL
```

```
<result expression> ::= <value expression>

<else clause> ::= ELSE <result>

<searched case> ::=
    CASE
      <searched when clause>...
      [ <else clause> ]
    END

<searched when clause> ::= WHEN <search condition> THEN <result>

<cast specification> ::=
    CAST <left paren> <cast operand> AS
        <cast target> <right paren>

<cast operand> ::=
      <value expression>
    | NULL

<cast target> ::=
      <domain name>
    | <data type>

<numeric value function> ::=
      <position expression>
    | <extract expression>
    | <length expression>

<position expression> ::=
    POSITION <left paren> <character value expression>
        IN <character value expression> <right paren>

<character value expression> ::=
      <concatenation>
    | <character factor>

<concatenation> ::=
    <character value expression> <concatenation operator>
        <character factor>

<character factor> ::=
    <character primary> [ <collate clause> ]

<character primary> ::=
      <value expression primary>
    | <string value function>
```

```
<string value function> ::=
      <character value function>
    | <bit value function>

<character value function> ::=
      <character substring function>
    | <fold>
    | <form-of-use conversion>
    | <character translation>
    | <trim function>

<character substring function> ::=
    SUBSTRING <left paren> <character value expression> FROM <start position>
                [ FOR <string length> ] <right paren>

<start position> ::= <numeric value expression>

<string length> ::= <numeric value expression>

<fold> ::= ( UPPER | LOWER )
     <left paren> <character value expression> <right paren>

<form-of-use conversion> ::=
    CONVERT <left paren> <character value expression>
        USING <form-of-use conversion name> <right paren>

<form-of-use conversion name> ::= <qualified name>

<character translation> ::=
    TRANSLATE <left paren> <character value expression>
        USING <translation name> <right paren>

<translation name> ::= <qualified name>

<trim function> ::=
    TRIM <left paren> <trim operands> <right paren>

<trim operands> ::=
    [ [ <trim specification> ] [ <trim character> ] FROM ] <trim source>

<trim specification> ::=
      LEADING
    | TRAILING
    | BOTH

<trim character> ::= <character value expression>
```

```
<trim source> ::= <character value expression>

<bit value function> ::=
    <bit substring function>

<bit substring function> ::=
    SUBSTRING <left paren> <bit value expression> FROM <start position>
        [ FOR <string length> ] <right paren>

<bit value expression> ::=
      <bit concatenation>
    | <bit factor>

<bit concatenation> ::=
    <bit value expression> <concatenation operator> <bit factor>

<bit factor> ::= <bit primary>

<bit primary> ::=
      <value expression primary>
    | <string value function>

<extract expression> ::=
    EXTRACT <left paren> <extract field>
        FROM <extract source> <right paren>

<extract field> ::=
      <datetime field>
    | <time zone field>

<datetime field> ::=
      <non-second datetime field>
    | SECOND

<time zone field> ::=
      TIMEZONE_HOUR
    | TIMEZONE_MINUTE

<extract source> ::=
      <datetime value expression>
    | <interval value expression>

<datetime value expression> ::=
      <datetime term>
    | <interval value expression> <plus sign> <datetime term>
    | <datetime value expression> <plus sign> <interval term>
    | <datetime value expression> <minus sign> <interval term>
```

```
<interval term> ::=
      <interval factor>
    | <interval term 2> <asterisk> <factor>
    | <interval term 2> <solidus> <factor>
    | <term> <asterisk> <interval factor>

<interval factor> ::=
    [ <sign> ] <interval primary>

<interval primary> ::=
      <value expression primary> [ <interval qualifier> ]

<interval term 2> ::= <interval term>

<interval value expression> ::=
      <interval term>
    | <interval value expression 1> <plus sign> <interval term 1>
    | <interval value expression 1> <minus sign> <interval term 1>
    | <left paren> <datetime value expression> <minus sign>
          <datetime term> <right paren> <interval qualifier>

<interval value expression 1> ::= <interval value expression>

<interval term 1> ::= <interval term>

<datetime term> ::=
      <datetime factor>

<datetime factor> ::=
      <datetime primary> [ <time zone> ]

<datetime primary> ::=
      <value expression primary>
    | <datetime value function>

<time zone> ::=
    AT <time zone specifier>

<time zone specifier> ::=
      LOCAL
    | TIME ZONE <interval value expression>

<length expression> ::=
      <char length expression>
    | <octet length expression>
    | <bit length expression>

<char length expression> ::=
```

```
    ( CHAR_LENGTH | CHARACTER_LENGTH )
        <left paren> <string value expression> <right paren>

<string value expression> ::=
      <character value expression>
    | <bit value expression>

<octet length expression> ::=
    OCTET_LENGTH <left paren> <string value expression> <right paren>

<bit length expression> ::=
    BIT_LENGTH <left paren> <string value expression> <right paren>

<null specification> ::=
    NULL

<default specification> ::=
    DEFAULT

<row value constructor list> ::=
    <row value constructor element>
        [ ( <comma> <row value constructor element> )... ]

<row subquery> ::= <subquery>

<comp op> ::=
      <equals operator>
    | <not equals operator>
    | <less than operator>
    | <greater than operator>
    | <less than or equals operator>
    | <greater than or equals operator>

<between predicate> ::=
    <row value constructor> [ NOT ] BETWEEN
      <row value constructor> AND <row value constructor>

<in predicate> ::=
    <row value constructor>
      [ NOT ] IN <in predicate value>

<in predicate value> ::=
      <table subquery>
    | <left paren> <in value list> <right paren>

<in value list> ::=
    <value expression> ( <comma> <value expression> )...
```

```
<like predicate> ::=
    <match value> [ NOT ] LIKE <pattern>
      [ ESCAPE <escape character> ]

<match value> ::= <character value expression>

<pattern> ::= <character value expression>

<escape character> ::= <character value expression>

<null predicate> ::= <row value constructor>
    IS [ NOT ] NULL

<quantified comparison predicate> ::=
    <row value constructor> <comp op> <quantifier> <table subquery>

<quantifier> ::= <all> | <some>

<all> ::= ALL

<some> ::= SOME | ANY

<exists predicate> ::= EXISTS <table subquery>

<unique predicate> ::= UNIQUE <table subquery>

<match predicate> ::=
    <row value constructor> MATCH [ UNIQUE ]
        [ PARTIAL | FULL ] <table subquery>

<overlaps predicate> ::=
    <row value constructor 1> OVERLAPS <row value constructor 2>

<row value constructor 1> ::= <row value constructor>

<row value constructor 2> ::= <row value constructor>

<truth value> ::=
      TRUE
    | FALSE
    | UNKNOWN

<constraint attributes> ::=
      <constraint check time> [ [ NOT ] DEFERRABLE ]
    | [ NOT ] DEFERRABLE [ <constraint check time> ]

<constraint check time> ::=
      INITIALLY DEFERRED
```

```
    | INITIALLY IMMEDIATE

<table constraint definition> ::=
    [ <constraint name definition> ]
    <table constraint> [ <constraint attributes> ]

<table constraint> ::=
      <unique constraint definition>
    | <referential constraint definition>
    | <check constraint definition>

<unique constraint definition> ::=
            <unique specification> even in SQL3)
    <unique specification>
      <left paren> <unique column list> <right paren>

<unique column list> ::= <column name list>

<referential constraint definition> ::=
    FOREIGN KEY
        <left paren> <referencing columns> <right paren>
      <references specification>

<referencing columns> ::=
    <reference column list>

<module contents> ::=
      <declare cursor>
    | <dynamic declare cursor>
    | <procedure>

<declare cursor> ::=
    DECLARE <cursor name> [ INSENSITIVE ] [ SCROLL ] CURSOR
      FOR <cursor specification>

<cursor name> ::= <identifier>

<cursor specification> ::=
    <query expression> [ <order by clause> ]
      [ <updatability clause> ]

<order by clause> ::=
    ORDER BY <sort specification list>

<sort specification list> ::=
    <sort specification> [ ( <comma> <sort specification> )... ]

<sort specification> ::=
```

```
    <sort key> [ <collate clause> ] [ <ordering specification> ]

<sort key> ::=
      <column name>
    | <unsigned integer>

<ordering specification> ::= ASC | DESC

<updatability clause> ::=
    FOR
        ( READ ONLY |
          UPDATE [ OF <column name list> ] )

<dynamic declare cursor> ::=
    DECLARE <cursor name> [ INSENSITIVE ] [ SCROLL ] CURSOR
        FOR <statement name>

<statement name> ::= <identifier>

<procedure> ::=
    PROCEDURE <procedure name>
        <parameter declaration list> <semicolon>
      <SQL procedure statement> <semicolon>

<procedure name> ::= <identifier>

<parameter declaration list> ::=
      <left paren> <parameter declaration>
          [ ( <comma> <parameter declaration> )... ] <right paren>
    | <parameter declaration>...

<parameter declaration> ::=
      <parameter name> <data type>
    | <status parameter>

<status parameter> ::=
    SQLCODE | SQLSTATE

<SQL procedure statement> ::=
      <SQL schema statement>
    | <SQL data statement>
    | <SQL transaction statement>
    | <SQL connection statement>
    | <SQL session statement>
    | <SQL dynamic statement>
    | <SQL diagnostics statement>

<SQL schema statement> ::=
```

```
      <SQL schema definition statement>
    | <SQL schema manipulation statement>

<SQL schema definition statement> ::=
      <schema definition>
    | <table definition>
    | <view definition>
    | <grant statement>
    | <domain definition>
    | <character set definition>
    | <collation definition>
    | <translation definition>
    | <assertion definition>

<schema definition> ::=
    CREATE SCHEMA <schema name clause>
      [ <schema character set specification> ]
      [ <schema element>... ]

<schema name clause> ::=
      <schema name>
    | AUTHORIZATION <schema authorization identifier>
    | <schema name> AUTHORIZATION
          <schema authorization identifier>

<schema authorization identifier> ::=
    <authorization identifier>

<schema character set specification> ::=
    DEFAULT CHARACTER
        SET <character set specification>

<schema element> ::=
      <domain definition>
    | <table definition>
    | <view definition>
    | <grant statement>
    | <assertion definition>
    | <character set definition>
    | <collation definition>
    | <translation definition>

<domain definition> ::=
    CREATE DOMAIN <domain name>
        [ AS ] <data type>
      [ <default clause> ]
      [ <domain constraint>... ]
      [ <collate clause> ]
```

```
<domain constraint> ::=
    [ <constraint name definition> ]
    <check constraint definition> [ <constraint attributes> ]

<table definition> ::=
    CREATE [ ( GLOBAL | LOCAL ) TEMPORARY ] TABLE
        <table name>
      <table element list>
      [ ON COMMIT ( DELETE | PRESERVE ) ROWS ]

<view definition> ::=
    CREATE VIEW <table name> [ <left paren> <view column list>
                                  <right paren> ]
      AS <query expression>
      [ WITH [ <levels clause> ] CHECK OPTION ]

<view column list> ::= <column name list>

<levels clause> ::=
    CASCADED | LOCAL

<grant statement> ::=
   GRANT <privileges> ON <object name>
     TO <grantee> [ ( <comma> <grantee> )... ]
       [ WITH GRANT OPTION ]

<privileges> ::=
      ALL PRIVILEGES
    | <action list>

<action list> ::= <action> [ ( <comma> <action> )... ]

<action> ::=
      SELECT
    | DELETE
    | INSERT [ <left paren> <privilege column list> <right paren> ]
    | UPDATE [ <left paren> <privilege column list> <right paren> ]
    | REFERENCES [ <left paren> <privilege column list> <right paren> ]
    | USAGE

<privilege column list> ::= <column name list>

<object name> ::=
      [ TABLE ] <table name>
    | DOMAIN <domain name>
    | COLLATION <collation name>
    | CHARACTER SET <character set name>
```

```
    | TRANSLATION <translation name>

<grantee> ::=
      PUBLIC
    | <authorization identifier>

<assertion definition> ::=
    CREATE ASSERTION <constraint name> <assertion check>
      [ <constraint attributes> ]

<assertion check> ::=
    CHECK
        <left paren> <search condition> <right paren>

<character set definition> ::=
    CREATE CHARACTER SET <character set name>
        [ AS ]
      <character set source>
      [ <collate clause> | <limited collation definition> ]

<character set source> ::=
      GET <existing character set name>

<existing character set name> ::=
      <standard character repertoire name>
    | <implementation-defined character repertoire name>
    | <schema character set name>

<schema character set name> ::= <character set name>

<limited collation definition> ::=
    COLLATION FROM <collation source>

<collation source> ::=
      <collating sequence definition>
    | <translation collation>

<collating sequence definition> ::=
      <external collation>
    | <schema collation name>
    | DESC <left paren> <collation name> <right paren>
    | DEFAULT

<external collation> ::=
    EXTERNAL <left paren> <quote> <external collation name> <quote> <right paren>

<external collation name> ::=
      <standard collation name>
```

```
    | <implementation-defined collation name>

<standard collation name> ::= <collation name>

<implementation-defined collation name> ::= <collation name>

<schema collation name> ::= <collation name>

<translation collation> ::=
    TRANSLATION <translation name>
        [ THEN COLLATION <collation name> ]

<collation definition> ::=
    CREATE COLLATION <collation name> FOR
        <character set specification>
      FROM <collation source>
        [ <pad attribute> ]

<pad attribute> ::=
      NO PAD
    | PAD SPACE

<translation definition> ::=
    CREATE TRANSLATION <translation name>
      FOR <source character set specification>
        TO <target character set specification>
      FROM <translation source>

<source character set specification> ::= <character set specification>

<target character set specification> ::= <character set specification>

<translation source> ::=
      <translation specification>

<translation specification> ::=
      <external translation>
    | IDENTITY
    | <schema translation name>

<external translation> ::=
    EXTERNAL <left paren> <quote> <external translation name> <quote> <right paren>

<external translation name> ::=
      <standard translation name>
    | <implementation-defined translation name>

<standard translation name> ::= <translation name>
```

```
<implementation-defined translation name> ::= <translation name>

<schema translation name> ::= <translation name>

<SQL schema manipulation statement> ::=
      <drop schema statement>
    | <alter table statement>
    | <drop table statement>
    | <drop view statement>
    | <revoke statement>
    | <alter domain statement>
    | <drop domain statement>
    | <drop character set statement>
    | <drop collation statement>
    | <drop translation statement>
    | <drop assertion statement>

<drop schema statement> ::=
    DROP SCHEMA <schema name> <drop behavior>

<drop behavior> ::= CASCADE | RESTRICT

<alter table statement> ::=
    ALTER TABLE <table name> <alter table action>

<alter table action> ::=
      <add column definition>
    | <alter column definition>
    | <drop column definition>
    | <add table constraint definition>
    | <drop table constraint definition>

<add column definition> ::=
    ADD [ COLUMN ] <column definition>

<alter column definition> ::=
    ALTER [ COLUMN ] <column name> <alter column action>

<alter column action> ::=
      <set column default clause>
    | <drop column default clause>

<set column default clause> ::=
    SET <default clause>

<drop column default clause> ::=
    DROP DEFAULT
```

```
<drop column definition> ::=
   DROP [ COLUMN ] <column name> <drop behavior>

<add table constraint definition> ::=
   ADD <table constraint definition>

<drop table constraint definition> ::=
   DROP CONSTRAINT <constraint name> <drop behavior>

<drop table statement> ::=
   DROP TABLE <table name> <drop behavior>

<drop view statement> ::=
   DROP VIEW <table name> <drop behavior>

<revoke statement> ::=
   REVOKE [ GRANT OPTION FOR ]
       <privileges>
       ON <object name>
     FROM <grantee> [ ( <comma> <grantee> )... ] <drop behavior>

<alter domain statement> ::=
   ALTER DOMAIN <domain name> <alter domain action>

<alter domain action> ::=
     <set domain default clause>
   | <drop domain default clause>
   | <add domain constraint definition>
   | <drop domain constraint definition>

<set domain default clause> ::= SET <default clause>

<drop domain default clause> ::= DROP DEFAULT

<add domain constraint definition> ::=
   ADD <domain constraint>

<drop domain constraint definition> ::=
   DROP CONSTRAINT <constraint name>

<drop domain statement> ::=
   DROP DOMAIN <domain name> <drop behavior>

<drop character set statement> ::=
   DROP CHARACTER SET <character set name>

<drop collation statement> ::=
```

```
    DROP COLLATION <collation name>

<drop translation statement> ::=
    DROP TRANSLATION <translation name>

<drop assertion statement> ::=
    DROP ASSERTION <constraint name>

<SQL data statement> ::=
      <open statement>
    | <fetch statement>
    | <close statement>
    | <select statement: single row>
    | <SQL data change statement>

<open statement> ::=
    OPEN <cursor name>

<fetch statement> ::=
    FETCH [ [ <fetch orientation> ] FROM ]
      <cursor name> INTO <fetch target list>

<fetch orientation> ::=
      NEXT
    | PRIOR
    | FIRST
    | LAST
    | ( ABSOLUTE | RELATIVE ) <simple value specification>

<simple value specification> ::=
      <parameter name>
    | <embedded variable name>
    | <literal>

<fetch target list> ::=
    <target specification> [ ( <comma> <target specification> )... ]

<target specification> ::=
      <parameter specification>
    | <variable specification>

<close statement> ::=
    CLOSE <cursor name>

<select statement: single row> ::=
    SELECT [ <set quantifier> ] <select list>
      INTO <select target list>
        <table expression>
```

```
<select target list> ::=
    <target specification> [ ( <comma> <target specification> )... ]

<SQL data change statement> ::=
      <delete statement: positioned>
    | <delete statement: searched>
    | <insert statement>
    | <update statement: positioned>
    | <update statement: searched>

<delete statement: positioned> ::=
    DELETE FROM <table name>
      WHERE CURRENT OF <cursor name>

<delete statement: searched> ::=
    DELETE FROM <table name>
      [ WHERE <search condition> ]

<insert statement> ::=
    INSERT INTO <table name>
      <insert columns and source>

<insert columns and source> ::=
      [ <left paren> <insert column list> <right paren> ]
            <query expression>
    | DEFAULT VALUES

<insert column list> ::= <column name list>

<update statement: positioned> ::=
    UPDATE <table name>
      SET <set clause list>
        WHERE CURRENT OF <cursor name>

<set clause list> ::=
    <set clause> [ ( <comma> <set clause> )... ]

<set clause> ::=
    <object column> <equals operator> <update source>

<object column> ::= <column name>

<update source> ::=
      <value expression>
    | <null specification>
    | DEFAULT
```

```
<update statement: searched> ::=
    UPDATE <table name>
      SET <set clause list>
      [ WHERE <search condition> ]

<SQL transaction statement> ::=
      <set transaction statement>
    | <set constraints mode statement>
    | <commit statement>
    | <rollback statement>

<set transaction statement> ::=
    SET TRANSACTION <transaction mode>
        [ ( <comma> <transaction mode> )... ]

<transaction mode> ::=
      <isolation level>
    | <transaction access mode>
    | <diagnostics size>

<isolation level> ::=
    ISOLATION LEVEL <level of isolation>

<level of isolation> ::=
      READ UNCOMMITTED
    | READ COMMITTED
    | REPEATABLE READ
    | SERIALIZABLE

<transaction access mode> ::=
      READ ONLY
    | READ WRITE

<diagnostics size> ::=
    DIAGNOSTICS SIZE <number of conditions>

<number of conditions> ::= <simple value specification>

<set constraints mode statement> ::=
    SET CONSTRAINTS <constraint name list>
        ( DEFERRED | IMMEDIATE )

<constraint name list> ::=
      ALL
    | <constraint name> [ ( <comma> <constraint name> )... ]

<commit statement> ::=
    COMMIT [ WORK ]
```

```
<rollback statement> ::=
    ROLLBACK [ WORK ]

<SQL connection statement> ::=
      <connect statement>
    | <set connection statement>
    | <disconnect statement>

<connect statement> ::=
    CONNECT TO <connection target>

<connection target> ::=
      <SQL-server name>
        [ AS <connection name> ]
          correspondence with Tony Gordon)
        [ USER <user name> ]
    | DEFAULT

<SQL-server name> ::= <simple value specification>

<connection name> ::= <simple value specification>

<user name> ::= <simple value specification>

<set connection statement> ::=
    SET CONNECTION <connection object>

<connection object> ::=
      DEFAULT
    | <connection name>

<disconnect statement> ::=
    DISCONNECT <disconnect object>

<disconnect object> ::=
      <connection object>
    | ALL
    | CURRENT

<SQL session statement> ::=
      <set catalog statement>
    | <set schema statement>
    | <set names statement>
    | <set session authorization identifier statement>
    | <set local time zone statement>

<set catalog statement> ::=
```

```
    SET CATALOG <value specification>

<value specification> ::=
      <literal>
    | <general value specification>

<set schema statement> ::=
    SET SCHEMA <value specification>

<set names statement> ::=
    SET NAMES <value specification>

<set session authorization identifier statement> ::=
    SET SESSION AUTHORIZATION
        <value specification>

<set local time zone statement> ::=
    SET TIME ZONE
        <set time zone value>

<set time zone value> ::=
      <interval value expression>
    | LOCAL

<SQL dynamic statement> ::=
      <system descriptor statement>
    | <prepare statement>
    | <deallocate prepared statement>
    | <describe statement>
    | <execute statement>
    | <execute immediate statement>
    | <SQL dynamic data statement>

<system descriptor statement> ::=
      <allocate descriptor statement>
    | <deallocate descriptor statement>
    | <set descriptor statement>
    | <get descriptor statement>

<allocate descriptor statement> ::=
    ALLOCATE DESCRIPTOR <descriptor name>
       [ WITH MAX <occurrences> ]

<descriptor name> ::=
    [ <scope option> ] <simple value specification>

<scope option> ::=
      GLOBAL
```

```
    | LOCAL

<occurrences> ::= <simple value specification>

<deallocate descriptor statement> ::=
    DEALLOCATE DESCRIPTOR <descriptor name>

<set descriptor statement> ::=
    SET DESCRIPTOR <descriptor name>
        <set descriptor information>

<set descriptor information> ::=
      <set count>
    | VALUE <item number>
        <set item information> [ ( <comma> <set item information> )... ]

<set count> ::=
    COUNT <equals operator> <simple value specification 1>

<simple value specification 1> ::= <simple value specification>

<item number> ::= <simple value specification>

<set item information> ::=
    <descriptor item name> <equals operator> <simple value specification 2>

<descriptor item name> ::=
      TYPE
    | LENGTH
    | OCTET_LENGTH
    | RETURNED_LENGTH
    | RETURNED_OCTET_LENGTH
    | PRECISION
    | SCALE
    | DATETIME_INTERVAL_CODE
    | DATETIME_INTERVAL_PRECISION
    | NULLABLE
    | INDICATOR
    | DATA
    | NAME
    | UNNAMED
    | COLLATION_CATALOG
    | COLLATION_SCHEMA
    | COLLATION_NAME
    | CHARACTER_SET_CATALOG
    | CHARACTER_SET_SCHEMA
    | CHARACTER_SET_NAME
```

```
<simple value specification 2> ::= <simple value specification>

<item number> ::= <simple value specification>

<get descriptor statement> ::=
    GET DESCRIPTOR <descriptor name> <get descriptor information>

<get descriptor information> ::=
      <get count>
    | VALUE <item number>
        <get item information> [ ( <comma> <get item information> )... ]

<get count> ::=
    <simple target specification 1> <equals operator>
         COUNT

<simple target specification 1> ::= <simple target specification>

<simple target specification> ::=
      <parameter name>
    | <embedded variable name>

<get item information> ::=
    <simple target specification 2> <equals operator> <descriptor item name>>

<simple target specification 2> ::= <simple target specification>

<prepare statement> ::=
    PREPARE <SQL statement name> FROM <SQL statement variable>

<SQL statement name> ::=
      <statement name>
    | <extended statement name>

<extended statement name> ::=
    [ <scope option> ] <simple value specification>

<SQL statement variable> ::= <simple value specification>

<deallocate prepared statement> ::=
    DEALLOCATE PREPARE <SQL statement name>

<describe statement> ::=
      <describe input statement>
    | <describe output statement>

<describe input statement> ::=
    DESCRIBE INPUT <SQL statement name> <using descriptor>
```

```
<using descriptor> ::=
    ( USING | INTO ) SQL DESCRIPTOR <descriptor name>

<describe output statement> ::=
    DESCRIBE [ OUTPUT ] <SQL statement name> <using descriptor>

<execute statement> ::=
    EXECUTE <SQL statement name>
      [ <result using clause> ]
      [ <parameter using clause> ]

<result using clause> ::= <using clause>

<using clause> ::=
      <using arguments>
    | <using descriptor>

<using arguments> ::=
    ( USING | INTO ) <argument> [ ( <comma> <argument> )... ]

<argument> ::= <target specification>

<parameter using clause> ::= <using clause>

<execute immediate statement> ::=
    EXECUTE IMMEDIATE <SQL statement variable>

<SQL dynamic data statement> ::=
      <allocate cursor statement>
    | <dynamic open statement>
    | <dynamic fetch statement>
    | <dynamic close statement>
    | <dynamic delete statement: positioned>
    | <dynamic update statement: positioned>

<allocate cursor statement> ::=
    ALLOCATE <extended cursor name> [ INSENSITIVE ]
        [ SCROLL ] CURSOR
      FOR <extended statement name>

<extended cursor name> ::=
    [ <scope option> ] <simple value specification>

<dynamic open statement> ::=
    OPEN <dynamic cursor name> [ <using clause> ]

<dynamic cursor name> ::=
```

```
      <cursor name>
    | <extended cursor name>

<dynamic fetch statement> ::=
    FETCH [ [ <fetch orientation> ] FROM ] <dynamic cursor name>
        <using clause>

<dynamic close statement> ::=
    CLOSE <dynamic cursor name>

<dynamic delete statement: positioned> ::=
    DELETE FROM <table name>
      WHERE CURRENT OF
          <dynamic cursor name>

<dynamic update statement: positioned> ::=
    UPDATE <table name>
      SET <set clause>
          [ ( <comma> <set clause> )... ]
        WHERE CURRENT OF
            <dynamic cursor name>

<SQL diagnostics statement> ::=
    <get diagnostics statement>

<get diagnostics statement> ::=
    GET DIAGNOSTICS <sql diagnostics information>

<sql diagnostics information> ::=
      <statement information>
    | <condition information>

<statement information> ::=
    <statement information item> [ ( <comma> <statement information item> )... ]

<statement information item> ::=
    <simple target specification> <equals operator> <statement information item name>

<statement information item name> ::=
      NUMBER
    | MORE
    | COMMAND_FUNCTION
    | DYNAMIC_FUNCTION
    | ROW_COUNT

<condition information> ::=
    EXCEPTION <condition number>
      <condition information item> [ ( <comma> <condition information item> )... ]
```

```
<condition number> ::= <simple value specification>

<condition information item> ::=
    <simple target specification> <equals operator> <condition information item name>

<condition information item name> ::=
      CONDITION_NUMBER
    | RETURNED_SQLSTATE
    | CLASS_ORIGIN
    | SUBCLASS_ORIGIN
    | SERVER_NAME
    | CONNECTION_NAME
    | CONSTRAINT_CATALOG
    | CONSTRAINT_SCHEMA
    | CONSTRAINT_NAME
    | CATALOG_NAME
    | SCHEMA_NAME
    | TABLE_NAME
    | COLUMN_NAME
    | CURSOR_NAME
    | MESSAGE_TEXT
    | MESSAGE_LENGTH
    | MESSAGE_OCTET_LENGTH

<embedded SQL host program> ::=
      <embedded SQL Ada program>
    | <embedded SQL C program>
    | <embedded SQL COBOL program>
    | <embedded SQL Fortran program>
    | <embedded SQL MUMPS program>
    | <embedded SQL Pascal program>
    | <embedded SQL PL/I program>

<embedded SQL Ada program> ::= !! <EMPHASIS>(See the Syntax Rules.)

<embedded SQL C program> ::=
      !! <EMPHASIS>(See the Syntax Rules.)

<embedded SQL COBOL program> ::= !! <EMPHASIS>(See the Syntax Rules.)

<embedded SQL Fortran program> ::=
    !! <EMPHASIS>(See the Syntax Rules.)

<embedded SQL MUMPS program> ::= !! <EMPHASIS>(See the Syntax Rules.)

<embedded SQL Pascal program> ::=
    !! <EMPHASIS>(See the Syntax Rules.)
```

```
<embedded SQL PL/I program> ::= !! <EMPHASIS>(See the Syntax Rules.)

<embedded SQL declare section> ::=
      <embedded SQL begin declare>
        [ <embedded character set declaration> ]
        [ <host variable definition>... ]
      <embedded SQL end declare>
    | <embedded SQL MUMPS declare>

<embedded SQL begin declare> ::=
    <SQL prefix> BEGIN DECLARE SECTION
        [ <SQL terminator> ]

<SQL prefix> ::=
      EXEC SQL
    | <ampersand>SQL<left paren>

<SQL terminator> ::=
      END-EXEC
    | <semicolon>
    | <right paren>

<embedded character set declaration> ::=
    SQL NAMES ARE <character set specification>

<host variable definition> ::=
      <Ada variable definition>
    | <C variable definition>
    | <COBOL variable definition>
    | <Fortran variable definition>
    | <MUMPS variable definition>
    | <Pascal variable definition>
    | <PL/I variable definition>

<Ada variable definition> ::=
    <Ada host identifier> [ ( <comma> <Ada host identifier> )... ] :
    <Ada type specification> [ <Ada initial value> ]

<Ada type specification> ::=
      <Ada qualified type specification>
    | <Ada unqualified type specification>

<Ada qualified type specification> ::=
      SQL_STANDARD.CHAR [ CHARACTER SET
         [ IS ] <character set specification> ]
          <left paren> 1 <double period> <length> <right paren>
    | SQL_STANDARD.BIT
```

```
            <left paren> 1 <double period> <length> <right paren>
    | SQL_STANDARD.SMALLINT
    | SQL_STANDARD.INT
    | SQL_STANDARD.REAL
    | SQL_STANDARD.DOUBLE_PRECISION
    | SQL_STANDARD.SQLCODE_TYPE
    | SQL_STANDARD.SQLSTATE_TYPE
    | SQL_STANDARD.INDICATOR_TYPE

<Ada unqualified type specification> ::=
      CHAR
          <left paren> 1 <double period> <length> <right paren>
    | BIT
          <left paren> 1 <double period> <length> <right paren>
    | SMALLINT
    | INT
    | REAL
    | DOUBLE_PRECISION
    | SQLCODE_TYPE
    | SQLSTATE_TYPE
    | INDICATOR_TYPE

<Ada initial value> ::=
    <Ada assignment operator> <character representation>...

<Ada assignment operator> ::= <colon><equals operator>

<C variable definition> ::=
      [ <C storage class> ]
      [ <C class modifier> ]
      <C variable specification>
    <semicolon>

<C storage class> ::=
      auto
    | extern
    | static

<C class modifier> ::= const | volatile

<C variable specification> ::=
      <C numeric variable>
    | <C character variable>
    | <C derived variable>

<C numeric variable> ::=
    ( long | short | float | double )
      <C host identifier> [ <C initial value> ]
```

```
            [ ( <comma> <C host identifier> [ <C initial value> ] )... ]

<C initial value> ::=
    <equals operator> <character representation>...

<C character variable> ::=
    char [ CHARACTER SET
             [ IS ] <character set specification> ]
      <C host identifier>
        <C array specification> [ <C initial value> ]
        [ ( <comma> <C host identifier>
          <C array specification>
                 [ <C initial value> ] )... ]

<C array specification> ::=
    <left bracket> <length> <right bracket>

<C derived variable> ::=
      <C VARCHAR variable>
    | <C bit variable>

<C VARCHAR variable> ::=
    VARCHAR [ CHARACTER SET [ IS ]
        <character set specification> ]
        <C host identifier>
            <C array specification> [ <C initial value> ]
          [ ( <comma> <C host identifier>
              <C array specification>
                      [ <C initial value> ] )... ]

<C bit variable> ::=
    BIT <C host identifier>
        <C array specification> [ <C initial value> ]
      [ ( <comma> <C host identifier>
        <C array specification>
                   [ <C initial value> ] )... ]

<COBOL variable definition> ::=
    (01|77) <COBOL host identifier> <COBOL type specification>
      [ <character representation>... ] <period>

<COBOL type specification> ::=
      <COBOL character type>
    | <COBOL bit type>
    | <COBOL numeric type>
    | <COBOL integer type>

<COBOL character type> ::=
```

```
    [ CHARACTER SET [ IS ]
          <character set specification> ]
    ( PIC | PICTURE ) [ IS ] ( X [ <left paren> <length> <right paren> ] )...

<COBOL bit type> ::=
    ( PIC | PICTURE ) [ IS ]
        ( B [ <left paren> <length> <right paren> ] )...

<COBOL numeric type> ::=
    ( PIC | PICTURE ) [ IS ]
      S <COBOL nines specification>
    [ USAGE [ IS ] ] DISPLAY SIGN LEADING SEPARATE

<COBOL nines specification> ::=
      <COBOL nines> [ V [ <COBOL nines> ] ]
    | V <COBOL nines>

<COBOL nines> ::= ( 9 [ <left paren> <length> <right paren> ] )...

<COBOL integer type> ::=
      <COBOL computational integer>
    | <COBOL binary integer>

<COBOL computational integer> ::=
    ( PIC | PICTURE ) [ IS ] S<COBOL nines>
      [ USAGE [ IS ] ] ( COMP | COMPUTATIONAL )

<COBOL binary integer> ::=
    ( PIC | PICTURE ) [ IS ] S<COBOL nines>
      [ USAGE [ IS ] ] BINARY

<Fortran variable definition> ::=
    <Fortran type specification>
    <Fortran host identifier>
        [ ( <comma> <Fortran host identifier> )... ]

<Fortran type specification> ::=
      CHARACTER [ <asterisk> <length> ]
          [ CHARACTER SET [ IS ]
                <character set specification> ]
    | BIT [ <asterisk> <length> ]
    | INTEGER
    | REAL
    | DOUBLE PRECISION

<MUMPS variable definition> ::=
    ( <MUMPS numeric variable> | <MUMPS character variable> )
        <semicolon>
```

```
<MUMPS numeric variable> ::=
    <MUMPS type specification>
      <MUMPS host identifier> [ ( <comma> <MUMPS host identifier> )... ]

<MUMPS type specification> ::=
      INT
    | DEC
          [ <left paren> <precision> [ <comma> <scale> ] <right paren> ]
    | REAL

<MUMPS character variable> ::=
    VARCHAR <MUMPS host identifier> <MUMPS length specification>
      [ ( <comma> <MUMPS host identifier> <MUMPS length specification> )... ]

<MUMPS length specification> ::=
    <left paren> <length> <right paren>

<Pascal variable definition> ::=
    <Pascal host identifier> [ ( <comma> <Pascal host identifier> )... ] <colon>
      <Pascal type specification> <semicolon>

<Pascal type specification> ::=
      PACKED ARRAY
          <left bracket> 1 <double period> <length> <right bracket>
        OF CHAR
          [ CHARACTER SET [ IS ]
                <character set specification> ]
    | PACKED ARRAY
          <left bracket> 1 <double period> <length> <right bracket>
        OF BIT
    | INTEGER
    | REAL
    | CHAR [ CHARACTER SET
                              [ IS ] <character set specification> ]
    | BIT

<PL/I variable definition> ::=
    (DCL | DECLARE)
        (   <PL/I host identifier>
          | <left paren> <PL/I host identifier>
                [ ( <comma> <PL/I host identifier> )... ] <right paren> )
    <PL/I type specification>
    [ <character representation>... ] <semicolon>

<PL/I type specification> ::=
      ( CHAR | CHARACTER ) [ VARYING ]
          <left paren><length><right paren>
```

```
          [ CHARACTER SET
                [ IS ] <character set specification> ]
    | BIT [ VARYING ] <left paren><length><right paren>
    | <PL/I type fixed decimal> <left paren> <precision>
          [ <comma> <scale> ] <right paren>
    | <PL/I type fixed binary> [ <left paren> <precision> <right paren> ]
    | <PL/I type float binary> <left paren> <precision> <right paren>

<PL/I type fixed decimal> ::=
      ( DEC | DECIMAL ) FIXED
    | FIXED ( DEC | DECIMAL )

<PL/I type fixed binary> ::=
      ( BIN | BINARY ) FIXED
    | FIXED ( BIN | BINARY )

<PL/I type float binary> ::=
      ( BIN | BINARY ) FLOAT
    | FLOAT ( BIN | BINARY )

<embedded SQL end declare> ::=
    <SQL prefix> END DECLARE SECTION
        [ <SQL terminator> ]

<embedded SQL MUMPS declare> ::=
    <SQL prefix>
      BEGIN DECLARE SECTION
        [ <embedded character set declaration> ]
        [ <host variable definition>... ]
      END DECLARE SECTION
    <SQL terminator>

<embedded SQL statement> ::=
    <SQL prefix>
      <statement or declaration>
    [ <SQL terminator> ]

<statement or declaration> ::=
      <declare cursor>
    | <dynamic declare cursor>
    | <temporary table declaration>
    | <embedded exception declaration>
    | <SQL procedure statement>

<embedded exception declaration> ::=
    WHENEVER <condition> <condition action>

<condition> ::=
```

```
    SQLERROR | NOT FOUND

<condition action> ::=
    CONTINUE | <go to>

<go to> ::=
    ( GOTO | GO TO ) <goto target>

<goto target> ::=
      <host label identifier>
    | <unsigned integer>
    | <host PL/I label variable>

<host label identifier> ::= !!<EMPHASIS>(See the Syntax Rules.)

<host PL/I label variable> ::= !!<EMPHASIS>(See the Syntax Rules.)

<preparable statement> ::=
      <preparable SQL data statement>
    | <preparable SQL schema statement>
    | <preparable SQL transaction statement>
    | <preparable SQL session statement>
    | <preparable implementation-defined statement>

<preparable SQL data statement> ::=
      <delete statement: searched>
    | <dynamic single row select statement>
    | <insert statement>
    | <dynamic select statement>
    | <update statement: searched>
    | <preparable dynamic delete statement: positioned>
    | <preparable dynamic update statement: positioned>

<dynamic single row select statement> ::= <query specification>

<dynamic select statement> ::= <cursor specification>

<preparable dynamic delete statement: positioned> ::=
   DELETE [ FROM <table name> ]
      WHERE CURRENT OF <cursor name>

<preparable dynamic update statement: positioned> ::=
   UPDATE [ <table name> ]
      SET <set clause list>
      WHERE CURRENT OF <cursor name>

<preparable SQL schema statement> ::=
      <SQL schema statement>
```

```
<preparable SQL transaction statement> ::=
      <SQL transaction statement>

<preparable SQL session statement> ::=
      <SQL session statement>

<preparable implementation-defined statement> ::=
    !! <EMPHASIS>(See the Syntax Rules.)

<direct SQL statement> ::=
    <directly executable statement> <semicolon>

<directly executable statement> ::=
      <direct SQL data statement>
    | <SQL schema statement>
    | <SQL transaction statement>
    | <SQL connection statement>
    | <SQL session statement>
    | <direct implementation-defined statement>

<direct SQL data statement> ::=
      <delete statement: searched>
    | <direct select statement: multiple rows>
    | <insert statement>
    | <update statement: searched>
    | <temporary table declaration>

<direct select statement: multiple rows> ::=
    <query expression> [ <order by clause> ]

<direct implementation-defined statement> ::=
    !!<EMPHASIS>(See the Syntax Rules)

<SQL object identifier> ::=
    <SQL provenance> <SQL variant>

<SQL provenance> ::= <arc1> <arc2> <arc3>

<arc1> ::= iso | 1 | iso <left paren> 1 <right paren>

<arc2> ::= standard | 0 | standard <left paren> 0 <right paren>

<arc3> ::= 9075

<SQL variant> ::= <SQL edition> <SQL conformance>

<SQL edition> ::= <1987> | <1989> | <1992>
```

```
<1987> ::= 0 | edition1987 <left paren> 0 <right paren>

<1989> ::= <1989 base> <1989 package>

<1989 base> ::= 1 | edition1989 <left paren> 1 <right paren>

<1989 package> ::= <integrity no> | <integrity yes>

<integrity no> ::= 0 | IntegrityNo <left paren> 0 <right paren>

<integrity yes> ::= 1 | IntegrityYes <left paren> 1 <right paren>

<1992> ::= 2 | edition1992 <left paren> 2 <right paren>

<SQL conformance> ::= <low> | <intermediate> | <high>

<low> ::= 0 | Low <left paren> 0 <right paren>

<intermediate> ::= 1 | Intermediate <left paren> 1 <right paren>

<high> ::= 2 | High <left paren> 2 <right paren>
```

38.3 Syntaxe de l'ANSI/ISO SQL 1998

```
Specification de la syntaxe SQL ANSI/ISO SQL 1998 egalement appele SQL-3
-
Voici une copie de SQL3 bnf.  SQL3 est un sur-ensemble de SQL-92
[qui est un sur-ensemble de SQL-89 niveau 2].  SQL3 n'est pas encore
un standard, alors que SQL-92 en est un.
ENTRY LEVEL SQL-92 represente l'etat de l'implantation pour la plupart
des vendeurs.  Il n'y a que peu de differences entre SQL-92 ENTRY LEVEL
et SQL-89 Niveau II, mais deux d'entre elles sont tres importantes:

   - Les identificateurs delimites

   - Le traitement de l'option WITH CHECK sur les vues CASCADE par defaut.
     Dans SQL-89 la valeur par defaut etait [en realite] LOCAL.

pour le langage, et effectuees aux environs du  1-SEP-1993 15:13:55.88.
La version BNF specifique incluse ici correspond a :
  ANSI et SQL3-seulement.


<SQL terminal character> ::=
    <SQL language character>

<SQL language character> ::=
```

```
      <simple Latin letter>
    | <digit>
    | <SQL special character>

<simple Latin letter> ::=
      <simple Latin upper case letter>
    | <simple Latin lower case letter>

<simple Latin upper case letter> ::=
          A | B | C | D | E | F | G | H | I | J | K | L | M | N | O
    | P | Q | R | S | T | U | V | W | X | Y | Z

<simple Latin lower case letter> ::=
          a | b | c | d | e | f | g | h | i | j | k | l | m | n | o
    | p | q | r | s | t | u | v | w | x | y | z

<digit> ::=
    0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<SQL special character> ::=
      <space>
    | <double quote>
    | <percent>
    | <ampersand>
    | <quote>
    | <left paren>
    | <right paren>
    | <asterisk>
    | <plus sign>
    | <comma>
    | <minus sign>
    | <period>
    | <solidus>
    | <colon>
    | <semicolon>
    | <less than operator>
    | <equals operator>
    | <greater than operator>
    | <question mark>
    | <left bracket>
    | <right bracket>
    | <circumflex>
    | <underscore>
    | <vertical bar>

<space> ::= !! <EMPHASIS>(space character in character set in use)

<double quote> ::= "
```

```
<percent> ::= %

<ampersand> ::= &

<quote> ::= '

<left paren> ::= (

<right paren> ::= )

<asterisk> ::= *

<plus sign> ::= +

<comma> ::= ,

<minus sign> ::= -

<period> ::= .

<solidus> ::= /

<colon> ::= :

<semicolon> ::= ;

<less than operator> ::= <

<equals operator> ::= =

<greater than operator> ::= >

<question mark> ::= ?

<left bracket> ::= [

<right bracket> ::= ]

<circumflex> ::= ^

<underscore> ::= _

<vertical bar> ::= |

<separator> ::= ( <comment> | <space> | <newline> )...

<comment> ::=
```

```
      <simple comment>
    | <bracketed comment>

<simple comment> ::=
    <simple comment introducer> [ <comment character>... ] <newline>

<simple comment introducer> ::= <minus sign><minus sign>[<minus sign>...]

<comment character> ::=
      <nonquote character>
    | <quote>

<nonquote character> ::= !! <EMPHASIS>(See the Syntax Rules.)

<newline> ::= !! <EMPHASIS>(implementation-defined end-of-line indicator)

<bracketed comment> ::=            !! (<EMPHASIS>(See the Syntax Rules))
    <bracketed comment introducer>
      <bracketed comment contents>
    <bracketed comment terminator>

<bracketed comment introducer> ::= <solidus><asterisk>

<bracketed comment contents> ::=
    [ ( <comment character> | <separator> )... ]

<bracketed comment terminator> ::= <asterisk><solidus>

<token> ::=
      <nondelimiter token>
    | <delimiter token>

<nondelimiter token> ::=
      <regular identifier>
    | <key word>
    | <unsigned numeric literal>
    | <national character string literal>
    | <bit string literal>
    | <hex string literal>
    | <user-defined operator symbol>

<regular identifier> ::= <identifier body>

<identifier body> ::=
    <identifier start> [ ( <underscore> | <identifier part> )... ]

<identifier start> ::= !! <EMPHASIS>(See the Syntax Rules)
```

```
<identifier part> ::=
      <identifier start>
    | <digit>

<key word> ::=
      <reserved word>
    | <non-reserved word>

<reserved word> ::=
      ABSOLUTE | ACTION | ACTOR | ADD | AFTER | ALIAS
    | ALL | ALLOCATE | ALTER
    | AND | ANY | ARE
    | AS | ASC | ASSERTION
    | ASYNC | AT
    | ATTRIBUTES
    | AUTHORIZATION | AVG
    | BEFORE | BEGIN | BETWEEN | BIT | BIT_LENGTH
    | BOOLEAN | BOTH | BREADTH | BY
    | CASCADE | CASCADED | CASE | CAST
    | CATALOG
    | CHAR | CHARACTER
    | CHAR_LENGTH | CHARACTER_LENGTH | CHECK | CLASS | CLOSE | COALESCE
    | COLLATE
    | COLLATION | COLUMN | COMMIT | COMPLETION
    | CONNECT | CONNECTION | CONSTRAINT
    | CONSTRAINTS | CONSTRUCTOR | CONTINUE | CONVERT | CORRESPONDING
    | COUNT
    | CREATE | CROSS | CURRENT | CURRENT_DATE
    | CURRENT_PATH
    | CURRENT_TIME
    | CURRENT_TIMESTAMP | CURRENT_USER | CURSOR | CYCLE
    | DATA | DATE | DAY | DEALLOCATE
    | DEC | DECIMAL | DECLARE | DEFAULT
    | DEFERRABLE | DEFERRED | DELETE | DEPTH
    | DESC | DESCRIBE
    | DESCRIPTOR
    | DESIGNATOR
    | DESTROY | DESTRUCTOR | DICTIONARY
    | DIAGNOSTICS | DISCONNECT | DISTINCT | DOMAIN
    | DOUBLE | DROP
    | EACH
    | ELEMENT
    | ELSE
    | END | END-EXEC | EQUALS
    | ESCAPE | EXCEPT
    | EXEC | EXECUTE | EXISTS | EXTERNAL | EXTRACT
    | FACTOR
    | FALSE | FETCH | FIRST | FLOAT | FOR | FOREIGN
```

```
| FOUND | FROM | FULL
| FUNCTION
| GENERAL | GET | GLOBAL | GO | GOTO
| GRANT | GROUP
| HAVING | HOUR
| IDENTITY | IGNORE | IMMEDIATE
| IN | INDICATOR
| INITIALLY | INNER | INOUT
| INPUT | INSENSITIVE | INSERT
| INSTEAD
| INT | INTEGER | INTERSECT | INTERVAL
| INTO | IS | ISOLATION
| JOIN
| KEY
| LANGUAGE | LAST | LEADING | LEFT
| LESS | LEVEL | LIKE | LIMIT
| LIST
| LOCAL | LOWER
| MATCH | MAX | MIN | MINUTE | MODIFY | MODULE
| MONTH
| MOVE | MULTISET
| NAMES | NATIONAL | NATURAL | NCHAR | NEW
| NEW_TABLE
| NEXT | NO
| NONE | NOT| NULL | NULLIF | NUMERIC
| OCTET_LENGTH | OF | OFF
| OID | OLD
| OLD_TABLE
| ON | ONLY | OPEN | OPERATION
| OPERATOR
| OPERATORS
| OPTION | OR | ORDER
| OUT | OUTER | OUTPUT | OVERLAPS
| PAD | PARAMETERS | PARTIAL
| PATH
| PENDANT | POSITION | POSTFIX | PRECISION | PREFIX
| PREORDER | PREPARE | PRESERVE | PRIMARY | PRIOR | PRIVATE
| PRIVILEGES | PROCEDURE | PROTECTED | PUBLIC
| READ | REAL | RECURSIVE
| REFERENCES | REFERENCING | RELATIVE | REPRESENTATION
| RESTRICT | REVOKE | RIGHT
| ROLE | ROLLBACK
| ROUTINE
| ROW | ROWS
| SAVEPOINT | SCHEMA | SCROLL | SEARCH | SECOND | SECTION
| SELECT
| SENSITIVE
| SEQUENCE
```

```
    | SESSION | SESSION_USER | SET
    | SIMILAR | SIZE | SMALLINT | SOME | SPACE | SPECIFIC
    | SQL | SQLCODE
    | SQLERROR | SQLEXCEPTION | SQLSTATE | SQLWARNING
    | START | STATE
    | STRUCTURE
    | SUBSTRING | SUM
    | SYMBOL
    | SYSTEM_USER
    | TABLE | TEMPLATE | TEMPORARY
    | TERM
    | TEST | THAN
    | THEN | THERE | TIME | TIMESTAMP
    | TIMEZONE_HOUR
    | TIMEZONE_MINUTE | TO | TRAILING | TRANSACTION
    | TRANSLATE | TRANSLATION
    | TRIGGER | TRIM | TRUE | TYPE
    | UNDER | UNION | UNIQUE | UNKNOWN
    | UPDATE | UPPER | USAGE | USER | USING
    | VALUE | VALUES | VARCHAR | VARIABLE | VARIANT
    | VARYING | VIEW
    | VIRTUAL | VISIBLE
    | WAIT | WHEN | WHENEVER | WHERE
    | WITH | WITHOUT
    | WORK | WRITE
    | YEAR
    | ZONE

<non-reserved word> ::=
      ADA
    | C | CATALOG_NAME
    | CHAIN
    | CHARACTER_SET_CATALOG | CHARACTER_SET_NAME
    | CHARACTER_SET_SCHEMA | CLASS_ORIGIN | COBOL | COLLATION_CATALOG
    | COLLATION_NAME
    | COLLATION_SCHEMA | COLUMN_NAME | COMMAND_FUNCTION
    | COMMITTED | CONDITION_NUMBER | CONNECTION_NAME | CONSTRAINT_CATALOG
    | CONSTRAINT_NAME | CONSTRAINT_SCHEMA | CURSOR_NAME
    | DATA | DATETIME_INTERVAL_CODE | DATETIME_INTERVAL_PRECISION
    | DYNAMIC_FUNCTION
    | FORTRAN
    | HOLD
    | KEY_MEMBER | KEY_TYPE
    | LENGTH
    | MESSAGE_LENGTH | MESSAGE_OCTET_LENGTH | MESSAGE_TEXT | MORE | MUMPS
    | NAME | NULLABLE | NUMBER
    | PASCAL | PLI
    | REPEATABLE | RETURNED_LENGTH | RETURNED_OCTET_LENGTH | RETURNED_SQLSTATE
```

```
    | ROUTINE_CATALOG | ROUTINE_NAME | ROUTINE_SCHEMA | ROW_COUNT
    | SCALE | SCHEMA_NAME | SERIALIZABLE | SERVER_NAME | SPECIFIC_NAME
    | SUBCLASS_ORIGIN
    | TABLE_NAME | TYPE
    | UNCOMMITTED | UNNAMED

<unsigned numeric literal> ::=
      <exact numeric literal>
    | <approximate numeric literal>

<exact numeric literal> ::=
      <unsigned integer> [ <period> [ <unsigned integer> ] ]
    | <period> <unsigned integer>

<unsigned integer> ::= <digit>...

<approximate numeric literal> ::= <mantissa> E <exponent>

<mantissa> ::= <exact numeric literal>

<exponent> ::= <signed integer>

<signed integer> ::= [ <sign> ] <unsigned integer>

<sign> ::= <plus sign> | <minus sign>

<national character string literal> ::=
    N <quote> [ <character representation>... ] <quote>
      [ ( <separator> <quote> [ <character representation>... ] <quote> )... ]

<character representation> ::=
      <nonquote character>
    | <quote symbol>

<quote symbol> ::= <quote><quote>

<bit string literal> ::=
    B <quote> [ <bit>... ] <quote>
      [ ( <separator> <quote> [ <bit>... ] <quote> )... ]

<bit> ::= 0 | 1

<hex string literal> ::=
    X <quote> [ <hexit>... ] <quote>
      [ ( <separator> <quote> [ <hexit>... ] <quote> )... ]

<hexit> ::= <digit> | A | B | C | D | E | F | a | b | c | d | e | f
```

```
<user-defined operator symbol> ::= !! <EMPHASIS>(See the Syntax Rules)

<delimiter token> ::=
      <character string literal>
    | <date string>
    | <time string>
    | <timestamp string>
    | <interval string>
    | <delimited identifier>
    | <SQL special character>
    | <not equals operator>
    | <greater than or equals operator>
    | <less than or equals operator>
    | <concatenation operator>
    | <double period>
    | <double colon>
    | <assignment operator>
    | <left bracket>
    | <right bracket>

<character string literal> ::=
    [ <introducer><character set specification> ]
    <quote> [ <character representation>... ] <quote>
      [ ( <separator> <quote> [ <character representation>... ] <quote> )... ]

<introducer> ::= <underscore>

<character set specification> ::=
      <standard character repertoire name>
    | <implementation-defined character repertoire name>
    | <user-defined character repertoire name>
    | <standard universal character form-of-use name>
    | <implementation-defined universal character form-of-use name>

<standard character repertoire name> ::= <character set name>

<character set name> ::= [ <schema name> <period> ]
      <SQL language identifier>

<schema name> ::=
    [ <catalog name> <period> ] <unqualified schema name>

<catalog name> ::= <identifier>

<identifier> ::=
    [ <introducer><character set specification> ] <actual identifier>

<actual identifier> ::=
```

```
      <regular identifier>
    | <delimited identifier>

<delimited identifier> ::=
    <double quote> <delimited identifier body> <double quote>

<delimited identifier body> ::= <delimited identifier part>...

<delimited identifier part> ::=
      <nondoublequote character>
    | <doublequote symbol>

<nondoublequote character> ::= !! <EMPHASIS>(See the Syntax Rules)

<doublequote symbol> ::= <double quote><double quote>

<unqualified schema name> ::= <identifier>

<SQL language identifier> ::=
    <SQL language identifier start>
       [ ( <underscore> | <SQL language identifier part> )... ]

<SQL language identifier start> ::= <simple Latin letter>

<SQL language identifier part> ::=
      <simple Latin letter>
    | <digit>

<implementation-defined character repertoire name> ::=
    <character set name>

<user-defined character repertoire name> ::= <character set name>

<standard universal character form-of-use name> ::=
    <character set name>

<implementation-defined universal character form-of-use name> ::=
    <character set name>

<date string> ::=
    <quote> <date value> <quote>

<date value> ::=
    <years value> <minus sign> <months value>
        <minus sign> <days value>

<years value> ::= <datetime value>
```

```
<datetime value> ::= <unsigned integer>

<months value> ::= <datetime value>

<days value> ::= <datetime value>

<time string> ::=
    <quote> <time value> [ <time zone interval> ] <quote>

<time value> ::=
    <hours value> <colon> <minutes value> <colon> <seconds value>

<hours value> ::= <datetime value>

<minutes value> ::= <datetime value>

<seconds value> ::=
      <seconds integer value> [ <period> [ <seconds fraction> ] ]

<seconds integer value> ::= <unsigned integer>

<seconds fraction> ::= <unsigned integer>

<time zone interval> ::=
    <sign> <hours value> <colon> <minutes value>

<timestamp string> ::=
    <quote> <date value> <space> <time value>
        [ <time zone interval> ] <quote>

<interval string> ::=
    <quote> ( <year-month literal> | <day-time literal> ) <quote>

<year-month literal> ::=
      <years value>
    | [ <years value> <minus sign> ] <months value>

<day-time literal> ::=
      <day-time interval>
    | <time interval>

<day-time interval> ::=
    <days value>
      [ <space> <hours value> [ <colon> <minutes value>
        [ <colon> <seconds value> ] ] ]

<time interval> ::=
      <hours value> [ <colon> <minutes value> [ <colon> <seconds value> ] ]
```

```
    | <minutes value> [ <colon> <seconds value> ]
    | <seconds value>

<not equals operator> ::= <>

<greater than or equals operator> ::= >=

<less than or equals operator> ::= <=

<concatenation operator> ::= ||

<double period> ::= ..

<double colon> ::= ::

<assignment operator> ::= :=

<SQL-client module definition> ::= <module>

<module> ::=
    <module name clause>
      <module remainder>
    [ END MODULE ]

<module name clause> ::=
    MODULE [ <module name> ]

<module name> ::=
      <SQL-server module name>
    | <SQL-client module name>

<SQL-server module name> ::= <qualified identifier>

<qualified identifier> ::= <identifier>

<SQL-client module name> ::= <identifier>

<module remainder> ::=
    [ <module character set specification> ]
    <language clause>
    <module authorization clause>
      [ <module path specification> ]
    <module contents>...

<module character set specification> ::=
    NAMES ARE <character set specification>

<language clause> ::=
```

```
    LANGUAGE <language name>

<language name> ::=
    ADA | C | COBOL | FORTRAN | MUMPS | PASCAL | PLI | SQL

<module authorization clause> ::=
      SCHEMA <schema name>
    | AUTHORIZATION <module authorization identifier>
    | SCHEMA <schema name> AUTHORIZATION <module authorization identifier>

<module authorization identifier> ::=
    <authorization identifier>

<authorization identifier> ::= <identifier>

<module path specification> ::=
    PATH <schema name list>

<schema name list> ::=
    <schema name> [ ( <comma> <schema name> )... ]

<module contents> ::=
      <global declaration> [ <semicolon> ]
    | <routine>

<global declaration> ::=
      <declare cursor>
    | <temporary abstract data type declaration>
    | <temporary table declaration>
    | <temporary view declaration>

<declare cursor> ::=
    DECLARE <cursor name> [ <cursor sensitivity> ]
      [ SCROLL ] CURSOR
      [ WITH HOLD ]
      FOR <cursor specification>
<cursor sensitivity> ::=
      SENSITIVE
    | INSENSITIVE

<cursor name> ::= <local qualified name>

<local qualified name> ::=
    [ <local qualifier> <period> ] <qualified identifier>

<local qualifier> ::= MODULE

<cursor specification> ::=
```

```
    <query expression> [ <order by clause> ]
      [ <updatability clause> ]

<query expression> ::=
    <possibly updatable query expression> [ <trigger definition>... ]

<possibly updatable query expression> ::=
      <non-join query expression>
    | <joined table>

<non-join query expression> ::=
      <non-join query term>
    | <query expression> UNION  [ ALL ]
          [ <corresponding spec> ] <query term>
    | <query expression> EXCEPT [ ALL ]
          [ <corresponding spec> ] <query term>

<non-join query term> ::=
      <non-join query primary>
    | <query term> INTERSECT [ ALL ]
          [ <corresponding spec> ] <query primary>
    | <recursive union>

<non-join query primary> ::=
      <simple table>
    | <left paren> <non-join query expression> <right paren>

<simple table> ::=
      <query specification>
    | <table value designator>
    | <explicit table>
    | <collection expression>

<query specification> ::=
    SELECT [ <set quantifier> ] <select list>
      <table expression>

<set quantifier> ::= DISTINCT | ALL

<select list> ::=
      <asterisk>
    | <select sublist> [ ( <comma> <select sublist> )... ]

<select sublist> ::=
      <derived column>
    | <qualifier> <period> <asterisk>

<derived column> ::=
```

```
    <value expression> [ <as clause> ]

<value expression> ::=
      <numeric value expression>
    | <string value expression>
    | <datetime value expression>
    | <interval value expression>
    | <enumerated value expression>
    | <boolean value expression>
    | <attributes function>
    | <abstract data type value expression>
    | <table value expression>
    | <collection value expression>

<numeric value expression> ::=
      <term>
    | <numeric value expression> <plus sign> <term>
    | <numeric value expression> <minus sign> <term>

<term> ::=
      <factor>
    | <term> <asterisk> <factor>
    | <term> <solidus> <factor>

<factor> ::=
    [ <sign> ] <numeric primary>

<numeric primary> ::=
      <value expression primary>
    | <numeric value function>

<value expression primary> ::=
      <unsigned value specification>
    | <column reference>
    | <row reference>
    | <set function specification>
    | <table subquery>
    | <case expression>
    | <left paren> <value expression> <right paren>
    | <cast specification>

<unsigned value specification> ::=
      <unsigned literal>
    | <general value specification>

<unsigned literal> ::=
      <unsigned numeric literal>
    | <general literal>
```

```
<general literal> ::=
      <character string literal>
    | <national character string literal>
    | <bit string literal>
    | <hex string literal>
    | <datetime literal>
    | <interval literal>
    | <enumeration literal>
    | <boolean literal>
    | <oid literal>

<datetime literal> ::=
      <date literal>
    | <time literal>
    | <timestamp literal>

<date literal> ::=
    DATE <date string>

<time literal> ::=
    TIME <time string>

<timestamp literal> ::=
    TIMESTAMP <timestamp string>

<interval literal> ::=
    INTERVAL [ <sign> ] <interval string> <interval qualifier>

<interval qualifier> ::=
      <start field> TO <end field>
    | <single datetime field>

<start field> ::=
    <non-second datetime field>
        [ <left paren> <interval leading field precision> <right paren> ]

<non-second datetime field> ::= YEAR | MONTH | DAY | HOUR | MINUTE

<interval leading field precision> ::= <unsigned integer>

<end field> ::=
      <non-second datetime field>
    | SECOND
        [ <left paren> <interval fractional seconds precision> <right paren> ]

<interval fractional seconds precision> ::= <unsigned integer>
```

```
<single datetime field> ::=
      <non-second datetime field>
          [ <left paren> <interval leading field precision> <right paren> ]
    | SECOND [ <left paren> <interval leading field precision>
          [ <comma> <interval fractional seconds precision> ] <right paren> ]

<enumeration literal> ::=
    <domain name> <double colon> <enumeration name>

<domain name> ::= <schema qualified name>

<schema qualified name> ::=
    [ <schema name> <period> ] <qualified identifier>

<enumeration name> ::= <identifier>

<boolean literal> ::=
      TRUE
    | FALSE

<oid literal> ::= OID <oid string>

<oid string> ::= <quote> <oid value> <quote>

<oid value> ::= <character representation>...

<general value specification> ::=
      <item reference>
    | USER
    | CURRENT_USER
    | SESSION_USER
    | SYSTEM_USER
    | CURRENT_PATH
    | VALUE
    | <function invocation>
    | <attribute reference>
    | <template parameter name>

<item reference> ::=
    <parameter name> [ <indicator parameter> ]

<parameter name> ::= <colon> <identifier>

<indicator parameter> ::= [ INDICATOR ] <parameter name>

<function invocation> ::= <routine invocation>

<routine invocation> ::=
```

```
    <routine name> <argument list>

<routine name> ::= <local or schema qualified name>

<local or schema qualified name> ::=
    [ <local or schema qualifier> <period> ] <qualified identifier>

<local or schema qualifier> ::=
      <schema name>
    | MODULE

<argument list> ::=
      <left paren> <positional arguments> <comma> <keyword arguments> <right paren>
    | <left paren> <positional arguments> <right paren>
    | <left paren> <keyword arguments> <right paren>
    | <left paren> <right paren>

<positional arguments> ::=
    <argument> [ ( <comma> <argument> )... ]

<argument> ::=
      <value expression>
    | <generalized expression>

<generalized expression> ::=
    <value expression> AS <abstract data type name>

<abstract data type name> ::= <local or schema qualified name>
<attribute name> ::=
      <identifier>
    | OID

<keyword arguments> ::=
    <keyword argument> [ ( <comma> <keyword argument> )... ]

<keyword argument> ::=
    <parameter name> <keyword parameter tag> <argument>

<keyword parameter tag> ::= =>

<attribute reference> ::=
    <value specification> <period> <attribute name>

<value specification> ::=
      <literal>
    | <general value specification>

<literal> ::=
```

```
      <signed numeric literal>
    | <general literal>

<signed numeric literal> ::=
    [ <sign> ] <unsigned numeric literal>

<template parameter name> ::= <colon> <identifier>

<column reference> ::= [ <qualifier> <period> ] <column name>

<qualifier> ::=
      <table name>
    | <correlation name>

<table name> ::=
      <local or schema qualified name>

<correlation name> ::= <identifier>

<column name> ::=
      <identifier>
    | OID

<row reference> ::= ROW <qualifier>

<set function specification> ::=
      COUNT <left paren> <asterisk> <right paren>
    | <general set function>

<general set function> ::=
      <set function type>
          <left paren> [ <set quantifier> ] <value expression> <right paren> ]

<set function type> ::=
    AVG | MAX | MIN | SUM | COUNT

<table subquery> ::= <subquery>

<subquery> ::= <left paren> <query expression> <right paren>

<case expression> ::=
      <case abbreviation>
    | <case specification>

<case abbreviation> ::=
      NULLIF <left paren> <value expression> <comma>
            <value expression> <right paren>
    | COALESCE <left paren> <value expression>
```

```
            ( <comma> <value expression> )... <right paren>

<case specification> ::=
      <simple case>
    | <searched case>

<simple case> ::=
    CASE <case operand>
      <simple when clause>...
      [ <else clause> ]
    END

<case operand> ::= <value expression>

<simple when clause> ::= WHEN <when operand> THEN <result>

<when operand> ::= <value expression>

<result> ::= <result expression> | NULL

<result expression> ::= <value expression>

<else clause> ::= ELSE <result>

<searched case> ::=
    CASE
      <searched when clause>...
      [ <else clause> ]
    END

<searched when clause> ::= WHEN <search condition> THEN <result>

<search condition> ::=
    <boolean value expression>

<boolean value expression> ::=
      <boolean term>
    | <boolean value expression> OR <boolean term>

<boolean term> ::=
      <boolean factor>
    | <boolean term> AND <boolean factor>

<boolean factor> ::=
    [ NOT ] <boolean primary>

<boolean primary> ::=
      <predicate>
```

```
    | <value expression primary>

<predicate> ::=
      <comparison predicate>
    | <between predicate>
    | <in predicate>
    | <like predicate>
    | <null predicate>
    | <quantified comparison predicate>
    | <exists predicate>
    | <unique predicate>
    | <match predicate>
    | <overlaps predicate>
    | <similar predicate>
    | <quantified predicate>
    | <there is predicate>
    | <distinct predicate>
    | <boolean predicate>

<comparison predicate> ::=
    <row value designator> <comp op> <row value designator>

<row value designator> ::=
       <row value designator element>
    | <left paren> <row value designator list> <right paren>
    | <row subquery>

<row value designator element> ::=
      <value expression>
    | <null specification>
    | <default specification>

<null specification> ::=
    NULL [ <left paren> <null state> <right paren> ]

<null state> ::= <identifier>

<default specification> ::=
    DEFAULT

<row value designator list> ::=
    <row value designator element>
        [ ( <comma> <row value designator element> )... ]

<row subquery> ::= <subquery>

<comp op> ::=
      <equals operator>
```

```
    | <not equals operator>
    | <less than operator>
    | <greater than operator>
    | <less than or equals operator>
    | <greater than or equals operator>

<between predicate> ::=
    <row value designator> [ NOT ] BETWEEN
      <row value designator> AND <row value designator>

<in predicate> ::=
    <row value designator>
      [ NOT ] IN <in predicate value>

<in predicate value> ::=
      <table subquery>
    | <left paren> <in value list> <right paren>

<in value list> ::=
    <value expression> ( <comma> <value expression> )...

<like predicate> ::=
    <match value> [ NOT ] LIKE <pattern>
      [ ESCAPE <escape character> ]

<match value> ::= <character value expression>

<character value expression> ::=
      <concatenation>
    | <character factor>

<concatenation> ::=
    <character value expression> <concatenation operator>
        <character factor>

<character factor> ::=
    <character primary> [ <collate clause> ]

<character primary> ::=
      <value expression primary>
    | <string value function>

<string value function> ::=
      <character value function>
    | <bit value function>

<character value function> ::=
      <character substring function>
```

```
    | <regular expression substring function>
    | <fold>
    | <form-of-use conversion>
    | <character translation>
    | <trim function>

<character substring function> ::=
    SUBSTRING <left paren> <character value expression> FROM <start position>
                [ FOR <string length> ] <right paren>

<start position> ::= <numeric value expression>

<string length> ::= <numeric value expression>

<regular expression substring function> ::=
    SUBSTRING <left paren> <character value expression> FROM
                <character value expression> FOR
                      <escape character> <right paren>

<escape character> ::= <character value expression>

<fold> ::= ( UPPER | LOWER )
     <left paren> <character value expression> <right paren>

<form-of-use conversion> ::=
    CONVERT <left paren> <character value expression>
        USING <form-of-use conversion name> <right paren>

<form-of-use conversion name> ::= <schema qualified name>

<character translation> ::=
    TRANSLATE <left paren> <character value expression>
        USING <translation name> <right paren>

<translation name> ::= <schema qualified name>

<trim function> ::=
    TRIM <left paren> <trim operands> <right paren>

<trim operands> ::=
    [ [ <trim specification> ] [ <trim character> ] FROM ] <trim source>

<trim specification> ::=
      LEADING
    | TRAILING
    | BOTH

<trim character> ::= <character value expression>
```

```
<trim source> ::= <character value expression>

<bit value function> ::=
    <bit substring function>

<bit substring function> ::=
    SUBSTRING <left paren> <bit value expression> FROM <start position>
        [ FOR <string length> ] <right paren>

<bit value expression> ::=
      <bit concatenation>
    | <bit factor>

<bit concatenation> ::=
    <bit value expression> <concatenation operator> <bit factor>

<bit factor> ::= <bit primary>

<bit primary> ::=
      <value expression primary>
    | <string value function>

<collate clause> ::= COLLATE <collation name>

<collation name> ::= <schema qualified name>

<pattern> ::= <character value expression>

<null predicate> ::= <row value designator>
    IS [ NOT ] NULL
        [ <left paren> <null values specification> <right paren> ]

<null values specification> ::= <asterisk> | <null state>

<quantified comparison predicate> ::=
    <row value designator> <comp op> <quantifier> <table subquery>

<quantifier> ::= <all> | <some>

<all> ::= ALL

<some> ::= SOME | ANY

<exists predicate> ::= EXISTS <table subquery>

<unique predicate> ::= UNIQUE <table subquery>
```

```
<match predicate> ::=
    <row value designator> MATCH [ UNIQUE ]
        [ PARTIAL | FULL ] <table subquery>

<overlaps predicate> ::=
    <row value designator 1> OVERLAPS <row value designator 2>

<row value designator 1> ::= <row value designator>

<row value designator 2> ::= <row value designator>

<row value designator 1> ::= <row value designator>

<row value designator 2> ::= <row value designator>

<similar predicate> ::=
    <match value> [ NOT ] SIMILAR TO
        <similar pattern>
      [ ESCAPE <escape character> ]

<similar pattern> ::= <character value expression>

<quantified predicate> ::=
      <existential clause> <left paren> <search condition> <right paren>
    | <universal clause> <left paren> <search condition> <right paren>
    | <quantified comparison predicate>

<existential clause> ::=
    FOR SOME <table reference list>

<table reference list> ::=
    <table reference> [ ( <comma> <table reference> )... ]

<table reference> ::=
      <table name> [ [ AS ] <correlation name>
          [ <left paren> <derived column list> <right paren> ] ]
    | <derived table> [ AS ] <correlation name>
          [ <left paren> <derived column list> <right paren> ]
    | <joined table>

<derived column list> ::= <column name list>

<column name list> ::=
    <column name> [ ( <comma> <column name> )... ]

<derived table> ::= <table subquery>

<joined table> ::=
```

```
      <cross join>
    | <qualified join>
    | <left paren> <joined table> <right paren>

<cross join> ::=
    <table reference> CROSS JOIN <table reference>

<qualified join> ::=
    <table reference> [ NATURAL ] [ <join type> ] JOIN
      <table reference> [ <join specification> ]

<join type> ::=
      INNER
    | <outer join type> [ OUTER ]
    | UNION

<outer join type> ::=
      LEFT
    | RIGHT
    | FULL

<join specification> ::=
      <join condition>
    | <named columns join>
    | <constraint join>

<join condition> ::= ON <search condition>

<named columns join> ::=
    USING <left paren> <join column list> <right paren>

<join column list> ::= <column name list>

<constraint join> ::=
    | USING PRIMARY KEY
    | USING FOREIGN KEY
    | USING CONSTRAINT <constraint name>

<constraint name> ::= <schema qualified name>

<universal clause> ::=
    FOR ALL <table reference list>

<there is predicate> ::=
    <left paren> <there is clause> <where clause> <right paren>

<there is clause> ::=
    THERE IS <table reference list>
```

```
<where clause> ::= WHERE <search condition>

<distinct predicate> ::=
    <row value designator 1> IS DISTINCT FROM
        <row value designator 2>

<boolean predicate> ::=
    <boolean value expression> [ IS [ NOT ]
          <truth value> ]

<truth value> ::=
      TRUE
    | FALSE
    | UNKNOWN

<cast specification> ::=
    CAST <left paren> <cast operand> AS
        <cast target> <right paren>

<cast operand> ::=
      <value expression>
    | NULL

<cast target> ::=
      <domain name>
    | <data type>

<data type> ::=
      <predefined type>
    | <abstract data type name>
    | <generated type reference>
    | <template parameter name>
    | <collection type>

<predefined type> ::=
      <character string type>
           [ CHARACTER SET <character set specification> ]
    | <national character string type>
    | <bit string type>
    | <numeric type>
    | <enumerated type>
    | <boolean type>
    | <datetime type>
    | <interval type>

<character string type> ::=
      CHARACTER [ <left paren> <length> <right paren> ]
```

```
    | CHAR [ <left paren> <length> <right paren> ]
    | CHARACTER VARYING <left paren> <length> <right paren>
    | CHAR VARYING <left paren> <length> <right paren>
    | VARCHAR <left paren> <length> <right paren>

<length> ::= <unsigned integer>

<national character string type> ::=
      NATIONAL CHARACTER [ <left paren> <length> <right paren> ]
    | NATIONAL CHAR [ <left paren> <length> <right paren> ]
    | NCHAR [ <left paren> <length> <right paren> ]
    | NATIONAL CHARACTER VARYING <left paren> <length> <right paren>
    | NATIONAL CHAR VARYING <left paren> <length> <right paren>
    | NCHAR VARYING <left paren> <length> <right paren>

<bit string type> ::=
      BIT [ <left paren> <length> <right paren> ]
    | BIT VARYING <left paren> <length> <right paren>

<numeric type> ::=
      <exact numeric type>
    | <approximate numeric type>

<exact numeric type> ::=
      NUMERIC [ <left paren> <precision> [ <comma> <scale> ] <right paren> ]
    | DECIMAL [ <left paren> <precision> [ <comma> <scale> ] <right paren> ]
    | DEC [ <left paren> <precision> [ <comma> <scale> ] <right paren> ]
    | INTEGER
    | INT
    | SMALLINT

<precision> ::= <unsigned integer>

<scale> ::= <unsigned integer>

<approximate numeric type> ::=
      FLOAT [ <left paren> <precision> <right paren> ]
    | REAL
    | DOUBLE PRECISION

<enumerated type> ::=
    <left paren> <enumeration name list> <right paren>

<enumeration name list> ::=
    <enumeration name> [ ( <comma> <enumeration name> )... ]

<boolean type> ::= BOOLEAN
```

```
<datetime type> ::=
      DATE
    | TIME [ <left paren> <time precision> <right paren> ]
          [ WITH TIME ZONE ]
    | TIMESTAMP [ <left paren> <timestamp precision> <right paren> ]
          [ WITH TIME ZONE ]

<time precision> ::= <time fractional seconds precision>

<time fractional seconds precision> ::= <unsigned integer>

<timestamp precision> ::= <time fractional seconds precision>

<interval type> ::= INTERVAL <interval qualifier>

<generated type reference> ::=
    <type template name> <template parameter list>

<type template name> ::= <schema qualified name>

<template parameter list> ::=
    <left paren>
      <template parameter> [ ( <comma> <template parameter> )... ]
    <right paren>

<template parameter> ::=
      <value specification>
    | <data type>

<collection type> ::=
      <set type>
    | <multiset type>
    | <list type>

<set type> ::= SET <left paren> <data type> <right paren>

<multiset type> ::= MULTISET <left paren> <data type> <right paren>

<list type> ::= LIST <left paren> <data type> <right paren>

<numeric value function> ::=
      <position expression>
    | <extract expression>
    | <length expression>

<position expression> ::=
    POSITION <left paren> <character value expression>
        IN <character value expression> <right paren>
```

```
<extract expression> ::=
    EXTRACT <left paren> <extract field>
        FROM <extract source> <right paren>

<extract field> ::=
      <datetime field>
    | <time zone field>

<datetime field> ::=
      <non-second datetime field>
    | SECOND

<time zone field> ::=
      TIMEZONE_HOUR
    | TIMEZONE_MINUTE

<extract source> ::=
      <datetime value expression>
    | <interval value expression>

<datetime value expression> ::=
      <datetime term>
    | <interval value expression> <plus sign> <datetime term>
    | <datetime value expression> <plus sign> <interval term>
    | <datetime value expression> <minus sign> <interval term>

<interval term> ::=
      <interval factor>
    | <interval term 2> <asterisk> <factor>
    | <interval term 2> <solidus> <factor>
    | <term> <asterisk> <interval factor>

<interval factor> ::=
    [ <sign> ] <interval primary>

<interval primary> ::=
      <value expression primary> [ <interval qualifier> ]

<interval term 2> ::= <interval term>

<interval value expression> ::=
      <interval term>
    | <interval value expression 1> <plus sign> <interval term 1>
    | <interval value expression 1> <minus sign> <interval term 1>
    | <left paren> <datetime value expression> <minus sign>
          <datetime term> <right paren> <interval qualifier>
```

```
<interval value expression 1> ::= <interval value expression>

<interval term 1> ::= <interval term>

<datetime term> ::=
      <datetime factor>

<datetime factor> ::=
      <datetime primary> [ <time zone> ]

<datetime primary> ::=
      <value expression primary>
    | <datetime value function>

<datetime value function> ::=
      <current date value function>
    | <current time value function>
    | <current timestamp value function>

<current date value function> ::= CURRENT_DATE

<current time value function> ::=
      CURRENT_TIME [ <left paren> <time precision> <right paren> ]

<current timestamp value function> ::=
      CURRENT_TIMESTAMP [ <left paren> <timestamp precision> <right paren> ]

<time zone> ::=
    AT <time zone specifier>

<time zone specifier> ::=
      LOCAL
    | TIME ZONE <interval primary>

<length expression> ::=
      <char length expression>
    | <octet length expression>
    | <bit length expression>

<char length expression> ::=
    ( CHAR_LENGTH | CHARACTER_LENGTH )
        <left paren> <string value expression> <right paren>

<string value expression> ::=
      <character value expression>
    | <bit value expression>

<octet length expression> ::=
```

```
    OCTET_LENGTH <left paren> <string value expression> <right paren>

<bit length expression> ::=
    BIT_LENGTH <left paren> <string value expression> <right paren>

<enumerated value expression> ::=
      <domain name> <left paren> <value expression> <right paren>
    | <enumerated primary>

<enumerated primary> ::=
      <value expression primary>

<attributes function> ::=
    ATTRIBUTES
      <left paren> <abstract data type value expression> <right paren>

<abstract data type value expression> ::= <ADT expression>

<ADT expression> ::=
    <ADT term>
    | <ADT expression> <term operator> <ADT term>

<term operator> ::= <ADT operator>

<ADT operator> ::= <user-defined operator symbol)

<ADT term> ::=
    <ADT factor>
    | <ADT term> <factor operator> <ADT factor>

<factor operator> ::= <ADT operator>

<ADT factor> ::=
    <ADT primary>
    | <prefix operator> <ADT primary>
    | <ADT primary> <postfix operator>

<ADT primary> ::=
    <value expression primary>

<prefix operator> ::= <ADT operator>

<postfix operator> ::= <ADT operator>

<table value expression> ::=
    <table type>
      <left paren>
        [ <value expression> [ ( <comma> <value expression> )... ] ]
```

```
      <right paren>

<table type> ::=
      TABLE
    | SET
    | LIST

<collection value expression> ::=
      <set value designator>
    | <multiset value designator>
    | <list value designator>

<set value designator> ::=
    SET <left paren> [ <collection list> ] <right paren>

<collection list> ::=
    <collection element> [ ( <comma> <collection element> )... ]

<collection element> ::= <value expression>

<multiset value designator> ::=
    MULTISET <left paren> [ <collection list> ] <right paren>

<list value designator> ::=
    LIST <left paren> [ <collection list> ] <right paren>

<as clause> ::= [ AS ] <column name>

<table expression> ::=
    <from clause>
    [ <where clause> ]
    [ <group by clause> ]
    [ <having clause> ]

<from clause> ::= FROM <table reference>
    [ ( <comma> <table reference> )... ]

<group by clause> ::=
    GROUP BY <grouping column reference list>

<grouping column reference list> ::=
    <grouping column reference>
        [ ( <comma> <grouping column reference> )... ]

<grouping column reference> ::=
    <column reference> [ <collate clause> ]

<having clause> ::= HAVING <search condition>
```

```
<table value designator> ::=
    VALUES <table value designator list>

<table value designator list> ::=
    <row value designator> [ ( <comma> <row value designator> )... ]

<explicit table> ::= <table type> <table name>

<collection expression> ::=
    <value expression>

<query term> ::=
      <non-join query term>
    | <joined table>

<corresponding spec> ::=
    CORRESPONDING [ BY <left paren> <corresponding column list> <right paren> ]

<corresponding column list> ::= <column name list>

<query primary> ::=
      <non-join query primary>
    | <joined table>

<recursive union> ::=
    <left paren> <initial expression>
      RECURSIVE UNION <correlation name list>
          [ <left paren> <recursive column list> <right paren> ]
      <iteration expression>
      [ <search clause> ]
      [ <cycle clause> ]
      [ <limit clause> ] <right paren>

<initial expression> ::= <query expression>

<correlation name list> ::=
    <correlation name> [ ( <comma> <correlation name> )... ]

<recursive column list> ::= <column name list>

<iteration expression> ::= <query expression>

<search clause> ::=
    SEARCH <search order> SET <sequence column>

<search order> ::=
      PREORDER
```

```
    | ( DEPTH | BREADTH ) FIRST BY <sort specification list>

<sort specification list> ::=
    <sort specification> [ ( <comma> <sort specification> )... ]

<sort specification> ::=
    <sort key> [ <collate clause> ] [ <ordering specification> ]

<sort key> ::=
    <value expression>

<ordering specification> ::= ASC | DESC

<sequence column> ::= <column name>

<cycle clause> ::=
    CYCLE [ <cycle column list> ]
      SET <cycle mark column>
          [ TO <cycle mark value> ]

<cycle column list> ::=
    <cycle column> [ ( <comma> <cycle column> )... ]

<cycle column> ::= <column name>

<cycle mark column> ::= <column name>

<cycle mark value> ::= <value expression>

<limit clause> ::=
    [ RETURN | EXCEPTION ] LIMIT
        <left paren> <value specification> <right paren>

<trigger definition> ::=
    [ CREATE ] TRIGGER [ <trigger name> ]
      <trigger action time> <trigger event>
        [ ON <table name> ]
        [ ORDER <order value> ]
        [ REFERENCING <old or new values alias list> ]
      <triggered action>

<trigger name> ::= <schema qualified name>

<trigger action time> ::=
      BEFORE
    | AFTER
    | INSTEAD OF
```

```
<trigger event> ::=
      INSERT
    | DELETE
    | UPDATE [ OF <trigger column list> ]

<trigger column list> ::= <column name list>

<order value> ::= <unsigned integer>

<old or new values alias list> ::=
    <old or new values alias>...

<old or new values alias> ::=
      OLD [ AS ]
          <old values correlation name>
    | NEW [ AS ]
          <new values correlation name>
    | OLD_TABLE [ AS ]
          <old values table alias>
    | NEW_TABLE [ AS ]
          <new values table alias>

<old values correlation name> ::= <correlation name>

<new values correlation name> ::= <correlation name>

<old values table alias> ::= <identifier>

<new values table alias> ::= <identifier>

<triggered action> ::=
    [ FOR EACH ( ROW | STATEMENT ) ]
      [ WHEN <left paren> <search condition> <right paren> ]
      <triggered SQL statement>

<triggered SQL statement> ::=
    <SQL procedure statement> <semicolon>

<SQL procedure statement> ::=
    [ ASYNC <left paren> <async statement identifier> <right paren> ]
          <SQL executable statement>

<async statement identifier> ::= <numeric value expression>

<SQL executable statement> ::=
      <SQL schema statement>
    | <SQL data statement>
    | <SQL transaction statement>
```

```
    | <SQL connection statement>
    | <SQL session statement>
    | <SQL diagnostics statement>

<SQL schema statement> ::=
      <SQL schema definition statement>
    | <SQL schema manipulation statement>

<SQL schema definition statement> ::=
      <schema definition>
    | <table definition>
    | <view definition>
    | <grant statement>
    | <role definition>
    | <grant role statement>
    | <domain definition>
    | <null class definition>
    | <character set definition>
    | <collation definition>
    | <translation definition>
    | <assertion definition>
    | <trigger definition>
    | <routine>
    | <abstract data type definition>
    | <type template definition>

<schema definition> ::=
    CREATE SCHEMA <schema name clause>
      [ <schema character set specification> ]
      [ <schema path specification> ]
      [ <schema element>... ]

<schema name clause> ::=
      <schema name>
    | AUTHORIZATION <schema authorization identifier>
    | <schema name> AUTHORIZATION <schema authorization identifier>

<schema authorization identifier> ::=
    <authorization identifier>

<schema character set specification> ::=
    DEFAULT CHARACTER SET <character set specification>

<schema path specification> ::=
    PATH <schema name list>

<schema element> ::=
      <table definition>
```

```
    | <view definition>
    | <domain definition>
    | <null class definition>
    | <character set definition>
    | <collation definition>
    | <translation definition>
    | <assertion definition>
    | <trigger definition>
    | <routine>
    | <abstract data type definition>
    | <type template definition>
    | <grant statement>
    | <role definition>
    | <grant role statement>

<table definition> ::=
    CREATE [ <table scope> ] <table type> <table name>
      [ <constant or updatable> ]
      ( <table element list> | <subtable clause> )
      [ ON COMMIT <table commit action> ROWS ]

<table scope> ::=
    <global or local> TEMPORARY

<global or local> ::=
      GLOBAL
    | LOCAL

<constant or updatable> ::=
      CONSTANT
    | UPDATABLE

<table element list> ::=
    <left paren> <table element> [ ( <comma> <table element> )... ] <right paren>

<table element> ::=
      <column definition>
    | <table constraint definition>
    | <like clause>

<column definition> ::=
    <column name>
    ( <data type> | <domain name> )
    [ <default clause> ]
    [ <column constraint definition>... ]
    [ <collate clause> ]
    [ <null clause> ]
```

```
<default clause> ::=
      ( DEFAULT | <assignment operator> ) <default option>

<default option> ::=
      <literal>
    | <datetime value function>
    | USER
    | CURRENT_USER
    | SESSION_USER
    | SYSTEM_USER
    | NULL [ <left paren> <null state> <right paren> ]
    | <function invocation>

<column constraint definition> ::=
    [ <constraint name definition> ]
    <column constraint> [ <constraint attributes> ]

<constraint name definition> ::= CONSTRAINT <constraint name>

<column constraint> ::=
      NOT NULL
    | <unique specification>
    | <references specification>
    | <check constraint definition>

<unique specification> ::=
      UNIQUE
    | PRIMARY KEY

<references specification> ::=
    REFERENCES [ PENDANT ] <referenced table and columns>
      [ MATCH <match type> ]
      [ <referential triggered action> ]

<referenced table and columns> ::=
     <table name> [ <left paren> [ <reference column list> ] <right paren> ]

<reference column list> ::= <column name list>

<match type> ::=
      FULL
    | PARTIAL

<referential triggered action> ::=
      <update rule> [ <delete rule> ]
    | <delete rule> [ <update rule> ]

<update rule> ::= ON UPDATE <referential action>
```

```
<referential action> ::=
      CASCADE
    | SET NULL [ <left paren> <null state> <right paren> ]
    | SET DEFAULT
    | RESTRICT
    | NO ACTION

<delete rule> ::= ON DELETE <referential action>

<check constraint definition> ::=
    CHECK <left paren> <search condition> <right paren>

<constraint attributes> ::=
      <constraint check time> [ [ NOT ] DEFERRABLE ]
    | [ NOT ] DEFERRABLE [ <constraint check time> ]

<constraint check time> ::=
      INITIALLY DEFERRED
    | INITIALLY IMMEDIATE

<null clause> ::=
    NULL IS <null class name>

<null class name> ::= <schema qualified name>

<null clause> ::=
    NULL IS <null class name>

<table constraint definition> ::=
    [ <constraint name definition> ]
    <table constraint> [ <constraint attributes> ]

<table constraint> ::=
      <unique constraint definition>
    | <referential constraint definition>
    | <check constraint definition>

<unique constraint definition> ::=
                                even in SQL3)
      <unique specification> [ <left paren> <unique column list>
          <right paren> ]
    | UNIQUE ( VALUE )

<unique column list> ::= <column name list>

<referential constraint definition> ::=
    FOREIGN KEY [ <left paren> <referencing columns> <right paren> ]
```

```
      <references specification>

<referencing columns> ::=
    <reference column list>

<like clause> ::= LIKE <table name>

<subtable clause> ::=
    UNDER <supertable clause>
        [ ( , <supertable clause> )... ]

<supertable clause> ::=
    <supertable name>
      [ WITH ( <member renaming element>
          [ ( , <member renaming element> )... ] ) ]

<supertable name> ::= <table name>

<member renaming element> ::=
    <supertable member name> AS <subtable member name>

<supertable member name> ::=
      <column name>
    | <routine name>

<subtable member name> ::=
      <column name>
    | <routine name>

<table commit action> ::=
      PRESERVE
    | DELETE

<view definition> ::=
    CREATE VIEW <table name> [ <left paren> <view column list> <right paren> ]
      AS <query expression>
      [ WITH [ <levels clause> ] CHECK OPTION ]

<view column list> ::= <column name list>

<levels clause> ::=
      CASCADED
    | LOCAL

<domain definition> ::=
    CREATE DOMAIN <domain name>
        [ AS ] <data type>
      [ <default clause> ]
```

```
      [ <domain constraint>... ]
      [ <collate clause> ]
      [ <null clause> ]

<domain constraint> ::=
    [ <constraint name definition> ]
    <check constraint definition> [ <constraint attributes> ]

<null class definition> ::=
    CREATE NULL CLASS <null class name> [ AS ]
        ( <null state list> )

<null state list> ::=
    <null state> [ ( <comma> <null state> )... ]

<character set definition> ::=
    CREATE CHARACTER SET <character set name>
        [ AS ]
      <character set source>
      <form-of-use specification>
      [ <collate clause> | <limited collation definition> ]

<character set source> ::=
      GET <existing character set name>
          [ <plus sign> <character set source> ]
    | <left paren> <character list> <right paren>

<character list> ::=
    <character specification> [ ( <comma> <character specification> )... ]

<character specification> ::=
      <character string literal>
    | <ISO 10646 position>
    | <ISO 10646 character name>

<ISO 10646 position> ::=
    <val> [ <sep> <val> [ <sep> <val> [ <sep> <val> ] ] ]

<val> ::= <unsigned numeric literal>

<sep> ::= <ampersand>

<ISO 10646 character name> ::= !! <EMPHASIS>(See the Syntax Rules)

<form-of-use specification> ::= <identifier>

<limited collation definition> ::=
    COLLATION FROM <collation source>
```

```
<collation source> ::=
      <collating sequence definition>
    | <translation collation>
    | <collation dictionary specification>
    | <collation routine specification>

<collating sequence definition> ::=
      <external collation>
    | <schema collation name>
    | <internal collation source>
    | DESC <left paren> <collation name> <right paren>
    | DEFAULT

<external collation> ::=
    EXTERNAL
        <left paren> <quote> <external collation name> <quote> <right paren>

<external collation name> ::=
      <standard collation name>
    | <implementation-defined collation name>

<standard collation name> ::= <collation name>

<implementation-defined collation name> ::= <collation name>

<schema collation name> ::= <collation name>

<internal collation source> ::= <left paren> <collation options> <right paren>

<collation options> ::=
    <collation option> [ ( <comma> <collation option> )... ]

<collation option> ::=
      USING <left paren> <collating basis> <right paren>
    | SEQUENCE <left paren> <enumerated collating sequence> <right paren>
    | MODIFY <left paren> <collating modifiers> <right paren>
    | WHEN NOT FOUND
        ( IGNORE | MAX | MIN )

<collating basis> ::=
    <collating foundation> [ ( <plus sign> <collating foundation> )... ]

<collating foundation> ::=
    <collating sequence definition>
    <collating sequence definition> [ ( <asterisk> <translation name> )... ]

<enumerated collating sequence> ::=
```

```
    <collating chars> [ ( <comma> <collating chars> )... ]

<collating chars> ::=
      <character specification>
    | <character range>

<character range> ::=
    <character specification> <minus sign> <character specification>

<collating modifiers> ::=
      <collating modifier> [ ( <comma> <collating modifier> )... ]

<collating modifier> ::=
    <collating chars>
      ( <less than operator> | <greater than operator> | <equals operator> )
    <collating chars>

<translation collation> ::=
    TRANSLATION <translation name>
        [ THEN COLLATION <collation name> ]

<collation dictionary specification> ::=
    DICTIONARY <dictionary name> [ <plus sign> <dictionary name> ]

<dictionary name> ::=
    <quote> <implementation-defined dictionary name> <quote>

<implementation-defined dictionary name> ::= !! <EMPHASIS>(See the Syntax Rules)

<collation routine specification> ::=
      ROUTINE <left paren> <implementation-defined routine name>
          <left paren> <params> <right paren> <right paren>

<implementation-defined routine name> ::= !! <EMPHASIS>(See the Syntax Rules)

<params> ::= !! <EMPHASIS>(Not yet defined)

<params> ::= !! <EMPHASIS>(Not yet defined)

<existing character set name> ::=
      <standard character repertoire name>
    | <implementation-defined character repertoire name>
    | <schema character set name>

<schema character set name> ::= <character set name>

<collation definition> ::=
    CREATE COLLATION <collation name> FOR
```

```
        <character set specification>
      FROM <collation source>
        [ <pad attribute> ]

<pad attribute> ::=
      NO PAD
    | PAD SPACE

<translation definition> ::=
    CREATE TRANSLATION <translation name>
      FOR <source character set specification>
        TO <target character set specification>
      FROM <translation source>

<source character set specification> ::= <character set specification>

<target character set specification> ::= <character set specification>

<translation source> ::=
      <translation specification>
    | <translation routine>

<translation specification> ::=
      <external translation>
    | IDENTITY
    | <schema translation name>
    | <internal translation source>

<external translation> ::=
    EXTERNAL
        <left paren> <quote> <external translation name> <quote> <right paren>

<external translation name> ::=
      <standard translation name>
    | <implementation-defined translation name>

<standard translation name> ::= <translation name>

<implementation-defined translation name> ::= <translation name>

<schema translation name> ::= <translation name>

<internal translation source> ::=
    <left paren> <translation options> <right paren>

<translation options> ::=
    <translation option> [ ( <comma> <translation option> )... ]
```

```
<translation option> ::=
      USING <left paren> <translation basis> <right paren>
    | MODIFY <left paren> <translation modifiers> <right paren>

<translation basis> ::=
    <translation definition> [ ( <asterisk> <translation definition> )... ]

<translation modifiers> ::=
      <translation modifier> [ ( <comma> <translation modifier> )... ]

<translation modifier> ::=
    <collating chars> <equals operator> <collating chars>

<translation routine> ::=
      ROUTINE <left paren> <implementation-defined routine name>
          <left paren> <params> <right paren> <right paren>

<assertion definition> ::=
    CREATE ASSERTION <constraint name>
      <assertion trigger>...
      <triggered assertion> [ <constraint attributes> ]

<assertion trigger> ::=
      <immediate assertion trigger>
    | <deferred assertion trigger>

<immediate assertion trigger> ::=
    AFTER ( <assertion trigger statement>
      [ ( <comma> <assertion trigger statement> )... ] ON
           <table name> )...

<assertion trigger statement> ::=
      INSERT
    | DELETE
    | UPDATE [ OF <left paren> <assertion column list> <right paren> ]

<assertion column list> ::= <column name list>

<deferred assertion trigger> ::= BEFORE COMMIT

<triggered assertion> ::=
    CHECK <left paren> <search condition> <right paren>
      [ FOR
          [ EACH [ ROW OF ] ] <table name> ]

<routine> ::=
    [ CREATE | DECLARE ] <routine header> <routine name> <parameter list>
    [ <returns clause> ]
```

```
    [ <caller language clause> ]
    [ SPECIFIC <specific name> ] <semicolon>
    <routine body> <semicolon>

<routine header> ::=
      PROCEDURE
    | [ <function type> ] FUNCTION

<function type> ::=
      CONSTRUCTOR
    | DESTRUCTOR
    | DESIGNATOR
    | ACTOR

<parameter list> ::=
    <left paren>
      [ <parameter declaration> [ (  <comma> <parameter declaration> )... ] ]
    <right paren>

<parameter declaration> ::=
      [ <parameter mode> ] [ <parameter name> ] <data type>
          [ <default clause> ]
    | <status parameter>

<parameter mode> ::=
      IN
    | OUT
    | INOUT

<status parameter> ::=
      SQLSTATE
    | SQLCODE

<returns clause> ::= RETURNS <returns data type> [ <result cast> ]

<returns data type> ::= <data type>

<result cast> ::= CAST FROM <data type>

<caller language clause> ::= <language clause>

<specific name> ::= <schema qualified name>

<routine body> ::=
      <SQL routine body>
    | <external body reference>

<SQL routine body> ::= <SQL procedure statement>
```

```
<external body reference> ::=
    EXTERNAL [ NAME <external routine name> ]
    <external routine language clause>
    [ <variant attribute> ]

<external routine name> ::= <identifier>

<external routine language clause> ::= <language clause>

<variant attribute> ::=
      VARIANT
    | NOT VARIANT

<abstract data type definition> ::=
      <distinct type definition>
    | <explicit abstract data type definition>

<distinct type definition> ::=
    CREATE DISTINCT TYPE <distinct type name>
      AS <data type>

<distinct type name> ::= <abstract data type name>

<explicit abstract data type definition> ::=
    CREATE TYPE <abstract data type name>
      <abstract data type body>

<abstract data type body> ::=
    [ <oid options> ]
    [ <subtype clause> ]
    [ [ <constant or updatable> ] [ <member list> ] ]

<oid options> ::=
      WITH OID [ [ NOT ] VISIBLE ]
    | WITHOUT OID

<subtype clause> ::=
    UNDER <supertype clause>
        [ ( <comma> <supertype clause> )... ]

<supertype clause> ::=
    <abstract data type name> [ <component renaming clause> ]

<component renaming clause> ::=
    WITH <left paren> <component renaming element>
        [ ( , <component renaming element> )... ] <right paren>
```

```
<component renaming element> ::=
    <supertype component name> AS <subtype component name>

<supertype component name> ::= <component name>

<component name> ::= <identifier>

<subtype component name> ::= <component name>

<member list> ::=
    <left paren> <member> [ ( <comma> <member> )... ] <right paren>

<member> ::=
      <attribute definition>
    | <routine declaration>
    | <operator name list>
    | <equals clause>
    | <less-than clause>
    | <cast clause>
    | <table constraint definition>

<attribute definition> ::=
      <stored attribute>
    | <virtual attribute>

<stored attribute> ::=
    [ <encapsulation level> ]
    <attribute name> [ <constant or updatable> ]
    ( <data type> | <domain name> )
    [ <default clause> ]
    [ <column constraint definition>... ]
    [ <collate clause> ]
    [ <null clause> ]

<encapsulation level> ::=
      PRIVATE
    | PROTECTED
    | PUBLIC

<virtual attribute> ::=
    [ <encapsulation level> ]
      <attribute name> <derivation clause>
    [ <check constraint definition>... ]
    [ <collate clause> ]

<derivation clause> ::=
    [ READ ONLY | CONSTANT | UPDATABLE ] <data type>
      VIRTUAL [ <derivation functions> ]
```

```
<derivation functions> ::=
      <get function> [ <set function> ]
    | <set function> [ <get function> ]

<get function> ::=
    GET WITH <routine name>

<set function> ::=
    SET WITH <routine name>

<routine declaration> ::=
    [ <encapsulation level> ] <routine>

<operator name list> ::=
    OPERATORS <specific routine designator>...

<specific routine designator> ::=
      SPECIFIC <specific name>
    | <member name>

<member name> ::= <routine name> [ <data type list> ]

<data type list> ::=
    <left paren> <data type> [ ( <comma> <data type> )... ] <right paren>

<equals clause> ::=
    EQUALS <equals function specification> <semicolon>

<equals function specification> ::=
      <routine name>
    | STATE
    | OID

<less-than clause> ::=
    LESS THAN <less-than function specification> <semicolon>

<less-than function specification> ::=
      <routine name>
    | NONE

<cast clause> ::=
    CAST <left paren> <operand data type> AS
        <result data type>
         WITH <cast function> <right paren> <semicolon>

<operand data type> ::= <data type>
```

```
<result data type> ::= <data type>

<cast function> ::=
    <routine name>

<type template definition> ::=
    CREATE TYPE TEMPLATE <type template name>
      <template parameter declaration list>
      <abstract data type body>

<template parameter declaration list> ::=
    <left paren>
      <template parameter declaration>
        [ ( <comma> <template parameter declaration>)... ]
    <right paren>

<template parameter declaration> ::=
    <template parameter name> <template parameter type>

<template parameter type> ::=
      <data type>
    | TYPE

<grant statement> ::=
    GRANT <privileges>
      TO <grantee> [ ( <comma> <grantee> )... ]
        [ WITH GRANT OPTION ]

<privileges> ::=
      ALL SCHEMA PRIVILEGES
    | <object privileges> ON <object name>

<object privileges> ::=
      ALL PRIVILEGES
    | <action> [ ( <comma> <action> )... ]

<action> ::=
      SELECT [ <left paren> <privilege column list> <right paren> ]
    | DELETE
    | INSERT [ <left paren> <privilege column list> <right paren> ]
    | UPDATE [ <left paren> <privilege column list> <right paren> ]
    | REFERENCES [ <left paren> <privilege column list> <right paren> ]
    | USAGE
    | TRIGGER
    | EXECUTE
    | UNDER

<privilege column list> ::= <column name list>
```

```
<object name> ::=
      [ <table type> ] <table name>
    | DOMAIN <domain name>
    | COLLATION <collation name>
    | CHARACTER SET <character set name>
    | TRANSLATION <translation name>
    | NULL CLASS <null class name>
    | DATA TYPE <abstract data type name>
    | MODULE <module name>
    | TYPE TEMPLATE <type template name>
    | EXTERNAL ROUTINE <specific routine designator>

<grantee> ::=
      PUBLIC
    | <authorization identifier>
    | <role name>

<role name> ::= <authorization identifier>

<role definition> ::= CREATE ROLE <role name>

<grant role statement> ::=
    GRANT <role granted> [ ( <comma> <role granted> )... ]
      TO <grantee> [ ( <comma> <grantee> )... ]
        [ WITH ADMIN OPTION ]

<role granted> ::= <role name>

<SQL schema manipulation statement> ::=
      <drop schema statement>
    | <alter table statement>
    | <drop table statement>
    | <drop view statement>
    | <revoke statement>
    | <revoke role statement>
    | <drop role statement>
    | <alter domain statement>
    | <drop domain statement>
    | <drop null class statement>
    | <drop character set statement>
    | <drop collation statement>
    | <drop translation statement>
    | <drop assertion statement>
    | <drop trigger statement>
    | <drop routine statement>
    | <drop data type statement>
    | <drop type template statement>
```

```
<drop schema statement> ::=
    DROP SCHEMA <schema name> <drop behavior>

<drop behavior> ::= CASCADE | RESTRICT

<alter table statement> ::=
    ALTER <table type> <table name> <alter table action>

<alter table action> ::=
      <add column definition>
    | <alter column definition>
    | <drop column definition>
    | <add supertable clause>
    | <drop supertable clause>
    | <add table constraint definition>
    | <drop table constraint definition>

<add column definition> ::=
    ADD [ COLUMN ] <column definition>

<alter column definition> ::=
    ALTER [ COLUMN ] <column name> <alter column action>

<alter column action> ::=
      <set column default clause>
    | <drop column default clause>
    | <drop column domain clause>

<set column default clause> ::=
    SET <default clause>

<drop column default clause> ::=
    DROP DEFAULT

<drop column domain clause> ::=
    DROP DOMAIN [ <constraint disposition> [ <constraint name list> ] ]

<constraint disposition> ::=
      KEEP COLUMN CONSTRAINT
    | DROP COLUMN CONSTRAINT

<constraint name list> ::=
      ALL
    | <constraint name> [ ( <comma> <constraint name> )... ]

<drop column definition> ::=
    DROP [ COLUMN ] <column name> <drop behavior>
```

```
<add supertable clause> ::=
    ADD <supertable clause>

<drop supertable clause> ::=
    DROP <supertable clause> <drop behavior>

<add table constraint definition> ::=
    ADD <table constraint definition>

<drop table constraint definition> ::=
    DROP CONSTRAINT <constraint name> <drop behavior>

<drop table statement> ::=
    DROP <table type> <table name> <drop behavior>

<drop view statement> ::=
    DROP VIEW <table name> <drop behavior>

<revoke statement> ::=
    REVOKE [ GRANT OPTION FOR ]
        <privileges>
      FROM <grantee> [ ( <comma> <grantee> )... ] <drop behavior>

<revoke role statement> ::=
    REVOKE <role revoked> [ ( <comma> <role revoked> )... ]
      FROM <grantee> [ ( <comma> <grantee> )... ]

<role revoked> ::= <role name>

<drop role statement> ::= DROP ROLE <role name>

<alter domain statement> ::=
    ALTER DOMAIN <domain name> <alter domain action>

<alter domain action> ::=
      <set domain default clause>
    | <drop domain default clause>
    | <add domain constraint definition>
    | <drop domain constraint definition>

<set domain default clause> ::= SET <default clause>

<drop domain default clause> ::= DROP DEFAULT

<add domain constraint definition> ::=
    ADD <domain constraint>
```

```
<drop domain constraint definition> ::=
    DROP CONSTRAINT <constraint name>
      [ <constraint disposition> ]

<drop domain statement> ::=
    DROP DOMAIN <domain name> <drop behavior>
      [ <constraint disposition> [ <constraint name list> ] ]

<drop null class statement> ::=
     DROP NULL CLASS <null class name>

<drop character set statement> ::=
    DROP CHARACTER SET <character set name>

<drop collation statement> ::=
    DROP COLLATION <collation name>
      <drop behavior>

<drop translation statement> ::=
    DROP TRANSLATION <translation name>

<drop assertion statement> ::=
    DROP ASSERTION <constraint name>

<drop trigger statement> ::= DROP TRIGGER <trigger name>

<drop routine statement> ::=
    DROP ( PROCEDURE | FUNCTION ) <specific routine designator> <drop behavior>

<drop data type statement> ::=
    DROP DATA TYPE
        <abstract data type name> <drop behavior>

<drop type template statement> ::=
    DROP TYPE TEMPLATE <type template name>
      <drop behavior>

<SQL data statement> ::=
      <open statement>
    | <fetch statement>
    | <close statement>
    | <select statement: single row>
    | <new statement>
    | <destroy statement>
    | <SQL data change statement>

<open statement> ::=
    OPEN <cursor name>
```

```
      [ <open cascade option> ]

<open cascade option> ::=
      CASCADE ON
    | CASCADE OFF

<fetch statement> ::=
    FETCH [ [ <fetch orientation> ] FROM ]
      <cursor name> INTO <fetch target list>

<fetch orientation> ::=
      NEXT
    | PRIOR
    | FIRST
    | LAST
    | ( ABSOLUTE | RELATIVE ) <simple value specification>

<simple value specification> ::=
      <item reference>
    | <literal>

<fetch target list> ::=
    <target specification> [ ( <comma> <target specification> )... ]

<target specification> ::=
      <item reference>
    | <template parameter name>

<close statement> ::=
    CLOSE <cursor name>

<select statement: single row> ::=
    SELECT [ <set quantifier> ] <select list>
      INTO <select target list>
        <table expression>

<select target list> ::=
    <target specification> [ ( <comma> <target specification> )... ]

<new statement> ::=
    NEW <item reference>

<destroy statement> ::=
    DESTROY <object parameter name>

<object parameter name> ::= <parameter name>

<SQL data change statement> ::=
```

```
      <delete statement: positioned>
    | <delete statement: searched>
    | <insert statement>
    | <update statement: positioned>
    | <update statement: searched>

<delete statement: positioned> ::=
    DELETE [ FROM <table name> ]
      WHERE CURRENT OF <cursor name>

<delete statement: searched> ::=
    DELETE FROM <table reference>
      [ WHERE <search condition> ]

<insert statement> ::=
    INSERT INTO ( <table reference> | CURSOR <cursor name> )
      <insert columns and source>
      [ <insert point> ]

<insert columns and source> ::=
      [ <left paren> <insert column list> <right paren> ]
            <query expression>
    | DEFAULT VALUES

<insert column list> ::= <column name list>

<insert point> ::=
    <relative insert point> ELEMENT <where clause>

<relative insert point> ::=
      BEFORE
    | AFTER

<update statement: positioned> ::=
    UPDATE [ <table reference> ]
      SET [ <update type> ] <set clause list>
        WHERE CURRENT OF <cursor name>

<update type> ::= ALL | SOME | NONE

<set clause list> ::=
    <set clause> [ ( <comma> <set clause> )... ]

<set clause> ::=
    <update target> <equals operator> <row value designator>

<update target> ::=
      <object column>
```

```
    | <left paren> <object column list> <right paren>

<object column> ::= <column name>

<object column list> ::=
    <object column> [ ( <comma> <object column> )... ]

<update statement: searched> ::=
    UPDATE <table reference>
      <update mechanism>
      [ WHERE <search condition> ]

<update mechanism> ::=
      <update by setting>
    | <update by moving>

<update by setting> ::=
    SET [ <update type> ] <set clause list>

<update by moving> ::=
    MOVE <insert point>

<SQL transaction statement> ::=
      <start transaction statement>
    | <set transaction statement>
    | <set constraints mode statement>
    | <test completion statement>
    | <savepoint statement>
    | <release savepoint statement>
    | <commit statement>
    | <rollback statement>

<start transaction statement> ::=
    START TRANSACTION <transaction mode>
        [ ( <comma> <transaction mode> )...]

<transaction mode> ::=
      <isolation level>
    | <transaction access mode>
    | <diagnostics size>

<isolation level> ::=
    ISOLATION LEVEL <level of isolation>

<level of isolation> ::=
      READ UNCOMMITTED
    | READ COMMITTED
    | REPEATABLE READ
```

```
    | SERIALIZABLE

<transaction access mode> ::=
      READ ONLY
    | READ WRITE

<diagnostics size> ::=
    DIAGNOSTICS SIZE <number of conditions>

<number of conditions> ::= <simple value specification>

<set transaction statement> ::=
    SET [ LOCAL ] TRANSACTION <transaction mode>
        [ ( <comma> <transaction mode> )... ]

<set constraints mode statement> ::=
    SET CONSTRAINTS <constraint name list>
        ( DEFERRED | IMMEDIATE )

<test completion statement> ::=
    ( TEST | WAIT )
       ( ALL | ANY | <async statement identifier list> )
    COMPLETION

<async statement identifier list> ::=
    <left paren> <async statement identifier>
        [ ( <comma> <async statement identifier> )... ] <right paren>

<savepoint statement> ::= SAVEPOINT <savepoint specifier>

<savepoint specifier> ::=
      <savepoint name>
    | <simple target specification>

<savepoint name> ::= <identifier>

<simple target specification> ::=
    <item reference>

<release savepoint statement> ::=
    RELEASE SAVEPOINT <savepoint specifier>

<commit statement> ::=
    COMMIT [ WORK ] [ AND [ NO ] CHAIN ]

<rollback statement> ::=
    ROLLBACK [ WORK ] [ AND[ NO ]  CHAIN ]
      [ <savepoint clause> ]
```

```
<savepoint clause> ::=
    TO SAVEPOINT <savepoint specifier>

<SQL connection statement> ::=
      <connect statement>
    | <set connection statement>
    | <disconnect statement>

<connect statement> ::=
    CONNECT TO <connection target>

<connection target> ::=
      <SQL-server name>
        [ AS <connection name> ]
          correspondence with Tony Gordon)
        [ USER <user name> ]
    | DEFAULT

<SQL-server name> ::= <simple value specification>

<connection name> ::= <simple value specification>

<user name> ::= <simple value specification>

<set connection statement> ::=
    SET CONNECTION <connection object>

<connection object> ::=
      DEFAULT
    | <connection name>

<disconnect statement> ::=
    DISCONNECT <disconnect object>

<disconnect object> ::=
      <connection object>
    | ALL
    | CURRENT

<SQL session statement> ::=
      <set session authorization identifier statement>
    | <set role statement>
    | <set local time zone statement>

<set session authorization identifier statement> ::=
    SET SESSION AUTHORIZATION <value specification>
```

```
<set role statement> ::=
    SET ROLE ( <role name> | NONE )

<set local time zone statement> ::=
    SET TIME ZONE
        <set time zone value>

<set time zone value> ::=
      <interval value expression>
    | LOCAL

<SQL diagnostics statement> ::=
    <get diagnostics statement>

<get diagnostics statement> ::=
    GET DIAGNOSTICS <sql diagnostics information>

<sql diagnostics information> ::=
      <statement information>
    | <condition information>

<statement information> ::=
    <statement information item> [ ( <comma> <statement information item> )... ]

<statement information item> ::=
    <simple target specification>
        <equals operator> <statement information item name>

<statement information item name> ::=
      NUMBER
    | MORE
    | COMMAND_FUNCTION
    | ROW_COUNT
    | TRANSACTIONS_COMMITTED
    | TRANSACTIONS_ROLLED_BACK
    | TRANSACTION_ACTIVE

<condition information> ::=
    EXCEPTION <condition number>
      <condition information item>
          [ ( <comma> <condition information item> )... ]

<condition number> ::= <simple value specification>

<condition information item> ::=
    <simple target specification>
        <equals operator> <condition information item name>
```

```
<condition information item name> ::=
      CONDITION_NUMBER
    | RETURNED_SQLSTATE
    | CLASS_ORIGIN
    | SUBCLASS_ORIGIN
    | SERVER_NAME
    | CONNECTION_NAME
    | CONSTRAINT_CATALOG
    | CONSTRAINT_SCHEMA
    | CONSTRAINT_NAME
    | TRIGGER_CATALOG
    | TRIGGER_SCHEMA
    | TRIGGER_NAME
    | CATALOG_NAME
    | SCHEMA_NAME
    | TABLE_NAME
    | COLUMN_NAME
    | CURSOR_NAME
    | ROUTINE_CATALOG
    | ROUTINE_SCHEMA
    | ROUTINE_NAME
    | SPECIFIC_NAME
    | MESSAGE_TEXT
    | MESSAGE_LENGTH
    | MESSAGE_OCTET_LENGTH

<order by clause> ::=
    ORDER BY <sort specification list>

<updatability clause> ::=
    FOR
        ( READ ONLY |
          UPDATE [ OF <column name list> ] )

<temporary abstract data type declaration> ::=
    DECLARE TEMPORARY TYPE <abstract data type name>
        <abstract data type body>

<temporary table declaration> ::=
    DECLARE LOCAL TEMPORARY <table type> <table name>
      <table element list>
      [ ON COMMIT <table commit action> ROWS ]

<temporary view declaration> ::=
    DECLARE TEMPORARY VIEW <table name>
        [ <left paren> <view column list> <right paren> ]
        AS <query expression>
```

```
<scalar subquery> ::= <subquery>

<regular expression> ::=
      <regular term>
    | <regular expression> <vertical bar> <regular term>

<regular term> ::=
      <regular factor>
    | <regular term> <regular factor>

<regular factor> ::=
      <regular primary>
    | <regular primary> <asterisk>
    | <regular primary> <plus sign>

<regular primary> ::=
      <character specifier>
    | <percent>
    | <regular character set>
    | <left paren> <regular expression> <right paren>

<character specifier> ::=
      <non-escaped character>
    | <escaped character>

<non-escaped character> ::= !! <EMPHASIS>(See the Syntax Rules)

<escaped character> ::= !! <EMPHASIS>(See the Syntax Rules)

<regular character set> ::=
      <underscore>
    | <left bracket> <character enumeration>... <right bracket>
    | <left bracket> <circumflex> <character enumeration>... <right bracket>
    | <left bracket> <colon> <regular character set identifier> <colon> <right bracket>

<character enumeration> ::=
      <character specifier>
    | <character specifier> <minus sign> <character specifier>

<regular character set identifier> ::= <identifier>

<SQL object identifier> ::=
    <SQL provenance> <SQL variant>

<SQL provenance> ::= <arc1> <arc2> <arc3>

<arc1> ::= iso | 1 | iso <left paren> 1 <right paren>
```

```
<arc2> ::= standard | 0 | standard <left paren> 0 <right paren>

<arc3> ::= 9075

<SQL variant> ::= <SQL edition> <SQL conformance>

<SQL edition> ::= <1987> | <1989> | <1992>

<1987> ::= 0 | edition1987 <left paren> 0 <right paren>

<1989> ::= <1989 base> <1989 package>

<1989 base> ::= 1 | edition1989 <left paren> 1 <right paren>

<1989 package> ::= <integrity no> | <integrity yes>

<integrity no> ::= 0 | IntegrityNo <left paren> 0 <right paren>

<integrity yes> ::= 1 | IntegrityYes <left paren> 1 <right paren>

<1992> ::= 2 | edition1992 <left paren> 2 <right paren>

<SQL conformance> ::= <low> | <intermediate> | <high>

<low> ::= 0 | Low <left paren> 0 <right paren>

<intermediate> ::= 1 | Intermediate <left paren> 1 <right paren>

<high> ::= 2 | High <left paren> 2 <right paren>
```

38.4 Extension Temporelle à SQL92

- Document pour l'Extension Temporelle à SQL-92 `(ftp://FTP.cs.arizona.edu/tsql/tsql2/)`
- Spécification SQL-3 Temporelle `(ftp://FTP.cs.arizona.edu/tsql/tsql2/sql3/)`

Ce répertoire contient les spécifications pour une extension temporelle au standard du langage SQL-92. Ce nouveau langage est appelé TSQL2.

Les spécifications du langage présentées ici correspondent à la version finale du langage.

La correspondance doit être adressée au bureau du Comité de Conception du Langage TSQL2, Richard T.Snodgrass, Department of Computer Science, University of Arizona, Tucson, AZ 85721, rts@cs.arizona.edu. Les affiliations et les adresses électroniques des membres du Comité de Conception du langage TSQL2 peuvent être trouvées, dans un chapitre séparé, à la fin des spécifications du langage.

Le contenu du répertoire est le suivant :

spec.dvi,.ps Spécifications du langage TSQL2, publié en Septembre, 1994 bookspec.ps Spécifications du langage TSQL2, telles qu'elles apparaissent dans le livre sur TSQL2, publié en Septembre, 1995 (voir ci-dessous).

sql3 proposition de modification soumise au comités ANSI et ISO SQL3.

Un ensemble de commentaires, associés aux spécifications du langage, parle des décisions de conception, fournit des exemples, et traite de la façon dont le langage peut être implanté. Ces commentaires sont, à l'origine, des propositions faites au Comité de Conception du Langage TSQL2. Ils poursuivent actuellement un objectif différent: fournir des exemples de syntaxe, expliquer les nombreuses décisions prises durant la conception du langage, et comparer TSQL2 aux nombreuses autres propositions de langage faites au cours des quinze dernières années. Il faut insister sur le fait que ces commentaires ne font pas partie intégrante des spécifications du langage TSQL2 en lui-même, mais plutôt qu'elles le complètent et constituent un apport. Le mot de la fin est donné par les spécifications de TSQL2 proprement dit.

Les commentaires, ainsi que les spécifications du langage, plusieurs index, et d'autres éléments de support ont été publiés dans un livre :

Snodgrass, R.T., éditeur, The TSQL2 Temporal Query Language, (Le Langage de Requêtes Temporel TSQL2) Kluwer Academic Publishers, 1995, 674+xxiv pages.

Les commentaires d'évaluation sont donnés sous forme abrégée dans le livre; La totalité des commentaires est fournie dans le fichier eval.ps situé dans ce répertoire

Le fichier tl2tsql2.pl est un programme prolog qui traduit la logique temporelle autorisée en TSQL2. Ce programme a été écrit par Michael Boehlen boehlen@iesd.auc.dk Il peut être contacté pour obtenir un papier qui décrit cette traduction. C'est une version tout à fait ancienne du programme. Les nouvelles versions sont disponibles à `(http://www.cs.auc.dk/general/DBS/tdb/TimeCenter/Software)` (the TimeDB and Tiger systems).

39 Didacticiel SQL

On peut trouver ce didacticiel SQL à `(http://w3.one.net/ jhoffman/sqltut.htm)`

Pour les commentaires ou suggestions, envoyer un courrier électronique à jhoffman@one.net

Vous pouvez également souhaiter jeter un oeil à `(http://w3.one.net/ jhoffman/index.html)`

John Hoffman suggère de visiter les sites suivants :

`(http://www.contrib.andrew.cmu.edu/ shadow/sql.html)`Référence SQL

`(http://www.inquiry.com/techtips/thesqlpro/)`Demandez le Pro. de SQL

`(http://www.inquiry.com/techtips/thesqlpro/usefulsites.html)`Sites utiles au Pro. du SGBD Relationnel SQL

`(http://infoweb.magi.com/ steve/develop.html)`Les Sites de sources du programmeur SGBD

`(http://info.itu.ch/special/wwwfiles)` Allez-y et regardez le fichier comp_db.html

`(http://www.compapp.dcu.ie/databases/f017.html)`Ingrédients pour SGBD

`(http://www.stars.com/Tutorial/CGI/)`Création Web (http://wfn-shop.princeton.edu/cgi-bin/foldoc)Dictionnaire d'Informatique

(http://www-ccs.cs.umass.edu/db.html)DBMS Lab/Liens

SQL FAQ (http://epoch.CS.Berkeley.EDU:8000/sequoia/dba/montage/FAQ) Allez-y et regardez le fichier SQL_TOC.html

(http://chaos.mur.csu.edu.au/itc125/cgi/sqldb.html)SGBD SQL

(http://www.it.rit.edu/ wjs/IT/199602/icsa720/icsa720postings.html)Page de Conception de Bases de Données RIT

(http://www.pcslink.com/ ej/dbweb.html)Site de liens vers des Bases de Données (http://www.eng.uc.edu/ jtilley/tutorial.html)Didacticiels de programmation sur le Web

(http://www.ndev.com/ndc2/support/resources.htp)Ressources pour le Développement

(http://ashok.pair.com/sql.htm)Liste de Requêtes

(http://jazz.external.hp.com/training/sqltables/main.html)IMAGE SQL Diverses

(http://www.eit.com/web/netservices.html)Liste de Ressources Internet

Voici, ci-dessous, un extrait de la page d'accueil du didacticiel SQL.

```
Introduction au Langage de Requete Structure

Version 3.31

Cette page contient un didacticiel du Langage de Requete Structure( Structured
Query Language, egalement connu sous le nom de SQL). Ce didacticiel constitue
une nouveaute sur le World Wide Web, car c'est le premier didacticiel SQL
complet disponible sur l'Internet. SQL permet l'acces aux donnees dans les
systemes  de gestion de bases de donnees relationnels tels que Oracle,
Sybase, Informix, Microsoft SQL Server, Access, et autres en permettant aux
utilisateurs de decrire les donnees qu'ils desirent obtenir. SQL permet aussi
aux utilisateurs de definir l'organisation des donnees dans la base et de les
manipuler. Le but de cette page est de decrire l'utilisation de SQL, et de
donner des exemples. C'est le langage ANSI SQL, ou standard SQL, qui sera
utilise dans ce document. Il ne sera pas question ici des fonctionnalites
specifiques a un SGBD particulier, qui seront traitees dans la section "SQL
non-standard". Nous vous recommandons d'imprimer cette page afin de pouvoir
vous referer facilement aux differents exemples.
----------------------------------------------------------------------------
Table des matieres

     Principes fondamentaux de l'instruction SELECT
     Selection Conditionnelle
     Operateurs Relationnels
     Conditions Composees
     IN & BETWEEN
     Utilisation de LIKE
```

```
     Jointures
     Cles
     Realisation d'une Jointure
     Elimination des Doubles
     Alias & In/Sous-requetes

     Fonctions d'Agregation
     Vues
     Creation de Nouvelles Tables
     Modification des Tables
     Ajout de Donnees
     Suppression de Donnees
     Mise a Jour des Donnees

     Index
     GROUP BY & HAVING
     Sous-requetes Supplementaires
     EXISTS & ALL
     UNION & Jointures Externes
     SQL Integre
     Questions Courantes sur SQL
     SQL Non-standard
     Resume de la Syntaxe
     Liens Importants

----------------------------------------------------------------------------
Principes fondamentaux de l'instruction SELECT

Dans une base de donnees relationnelle, les donnees sont stockees dans des
tables. Par exemple,  une table pourrait mettre en relation le Numero de
Securite Sociale, le Nom et l'Adresse:

                        TableAdresseEmploye

 NSS      Prenom    Nom       Addresse        Ville       Etat
 512687458Joe       Smith     83 First Street Howard      Ohio
 758420012Mary      Scott     842 Vine Ave.   LosantivilleOhio
 102254896Sam       Jones     33 Elm St.      Paris       New York
 876512563Sarah     Ackerman  440 U.S. 110    Upton       Michigan

Maintenant, supposons que nous voulions obtenir l'adresse de chaque employe.
On utilisera SELECT, comme ceci :

SELECT Prenom, Nom, Adresse, Ville, Etat
FROM TableAdresseEmploye;

Voici le resultat de l'interrogation de notre base de donnees :
```

Prenom	Nom	Adresse	Ville	Etat
Joe	Smith	83 First Street	Howard	Ohio
Mary	Scott	842 Vine Ave.	Losantiville	Ohio
Sam	Jones	33 Elm St.	Paris	New York
Sarah	Ackerman	440 U.S. 110	Upton	Michigan

Explication de ce que l'on vient de faire : on vient de rechercher dans toutes les donnees de la table TableAdresseEmploye les colonnes nommees Prenom, Nom, Adresse, Ville et Etat. Noter que les noms de colonnes et les noms de tables sont sans espaces... ils doivent etre saisis en un seul mot; et que l'instruction se termine par un point-virgule (;). La forme generale d'une instruction SELECT, qui permet de retrouver toutes les lignes d'une table est :

```
SELECT NomColonne, NomColonne, ...
FROM NomTable;
```

Pour obtenir toutes les colonnes d'une table sans avoir a taper tous les noms de colonne, utiliser :

```
SELECT * FROM NomTable;
```

Chaque Systeme de Gestion de Base de Donnees (SGBD) et chaque logiciel de base de donnees utilisent differentes methodes pour se connecter a la base de donnee et pour entrer les instructions SQL; consultez le "guru" de votre ordinateur pour qu'il vous aide a vous connecter de facon a pouvoir utiliser SQL.

Selection Conditionnelle

Pour etudier plus avant l'instruction SELECT , jetons un oeil a un nouvel exemple de table (exemple uniquement hypothetique) :

EmployeeStatisticsTable

EmployeeIDNo	Salary	Benefits	Position
010	75000	15000	Manager
105	65000	15000	Manager
152	60000	15000	Manager
215	60000	12500	Manager
244	50000	12000	Staff
300	45000	10000	Staff
335	40000	10000	Staff
400	32000	7500	Entry-Level
441	28000	7500	Entry-Level

Operateurs Relationnels

Il y a six Operateurs Relationnels en SQL, et, apres les avoir presentes,
nous verrons comment les utiliser :

```
 =            Egal
 <> or !=
 (voir le manuel) Different
 <            Plus Petit Que
 >            Plus Grand Que
 <=           Plus Petit Que ou Egal a

 >=           Plus Grand Que ou Egal a
```

La clause WHERE est utilisee pour specifier que l'on affichera seulement
certaines ligne de la table, selon un critere definit par cette clause WHERE.
Ce sera plus clair en prenant une paire d'exemples.

Si l'on desire voir les numeros d'identification des employes (EMPLOYEEIDNO)
dont le salaire est egal ou superieur a 50 000, on utilisera la requete
suivante :

```
SELECT EMPLOYEEIDNO
FROM EMPLOYEESTATISTICSTABLE
WHERE SALARY >= 50000;
```

Noter que le symbole >= (plus grand que ou egal a) est utilise, puisque l'on
desire voir tout ceux qui gagnent plus de 50 000, ou 50 000, sur la meme
liste. On aura l'affichage :

```
EMPLOYEEIDNO
------------
010
105
152
215
244
```

La description de WHERE, SALARY >= 50000, est appelee une condition. On pourrait
effectuer le meme traitement sur des colonnes de texte :

```
SELECT EMPLOYEEIDNO
FROM EMPLOYEESTATISTICSTABLE
WHERE POSITION = 'Manager';
```

Ceci entrainera l'affichage des Numeros d'Identification de tous les Managers.
En general, avec les colonnes contenant du texte, n'utiliser que egal a ou
different de, et assurez vous que tout texte apparaissant dans l'instruction
est entoure d'apostrophes (').

Conditions plus complexes: Conditions Composees

L'operateur AND (ET) combine deux ou plusieurs conditions et n'affiche une ligne que si cette ligne satisfait TOUTES les conditions requises (i.e. ou toutes les conditions sont realisees). Par exemple, pour afficher tout le personnel gagnant plus 40 000, ecrire :

```
SELECT EMPLOYEEIDNO
FROM EMPLOYEESTATISTICSTABLE
WHERE SALARY > 40000 AND POSITION = 'Staff';
```

L'operateur OR (OU) combine deux ou plusieurs conditions mais retourne cette ligne si N'IMPORTE LAQUELLE des conditions requises est remplie. Pour visualiser tous ceux qui gagnent moins de 40 000 ou qui recoivent moins de 10 000 en participation aux benefices, utilisez la requete suivante :

```
SELECT EMPLOYEEIDNO
FROM EMPLOYEESTATISTICSTABLE
WHERE SALARY < 40000 OR BENEFITS < 10000;
```

Les operateurs AND et OR peuvent etre combines, par exemple :

```
SELECT EMPLOYEEIDNO
FROM EMPLOYEESTATISTICSTABLE
WHERE POSITION = 'Manager' AND SALARY > 60000 OR BENEFITS > 12000;
```

En premier lieu, SQL recherche les lignes pour lesquelles la valeur de la colonne salaire est superieure a 60 000 et celle de position est egale a Manager, puis, a partir de cette liste de lignes, SQL recherche alors celles qui satisfont a la condition AND (ET) precedente ou a la condition specifiant que la colonne Indemnites est superieure a 12 000. En consequence, SQL n'affiche seulement que cette seconde liste de lignes, en gardant a l'esprit que tous ceux dont les Indemnites sont superieures a 12 000 en feront partie puisque l'operateur OR (OU) inclue la ligne si l'une des conditions est verifiee. Notez en passant que l'operation AND (ET) est effectuee en premier.

Pour generaliser ce processus, SQL effectue l(es) operation(s) AND pour determiner les lignes ou l(es) operation(s) AND sont verifiees (souvenez-vous bien : toutes les conditions sont verifiees), puis ces resultats sont utilises pour tester les conditions OR, et, ne seront affichees que les lignes ou les conditions requises par l'operateur OR seront verifiees.

Pour que les OR's soient effectues avant les AND's, par exemple si vous vouliez voir une liste des employes dont le salaire est eleve (>50 000) OU beneficiant d'indemnites importantes (>10 000), ET qui soient cadres, utilisez des parentheses :

```
SELECT EMPLOYEEIDNO
FROM EMPLOYEESTATISTICSTABLE
WHERE POSITION = 'Manager' AND (SALARY > 50000 OR BENEFIT > 10000);

----------------------------------------------------------------------------
IN et BETWEEN

Une methode plus facile pour utiliser les conditions composees consiste a utiliser
IN ou BETWEEN. Par exemple, si vous desirez une liste des cadres et du personnel :

SELECT EMPLOYEEIDNO
FROM EMPLOYEESTATISTICSTABLE
WHERE POSITION IN ('Manager', 'Staff');

ou une liste de ceux dont le salaire est superieur ou egal a 30 000, mais inferieur
ou egal a 50 000, utilisez:

SELECT EMPLOYEEIDNO
FROM EMPLOYEESTATISTICSTABLE
WHERE SALARY BETWEEN 30000 AND 50000;

Pour obtenir la liste de ceux qui n'entrent pas dans ces criteres, essayez :

SELECT EMPLOYEEIDNO
FROM EMPLOYEESTATISTICSTABLE
WHERE SALARY NOT BETWEEN 30000 AND 50000;

De la meme facon, NOT IN donne la liste de toutes les lignes exclues de la liste
obtenue par l'operateur IN.

----------------------------------------------------------------------------
Utilisation de LIKE

Regardons la table EmployeeStatisticsTable, et disons que l'on veut voir tous les
gens dont le nom commence par "L"; essayons :

SELECT EMPLOYEEIDNO
FROM EMPLOYEEADDRESSTABLE
WHERE LASTNAME LIKE 'L%';

Le signe pourcentage (%) est utilise pour representer n'importe quel caractere
possible (nombre, lettre, ou signe de ponctuation) ou ensemble de caracteres
qui peut apparaitre apres le "L". Pour trouver les gens dont le Nom se termine
avec "L", utiliser '%L', ou si vous desirez le "L" au milieu du mot, essayez
'%L%'. Le symbole '%' peut etre utilise pour n'importe quel caractere, et
dont la position est relative par rapport a des caracteres donnes. NOT LIKE
affiche les lignes qui ne correspondent pas a la description donnee. Il y a
d'autres manieres d'utiliser LIKE, de meme que n'importe lesquelles des
```

```
conditions composees dont nous venons de parler, bien que cela depende du SGBD
que vous utilisez; comme d'habitude, consultez un manuel ou le gestionnaire ou
administrateur de votre systeme pour en connaitre les fonctionnalites, ou
simplement, assurez vous que ce que vous essayer de faire est possible et
autorise. Cet avertissement est aussi valable pour les fonctionnalites de SQL
exposees ci-dessous. Cette section est donnee a titre d'exemple des requetes
qui peuvent etre ecrites en SQL.
----------------------------------------------------------------------------
Jointures

Dans cette section, nous allons parler uniquement des jointures internes et
des equi-jointures, dans la mesure ou ce sont les plus utiles. Pour avoir plus
d'informations, voyez les liens sur des sites SQL au bas de cette  page.

On suggere qu'une bonne maniere de concevoir une base de donnees implique que
chaque table ne contienne des donnees qui ne concernent qu'une seule entite,
et que des informations detaillees peuvent etre obtenues, dans une base de
donnees relationnelle, en utilisant des tables supplementaires et en effectuant
une jointure.

Premierement, jetons un oeil a ces exemples de tables :

            AntiqueOwners

 OwnerIDOwnerLastName OwnerFirstName
 01     Jones         Bill
 02     Smith         Bob
 15     Lawson        Patricia
 21     Akins         Jane
 50     Fowler        Sam

---------------------------------------------------------

       Orders

 OwnerIDItemDesired
 02     Table
 02     Desk
 21     Chair
 15     Mirror

--------------------------------------

           Antiques

 SellerIDBuyerID Item
 01      50      Bed
 02      15      Table
```

```
 15      02      Chair
 21      50      Mirror
 50      01      Desk
 01      21      Cabinet
 02      21      Coffee Table
 15      50      Chair
 01      15      Jewelry Box
 02      21      Pottery
 21      02      Bookcase
 50      01      Plant Stand
```

```
-----------------------------------------------------------------------------
Cles

En premier lieu, nous allons parler du concept de cles. Une cle primaire est
une colonne ou en ensemble de colonnes qui identifie de maniere unique les
autres donnees d'une ligne donnee. Par exemple, dans la table AntiqueOwners,
la colonne OwnerID identifie de maniere unique cette ligne. Ceci signifie deux
choses: que deux lignes ne peuvent avoir le meme OwnerID, et que, meme si
deux proprietaires les memes noms et prenoms la colonne OwnerID garantit que
ces deux proprietaires ne seront pas confondus l'un avec l'autre, puisque la
colonne OwnerID unique sera utilisee a travers la base de donnees pour se
referer a un proprietaire, plutot que son nom.

Une cle externe est une colonne d'une table qui est cle primaire d'une autre
table, ce qui signifie que toutes les donnees d'une cle externe doivent avoir
des donnees correspondantes dans l'autre table, ou cette colonne est la cle
primaire.
Pour parler SGBD, cette correspondance est connue sous le nom d'integre
referentielle. Par exemple, dans la table Antiques, BuyerID et SellerID sont
tous les deux des cles externes a la cle primaire de la table AntiqueOwners
(OwnerID; pour les besoins de notre argumentation, on doit d'abord etre
reference dans la table AntiqueOwners avant de pouvoir acheter ou vendre quoi
que ce soit), puisque, dans les deux tables, les colonnes ID sont utilisees
pour identifier les proprietaires, les acheteurs ou les vendeurs, et que
OwnerID est la cle primaire de la table AntiqueOwners. En d'autres termes,
toutes ces donnees "ID" sont utilisees pour se referer aux proprietaires,
acheteurs et vendeurs sans avoir a utiliser les noms effectifs.

-----------------------------------------------------------------------------
Realisation d'une jointure

Le but de ces cles est ainsi de pouvoir mettre en relation les donnees a
travers les tables sans avoir a repeter les donnees dans chaque tables,
--c'est toute la puissance des bases de donnees relationnelles. Par exemple,
on peut trouver les noms de ceux qui ont achete une chaise sans avoir a lister
la totalite du nom de l'acheteur dans la table Antiques... vous pouvez trouver
ce nom en mettant en relation ceux qui ont achete une chaise avec les noms
```

```
dans la table AntiqueOwners en utilisant le OwnerID, qui met en relation les
donnees dans les deux tables. Pour trouver les noms de ceux qui ont achete
une chaise, utilisez la requete suivante :

SELECT OWNERLASTNAME, OWNERFIRSTNAME
FROM ANTIQUEOWNERS, ANTIQUES
WHERE BUYERID = OWNERID AND ITEM = 'Chair';

Notez ce qui suit au sujet de cette requete... notez que les deux tables mise
en jeux dans cette relation sont listees dans la clause FROM de l'instruction.
Dans la clause WHERE, notez, en premier lieu, que la partie ITEM = 'Chair' de
la clause limite la liste a ceux qui ont achete (et, dans cet exemple, de ce
fait possedent) une chaise. En second lieu, notez comment les colonnes ID sont
mises en relation d'une table a la suivante par l'utilisation de la clause
BUYERID = OWNERID. Ne seront listes que les noms  de la table AntiqueOwners
dont les ID correspondent a travers les tables et dont l'article achete est
une chaise (a cause du AND). Parce que la condition de jointure utilisee est
un signe egal, cette jointure est appelee une equi-jointures. le resultat de
cette requete donnera deux noms: Smith, Bob et Fowler, Sam.

La notation avec un point (.) fait reference a l'utilisation du nom de colonne
en suffixe du nom de table pour eviter toute ambiguite, comme par exemple:

SELECT ANTIQUEOWNERS.OWNERLASTNAME, ANTIQUEOWNERS.OWNERFIRSTNAME
FROM ANTIQUEOWNERS, ANTIQUES
WHERE ANTIQUES.BUYERID = ANTIQUEOWNERS.OWNERID AND ANTIQUES.ITEM = 'Chair';

Cependant, puisque les noms de colonnes sont differents dans chaque table,
cela n'etait pas necessaire.

----------------------------------------------------------------------------
DISTINCT et l'Elimination des Doubles

Disons que vous ne vouliez seulement que la liste des Identificateurs (ID) et
des noms des gens qui ont vendu une antiquite. Evidemment, vous ne desirez
une liste ou chaque vendeur n'apparait qu'une fois--vous ne voulez pas savoir
combien d'antiquites ont ete vendues par une personne, mais uniquement le fait
que cette personne en a vendu une (pour les comptages, voir la fonction
d'Agregation ci-dessous). Cela signifie qu'il vous faudra dire a SQL d'eliminer
les doubles des lignes des ventes, et de ne seulement lister chaque personne
qu'une fois. Pour realiser cela, utilisez le mot-cle DISTINCT.

Premierement, vous aurez besoin de faire un equi-jointures sur la table
AntiqueOwners pour obtenir les donnees concernant le Nom et le Prenom de la
personne. Cependant, gardez a l'esprit que, puisque la colonne SellerID dans
la table Antiques est une cle externe de la table AntiqueOwners, un vendeur
ne sera liste que s'il y a une ligne dans la table AntiqueOwners contenant les
ID et les noms. Nous voulons egalement eliminer les multiples occurrences du
```

SellerID dans notre liste, donc, nous utiliserons le mot-cle DISTINCT pour les colonnes ou les repetitions peuvent se produire.

Pour ajouter une difficulte, nous voulons aussi que cette liste soit classee par ordre alphabetique des Noms, puis des Prenoms (a l'interieur des noms), puis des OwnerID (a l'interieur des noms et des prenoms). Pour cela, nous utiliserons la clause ORDER BY :

```
SELECT DISTINCT SELLERID, OWNERLASTNAME, OWNERFIRSTNAME
FROM ANTIQUES, ANTIQUEOWNERS
WHERE SELLERID = OWNERID
ORDER BY OWNERLASTNAME, OWNERFIRSTNAME, OWNERID;
```

Dans cet exemple, puisque tout le monde a vendu un article, nous aurons une liste de tous les proprietaires, classes par ordre alphabetique sur les noms. Pour reference ulterieure (au cas ou quelqu'un le demande), ce type de jointure est considere comme appartenant a la categorie des jointures internes.

Alias et In/Sous-requetes

Dans cette section, nous parlerons des Alias, In et de l'utilisation des sous-requetes, et de la maniere de les utiliser dans un exemple de 3-table. En premier lieu, regardez cette requete qui imprime le nom des proprietaires qui ont passe une commande et la nature de cette commande, en ne listant seulement que les commandes qui peuvent etre satisfaites (c'est a dire qu'il y a un vendeur qui possede l'article commande) :

```
SELECT OWN.OWNERLASTNAME Last Name, ORD.ITEMDESIRED Item Ordered
FROM ORDERS ORD, ANTIQUEOWNERS OWN
WHERE ORD.OWNERID = OWN.OWNERID
AND ORD.ITEMDESIRED IN

     (SELECT ITEM
     FROM ANTIQUES);
```

Ce qui donne :

```
Last name Item Ordered
--------- ------------
Smith     Table
Smith     Desk
Akins     Chair
Lawson    Mirror
```

Il y a plusieurs choses a noter a propos de cette requete :

1. Tout d'abord, les mots "Last Name" et "Item Ordered" dans les lignes

SELECT donnent les en-tetes du rapport.

2. Les mots OWN et ORD sont des alias; ce sont de nouveaux noms pour les deux tables donnees dans la clause FROM qui sont utilises comme prefixes pour toutes les notations point (.) de noms de colonnes dans les requetes (voir ci-dessus). Ceci elimine les risques ambiguite, specialement dans l'equi-jointure de la clause WHERE ou les deux tables ont une colonne nommee OwnerID, et cette notation point (.) precise a SQL que nous designons deux OwnerID differents de deux tables differentes.
3. Notez que la table des commandes (ORDERS) est indiquee la premiere dans la clause FROM; ceci apporte la certitude que la liste sera realisee a partir de cette table, et que la table AntiqueOwners est utilisee uniquement pour obtenir les informations detaillees (Last Name / Nom).
4. Plus important, la clause AND dans la clause WHERE (OU) force l'utilisation de la Sous-requete In ("= ANY" ou "= SOME" sont deux utilisations equivalentes de IN). Cela entraine que la sous-requete est effectuee, retournant une liste de tous les articles (Items) appartenant a la table Antiques, comme s'il n'y avait pas de clause WHERE (OU). Donc, pour lister une ligne de la table ORDERS, le ItemDesired (article_desire) doit etre dans la liste retournee des articles appartenant a la table Antiques, et donc un article ne sera liste que si la commande ne peut etre honoree que par un autre proprietaire. On peut se le representer comme ceci: la sous-requete retourne un ensemble d'articles (Items) auquel chaque ItemDesired (Article_Desire) dans la table des commandes (ORDERS) est compare; la condition IN (DANS) n'est vraie que si le ItemDesired appartient a l'ensemble provenant de la table ANTIQUES.
5. Notez egalement, comme c'est le cas ici, qu'il y a un objet ancien pour chaque demande, ce qui, evidemment, n'est pas toujours le cas... De plus, notez aussi que, lorsque IN, "= ANY", ou "= SOME" est utilise, ces mots-cles font reference a toutes les lignes qui conviennent, pas aux colonnes... c'est a dire que vous ne pouvez pas mettre de multiples colonnes dans un clause SELECT de sous-requete, pour tenter de faire correspondre la colonne de la clause WHERE externe avec l'une des multiples valeurs de colonnes possibles de la sous-requete; une seule colonne peut etre indiquee dans la sous-requete, et la correspondance possible provient de multiples valeurs de lignes, dans cette colonne unique, et non pas l'inverse.

Ouf! Ce sera tout sur ce sujet des requetes SELECT complexes pour l'instant. Maintenait, voyons d'autres instructions SQL.

Instructions SQL Diverses

Fonctions d'Agregation

Je parlerai de cinq fonctions d'agregation importantes: SUM, AVG, MAX, MIN, et COUNT. On les appelle fonctions d'agregation parce qu'elles resument les resultats d'une requete, plutot que de donner une liste de toutes les lignes.

```
   * SUM () donne la somme, pour une colonne donnee, de toutes les lignes qui
satisfont aux conditions requises, et ou la colonne donnee est numerique.
   * AVG () donne la moyenne de la colonne donnee.
   * MAX () donne la plus grande valeur dans la colonne donnee.
   * MIN () donne la plus petite valeur dans la colonne donnee.
   * COUNT(*) donne le nombre de lignes qui satisfont aux conditions.

En utilisant les tables du debut de ce document, regardons trois exemples :

SELECT SUM(SALARY), AVG(SALARY)
FROM EMPLOYEESTATISTICSTABLE;

Cette requete donne la somme des de salaires tous les salaries presents dans
la table et le salaire moyen.

SELECT MIN(BENEFITS)
FROM EMPLOYEESTATISTICSTABLE
WHERE POSITION = 'Manager';

Cette requete donne le chiffre de la colonne indemnites le plus faible des
employes qui sont Managers, cette valeur est 12 500.

SELECT COUNT(*)
FROM EMPLOYEESTATISTICSTABLE
WHERE POSITION = 'Staff';

Cette requete vous donne le nombre d'employes ayant le statut de cadre (Staff,
i.e. 3).

----------------------------------------------------------------------------
Les Vues

En SQL, vous pouvez (verifiez aupres de votre Administrateur de Base de Donnees,
DBA) avoir acces a la creation de vues par vous-meme. Une vue vous permet d'affecter
les resultats d'une requete a une nouvelle table personnelle, que vous pourrez
utiliser dans d'autres requetes, pour laquelle vous donnez le nom de la vue dans
votre clause FROM. Quand vous accedez a une vue, la requete qui est definie dans
l'instruction de creation de la vue est effectuee (generalement), et les resultats
de cette requete ont la meme allure qu'une autre table dans la requete que vous
avez ecrit en invoquant la vue. Par exemple, pour creer une vue :

CREATE VIEW ANTVIEW AS SELECT ITEMDESIRED FROM ORDERS;

Maintenant, ecrivons une requete utilisant cette vue comme une table, ou la
table est seulement une liste de tous les articles desires (ITEMDESIRED) de
la table ORDERS :
```

```
SELECT SELLERID
FROM ANTIQUES, ANTVIEW
WHERE ITEMDESIRED = ITEM;

Cette table montre tous les Identifiants de Vendeurs (SellerID) de la table
ANTIQUES ou l'article (Item) dans cette table apparait dans la vue ANTVIEW,
qui consiste justement en tous les Articles Desires (Items Desired) dans la
table ORDERS. La liste est cree en parcourant les articles AntiquesItems un
par un jusqu'a ce qu'il y ait correspondance avec la vue ANTVIEW. Les vues
peuvent etre utilisees pour restreindre les acces a la base de donnees,
ainsi que, dans ce cas, pour simplifier une requete complexe.

----------------------------------------------------------------------------
Creation de Nouvelles Tables

Toutes les tables, dans une base de donnees doivent etre creees a un certain
moment... voyons comment mous pourrions creer la table des commandes (ORDERS) :

CREATE TABLE ORDERS
(OWNERID INTEGER NOT NULL,
ITEMDESIRED CHAR(40) NOT NULL);

Cette instruction donne un nom a la table et renseigne le SGBD sur la nature
de chaque colonne de la table. Veuillez noter que cette instruction utilise
des types de donnees generiques, et que les types de donnees peuvent etre
differents, selon le SGBD que vous utilisez. Comme d'habitude, verifiez vos
conditions locales. Voici quelques types de donnees generiques courants:

   * Char(x) - Une colonne de caracteres, ou x est un nombre indiquant le
     nombre maximum de caracteres permis (taille maximale) de la colonne.
   * Integer - Une colonne de nombres entiers, positifs ou negatifs.
   * Decimal(x, y) - Une colonne de nombre decimaux, ou x est la taille
     maximum, en digits, des nombres decimaux dans cette colonne, et y le
     nombre maximal de digits autorises apres la virgule. Le nombre maximal
     (4,2) sera 99.99.
   * Date - Une colonne de date dans un format specifique au SGBD.
   * Logical - Une colonne qui ne peut contenir que deux valeurs: VRAI ou FAUX.

Autre remarque, l'indication NOT NULL (non nul) signifie que la colonne doit
avoir une valeur pour chacune des lignes. Si l'on avait utilise NULL (nul),
cette colonne peut etre laissee vide dans certaines lignes.

----------------------------------------------------------------------------
Modification des tables

Ajoutons une colonne a la table ANTIQUES pour permettre la saisie du prix d'un
article donne :
```

```
ALTER TABLE ANTIQUES ADD (PRICE DECIMAL(8,2) NULL);

On verra plus tard comment les donnees pour cette nouvelle colonne peuvent
etre mises a jour ou ajoutees.

----------------------------------------------------------------------------
Ajout de Donnees

Pour inserer des lignes dans une table, faites ce qui suit :

INSERT INTO ANTIQUES VALUES (21, 01, 'Ottoman', 200.00);

Ceci insere les donnees dans la table, en tant que nouvelle ligne, colonne
par colonne, dans un ordre predefinit. Au lieu de cela, changeons cet ordre
et laissons le Prix vide:

INSERT INTO ANTIQUES (BUYERID, SELLERID, ITEM)
VALUES (01, 21, 'Ottoman');

----------------------------------------------------------------------------
Suppression de donnees

Supprimons cette nouvelle ligne de la base de donnees :

DELETE FROM ANTIQUES
WHERE ITEM = 'Ottoman';

Mais s'il y a une autre ligne qui contient 'Ottoman', cette ligne sera
egalement supprimee. Supprimons toutes les lignes (une, dans ce cas) qui
contient les donnees specifiques que nous avons ajoute plus
 tot :

DELETE FROM ANTIQUES
WHERE ITEM = 'Ottoman' AND BUYERID = 01 AND SELLERID = 21;

----------------------------------------------------------------------------
Mise a Jour des Donnees

Mettons un Prix a jour dans une ligne qui n'en contient pas encore :

UPDATE ANTIQUES SET PRICE = 500.00 WHERE ITEM = 'Chair';

Ceci met le Prix de toutes les Chaises (Chair) a 500.00. Comme indique
ci-dessus, conditions WHERE supplementaires, utilisation de AND, il faut
utiliser ces conditions pour limiter la mise a jour a des lignes specifiques.
De plus, des colonnes supplementaires peuvent etre renseignees en separant
les instructions "egal" par des virgules.
```

```
----------------------------------------------------------------------------
Considerations Diverses

Index

Les Index permettent a un SGBD d'acceder au donnees plus rapidement (veuillez
noter que cette fonctionnalite est non-standard/indisponible sur certains
systemes). Le systeme cree une structure de donnee interne (l'index) qui
entraine une selection de lignes beaucoup plus rapide, quand la selection
est basee sur des colonnes indexees. Cet index indique au SGBD ou se trouve
une certaine ligne dans une table etant donne une valeur de colonne indexee,
exactement comme l'index d'un livre vous indique a quelle page un mot donne
se trouve. Creons un index pour le OwnerID dans la colonne AntiqueOwners :

CREATE INDEX OID_IDX ON ANTIQUEOWNERS (OWNERID);

Maintenant sur les noms:

CREATE INDEX NAME_IDX ON ANTIQUEOWNERS (OWNERLASTNAME, OWNERFIRSTNAME);

Pour etre debarrasse d'un index, supprimez le :

DROP INDEX OID_IDX;

A propos, vous pouvez aussi bien "supprimer" une table (attention!--cela
signifie que votre table est detruite). Dans le second exemple, l'index est
construit a partir des deux colonnes, agregees ensembles--un comportement
bizarre peut resulter de cette situation... verifiez dans votre manuel avant
d'effectuer une telle operation.

Quelques SGBD n'imposent pas l'utilisation de cles primaires; en d'autres
termes, l'unicite d'une colonne n'est pas imposee automatiquement. Cela
signifie que, par exemple, j'aurais pu essayer d'inserer une autre ligne
dans la table  AntiqueOwners avec un OwnerID de 02, quelques systemes me
permettent de faire cela, bien qu'il ne le faille pas, puisque cette colonne
est supposee etre unique dans cette table (chaque valeur de ligne est supposee
etre differente). Une maniere de contourner cela est de creer un index unique
sur la colonne que nous souhaitons voir etre la cle primaire pour forcer le
systeme a interdire les doubles :

CREATE UNIQUE INDEX OID_IDX ON ANTIQUEOWNERS (OWNERID);

----------------------------------------------------------------------------
GROUP BY et HAVING

Une utilisation speciale de la clause GROUP BY est l'association d'une fonction
agregee (specialement COUNT; qui compte le nombre de lignes dans chaque groupe)
avec des groupes de lignes. Premierement, supposons que la table ANTIQUES
```

```
possede la colonne Prix (Price)t, et que chaque ligne contienne une valeur
dans cette colonne. Nous voulons voir le prix de l'article le plus cher achete
par chaque proprietaire. Il nous faut donc dire a SQL de grouper les achats de
chacun des proprietaires, et de nous dire le prix d'achat maximum :

SELECT BUYERID, MAX(PRICE)
FROM ANTIQUES
GROUP BY BUYERID;

Maintenant, disons que nous ne voulons voir que le prix maximum si l'achat
depasse 1000, nous devrons utiliser la clause HAVING :

SELECT BUYERID, MAX(PRICE)
FROM ANTIQUES
GROUP BY BUYERID
HAVING PRICE > 1000;

----------------------------------------------------------------------------
Sous-requetes Supplementaires

Un autre usage commun des sous-requetes amene a l'utilisation des operateurs
pour permettre a une condition WHERE d'inclure la sortie SELECT d'une
sous-requete. En premier, demandons la liste des acheteurs ayant achete un
article cher (le prix de cet article est superieur de 100 au prix moyen de tous
les articles achetes) :

SELECT OWNERID
FROM ANTIQUES
WHERE PRICE >

     (SELECT AVG(PRICE) + 100
     FROM ANTIQUES);

La sous-requete calcule le Prix moyen, ajoute 100, et, et en utilisant ce
chiffre on imprime un OwnerID pour chaque article coutant plus que ce chiffre.
On peut utiliser DISTINCT OWNERID, pour eliminer les doubles.

Listons les Noms (Last Names) de ceux qui sont dans la table AntiqueOwners,
SEULEMENT s'ils ont achete un article :

SELECT OWNERLASTNAME
FROM ANTIQUEOWNERS
WHERE OWNERID =

     (SELECT DISTINCT BUYERID
     FROM ANTIQUES);

Cette sous-requete retourne une liste des acheteurs, et le Nom du proprietaire
```

d'un objet ancien est imprime seulement si l'identificateur du Proprietaire (Owner's ID) dans la liste obtenue par la sous-requete (appelee quelquefois liste des candidats).

Voici un exemple de mise a jour: nous savons que la personne qui a achete la bibliotheque a un Prenom errone dans la base de donnees... Ce devrait etre John :

```
UPDATE ANTIQUEOWNERS
SET OWNERFIRSTNAME = 'John'
WHERE OWNERID =

     (SELECT BUYERID
     FROM ANTIQUES
     WHERE ITEM = 'Bookcase');
```

Tout d'abord, la sous-requete trouve le BuyerID pour la (les) personne(s) qui a (ont) achete(s) la bibliotheque, puis la requete externe met a jour son Prenom.

Souvenez vous de cette regle a propos des sous-requetes: quand vous avez une sous-requete faisant partie d'une condition WHERE, la clause SELECT dans la sous-requete doit avoir des colonnes qui correspondent en nombre et en type a celle de la clause WHERE de la requete externe. En d'autres termes, si vous avez "WHERE ColumnName = (SELECT...);", le SELECT ne peut faire reference qu'a une seule colonne, pour pouvoir correspondre a la clause WHERE externe, et elles doivent etre du meme type (les deux etant soit entiers, soit chaines de caracteres, etc.).

EXISTS et ALL

EXISTS utilise une sous-requete comme condition, ou cette condition est Vraie si la sous-requete retourne au moins une ligne et Fausse si la sous-requete n'en retourne aucune; c'est une fonctionnalite qui n'est pas intuitive et n'est utilisee que dans peu de cas. Cependant, si un client eventuel voulait consulter la liste des proprietaires pour voir s'il y a des chaises (Chairs), essayez :

```
SELECT OWNERFIRSTNAME, OWNERLASTNAME
FROM ANTIQUEOWNERS
WHERE EXISTS

     (SELECT *
     FROM ANTIQUES
     WHERE ITEM = 'Chair');
```

S'il y a des Chaises (Chair) dans une colonne de la table ANTIQUES, la

```
sous-requete renverra une ou plusieurs lignes, rendant la clause EXISTS
vraie, ce qui amenera SQL a donner une liste des proprietaires dans
ANTIQUEOWNERS. S'il n'y avait eu aucune Chaise, la requete externe n'aurait
pas renvoye de ligne.

ALL est une autre fonctionnalite peu commune, et en general, on peut realiser
une requete avec ALL de manieres differentes et eventuellement plus simples;
regardons cet exemple de requete :

SELECT BUYERID, ITEM
FROM ANTIQUES
WHERE PRICE >= ALL

     (SELECT PRICE
     FROM ANTIQUES);

Ceci va nous retourner l'article de prix le plus eleve (ou plus d'un article
s'il y a des ex-aequo), et son acheteur. La sous-requete renvoie la liste de
tous les Prix (PRICE) dans la table ANTIQUES, puis la requete externe examine
chaque ligne de la table ANTIQUES et si son Prix est superieur ou egal a chacun
(ou ALL, TOUS) des Prix de cette liste, il est affiche, donnant ainsi l'article
de prix le plus eleve. La raison pour laquelle ">=" doit etre utilise est que
l'article de prix le plus eleve sera egal au prix le plus eleve, puisque cet
Article est dans la liste de Prix.

----------------------------------------------------------------------------
UNION et Jointure Externe

Il y a des occasions  ou vous pouvez desirer voir ensembles les resultats de
requetes multiples, leurs sorties etant combinees; pour cela utilisez UNION.
Pour fusionner la sortie des deux requetes suivantes, en affichant l'identificateur
de tous les Acheteurs plus tous ceux qui ont passe une Commande :

SELECT BUYERID
FROM ANTIQUEOWNERS
UNION
SELECT OWNERID
FROM ORDERS;

Il faut noter que SQL necessite que les types de donnees des listes des clauses
SELECT correspondent colonne par colonne. Dans cet exemple, les identificateurs
BuyerID et OwnerID sont du meme type (entier). Notez egalement que SQL
effectue automatiquement une elimination des doubles quand on utilise la clause
UNION (comme si c'etaient deux "ensembles"); dans une requete simple, il faut
utiliser la clause DISTINCT.

La jointure externe est utilisee quand une requete de jointure est "unifiee",
les lignes n'etant pas incluses dans la jointure. Ceci est particulierement
```

```
utile si des "balises" de type constante texte sont inclus. D'abord,
regardez la requete :

SELECT OWNERID, 'is in both Orders & Antiques'
FROM ORDERS, ANTIQUES
WHERE OWNERID = BUYERID
UNION
SELECT BUYERID, 'is in Antiques only'
FROM ANTIQUES
WHERE BUYERID NOT IN

     (SELECT OWNERID
     FROM ORDERS);

La premiere requete effectue une jointure pour lister tous les proprietaires
qui sont dans les deux tables, et met une balise apres l'identificateur (ID)
en inserant le texte correspondant a la balise. La clause UNION fusionne cette
liste avec la liste suivante. La seconde liste est generee premierement en
listant les identificateurs (ID) qui ne sont pas dans la table ORDERS, c'est a
dire en generant une liste des ID exclus de la requete de jointure.. Puis,
chaque ligne de la table ANTIQUES est analysee, et, si l'identifiant de
l'acheteur (BuyerID) n'est pas dans cette liste d'exclusion, il est liste avec
le texte correspondant a sa balise. Il y aurait peut-etre une meilleure
maniere de creer cette liste, mais c'est difficile de generer des balises
informationnelles.

Ce concept est utile dans des situations ou une cle primaire est en relation
avec une cle externe, et ou la valeur de la cle externe est NULLE (NULL) pour
quelques cles primaires. Par exemple, dans une table, la cle primaire est
vendeur, et dans une autre table client, avec le vendeur enregistre dans la
meme ligne. Cependant, si un vendeur n'a pas de clients, le nom de cette
personne n'apparaitra pas dans la table des clients. Une jointure externe sera
utilisee pour imprimer une liste de tous les vendeurs, avec leurs clients, que
le vendeur ait un client ou pas--c'est a dire qu'il n'y aura pas de client
imprime (valeur logique NULL) si le vendeur n'a pas de client, mais existe dans
la table des vendeurs. Autrement, le vendeur sera liste avec chaque client.

ASSEZ DE REQUETES!!! Qu'est-ce que vous dites?...Eh bien, maintenant voyons
quelque chose de completement different...

----------------------------------------------------------------------------
SQL incorpore--un vilain exemple (n'ecrivez pas un programme comme cela... il
est la UNIQUEMENT a titre d'exemple)

/* -Voici un exemple de programme qui utilise le SQL incorpore (Embedded
    SQL). Le SQL incorpore permet aux programmeurs de se connecter a une
    base de donnees et d'inclure du code SQL en plein programme, et ainsi,
    leurs programmes peuvent utiliser, manipuler, et traiter les donnees
```

```
    d'une base de donnees.
   -Cet exemple de Programme C (qui utilise du SQL incorpore) doit imprimer
    un rapport.
   -Les instructions SQL devront etre precompilees avant d'effectuer la
    compilation normale.
   -Si vous utilisez un langage different les parties EXEC SQL seront les
    memes (standard), mais le code C qui les entourent devront etre
    modifiees, y compris les declarations de variables hotes.
   -Le SQL incorpore differe de systeme a systeme, aussi, encore une fois,
    verifiez la documentation locale, specialement les declarations de
    variables et les procedures de connexion pour lesquelles le reseau, le
    SGBD, et le systeme d'exploitation sont cruciaux. */

/***************************************************/
/* CE PROGRAMME N'EST PAS COMPILABLE OU EXECUTABLE */
/* IL EST UNIQUEMENT DONNE A TITRE D'EXEMPLE       */
/***************************************************/

#include <stdio.h>

/* Section de declaration des variables hotes; ce seront les variables
utilisees par votre programme, mais egalement celles utilisees par SQL
pour y mettre ou y lire des valeurs,. */
EXEC SQL BEGIN DECLARE SECTION;
  int BuyerID;
  char Prenom[100], Nom[100], Item[100];
EXEC SQL END DECLARE SECTION;

/* Cette section,insere les variables SQLCA, de facon a pouvoir tester les erreurs. */
EXEC SQL INCLUDE SQLCA;

main() (

/* Ceci est une possibilite pour se connecter a la base de donnees */
EXEC SQL CONNECT UserID/Password;

/* Cette partie de code soit vous indique que vous etes connecte  soit
teste si un code erreur a ete genere, signifiant que la connexion etait
incorrecte ou impossible. */
  if(sqlca.sqlcode) (
    printf(Printer, "Erreur de connexion au serveur de base de donnees.\n");
    exit();
  )
  printf("Connecte au serveur de base de donnees.\n");

/* Ici, on declare un "Curseur". C'est utilise lorsqu'une requete retourne
   plus d'une ligne, et qu'on doit effectuer un traitement sur chaque ligne
   obtenue de la requete. Je vais utiliser pour le rapport, chaque ligne obtenue
```

```
   par cette requete. Ensuite, on utilisera "FETCH" (va chercher) pour recuperer
   les lignes, une par une, mais pour que la requete soit effectivement executee,
   il faut utiliser l'instruction "OPEN". La "Declaration" (Declare) sert
   uniquement a construire la requete. */
EXEC SQL DECLARE ItemCursor CURSOR FOR
  SELECT ITEM, BUYERID
  FROM ANTIQUES
  ORDER BY ITEM;
EXEC SQL OPEN ItemCursor;

/*
 +-- Inserer ici un test d'erreur similaire au precedent si vous le desirez --+
*/

/* L'instruction FETCH insere les valeurs de la ligne suivante respectivement
dans chacune des variables hotes. Cependant un "priming fetch" (technique de
programmation) doit etre execute en premier. Lorsque le curseur n'a plus de
donnees un code (sqlcode) est genere nous permettant de sortir de la boucle.
Notez que, pour des raisons de simplicite, on abandonne la boucle pour n'importe
quel sqlcode, meme s'il correspond a un code erreur. Autrement, il faut
effectuer un test d'erreur specifique. */
EXEC SQL FETCH ItemCursor INTO :Item, :BuyerID;
  while(!sqlca.sqlcode) (

/* Nous effectuerons egalement deux traitements pour chaque ligne. Premierement,
augmenter le prix de 5 (retribution du marchand) et ensuite, lire le nom de
l'acheteur pour le mettre dans le rapport. Pour faire cela, j'utiliserai les
instructions Update et Select, avant d'imprimer la ligne sur l'ecran. La mise a
jour suppose, cependant, qu'un acheteur donne n'a achete qu'un seul article,
ou, sinon, le prix sera augmente de trop nombreuses fois. Sinon, il aurait
fallu utiliser une logique "RowID" (consulter la documentation). De plus,
notez la presence du signe : (deux points) qui doit etre place devant les
noms de variables hotes quand elles sont utilisees dans des instructions
SQL. */

EXEC SQL UPDATE ANTIQUES
  SET PRICE = PRICE + 5
  WHERE ITEM = :Item AND BUYERID = :BuyerID;

EXEC SQL SELECT OWNERFIRSTNAME, OWNERLASTNAME
  INTO :Prenom, :Nom
  FROM ANTIQUEOWNERS
  WHERE BUYERID = :BuyerID;

    printf("%25s %25s %25s", Prenom, Nom, Item);

/* Rapport grossier--uniquement a but d'exemple! Aller chercher la ligne suivante */
EXEC SQL FETCH ItemCursor INTO :Item, :BuyerID;
```

```
  )

/* Fermer le curseur, enregistrer les modifications (voir ci-dessous),
et quitter le programme. */
EXEC SQL CLOSE DataCursor;
EXEC SQL COMMIT RELEASE;
  exit();
)

----------------------------------------------------------------------------
Questions courantes sur SQL--Sujets avances
(Consulter les liens FAQ pour en avoir plusieurs autres)

  1. Pourquoi ne puis-je pas demander uniquement les trois premieres lignes
     d'une table? --Parce que, dans une base de donnees relationnelle, les
     lignes sont inserees sans aucun ordre particulier, c'est a dire que le
     systeme les inserent dans un ordre arbitraire; de ce fait, vous ne pouvez
     demander des lignes qu'en utilisant des fonctionnalites SQL valides,
     telles que ORDER BY, etc.
  2. Que sont ces DDL et DML dont j'entends parler ? --DDL (Data
     Definition Language - Langage de Definition de Donnees) fait reference a
     (en SQL) l'instruction de Creation de Table (Create Table)...DML (Data
     Manipulation Language - Langage de Manipulation de Donnees) fait reference
     aux instructions Select, Update, Insert, et Delete.
  3. Les tables des base de donnees ne sont elles pas simplement des fichiers?
     --Eh bien, les SGBD stockent les donnees dans des fichiers declares par
     le gestionnaire du systeme avant que de nouvelles tables ne soient creees
     (sur les grands systemes), mais le systeme stocke les donnees dans un
     format special, et peut repartir les donnees d'une table dans plusieurs
     fichiers. Dans le monde des bases de donnees, un ensemble de fichiers
     crees pour une base de donnees est appele un "espace de tables". En
     general, sur les petits systemes, tout ce qui concerne une base de donnees
     (definitions et toutes les tables de donnees) est stocke dans un seul
     fichier.
  4. (Question en relation avec la precedente) Les bases de donnees ne sont
     elles pas simplement que des tableurs? --Non, et ceci pour deux raisons.
     Premierement, les tableurs peuvent avoir des donnees dans une cellule, mais
     une cellule est plus qu'une intersection ligne-colonne. Selon votre
     tableur, une cellule peut aussi contenir des formules et un formatage,
     ce que les bases de donnees (actuelles) ne permettent pas. Deuxiemement,
     les cellules dependent souvent des donnees presentes dans d'autres
     cellules. Dans les bases de donnees, les "cellules" sont independantes,
     sauf que les colonnes sont en relation logique (heureusement: ensembles,
     une ligne et une colonne decrivent une entite), et, en dehors des
     contraintes de cle primaire et de cle externe, chaque ligne d'une table
     est independante des autres.
  5. Comment puis-je importer un fichier texte de donnees dans une base de
     donnees? --Eh bien, vous ne pouvez pas le faire directement...il vous faut
```

```
   utiliser un programme utilitaire, tel que le SQL*Loader pour Oracle,
   ou ecrire un programme pour charger les donnees dans la base de donnees.
   Un programme pour realiser cela lit simplement chaque enregistrement du
   fichier texte, le separe en colonnes, et effectue une insertion (INSERT)
   dans la base de donnees.
6. Qu'est-ce qu'un schema? --Un schema est un ensemble logique de tables, tel
   que la base de donnees ANTIQUES ci-dessus...habituellement, on s'y refere
   simplement en tant que "base de donnees", mais une base de donnees peut
   contenir plus d'un schema. Par exemple, un schema en etoile est un ensemble
   de tables ou une table centrale, de grande taille, contient toutes les
   informations importantes, et est liee, via des cles externes, a des tables
   de dimension qui contiennent l'information detaillee, et qui peuvent etre
   utilisees dans une jointure pour creer des rapports detailles.
7. Quels conseils generaux pourriez vous donner pour rendre mes requetes SQL
   et mes bases de donnees meilleures et plus rapides (optimisees)?
      o Vous devriez essayer, si vous le pouvez, d'eviter, dans les clauses
        SELECT, des expressions telles que SELECT ColumnA + ColumnB, etc.
        L'optimiseur de requetes de la base de donnees, partie du SGBD qui
        determine la meilleure maniere d'extraire les donnees requises de la
        base de donnees elle-meme, traite les expressions d'une facon telle
        que cela demande en general plus de temps pour extraire les donnees
        que si les colonnes avaient ete selectionnees normalement, et que
        l'expression elle-meme calculee par programme.
      o Minimisez le nombre de colonnes incluses dans une clause Group By.
      o Si vous effectuez une jointure, tachez d'avoir les colonnes, de cette
        jointure, indexees (dans les deux tables).
      o Si vous avez un doute, creez un index.
      o A moins que vous ne fassiez de multiple comptages ou une requete
        complexe, utilisez COUNT(*) (le nombre de lignes sera genere par la
        requete) plutot que COUNT(Column_Name).
8. Qu'est-ce que la Normalisation? --La Normalisation est une technique de
   conception de base de donnees qui suggere qu'un certain critere doit etre
   pris en compte quand on definit l'organisation des tables (prise de
   decision sur le nombre de colonnes de chaque table, et creation de la
   structure des cles), ou l'idee est d'eliminer la redondance a travers les
   tables des donnees qui ne sont pas des cles. On parle de la normalisation
   en s'y referant habituellement en termes de "formes", et j'introduirais ici
   uniquement les trois premieres, bien qu'il soit tout a fait courant d'en
   utiliser d'autres formes, plus avancees (quatrieme, cinquieme, Boyce-Codd;
   consultez la documentation).

   La Premiere Forme Normale consiste a placer les donnees dans des tables
   separees ou les donnees dans chaque tables sont de type similaire, et a
   donner a chaque table une cle primaire.

   Mettre les donnees en Seconde Forme Normale consiste a mettre les donnees
   dans les tables ou elle ne dependent uniquement que d'une partie de la cle.
   Par exemple, si j'avais laisse les noms des proprietaires d'objets anciens
```

dans la table des articles, ce n'aurait pas ete une seconde forme normale puisque les donnees auraient ete redondantes; le nom aurait du etre repete pour chaque article possede, donc, les noms ont ete places dans leur propre table. Les noms en eux-memes n'ont rien a voir avec les articles, seules les identites des acheteurs et des vendeurs sont concernees.

La Troisieme Forme Normale consiste a se debarrasser, dans une table, a tout ce qui ne depend pas uniquement de la cle primaire. On met uniquement l'information qui depend de la cle, et l'on deplace, dans d'autres tables, tout ce qui est independant de la cle primaire, et l'on cree des cles primaires pour les nouvelles tables.

Il y a une certaine forme de redondance dans chaque forme, et si les donnees sont en 3NF (abrege pour 3ieme forme normale), elles sont deja en 1NF et 2NF. En terme d'organisation des donnees, organisez vos donnees de facon que les colonnes qui ne sont pas des cles primaires dependent seulement de la cle primaire entiere. Si vous jetez un oeil sur la base de donnees en exemple, vous verrez que, lorsque vous naviguez a travers la base de donnees, c'est au travers de jointures qui utilisent les colonnes de cle commune.

Deux autre points importants dans la conception de bases de donnees sont l'utilisation de noms significatifs, bien choisis, coherents et logiques pour les tables et les colonnes et l'utilisation de noms significatifs pour la base de donnees elle-meme. Sur le dernier point, ma base de donnees peche, puisque j'utilise des codes numeriques comme identificateurs. C'est en general bien meilleur d'utiliser, si possible, des cles qui ont, en elles-memes, un sens; par exemple, une meilleure cle pourrait consister des quatre premieres lettres du nom et de la premiere initiale du prenom, comme JONEB pour Bill Jones (ou pour eviter les doubles, ajoutez-y des nombres a la fin pour differencier deux ou plusieurs personnes ayant le meme nom, ainsi, on pourrait essayer JONEB1, JONEB2, etc.).

9. Quelle est la difference entre une requete simple ligne et une requete multi-lignes et pourquoi est-ce important de connaitre cette difference? --Premierement, pour parler de ce qui est evident, une requete simple ligne est une requete qui retourne une ligne unique comme resultat, et le resultat d'une requete multi-lignes est constitue de plusieurs lignes. Qu'une requete retourne une ligne ou plusieurs depend completement de la conception (ou schema) des tables qui constituent la base de donnees. Assurez-vous d'inclure suffisamment de conditions, et structurez vos instructions SQL correctement, de facon a obtenir le resultat desire (soit une ligne, soit plusieurs). Par exemple, si vous vouliez etre sur qu'une requete sur la table AntiqueOwners ne retourne qu'une ligne, employez une condition d'egalite sur la cle primaire, OwnerID.

   Trois raisons, concernant l'importance de ce sujet, viennent immediatement a l'esprit.

Premierement, l'obtention de plusieurs lignes alors que vous n'en attendez qu'une, ou vice-versa, peut signifier que la requete est erronee, que la base de donnees est incomplete, ou simplement que vous decouvrez quelque chose de nouveau concernant vos donnees.
Deuxiemement, si vous utilisez une instruction de mise a jour (UPDATE) ou de suppression (DELETE), il vaudrait mieux vous assurer que l'instruction que vous ecrivez effectue bien l'operation sur la ligne desiree (ou les lignes)...ou sinon, vous pourriez supprimer ou mettre a jour plus de lignes que vous ne le desirez.
Troisiemement, il faut soigneusement penser au nombre de lignes qui seront retournees pour toutes les requetes redigees en SQL incorpore. Si vous ecrivez une requete simple ligne, une seule instruction SQL peut suffire pour satisfaire a la logique du programme. D'un autre cote, si votre requete retourne de multiples lignes, il vous faudra utiliser l'instruction FETCH, et tres certainement quelque chose comme une structure de boucle sera necessaire dans votre programme pour traiter chaque ligne retournee par la requete.

10. Que sont les relations? --C'est une autre question de conception...le terme "relation" fait habituellement reference aux relations entre cles primaires et externes entre les tables. Ce concept est important parce que, quand les tables d'une base de donnees relationnelle sont concues, ces relations doivent etre definies parce que cela determine quelles colonnes sont ou ne sont pas des cles primaires ou externes. Vous avez peut-etre entendu parler des diagrammes Entites-Relations, qui sont une representation graphique des tables dans les schema de la base de donnees. Voyez l'exemple de diagramme a la fin de cette section ou consultez quelques sites indiques ci-dessous concernant ce sujet, car il y a de nombreuses manieres de dessiner les diagrammes E-R. Mais d'abord, jetons un oeil a chaque type de relation...

    Une relation 1-a-1 (ou 1:1, ou 1-1) signifie que vous avez une colonne cle primaire et que chaque cle primaire est en relation avec une cle externe. Par exemple, dans le premier exemple, dans la table des adresses des employes TableAdresseEmploye nous avons une colonne numero d'identification de l'employe (EmployeeIDNo). Donc, la table TableAdresseEmploye est en relation avec la table EmployeeStatisticsTable (deuxieme exemple de table) par l'intermediaire du numero EmployeeIDNo. Plus precisement, chaque employe, de la table TableAdresseEmploye possede des statistiques (une ligne de donnees) dans la table EmployeeStatisticsTable. Meme si c'est un exemple invente, c'est une relation "1-1". Inscrivez en caracteres gras le "has" ("a" ou "possede")...quand on decrit une relation, il est important de decrire une relation en utilisant un verbe.

    Les deux autres types de relations peuvent ou pas utiliser une cle primaire logique et des contraintes par rapport aux cles externes...cela depend strictement des souhaits du concepteur. La premiere de ces relations est la relation un-a-plusieurs ("1-M").
    Cela signifie que pour chaque valeur d'une colonne dans une table, il y a

une ou plusieurs valeurs correspondantes dans une autre table. Des contraintes de cle peuvent etre ajoutees au modele, ou eventuellement une colonne d'identification peut etre utilisee pour etablir une relation. Un exemple serait que pour chaque OwnerID dans la table AntiqueOwners, il y ait un ou plusieurs (la valeur zero est egalement autorisee) articles (Items) achetes dans la table ANTIQUES (verbe: acheter).

Finalement, la relation plusieurs-a plusieurs ("M-M") n'utilise generalement pas de cles, et habituellement utilise des identificateurs de colonnes. L'apparition inhabituelle d'une relation "M-M" signifie qu'une colonne, dans une table est en relation avec une autre colonne dans une autre table, et que pour chaque valeur de l'une de ces deux colonnes, il y a une ou plusieurs valeurs correspondantes dans la colonne correspondante de l'autre table (et vice-versa), ou possibilite plus courante, les deux tables ont une relation 1-M avec l'autre (deux relations, une 1-M dans chaque direction). Un [mauvais] exemple de la situation la plus courante consisterait, si vous avez une bases de donnees de fonctions, a avoir une table possedant une ligne pour chaque employe et sa fonction, et d'avoir une autre table contenant une ligne pour chaque fonction avec un des employes l'occupant. Dans ce cas, vous auriez plusieurs lignes pour chaque employe dans la premiere table, une pour chaque fonction, et plusieurs lignes pour chaque fonction dans le seconde table, une pour chaque employe ayant cette fonction.
Ces tables sont en relation M-M: chaque employe, dans la premiere table a plusieurs fonctions dans la seconde table, et chaque fonction, dans la seconde table, a plusieurs attributaires dans la premiere table. Ceci est la partie emergee de l'iceberg concernant ce sujet...consultez les liens ci-dessous pour avoir de plus amples informations et regardez le diagramme ci-dessous donnant un exemple simplifie de diagramme E-R.
[Exemple Simplifie de Diagramme Entites-Relations]

11. Quelles sont quelques unes des fonctionnalites importantes, non standard, de SQL (Question extremement courante)? --Eh bien, nous allons voir ca dans la section suivante...

SQL Non-standard..."A verifier pour votre site"

* INTERSECT et MINUS sont comme des instructions UNION , sauf que INTERSECT produits des lignes qui apparaissent dans les deux requetes, et que MINUS produit des lignes provenant de la premiere requete mais pas de la seconde.
* Fonctionnalites de la Generation de Rapport: la clause COMPUTE est placee a la fin d'une requete pour placer le resultat d'une fonction agregee a la fin d'une liste, comme COMPUTE SUM (PRICE); Une autre solution est d'utiliser une logique d'interruption: definir une interruption pour diviser les resultats de la requete en groupes bases sur une colonne, comme BREAK ON BUYERID. Alors, pour sortir un resultat apres la liste d'un

groupe, utilisez COMPUTE SUM OF PRICE ON BUYERID. Si, par exemple, vous avez utilise ces trois clauses ("BREAK" en premier, "COMPUTE on break" en second, "COMPUTE overall sum" en troisieme), vous obtiendrez un rapport qui regroupera les articles par acheteurs, listera la somme de Prix pour chaque groupe d'articles d'un acheteur, puis, apres que tous les groupes aient ete listes, listera la somme de tous les Prix, le tout, avec des en-tetes et des lignes generes par SQL.

* En plus des fonctions agregees indiquees ci-dessus, quelques SGBD ont des fonctions supplementaires qui peuvent etre utilisees dans des listes de selection (SELECT), sauf que ces fonctions (quelques fonctions caracteres autorisent des resultats de plusieurs lignes) doivent etre utilisees avec une valeur individuelle (pas de groupes), pour des requetes simple ligne. De plus, les fonctions ne doivent etre utilisees qu' avec les types de donnees appropries. Voici quelques Fonctions Mathematiques:

```
ABS(X)       Valeur A-convertit les nombres negatifs en nombres positifs
             et laisse les positifs inchanges
CEIL(X)      X est une valeur decimale qui sera arrondie a la valeur
             superieure.
FLOOR(X)     X est une valeur decimale qui sera arrondie a la valeur
             inferieure.
GREATEST(X,Y)Retourne la plus grande des deux valeurs.
LEAST(X,Y)   Retourne la plus petite des deux valeur.
MOD(X,Y)     Retourne le reste de X / Y.
POWER(X,Y)   Retourne X a la puissance Y.

ROUND(X,Y)   Arrondit X a Y positions decimales. i Y n'est pas donne, X
             est arrondi a la valeur de l'entier le plus proche.
SIGN(X)      Retourne le signe - si X < 0, sinon retourne un signe plus.
SQRT(X)      Retourne la racine carree de  X.

                        Fonctions Caracteres

LEFT(<string>,X)
                    Retourne les X caracteres les plus a gauche de la
                    chaine de caracteres.

RIGHT(<string>,X)
                    Retourne les X caracteres les plus a droite de la
                    chaine de caracteres.
UPPER(<string>)
                    Convertit tous les caracteres de la chaine en
                    majuscules.
LOWER(<string>)
                    Convertit tous les caracteres de la chaine en
                    minuscules.
INITCAP(<string>)
                    Convertit les caracteres initiaux de la chaine en
```

```
                      Capitales.
      LENGTH(<string>)
                      Retourne le nombre de caracteres de la chaine.

      <string>||<string>
                      Combine les deux chaines en une seule chaine,
                      concatenee, ou la premiere chaine est immediatement
                      suivie par la seconde.

      LPAD(<string>,X,'*')
                      Insere des caracteres * (ou n'importe quel autre, mis
                      entre guillemets) a gauche de la chaine de
                      caracteres pour lui donner une longueur de X
                      caracteres.

      RPAD(<string>,X,'*')
                      Insere des caracteres * (ou n'importe quel autre, mis
                      entre guillemets) a droite de la chaine de
                      caracteres pour lui donner une longueur de X
                      caracteres.

      SUBSTR(<string>,X,Y)
                      Extrait Y lettres de la chaine a partir de la
                      position X.

      NVL(<column>,<value>)
                      La fonction NVL va substituer la valeur <value>
                      pour chaque valeur nulle dans la colonne <column>. Si
                      la valeur courante dans la colonne <column> n'est pas
                      nulle (NULL), NVL est sans effet.

----------------------------------------------------------------------------
Resume de la Syntaxe--Pour Utilisateurs Avances Seulement

Voici la forme generale des instructions dont il a ete question dans ce
didacticiel, avec, en plus, quelques autres, (des explications sont donnees).
SOUVENEZ-VOUS que toutes ces instructions peuvent ne pas etre disponibles sur
votre systeme, aussi verifiez leur disponibilite dans la documentation:

ALTER TABLE <TABLE NAME> ADD|DROP|MODIFY (COLUMN SPECIFICATION[S]...voir
Create Table); --vous permet d'ajouter ou de supprimer une ou plusieurs colonnes
d'une table, ou de changer les specifications(donnees, types, etc.) d'une colonne
existante;
cette instruction est egalement utilisee pour changer les specifications
physiques d'une table
(comment la table est stockee, etc.), mais ces definitions sont specifiques au
SGBD, aussi, lire la documentation. De meme, ces specifications physiques sont
utilisees avec les instructions de creation de table, lorsqu'une table est creee
```

```
pour la premiere fois. De plus, seulement une seule option peut etre executee
par instruction de modification: (Alter Table)--soit add (ajout), drop
(suppression), OU modify (modification) dans une simple instruction.

COMMIT; --rend les modifications effectuees sur une bases de donnees permanentes
(depuis le dernier COMMIT; connu sous le nom de transaction)

CREATE [UNIQUE] INDEX <INDEX NAME>
ON <TABLE NAME> (<COLUMN LIST>); --UNIQUE est optionnel; entre parentheses.

CREATE TABLE <TABLE NAME>
(<COLUMN NAME> <DATA TYPE> [(<SIZE>)] <COLUMN CONSTRAINT>,
...autres colonnes); (valide egalement avec ALTER TABLE)
--ou SIZE est uniquement utilise avec certains types de donnees (voir
ci-dessus), et les contraintes incluent les possibilites suivantes (impose
automatiquement par le SGBD;
(un non respect entraine la generation d'une erreur) :

  1. NULL ou NOT NULL (voir ci-dessus)
  2. UNIQUE impose que deux lignes ne peuvent avoir la meme valeur pour cette
colonne
  3. PRIMARY KEY indique a la base de donnees que cette colonne est la colonne
cle primaire (utilise uniquement si la cle est une colonne cle, autrement une
instruction
     PRIMARY KEY (column, column, ...) apparait apres la derniere definition de
colonne.
  4. CHECK permet de tester une condition quand on insere ou on met a jour une
donnee dans cette colonne; par exemple, CHECK (PRICE > 0) amene le systeme a
tester si la colonne Prix est superieure ou egale a zero avant d'accepter la
valeur...quelquefois implante comme instruction CONSTRAINT.
  5. DEFAULT insere une valeur par defaut dans la base de donnees si l'on veut
inserer une ligne qui ne contienne pas de valeur pour cette colonne; par
exemple,
     BENEFITS INTEGER DEFAULT = 10000
  6. FOREIGN KEY fonctionne comme la Cle Primaire, mais est suivi par:
     REFERENCES <TABLE NAME> (<COLUMN NAME>), qui fait reference a la cle primaire
     de reference.

CREATE VIEW <TABLE NAME> AS <QUERY>;

DELETE FROM <TABLE NAME> WHERE <CONDITION>;

INSERT INTO <TABLE NAME> [(<COLUMN LIST>)]
VALUES (<VALUE LIST>);

ROLLBACK; --Annule toutes les modifications effectuees dans la base de donnees,
celles qui ont ete faites depuis la derniere commande COMMIT...Attention!
Quelques logiciels travaillant en transactions, donc, la commande ROLLBACK peut
```

ne pas fonctionner.

```
SELECT [DISTINCT|ALL] <LIST OF COLUMNS, FUNCTIONS, CONSTANTS, ETC.>
FROM <LIST OF TABLES OR VIEWS>
[WHERE <CONDITION(S)>]
[GROUP BY <GROUPING COLUMN(S)>]
[HAVING <CONDITION>]
[ORDER BY <ORDERING COLUMN(S)> [ASC|DESC]]; --ou ASC|DESC permet le classement en
ordre ascendant (ASCending) ou descendant (DESCending)

UPDATE <TABLE NAME>
SET <COLUMN NAME> = <VALUE>
[WHERE <CONDITION>]; --si la clause WHERE n'est pas donnee, toutes les lignes
seront mises a jour selon l'instruction SET
```

Liens Importants

Liens Informatique & SQL/DB : Netscape -- Oracle -- Sybase -- Informix --Microsoft
Page de Reference -- Ask the SQL Pro -- SQL Pro's Relational DB
Sites Utiles
Programmer's Source -- DBMS Sites -- inquiry.com -- DB Ingredients
Web Authoring -- Computing Dictionary -- DBMS Lab/Links -- SQL FAQ -- SQL Databases
RIT Database Design Page -- Database Jump Site -- Didacticiels de programmation sur le Web
Ressources pour le Developpement -- Query List -- IMAGE SQL

Divers: CNN -- USA Today -- Pathfinder -- ZDNet -- Metroscope -- CNet

Liste de ressources sur Internet -- Netcast Weather -- TechWeb -- LookSmart

Moteurs de Recherche: Yahoo -- Alta Vista -- Excite -- WebCrawler -- Lycos -- Infoseek -- search.com

L'auteur n'est pas reponsable de ces sites.

Avertissement

J'espere que vous aurez appris quelque chose de ce premier regard sur un langage tres important qui est en train de devenir plus repandu dans le mode de l'informatique client-serveur. J'ai redige cette page web pour apporter quelque chose d'utile au web et a la communaute des utilisateurs du web. En realite, j'ai appris que ce document est utilise dans plusieurs colleges pour des cours sur les bases de donnees et par des chercheurs. En outre, lisez cette page dans le nouveau livre, edite par Waite Publishing, sur Borland C++ Builder, qui sera publie cet ete et dans une version a venir chez Sams Publishing. De plus, je voudrais remercier tous les gens, sur les cinq continents, qui m'ont contactes

```
a propos de cette page.

J'espere aussi continuer a ajouter plus d'elements a ce didacticiel, tels que
des articles sur la conception d'une base de donnees et les extensions SQL non
standard, meme si je souhaite me tenir a l'ecart de particularites propres a
un Systeme de Gestion de Base de Donnee. Bonne chance pour vos developpements
en SQL et autres aventures informatiques.

Jim Hoffman
---------------------------------------------------------------------------
Commentaires ou Suggestions? Envoyez-moi un courrier electronique a jhoffman@one.net.

Ou vous pouvez desirer jeter un oeil aux pages Web de  Jim Hoffman's Web Pages
pour plus d'informations me concernant.

Copyright 1996-1997, James Hoffman. Ce document peut etre utilise gratuitement
par n'importe quel utilisateur d'Internet, mais ne peut pas etre inclus dans
un autre document, publie sous une autre forme, ou produit en masse de quelque
facon que ce soit.

Netscape Navigator donne le meilleur affichage de cette page; celui-ci n'est pas
tres bon si l'on utilise Microsoft Internet Explorer.

Derniere mise a jour : 8-25-1997; ajout de quelques elements.
```

40 Liste des autre Bases de Données

On trouvera ci-dessous la liste des autres bases de données SQL pour Unix, Linux.

– Cliquez et allez à Applications-¿databases. (http://www.caldera.com/tech-ref/linuxapps/linapps.html)

– Click et allez à Applications-¿databases. (http://www.xnet.com/ blatura/linapps.shtml)

– Ressources pour Bases de Données (http://linas.org/linux/db.html)

– Liste des Bases de Données libres (http://cuiwww.unige.ch:80/ scg/FreeDB/FreeDB.list.html)

– Liste des SGBDR de Browne (http://www.hex.net/ cbbrowne/rdbms.html)

– Liste des SGBDR de SAL (http://SAL.KachinaTech.COM/H/1/)

– Liste des SGBD Orientés Objets de SAL (http://SAL.KachinaTech.COM/H/2/)

– Liste des Utilitaires et d'Autres SGDB de SAL (http://SAL.KachinaTech.COM/H/3/)

– Index des Logiciels de Base de Données Publics de l'ACM SIGMOD (http://bunny.cs.uiuc.edu/sigmod/databaseSoftware/)

41 Conclusion

Après avoir cherché toutes les bases de données disponibles qui soient libres et dont le code source est disponible, j'ai trouvé que SEUL PostgreSQL, le SGDBR (Relationnel Objet)SQL libre, est le plus mature et le plus largement utilisé au monde. PostgreSQL est très attractif en ce que beaucoup de travail a déjà été effectué et qu'il possède des pilotes ODBC et JDBC permettant d'écrire des applications indépendantes des bases de données. Les applications écrites en PostgreSQL qui utilisent les pilotes ODBC, JDBC sont facilement portables sur d'autres systèmes tels que Oracle, Sybase et Informix. Et les applications écrites pour Oracle, Sybase et Informix qui utilisent les pilotes ODBC, JDBC sont facilement portables sur la base de données PostgreSQL.

42 FAQ - Questions sur PostgreSQL

42.1 Foire Aux Questions (FAQ) sur PostgreSQL

Dernière mise à jour: Mercredi 11 Juin 10:44:40 EDT 1997 Version: 6.1.1

Gestionnaire actuel: Bruce Momjian (maillist@candle.pha.pa.us)

La version la plus récente de ce document peut être consultée sur le site Web de postgreSQL, `(http://www.postgreSQL.org)`

On trouvera les réponses aux questions spécifiques à Linux à `(http://www.postgreSQL.org/docs/FAQ-Linux.phtml)`

On trouvera les réponses aux questions spécifiques à Irix à `(http://www.postgreSQL.org/docs/FAQ-Irix.phtml)`

Modifications dans cette version (* = modifié, + = nouveau) :

* 3.42) Qu'est-ce qu'une Optimisation de Requête Génétique? * 3.43) J'utilise Solaris et l'affichage de mes dates est faux. Pourquoi?

———————————————————————————————

42.2 Qu'est-ce que PostgreSQL?

PostgreSQL est une amélioration du système de gestion de base de données POSTGRES, un prototype de recherche de SGBD de la nouvelle génération. Alors que PostgreSQL garde le puissant modèle de données et les riches types de données de POSTGRES, il remplace le langage de requêtes PostQuel par un sous-ensemble étendu du langage SQL. PostgreSQL est libre et le code source est disponible.

Le développement de PostgreSQL est effectué par une équipe de développeurs Internet qui sont tous inscrits à la liste de diffusion PostgreSQL développement. Le coordinateur actuel est Marc G. Fournier (scrappy@postgreSQL.org). (Voir ci-dessous comment le joindre). Cette équipe est maintenant responsable de tous les développements actuels et futurs de PostgreSQL.

Les auteurs de PostgreSQL 1.01sont Andrew Yu et Jolly Chen. Beaucoup d'autres ont contribué au portage, aux tests, au débogage et aux améliorations du code; Le code original de Postgres, à partir duquel PostgreSQL est dérivé, est le résultat de l'effort de nombreux étudiants de troisième cycle, de deuxième cycle et d'enseignants sous la direction du Professeur Michael Stonebraker de l'université de Californie, Berkeley.

Le nom original du logiciel à Berkeley était Postgres. Lors de l'ajout des fonctionnalités SQL en 1995, il fut renommé Postgres95. Ce nom fut changé à la fin de 1996 en PostgreSQL.

42.3 Sur quelles machines tourne PostgreSQL?

Les auteurs ont compilé et testé PostgreSQL sur les plates-formes suivantes(quelques unes de ces compilations requièrent gcc 2.7.0) :

* aix - IBM sous AIX 3.2.5

* alpha - DEC Alpha AXP sous OSF/1 2.0

* BSD44_derivé - OSs dérivé de 4.4-lite BSD (NetBSD, FreeBSD)

* bsdi - BSD/OS 2.0, 2.01, 2.1, 3.0

* dgux - DG/UX 5.4R3.10

* hpux - HP PA-RISC sous HP-UX 9.0

* i386_solaris - i386 Solaris

* irix5 - SGI MIPS sous IRIX 5.3

* linux - Intel x86 sous Linux 1.2 et Linux ELF (Pour le Linux non-ELF Linux, voir LINUX_ELF ci-dessous).

* sparc_solaris - SUN SPARC sous Solaris 2.4

* sunos4 - SUN SPARC sous SunOS 4.1.3

* svr4 - Intel x86 sous Intel SVR4 * ultrix4 - DEC MIPS sous Ultrix 4.4

Pour les plates-formes suivantes, il y a des problèmes/bogues connus :

* nextstep - Motorola MC68K ou Intel x86 sous NeXTSTEP 3.2

42.4 Où puis-je trouver PostgreSQL?

Le premier site ftp anonyme pour PostgreSQL est :

* `(ftp://ftp.postgreSQL.org/pub)`

Il existe un site miroir à :

* `(ftp://postgres95.vnet.net/pub/postgres95)` * `(ftp://ftp.luga.or.at/pub/postgres95)` * `(ftp://cal011111.student.utwente.nl/pub/postgres95)` * `(ftp://ftp.uni-trier.de/pub/database/rdbms/postgres/postgres95)` * `(ftp://rocker.sch.bme.hu)`

42.5 Quel est le régime de copyright de PostgreSQL?

PostgreSQL est sujet au COPYRIGHT suivant.

Système de Gestion de Base de Données PostgreSQL

Copyright (c) 1994-6 Université de Californie. Tous droits réservés

42.6 Support pour PostgreSQL

Il n'y a aucun support officiel pour PostgreSQL de la part des de l'équipe de support initiale de l'Université de Californie, Berkeley. Il est maintenu uniquement grâce à l'effort de volontaires.

La liste de diffusion est: questions@postgreSQL.org. Elle est disponible pour des discussions sur des sujets concernants PostgreSQL, incluant, mais non limité aux rapports d'erreurs et corrections. Pour avoir des informations sur la manière de s'inscrire, envoyer un courrier électronique avec les lignes suivantes dans le corps du message (pas dans la ligne sujet)

subscribe end

à questions-request@postgreSQL.org.

Il y a aussi une liste résumée. Pour s'inscrire à cette liste, envoyez un courrier électronique à: questions-digest-request@postgreSQL.org avec, dans le CORPS du message :

subscribe end

Des résumés sont envoyés aux membres de cette liste chaque fois que la liste principale a reçu environ 30k de messages.

Il y a une liste de diffusion des erreurs. Pour s'inscrire à cette liste, envoyez un courrier électronique à bugs-request@postgreSQL.org avec, dans le CORPS du message :

Il y a aussi un forum de discussion pour les développeurs. Pour s'inscrire à cette liste, envoyer un courrier électronique à hackers-request@postgreSQL.org avec, dans le CORPS du message :

subscribe end

Des informations complémentaires sur PostgreSQL peuvent être trouvées via la page d'accueil WWW de PostgreSQL à :

`(http://www.postgreSQL.org)`

42.7 Dernière version dePostgreSQL

La dernière version de PostgreSQL est la version 6.0, qui est disponible depuis le 31 Janvier 1997. La version 6.1 est programmée pour bientôt. Pour avoir des informations sur les nouveautés de la version 6.1, consulter le fichier TODO sur notre page WWW.

Nous attendons la version 7.0 dans quelques mois qui sera plus rapide et réduira de 50 % la taille des colonnes système, sur disque, nécessaire pour chaque ligne d'une table. Cette version disposera aussi des fonctionnalités de vidage (dump) et de restauration (restore).

42.8 Existe-t-il une version commerciale de PostgreSQL?

Illustra Information Technology (filiale possédée totalement par Informix Software, Inc.) vend un SGBD Relationnel Objet appelé Illustra qui était, à l'origine, basé sur postgres. Illustra a des similarités d'aspect avec PostgreSQL mais possède plus de fonctionnalités, est plus robuste, plus performant et dispose de documentation et d'un support réels. D'un autre côté, il coûte de l'argent. Pour de plus informations, contactez sales@illustra.com

42.9 Quelle documentation est disponible pour PostgreSQL?

Un manuel utilisateur, des pages de manuel, et quelques petits exemples de test sont compris dans la distribution. Les pages de manuel concernant sql et les fonctions internes (built-in) sont particulièrement importantes.

La page www contient des liens sur un guide d'implémentation et sur cinq papiers concernants les concepts d'architecture et les fonctionnalités de postgres .

42.10 Quelle version de SQL PostgreSQL utilise-t-il?

PostgreSQL supporte un sous-ensemble de SQL-92. Ils en possède les constructions les plus importantes mais manque de certaines fonctionnalités. Les différences les plus visibles sont :

* les sous-requêtes imbriquées ne sont pas supportées * pas de clause HAVING sous un GROUP BY

D'un autre côté, vous pouvez créer des types définis par l'utilisateur, des fonctions, de l'héritage etc. Si vous souhaitez participer à la programmation de PostgreSQL, vous pouvez ajouter les fonctionnalités manquantes énumérées ci-dessus.

42.11 PostgreSQL fonctionne-t-il avec les versions précédentes de postgres?

PostgreSQL v1.09 est compatible avec les bases de données créées avec la version v1.01. Ceux qui veulent faire une mise à niveau de la version 1.0 doivent lire les indications dan le répertoire MIGRATION_1.0_TO_1.02.

Une mise à niveau vers la version 6.0 nécessite de faire un vidage et une restauration à partir des versions précédentes.

Une mise à niveau vers la version 6.1 nécessite de faire un vidage et une restauration à partir des versions précédentes.

Ceux qui font une mise à niveau de versions antérieures à 1.09 doivent d'abord faire une mise à niveau vers la version 1.09 sans vidage/restauration, puis vider les données de la version 1.09, et enfin les recharger dans la version 6.0 ou 6.1.

42.12 Combien de personnes utilisent PostgreSQL?

Puisque nous n'avons aucune procédure de licence ou d'enregistrement, c'est impossible à dire. Nous savons que des centaines de copies de PostgreSQL v1.* ont été rapatriées, et qu'il y a plusieurs centaines d'inscrits aux listes de diffusion.

——————————————————————————————

43 FAQ - Questions concernant l'Installation

43.1 Initdb ne tourne pas

* vérifiez que vous avez les chemins appropriés * vérifiez que l'utilisateur 'postgres' dispose des droits nécessaires sur les fichiers * assurez-vous qu'il y a des fichiers dans $ PGDATA/files, et qu'ils ne sont pas vides. S'ils n'y sont pas, c'est que "gmake install" à échoué pour une raison quelconque

43.2 Quand je démarre le postmaster, j'ai le message "FindBackend- could not find a backend to execute..." "postmaster- could not find backend to execute..."

Vous n'avez probablement pas le bon chemin de spécifié. L'exécutable 'postgres' a besoin d'être dans votre chemin.

43.3 Le système paraît perturbé par les virgules, les points décimaux, et le format des dates.

Vérifiez votre configuration locale. PostgreSQL utilise les options locales choisies par l'utilisateur qui a lancé le processus postmaster. Adaptez les selon votre environnement de travail.

43.4 Comment puis-je installer PostgreSQL ailleurs que dans /usr/local/pgsql?

Il vous faut éditer le fichier Makefile.global et modifier POSTGRESDIR suivant vos souhaits, ou créer un fichier Makefile.custom et définir POSTGRESDIR ici.

43.5 Quand je lance postmaster, j'ai un message "Bad System Call core dumped".

Cela peu être dû à de nombreux problèmes, mais, en premier lieu, vérifiez que vous disposez bien des extensions system V dans votre noyau. PostgreSQL nécessite un support de la mémoire partagé dans le noyau.

43.6 Quand j'essaie de lancer postmaster, j'ai des messages d'erreur "IpcMemoryCreate".

Soit votre gestion de la mémoire partagée n'est pas configurée correctement dans le noyau ou vous avez besoin d'étendre la mémoire partagée disponible dans le noyau. La quantité dont vous avez besoin dépend de l'architecture de votre système et du nombre de tampons avec lesquels postmaster a été configuré pour tourner. Pour la plupart des systèmes, avec des tampons de la taille par défaut, vous avez besoin d'un minimum de ˜ 760K.

43.7 J'ai modifié un fichier source, mais une recompilation ne reflète pas les modifications?

Les fichiers Makefile ne contiennent pas les dépendances correctes pour les fichiers inclus. Il vous faut exécuter 'make clean' puis un autre 'make'.

———————————————————————————

44 FAQ - Fonctionnalités de PostgreSQL

44.1 Comment dois-je spécifier une CLE ou d'autres contraintes sur une colonne?

Les contraintes sur les colonnes ne sont par supportées dans PostgreSQL. En conséquence, le systèmes ne vérifie pas l'existence de doublons.

Sous 6.0, créez un index unique sur la colonne. Les tentatives de créer des doubles dans cette colonne donneront une erreur.

44.2 Est-ce que PostgreSQL supporte les sous-requêtes imbriquées?

Les sous-requêtes imbriquées ne sont pas encore implantées, mais elles peuvent être simulées en utilisant des fonctions sql.

44.3 Comment dois-je définir des index "unique"?

PostgreSQL 6.0 supporte les index "unique".

44.4 J'ai eu un tas de problèmes dans l'utilisation des règles.

Actuellement, le système de règle dans PostgreSQL est pour la plupart non fonctionnel. Il fonctionne suffisamment pour supporter le mécanismes de vue, mais c'est tout. Vous utilisez les règles PostgreSQL à vos risques et périls.

44.5 Il ne me semble pas possible d'écrire au milieu de grands objets de manière fiable.

Le système de gestion des grands objets est également non fonctionnel dans PostgreSQL. Il fonctionne juste assez pour stocker de grandes paquets de données et de les relire, mais l'implémentation souffre de quelques problèmes sous-jacents. Vous utilisez les grands objets PostgreSQL à vos risques et périls.

44.6 PostgreSQL a-t-il un interface utilisateur graphique? Un générateur de rapport? Un interface de langage d'interrogation intégré?

Non. Non. Non. Au moins, pas dans la distribution officielle. Quelques utilisateurs ont indiqué quelques succès dans l'utilisation de 'pgbrowse' et de 'onyx' comme frontaux à PostgreSQL. Plusieurs contributeurs travaillent sur des outils frontaux basés sur Tk. Posez la question dans la liste de diffusion.

44.7 Comment puis-je écrire une application cliente pour PostgreSQL?

PostgreSQL supporte une bibliothèque d'interfaçage que l'on peut appeler à partir de C nommée libpq de même qu'une bibliothèque d'interfaçage basée sur Tcl appelée libtcl.

D'autres ont contribué à une interface perl et à une passerelle WWW à PostgreSQL. Consulter la page d'accueil PostgreSQL pour plus de détails.

44.8 Comment puis-je me prémunir d'accès à mon processus d'arrière plan PostgreSQL par d'autres hôtes?

Utilisez une authentification basée sur l'hôte en modifiant le fichier $ PGDATA/pg_hba en conséquence.

44.9 Comment puis-je mettre en place un pg_group?

Actuellement, il n'y a pas d'interface aisée pour mettre en place des groupes d'utilisateurs. Vous devez explicitement insérer/mettre à jour la table pg_group . Par exemple :

```
jolly=> insert into pg_group (groname, grosysid, grolist)
```

```
jolly=>     values ('posthackers', '1234', '5443, 8261');
INSERT 548224
jolly=> grant insert on foo to group posthackers;
CHANGE
jolly=>
```

Les champs dans le fichier pg_group sont :

* groname: le nom de groupe. C'est un char16 et il doit être purement alphanumérique. Ne pas inclure de soulignés ou d'autres ponctuations. * grosysid: l'identificateur de groupe. C'est un int4. Il doit être unique pour chaque groupe. * grolist: la liste des identificateurs de pg_user qui appartiennent au groupe.

```
C'est un int4[].
```

44.10 Quelle l'exacte différence entre les curseurs binaires et les curseurs normaux?

Les curseurs normaux retournent les données en format ASCII. Puisque les données sont stockées d'origine en format binaire, le système doit effectuer une conversion pour produire le format ASCII. De plus, les formats ASCII sont souvent plus grands en taille que le format binaire. Une fois que les attributs sont retournés en ASCII, souvent l'application cliente doit alors les convertir en format binaire pour les manipuler.

Les curseurs binaires vous renvoient les données dans le format binaire, natif, de représentation. Donc, les curseurs binaires ont tendance à être plus rapides puisqu'il y a moins de temps de traitement dû à la conversion.

Toutefois, l'ASCII est neutre en ce qui concerne l'architecture alors que la représentation binaire peut différer entre différentes architectures de machines. Donc, si votre machine cliente utilise une représentation des données différente de votre machine serveur, recevoir les attributs en format binaire ne correspond certainement pas à ce que vous désirez. Donc, si votre objectif principal des d'afficher les données en ASCII, alors les recevoir en ASCII vous évitera quelques efforts côté client.

44.11 Pourquoi l'opérateur != ne fonctionne-t-il pas?

SQL spécifie les opérateurs d'inégalité $<>$, et c'est ce que nous avons définit comme types internes.

En 6.0, != est équivalent à $<>$.

44.12 Qu'est-ce qu'un index R-tree et à quoi est-il utilisé?

Un index R-tree est utilisé pour indexer les données spatiales. Un index obtenu par hachage ne peut effectuer de recherches dans une gamme de données. Un index B-tree peut seulement effectuer des recherches dans une gamme de données sur une dimension. Les index R-tree permettent de traiter des données multidimensionnelles. Par exemple, si un index R-tree peut être construit sur un attribut de type 'point' le système peut répondre plus efficacement à des requêtes telles que sélectionner tous les points à l'intérieur d'un rectangle.

Le papier de référence qui décrit la conception originale du R-Tree est :

Guttman, A. "R-Trees: A Dynamic Index Structure for Spatial Searching." (R-Trees: une Structure d'Index Dynamique pour les Recherches Spatiales) Proc of the 1984 ACM SIGMOD Int'l Conf on Mgmt of Data, 45-57.

Vous pouvez également trouver ce papier dans le livre de Stonebraker "Readings in Database Systems" (Lectures sur les Systèmes de Gestion de Base de Données)

44.13 Quelle est la taille maximum d'un tuple?

Les Tuples sont limités à 8K octets. En tenant compte des attributs du système et autre surcharge, on devrait rester bien à l'écart des 8,000 octets pour être tranquille. Pour utiliser des attributs supérieurs à 8K, essayez d'utiliser l'interface pour les grands objets.

Les Tuples ne doivent pas franchir la frontière des 8k donc un tuple de 5k nécessitera un espace de stockage de 8k.

44.14 J'ai défini des index, mais mes requêtes ne paraissent pas en faire usage. Pourquoi?

PostgreSQL ne maintient pas automatiquement de statistiques. On doit réaliser un appel à une "réorganisation" explicite pour mettre les statistiques à jour. Après que les statistiques aient été mises à jour, l'optimiseur fera un meilleur usage des index. Notez que l'optimiseur des limité et n'utilise pas les index dans certaines circonstances (telles que les dans les clauses OR).

Si le système ne voit toujours pas l'index, c'est probablement parce que vous avez créé un index sur un champ de type *_ops incorrect. Par exemple, vous avez créé un champ CHAR(4), mais vous avez spécifié un index de type_class char_ops.

Consultez la page du manuel sur create_index pour avoir des informations sur les types de classes disponibles. Il doit correspondre au type de champ.

Postgres ne prévient pas l'utilisateur quand un index incorrect est créé.

Les index ne sont pas utilisés dans les opérations ORDER BY.

44.15 Existe-t-il des pilotes ODBC pour PostgreSQL?

Il y a deux pilotes ODBC disponibles, PostODBC et OpenLink ODBC.

Pour tous ceux qui sont intéressés par PostODBC, il y a maintenant deux listes de diffusion consacrées aux discussions sur PostODBC. Ce sont :

* postodbc-users@listserv.direct.net * postodbc-developers@listserv.direct.net

ces listes sont des listes de diffusion ordinaires gérées par majordomo. Vous pouvez vous inscrire en envoyant un courrier électronique à :

* majordomo@listserv.direct.net

OpenLink ODBC est actuellement en version bêta sous Linux. Vous pouvez le rapatrier de (`http://www.openlinksw.com/postgres.html`) Il fonctionne avec notre logiciel client ODBC standard vous aurez donc Postgres ODBC disponible sur toutes les plates formes clients que nous supportons (Win, Mac, Unix, VMS).

Nous vendrons probablement ce produit aux gens qui ont besoin d'un support de qualité commerciale, mais une version libre sera toujours disponible. Adressez vos questions à postgres95@openlink.co.uk.

44.16 Comment puis-je utiliser postgres avec un indexage multidimensionnel (¿2 dimensions)?

Les R-Trees intégrés peuvent traiter les polygones et les boîtes. En théorie, les R-trees peuvent être étendus pour prendre en charge un plus grand nombre de dimensions. En pratique, l'extension des R-trees demande un peu de travail et n'avons actuellement aucun document sur la manière de le faire.

44.17 Comment puis-je effectuer des recherches d'expressions régulières? une recherche d'expression régulière sans tenir compte des majuscules ou des minuscules?

```
PostgreSQL supporte une syntaxe de type SQL LIKE de meme qu'un systeme de recherche
d'expressions regulieres plus general avec l'operateur ~. L'operateur  !~ est
l'operateur de negation de recherche d'expressions regulieres. ~* et !~* sont les
operateurs d'expression regulieres indifferents aux majuscules et aux minuscules.
```

44.18 Je ne peux pas accéder à la base de données en tant qu'utilisateur 'root'.

Vous ne devriez pas créer d'utilisateurs de base de données avec l'identificateur d'utilisateur 0(root). Ils seront incapables d'accéder à la base de données. C'est une mesure de précaution à cause de la possibilité pour chaque utilisateur de lier dynamiquement des modules objets au moteur de base de données.

44.19 J'ai subit un crash du serveur pendant une réorganisation. Comment puis-je supprimer le fichier de verrouillage?

Si le serveur se crash pendant une commande de réorganisation, il y a des chances pour qu'il laisse un fichier de verrouillage en suspend quelque part. Une tentative de relancer la commande de réorganisation résultera dans le

WARN:can't create lock file – another vacuum cleaner running?

Si vous êtes sûr qu'il n'y a pas de réorganisation en cours, vous pouvez supprimer le fichier appelé "pg_vlock" dans le répertoire de votre base de données (qui est $ PGDATA/base/< dbName >)

44.20 Quelle et la différence entre les différents types caractères?

```
Type            Nom Interne     Notes
```

```
--------------------------------------------------

CHAR            char            1 caractere

CHAR2           char2           2 caracteres

CHAR4           char4           4 caracteres  optimise pour une longueur fixe

CHAR8           char8           8 caracteres

CHAR16          char16          16 caracteres

CHAR(#)         bpchar          complete par des blancs jusqu'a la longueur fixe specifiee

VARCHAR(#)      varchar         la taille specifie la longueur maximum, pas de remplissage

TEXT            text            longueur limitee seulement par la longueur maximum du tuple

BYTEA           bytea           tableau d'octets de longueur variable

Rappelez vous, il vous faut utiliser le nom interne lorsque vous  creez des index
sur ces champs ou quand vous effectuez d'autres operations internes.

Les quatre derniers types ci-dessus sont des types "varlena" (i.e. les quatre
premiers octets donnent la longueur, suivie par les donnees). CHAR(#) et VARCHAR(#)
allouent le nombre maximum d'octets quelle que soient les donnees stockees dans le
champ.
TEXT et BYTEA sont les seuls types caracteres qui ont une longueur variable sur le
disque.
```

44.21 Dans une requête, comment puis-je détecter si un champ est NULL?

PostgreSQL possède deux mots-clés intrinsèques, "isnull" et "notnull" (notez qu'il n'y a pas d'espaces). La Version 1.05 et suivante ainsi que la 6.* comprennent IS NULL et IS NOT NULL.

44.22 Comment puis-je voir que l'optimiseur de requête est en train d'évaluer ma requête?

Insérez le mot 'EXPLAIN' au début de la requête, par exemple :

EXPLAIN SELECT * FROM table1 WHERE age = 23;

44.23 Comment puis-je créer un champ série?

Postgres ne permet pas à l'utilisateur de spécifier une colonne utilisateur comme ayant le type SERIAL. A la place, vous pouvez utiliser le champ oid de chaque ligne comme valeur unique. Cependant, si vous avez besoin de vider et de recharger la base de données, vous serez obligés d'utiliser pgdump de postgres version 1.07 ou ultérieur ou 6.* avec l'option -o'ou COPY avec l'option WITH OIDS pour conserver les oids.

Une autre manière valide de le faire est de créer une fonction :

create table my_oids (f1 int4);

insert into my_oids values (1);

create function new_oid () returns int4 as 'update my_oids set f1 = f1 + 1; select f1 from my_oids; ' language 'sql';

puis:

create table my_stuff (my_key int4, value text);

insert into my_stuff values (new_oid(), 'hello');

Cependant, il faut garder à l'esprit qu'il y a compétition ici où un serveur peut effectuer une mise à jour, puis un autre serveur effectuer une mise à jour et que les deux sélectionnent le même nouvel identificateur. Cette instruction doit être effectuée dans le cadre d'une transaction.

Les Séquences sont implantées dans la version 6.1

44.24 Comment puis-je créer un index multicolonnes?

Dans la version 6.0, vous ne pouvez pas directement créer un index multicolonnes en utilisant l'instruction "create index". Il vous faut définir une fonction qui agit sur les colonnes multiples, puis, utiliser "create index" avec cette fonction.

En 6.1, cette fonctionnalité est disponible.

44.25 Que sont les fichiers temp_XXX dans le répertoire de ma base de données?

Ce sont les fichiers temporaires "temp_ files" créés par l'exécuteur de requête. Par exemple, si un tri doit être effectué pour réaliser une instruction ORDER BY, quelques fichiers temporaires sont crées en résultat du tri.

Si vous n'avez pas de transactions ou de tri en cours à cet instant, vous pouvez supprimer ces fichiers temporaires "temp_ files" sans risques.

44.26 Pourquoi mes fichiers ne deviennent-ils pas petits après une suppression?

Si vous exécuter une réorganisation avec une version pre-6.0, les lignes non utilisées seront marquées pour être réutilisées, mais les blocs du fichier ne seront pas libérés.

En 6.0, une réorganisation réduit la taille des tables correctement.

44.27 Pourquoi je ne peux pas me connecter à ma base de données à partir d'une autre machine?

La configuration par défaut permet uniquement de se connecter à partir de l'hôte tcp/ip localhost (machine locale). Vous devez ajouter une entrée pour un hôte supplémentaire dans le fichier pgsql/data/pg_hba.

44.28 J'ai l'erreur 'default index class unsupported' (index par défaut, classe non supportée) en créant un index. Comment dois-je faire?

Vous avez probablement utilisé :

create index idx1 on person using btree (name);

Les index PostgreSQL sont extensibles, et par conséquent en pre-6.0, vous devez spécifier un type de classe (class_type) quand vous créez un index. Lisez la page de manuel de create index (nommée create_index).

Avec la version 6.0, si vous ne spécifiez pas de type de classe (class_type), il prend par défaut le type correspondant à la première colonne.

44.29 Pourquoi la création d'un index entraîne un crash de ma tâche de fond serveur?

Vous avez probablement définit un type de classe *_ops type incorrect pour le champ que vous voulez indexer.

44.30 Comment puis-je trouver quels index et quelles opérations sont définis dans la base de données?

Exécutez le fichier pgsql/src/tutorial/syscat.source. Il donne l'illustration de beaucoup de 'sélections' ('select's) à faire pour obtenir des informations sur les tables du système de gestion de base de données.

44.31 Pourquoi les instructions ont-elles besoin d'un caractère supplémentaire terminal? Pourquoi 'createuser' retourne 'unexpected last match in input()'? Pourquoi pg_dump ne fonctionne-t-il pas?

Vous avez compilé postgres avec flex version 2.5.3. Il y a un bogue dans cette version de flex. Utilisez la version 2.5.2 ou 2.5.4 de flex à la place. Il y a un fichier doc/README.flex qui doit faire un colmatage correct du code source flex 2.5.3.

44.32 Tous mes serveurs se plantent lors d'un accès concurrent à une table. Pourquoi?

Ce problème peut avoir son origine dans le noyau qui n'est pas configuré pour supporter les sémaphores.

44.33 Quels sont les outils disponibles pour ajouter des fonctionnalités aux pages Web de?

Pour une intégration au web, PHP/FI est une excellente interface. L'URL pour cela est `(http://www.vex.net/php/)`

PHP est très bien pour des choses simples, mais pour si c'est plus compliqué, certains utilisent encore l'interface perl et CGI.pm.

Un exemple d'utilisation de WWW avec C pour dialoguer avec Postgres peut être essayé, il est à:

* `(http://www.postgreSQL.org/ mlc)`

Une passerelle WWW basée sur WDB utilisant perl peut être téléchargée de :

* `(http://www.eol.ists.ca/ dunlop/wdb -p95)`

44.34 Qu'est-ce que la fonctionnalité "time-warp" et en quoi cela concerne-t-il la réorganisation?

PostgreSQL traite les modifications de données de manière différente de la plupart des systèmes de gestion de bases de données. Quand une ligne est modifiée dans une table, la ligne originale est marquée avec un horodatage du moment où cette modification est intervenue, et une nouvelle ligne est créée avec les données actuelles. Par défaut, seulement les données actuelles sont utilisées dans une table. Si vous spécifiez une donnée date/heure à la suite du nom de la table dans une clause FROM, vous pouvez avoir accès aux données qui étaient "actuelles" à ce moment là, i.e.

SELECT * FROM employees $'July 24, 1996 09:00:00'$

affiche les lignes d'employés dans la ligne au moment spécifié. Vous pouvez spécifier des intervalles tels que $date, date$, $date$,, , $date$, ou ,. Cette dernière option permet d'accéder à toutes les lignes qui ont jamais existé.

L'insertion (INSERT) de lignes reçoit également un horodatage, donc, les lignes qui n'étaient pas dans la table à l'instant désiré ne seront pas affichées.

La réorganisation supprime les lignes qui ne sont plus "actuelles" (courantes). La fonctionnalité "time-warp" est utilisée par le moteur du SGBD pour l'annulation des modifications et pour la restauration après un plantage. Les dates/heures d'expiration peut être fixés avec la commande purge.

En 6.0, après le nettoyage d'une table, la date de création d'une ligne peut être incorrecte, causant l'échec d'un parcours de la base de données basé sur les dates (time-travel).

Cette fonctionnalité de "time-travel" a été supprimé dans la version 7.0.

44.35 Comment dois-je ajuster mon moteur de SGBD pour avoir de meilleures performances?

On peut faire deux choses. Vous pouvez utilisez l'option "Openlink" pour désactiver fsync() en lançant le postmaster avec l'option '-o -F'. Ceci évitera que des fsync() mettent à jour les données sur disque après chaque transaction.

Vous pouvez également utiliser l'option -B du postmaster pour accroître le nombre de mémoires tampon partagées que doivent se partager les différents processus en tâche de fond. Si vous donnez une valeur trop élevée à ce paramètre, le processus ne sera pas lancé ou se plantera de manière inattendue. Chaque tampon fait 8K et leur nombre est de 64 par défaut.

44.36 Quelles sont les fonctionnalités disponibles pour la mise au point dans PostgreSQL?

PostgreSQL dispose de plusieurs fonctions qui donnent des informations sur son état qui peuvent être utiles pour des besoins de débogage.

Tout d'abord, en compilant avec DEBUG définit, de nombreux assert()'s permettent de suivre l'avancement du processus de fond de plan et arrêtent le programme quand quelque chose d'inattendu se produit.

A la fois le postmaster et postgres ont plusieurs options de débogage. Premièrement, à chaque fois que lancez le postmaster, assurez-vous de rediriger la sortie standard et les messages d'erreurs vers un fichier d'enregistrement, comme :

```
cd /usr/local/pgsql
./bin/postmaster >server.log 2>&1 &
```

Ceci créera un fichier server.log en tête de l'arborescence du répertoire de PostgreSQL. Ce fichier peut contenir des informations utiles concernant les problèmes et les erreurs rencontrés par le serveur. Postmaster possède une option -d qui permet d'obtenir des informations plus détaillées. L'option -d peut prendre une valeur de 1 à 3 qui spécifie le niveau de débogage. On obtient un fichier de débogage bavard qui peut être formaté en utilisant le programme 'indent'. (Il vous faudra peut être supprimer les lignes '====' dans les versions 1.*.) Faites attention au fait qu'un niveau de débogage supérieur à un entraîne la génération de fichiers d'enregistrement volumineux dans les versions 1.*.

Vous pouvez , en réalité, faire tourner le processus de fond de plan postgres à partir de la ligne de commande, et taper vos instructions SQL directement. Ceci n'est recommandé que SEULEMENT dans des situations de débogage. Notez qu'un caractère saut de ligne termine la requête, pas un point-virgule. Si vous avez compilé en utilisant les symboles de débogage, vous pouvez peut-être utiliser un débogueur pour voir ce qui se passe. Le processus de fond de plan n'ayant pas été lancé par le postmaster, il ne tourne pas dans un environnement identique et les problèmes d'interaction processus de fond de plan/verrouillage peuvent ne pas être reproduits. Quelques systèmes d'exploitation peuvent se connecter à un processus de fond de plan directement pour diagnostiquer les problèmes.

Le programme postgres possède les options a -s, -A, -t qui peuvent être très utiles pour la mise au point et pour la mesure de performances.

La commande EXPLAIN (Voir dans cette FAQ) permet de voir comment PostgreSQL interprète votre requête.

44.37 Qu'est-ce qu'un oid? Qu'est-ce qu'un tid?

Les Oid sont la réponse de Postgres à des identificateurs de ligne uniques ou à des colonnes sérielles. Chaque ligne qui est créée dans Postgres reçoit un oid unique. Tous les oids générés par initdb sont inférieurs à 16384 (voir backend/access/transam.h). Tous les oids post-initdb (créés par un utilisateur) sont égaux ou supérieurs à cette valeur. Tous ces oids sont uniques, non pas seulement dans une table, ou une base de données, mais uniques à l'intérieur de l'installation postgres toute entière.

Postgres utilise les oids dans ses tables système internes pour relier les lignes de tables séparées. Ces oids peuvent être utilisés pour identifier des lignes spécifiques à un utilisateur et dans les jointures. On recommande l'utilisation de colonnes de type oid pour stocker des valeurs d'oids. Consulter les pages de manuel sql(l) pour voir les autres colonnes internes.

Les tids sont utilisés pour identifier des lignes physiques spécifiques contenant des valeurs de bloc ou de décalage. Les tids changent après une modification ou un rechargement des lignes. Ils sont utilisés par des enregistrements d'index pour pointer sur des lignes physiques. On ne peut pas y accéder par sql.

44.38 Quel est le sens de quelques termes utilisés dans?

Des parties du code source et d'anciennes documentations utilisent des termes qui sont d'un usage plus commun. En voici quelques uns :

* row, record, tuple ligne, enregistrement, tuple * attribute, field, column attribut, champ, colonne * table, class table, classe * retrieve, select retrouver, sélectionner * replace, update remplacer, mettre à jour * append, insert ajouter, insérer * oid, serial value oid, valeur sérielle * portal, cursor portail, curseur * range variable, table name, table alias variable indiquant une étendue, nom de la table, alias de la table

Merci de me faire savoir si vous pensez à d'autres.

44.39 Qu'est-ce que l'Optimisation de Requête Génétique?

Le module GEQO dans PostgreSQL est sensé résoudre le problème d'optimisation de requête sur la jointure de plusieurs tables au moyen d'un Algorithme Génétique (GA). Il permet le traitement de grandes jointures par l'intermédiaire d'une recherche non–exhaustive.

Pour de plus amples informations voir README.GEQO utesch@aut.tu-freiberg.de.

44.40 J'utilise Solaris et l'affichage de mes dates est faux. Pourquoi?

Il y avait un bogue dans 6.0 qui causait ce problème sous Solaris avec l'optimisation -O2. Passez à la version 6.1.

44.41 Comment puis-je avoir plus de 32 processus d'arrière plan concurrents?

Editez include/storage/sinvaladt.h, et changez la valeur de MaxBackendId. Dans le futur, nous avons prévu de rendre ce paramètre configurable.

---

45 FAQ - Etendre PostgreSQL

45.1 J'ai écrit une fonction définie par l'utilisateur et quand je la fait tourner sous psql, j'ai un plantage avec vidage mémoire.

Ce problème peut provenir d'un certains nombre de causes. Tout d'abord, essayez de tester votre fonction dans un programme de test indépendant. Assurez-vous également que vous n'êtes pas en train d'envoyer des NOTICES elog quand le frontal attend des données, comme dans les fonctions type_in() or type_out()

45.2 J'ai des messages du type "NOTICE-PortalHeapMemoryFree- 0x402251d0 not in alloc set"!

Vous êtes en train de libérer (pfree) quelque chose qui n'a pas été alloué (palloc). Quand vous écrivez des fonctions définies par l'utilisateur, n'incluez pas le fichier "libpq-fe.h". Si vous faites cela, va entraîner que votre palloc soit un malloc à la place d'un free. Puis, quand la tâche de fond libère (pfrees) la mémoire, vous recevez le message NOTICE.

45.3 J'ai écrit quelques nouveaux types et quelques fonctions astucieux pour PostgreSQL.

S'il vous plaît, partagez-les avec les autre utilisateurs de PostgreSQL. Envoyez vos extensions dans la liste de diffusion, et elles finiront peut-être dans le sous-répertoire contrib/.

45.4 Comment dois-je écrire une fonction C qui retourne un tuple?

Cela nécessite beaucoup de génie, tellement que les auteurs n'ont jamais essayer, bien que, en principe, cela soit possible. Une réponse brève sera ... vous ne pouvez pas. Cette possibilité est envisagée dans le future.

---

46 FAQ - Bogues

46.1 Comment dois faire un rapport de bogue?

Regardez la FAQ courante à `(http://www.postgreSQL.org)`

Regardez aussi à notre site ftp (ftp://ftp.postgreSQL.org/pub) pour voir s'il n'y a pas une version plus récente de PostgreSQL.

Vous pouvez aussi remplir le fichier formulaire "bug-template" et l'envoyer à :

* bugs@postgreSQL.org

C'est l'adresse de la liste de diffusion des développeurs.

47 FAQ Linux-PostgreSQL - Compilation de PostgreSQL

==

Foire Aux Questions (FAQ) de PostgresSQL V6.0 Spécifique Linux A LIRE EN PARALLE AVEC LA FAQ NORMALE

==

Dernière mise à jour: Wed Jan 29 20:16:00 GMT 1997

Gestionnaire actuel: Andrew C.R. Martin (martin@biochem.ucl.ac.uk)

Auteur original: Andrew C.R. Martin (martin@biochem.ucl.ac.uk)

47.1 Quelles modifications dois-je apporter à src/Makefile.global ou src/Makefile.custom?

Ces modifications sont réalisées plus facilement en exécutant le script d'adaptation dans le répertoire src qui créera un fichier Makefile.custom pour vous.

Si vous le faites par vous-même, vous *devez* renseigner la variable suivante : PORTNAME= linux

Il vous faudra aussi modifier celle-ci pour refléter votre propre installation: POSTGRESDIR

Si vous validez l'option USE_TCL option, vous devrez renseigner celles-ci : TCL_INCDIR= TCL_LIBDIR= TCL_LIB= TK_INCDIR= TK_LIBDIR= TK_LIB= X11_INCDIR= X11_LIBDIR= X11_LIB=

Sur mon système Slackware3.0, ces données sont : TCL_INCDIR= /usr/include/tcl TCL_LIBDIR= /usr/lib TCL_LIB= -ltcl TK_INCDIR= /usr/include/tcl TK_LIBDIR= /usr/lib TK_LIB= -ltk X11_INCDIR= /usr/include/X11 X11_LIBDIR= /usr/X386/lib X11_LIB= -lX11

Vous pouvez aussi effectuer d'autres modifications dont vous avez besoin, ceci étant documenté dans les fichiers the INSTALL file and in Makefile.global

47.2 Pourquoi ai-je des problèmes avec "libreadline" manquant?

Les systèmes Linux ne sont généralement pas livrés avec la bibliothèque GNU readline installée. Soit vous vous assurez que vous n'avez pas activé les options readline dans src/Makefile.global ou src/Makefile.custom soit vous installez la bibliothèque GNU readline.

Notez que Debian Linux (comme FreeBSD) est livré avec readline installé.

47.3 *REDHAT* Pourquoi ai-je des problèmes avec libdl and dlfcn.h manquants?

La bibliothèque libdl est utilisée pour la liaison dynamique de fonctions fournies par l'utilisateurs au moment de l'exécution. Pour une raison quelconque, cette bibliothèque a été oubliée dans la distribution Redhat. Il semble que ce problème a été résolu dans la version Redhat 4.0 (Colgate).

RedHat possède maintenant un nouveau paquetage ld.so RPM sur son site FTP. Récupérez simplement :

`(ftp://ftp.redhat.com/pub/redhat/devel/i386/RedHat/RPMS/ld.so-1.7.14-4.i386.rpm)`

Installez le fichier RPM de la manière habituelle et c'est tout!

Il y a eu des rapports indiquant que des systèmes ont été abîmés suite à l'accès à ces bibliothèques par certains programmes pendant leur mise à jour (ce qui n'est pas tout à fait surprenant). En conséquence, c'est une bonne idée de redémarrer le système avant l'installation de nouvelle bibliothèques et d'avoir le minimum de choses en fonction pendant ces mises à niveau. Passer en mode mono-utilisateur est certainement aussi une bonne idée!

Si vous voulez le faire de la manière difficile, vous pouvez obtenir la bibliothèque et le fichier en-tête à :

`(ftp://tsx-11.mit.edu/pub/linux/packages/GCC/ld.so-1.7.14.tar.gz)`

Autrement, vous pouvez obtenir les binaires précompilés dans distributions/debian/buzz/binary-i386/base/ld.so-1.7.14-4.deb sur le même site, ou suivre les instructions données pour la question 1.2 de correction de la même erreur avec les premières versions de Slackware 3.1. Ne pas utiliser cette méthode, à moins que vous sachiez ce que vous faites!

47.4 $SLACKWARE3.1$ Pourquoi ai-je des problèmes avec libdl and dlfcn.h manquants ?

Voir la réponse à la question 1.3. Slackware jusqu'à la version 3.0 était fourni avec cette bibliothèque et le fichier "include" et ils ont semblé revenir en arrière avec les dernières versions 3.1, mais les premières livraisons 3.1 (avant le 9 Septembre 1996) ne les avaient pas et beaucoup de versions CD-ROM ont été réalisés à partir de ces premières versions 3.1.

Il y a eu des rapports indiquant que des systèmes ont été abîmés suite à l'accès à ces bibliothèques par certains programmes pendant leur mise à jour (ce qui n'est pas tout à fait surprenant). En conséquence, c'est une bonne idée de redémarrer le système avant l'installation de nouvelle bibliothèques et d'avoir le minimum de choses en fonction pendant ces mises à niveau. Passer en mode mono-utilisateur est certainement aussi une bonne idée!

La meilleure manière de corriger cela est d'obtenir le fichier ldso.tgz du disque a4 d'une distribution plus récente de Slackware 3.1 de décompacter ce fichier à partir du répertoire root (/), puis de faire

sh install/doinst.sh

pour terminer l'installation, à la suite de cela, faites

ldconfig

Si vous voulez l'installer manuellement, vous devez d'abord installer le fichier dlfcn.h in /usr/include.

Puis, installer le fichier libdl.so.1.7.14 (ou une version plus récente) dans /lib, puis faire :

cd /lib ln -sf libdl.so.1.7.14 libdl.so.1 ln -sf libdl.so.1 libdl.so

Sur quelques systèmes (dépendant de la configuration de GCC) il peut être nécessaire de faire :

cd /usr/lib ln -sf /lib/libdl.so .

Finalement

ldconfig

47.5 Ma compilation du processus d'arrière plan échoue réclamant le "fichier include" dlfcn.h missing

Voir la réponse aux questions 1.3/1.4. N'oubliez pas que si vous utilisez un système a.out vous devez auparavant avoir installé le paquetage dld (qui n'est pas fourni avec la plupart des systèmes a.out) pour avoir dlfcn.h en entier. Voir Question 1.11.

47.6 GCC se plaint d'une option ignorée -fpic

Le première versions de GCC acceptaient soit -fpic soit -fPIC. Il apparaît que les plus récentes versions (V2.7.2?) requièrent -fPIC. Si vous utilisez une version ELF de Linux, on peut ignorer cela sans problème, puisque -fPIC est la valeur par défaut.

Vous pouvez apporter une correction à cela en éditant le fichier src/Makefile.global et en changeant CFLAGS_SL

47.7 J'ai des messages d'avertissement du type "warning- cast from pointer to integer of different size"

Ils apparaissaient dans les premières versions de Postgres95 et peuvent, sans risque, être ignorés. PostgreSQL V6.0 doit pouvoir se compiler sans messages d'avertissement sauf ceux relatifs aux fichiers en-têtes système (qui peuvent également être ignorés sans risque).

47.8 $SuSE - Linux4.2$ Où sont curses et termcap?

SuSE-Linux a ncurses et non pas curses. Fixer la valeur de CURSES_LIB dans src/Makefile.custom à -lncurses (ou faites ceci par l'intermédiaire d'un script personnalisé).

SuSE-Linux possède la bibliothèque termcap library dans /usr/lib/termcap au lieu de /usr/lib. Si vous avez un problème, il vous faut ajouter la ligne suivante à src/Makefile.custom:

LDADD_BE+= -L/usr/lib/termcap

Vous pouvez avoir besoin d'éditer src/bin/psql/Makefile et d'effectuer la modification de:

ifeq $((PORTNAME), linux) LD\_ADD+ = -ltermcapen : ifeq($(PORTNAME), linux) LD_ADD+=

47.9 Pourquoi ai-je des problèmes avec ld.so?

Si vous avez des problèmes avec ld.so, une autre bibliothèque nécessaire en format ELF pour les liens dynamiques, alors c'est que vous avez bricolé votre système ou (plus vraisemblablement) effectué une mise à niveau de Linux.

Voyez les réponses aux questions 1.3/1.4. Vous pouvez avoir besoin d'installer ld.so.x.y.z dans /lib et de faire tourner ldconfig.

La version stable la plus récente du paquetage ld package est 1.7.14 Au moment de la rédaction de ce texte, les versions 1.8.x de ld sont expérimentales.

47.10 Pourquoi ai-je les erreurs 'yy_flush_buffer undefined'?

Ce n'est pas réellement spécifique à Linux, mais c'est courant sur les installations Linux les plus anciennes. Il vous faut une version récente de flex (2.5.2 ou suivantes) pour compiler PostgreSQL. Notez que flex 2.5.3 a un bogue: voir la Question 3.4.

47.11 Comment dois-je compiler PostgreSQL sur un système a.out?

Premièrement, vous devez installer la bibliothèque dld. On peut l'obtenir sur le site Sunsite sous le nom : Linux/libs/dld.3.2.7.tar.gz

Deuxièmement, ajouter la ligne suivante au fichier src/Makefile.custom: LINUX_ELF= (ou utilisez un script personnalisé)

47.12 Pourquoi make échoue-t-il avec - yacc -d /disk2/PostgreSQL/src/backend/parser/gram.y make- /usr/bin/make- cannot execute binary file

Ce fut un problème des premières versions de Postgres95. Pour PostgreSQL on utilise bison -y par défaut plutôt que yacc.

yacc est généralement implanté comme script qui invoque bison -y Pour une raison quelconque (certaines versions de make? certaines versions de bash?) make est incapable d'exécuter ce fichier script.

Pour corriger cela, éditez simplement src/mk/port/postgres.mk.linux et, à la fin du fichier, changez : # YACC = bison -y en YACC = bison -y

47.13 Que sont les références à libsocket dans X11_LIB et libnsl dans src/Makefile.global?

Ce fut un problème dans la version 1.08 (elles sont spécifiques à Sun Solaris). Cela a été corrigé dans 1.09 et 6.0

47.14 *DEBIAN* Où se trouve libtermcap?

Le Linux de Debian est livré sans bibliothèque termcap et utilise ncurses (qui à la place utilise terminfo). Il n'y a pas besoin de modifier la variable CURSES_LIB dans src/bin/psql/Makefile puisque Debian fournit un lien de libncurses vers libcurses (à la différence de SuSE-Linux — voir la Question 1.8).

Vous pouvez avoir besoin d'éditer src/bin/psql/Makefile et de changer:

ifeq $((PORTNAME), linux) LD\_ADD+ = -ltermcap en : ifeq($(PORTNAME), linux) LD_ADD+=

48 FAQ Linux-PostgreSQL - Compilation des programmes accessoires

48.1 L'éditeur de liens n'arrive pas à trouver libX11 quand on compile pgtclsh

Ajoutez ce qui suit à src/Makefile.custom X11_LIBDIR = /usr/X11R6/lib

49 FAQ Linux-PostgreSQL - Problèmes d'exécution

49.1 J'ai une erreur indiquant _fUnKy_POSTPORT_sTuFf_ non défini quand j'exécute les scripts tels que createuser

C'est un bogue dans la V1.06-V1.07 de Postgres et il est corrigé dans la version V1.08 et au-dessus.

49.2 Je lance postmaster et ensuite le système affiche 'Bad system call(Core dumped)'

Ceci indique que vous n'avez pas compilé le support de la mémoire partagée dans votre noyau. Il vous faut recompiler le noyau Linux en incluant cette fonctionnalité.

49.3 Quand j'essaie de lancer le Postmaster, pourquoi ai-je une erreur de la forme Failed Assertion("!(file != 0) - (null)", File

Failed Assertion("!(file != 0):(null)", File: "/usr/local/PostgreSQL/src/backend/storage/file/fd.c", Line: 257) !(file != 0) (0) initdb: could not create template database initdb: cleaning up.

Vos permissions sur le fichier /dev/null ne sont pas bonnes.

ls -l /dev/null devrait vous donner quelque chose comme:

crw-rw-rw- 1 root wheel 2, 2 Oct 8 18:41 /dev/null

Corrigez les permissions en utilisant :

chmod a+rw /dev/null

49.4 Pourquoi createuser ne fonctionne-t-il pas?

Il y a un problème avec la Version 2.5.3 de GNU flex et de createuser. Vos options sont de revenir à la version V2.5.2 de flex, d'appliquer une rustine à la version V2.5.3 (fournie dans doc/README.flex) ou d'attendre la version V2.5.4 qui corrigera ce bogue.

49.5 Pourquoi ai-je une erreur comme IpcMemoryCreate- memKey=155356396 , size=760632 ,

permission=384IpcMemoryCreate: shmget(..., create, ...) failed: Invalid argument

Vous n'avez pas inclus le support IPC dans votre noyau Linux. Il vous faudra reconstruire le noyau et valider cette option.

49.6 Pourquoi psql échoue avec psql- can't load library 'libpq.so.1'

Psql a été compilé pour être lié dynamiquement avec la bibliothèque libpq.

Pour remédier à cela, vous devez vous connecter comme root et éditer le fichier /etc/ld.so.conf Ajouter une simple ligne à la fin qui donne le nom du répertoire de la bibliothèque PostgreSQL (le sous-répertoire lib de votre installation pour PostgreSQL) et exécuter /sbin/ldconfig

50 FAQ IRIX-PostgreSQL - Compilation de PostgreSQL

=== Foire Aux Questions (FAQ) for PostgresSQL V6.1 Spécifique IRIX A LIRE EN PARALLELE AVEC LA FAQ NORMALE === Dernière mise à jour: Vendredi 13 Juin 09:54:00 BST 1997

Gestionnaire actuel: Andrew C.R. Martin (martin@biochem.ucl.ac.uk) Auteur original author: Andrew C.R. Martin (martin@biochem.ucl.ac.uk)

Modifications dans cette version (* = modifié, + = nouveau, - = supprimé): +1.5) Puis-je installer PostgreSQL sous Irix 6.4?

Ce fichier est divisé approximativement comme suit :

- 1.*) Installation de Postgres95
- 2.*) Désinstallation de Postgres95
- 3.*) Extension de Postgres95

51 FAQ-IRIX Installation de Postgres95

51.1 De quels éléments supplémentaires ai-je besoin pour installer Postgres95 sous Irix?

Vous *devez* avoir les éléments suivants installés:

- a) Gnu make (installé sous le nom gmake)

Nous vous recommandons d'installer ce qui suit :

- a) GNU install (installé sous le nom ginstall)

Vous pouvez choisir d'installer :

- a) La bibliothèque GNU readline (si vous désirez que psql ait le support de readline).
- b) Tcl/Tk (si vous souhaitez compiler pgtclsh)

51.2 Quelles modifications dois-je apporter à src/Makefile.global ou src/Makefile.custom?

La manière la plus simple de faire cela est d'utiliser le script personnalisé dans le répertoire src.

Vous *devez* renseigner les variables suivantes : PORTNAME= irix5

Vous aurez aussi besoin de changer ce qui suit pour s'adapter à votre propre installation : POSTGRESDIR

Si vous activez l'option USE_TCL, vous aurez besoin de renseigner: TCL_INCDIR= TCL_LIBDIR= TCL_LIB = TK_INCDIR= TK_LIBDIR= TK_LIB =

Vous pouvez également faire toutes les autres modifications dont vous avez besoin comme indiqué dans le fichier INSTALL file et dans Makefile.global

51.3 Que sont les références a libsocket dans X11_LIB et à libnsl dans src/Makefile.global?

Ce fut un problème dans la version 1.08 (elles sont spécifiques à Sun Solaris). Cela a été corrigé dans 1.09 et 6.0

51.4 Are there any other changes I should make?

Si vous avez installé le programme GNU install (ginstall), vous devriez ajouter la ligne suivante à src/Makefile.custom: CUSTOM_INSTALL=ginstall

Pour une explication sur ce point, voir la Question 2.1

51.5 Puis-je installer PostgreSQL sous Irix 6.4?

Irix 6.4 a un bogue dans ld qui l'empêche de prendre en compte correctement les adresses des procédures statiques quand les fichiers objets sont assemblés en fichiers objets plus grands en utilisant 'ld -r'. Ce bogue a été rapporté à Silicon Graphics. La rustine suivante peut être appliquée pour le contourner. (Fourni par Bob Bruccoleri bruc@bms.com)

*** ./backend/Makefile.orig Thu May 22 00:00:15 1997 — ./backend/Makefile Thu Jun 5 16:47:27 1997 *************** *** 54,60 **** all: postgres $ (POSTGRES_IMP) global1.bki.source local1_template1.bki.source

postgres: $ (OBJS) ../utils/version.o ! $ (CC) -o postgres $ (OBJS) ../utils/version.o $ (LDFLAGS)

$ (OBJS): $ (DIRS:%=%.dir)

— 54,64 —- all: postgres $ (POSTGRES_IMP) global1.bki.source local1_template1.bki.source

postgres: $ (OBJS) ../utils/version.o

```
! #      \$ (CC) -o postgres  \$ (OBJS) ../utils/version.o  \$ (LDFLAGS)
!       -rm -f *.o
!       find . -name "*.o" -exec cp (flower-brackets) . \;
```

! rm -f SUBSYS.o ! $ (CC) -o postgres *.o ../utils/version.o $ (LDFLAGS)

$ (OBJS): $ (DIRS:%=%.dir)

52 FAQ-IRIX Désinstallation de Postgres95

52.1 Pourquoi ne puis-je pas déplacer les fichiers exécutables?

Par défaut, le portage IRIX utilise la version compatible BSD du programme install de /usr/bin/X11. Si vous lisez la page de manuel pour cette version d'install, vous verrez que son utilisation n'est pas destinée à l'utilisateur final; il possède l'effet de bord intéressant de changer le propriétaire des fichiers qu'ils installe en root.

Vous devriez encore pouvoir détruire les fichiers puisque vous (utilisateur postgres) êtes propriétaire du répertoire dans lequel ils sont stockés.

Le programme install normal de IRIX ne peut pas être utilisé facilement car il prend ses arguments en ordre inverse. De ce fait, on recommande d'utiliser la version GNU du programme install (ginstall). Voir la Question 1.4

53 FAQ-IRIX Extending Postgres95

53.1 Comment dois-je compiler un programme C pour créer une fonction d'extension à Postgres95

Voici une ligne de commande en exemple :

cc -I/usr/local/postgres95/include/ -I/usr/local/postgres95/src/backend -shared -o funcs.so funcs.c

54 FAQ-IRIX PostgreSQL contact des AUTEURS

Dr. Andrew C.R. Martin University College Londres

EMAIL: (Work) martin@biochem.ucl.ac.uk

(Home) andrew@stagleys.demon.co.uk

URL: `(http://www.biochem.ucl.ac.uk/ martin)`

Tel: (Work) +44(0)171 419 3890

Tel: (Home) +44(0)1372 275775

55 Notice de Copyright