1

Bienvenue dans le monde SQL

Bienvenue dans le monde en perpétuel mouvement du langage SQL et des technologies de bases de données. En lisant ce livre, vous avez pris le parti d'accepter qu'elles seront bientôt indispensables pour survivre dans le monde actuel des bases de données relationnelles et de l'administration des données. Malheureusement, avant d'entrer dans le vif du sujet, il est important de commencer par poser les bases de SQL et de traiter quelques concepts préliminaires indispensables.

Au sommaire de ce chapitre

- SQL : introduction et bref historique
- Introduction à la notion de SGBD (système de gestion de base de données)
- Définition de termes et de concepts-clés
- Présentation de la base de données utilisée dans cet ouvrage

1.1 SQL : introduction et bref historique

Toute activité commerciale s'appuie sur des données, qui ont besoin d'une méthode d'organisation ou de conservation. Lorsque, pour ce faire, on utilise une base de données, le mécanisme est appelé *système de gestion de base de données* (SGBD, en anglais DBMS pour *DataBase Management System*). Les SGBD existent depuis de nombreuses années. À l'origine, la plupart étaient des systèmes traitant des fichiers non relationnels sur mainframe. Aujourd'hui, l'accroissement des activités commerciales et des volumes de données ainsi que les technologies de l'Internet étendent les fonctionnalités des systèmes de gestion de base de données dans d'autres directions.

La nouvelle vague du traitement de l'information est conduite essentiellement par les *systèmes de gestion de base de données relationnelle* (SGBDR, en anglais RDBMS pour *Relational DataBase Management System*), dérivés des traditionnels SGBD. Les entreprises d'aujourd'hui font appel aux bases de données relationnelles, aux technologies client-serveur et bien sûr aux technologies de l'Internet pour gérer leurs données et assurer leur compétitivité. Les prochaines sections traitent du langage SQL et des bases de données relationnelles. Une bonne connaissance du concept de bases de données relationnelles et de l'utilisation de SQL

dans le monde actuel des technologies de l'information est indispensable à une bonne compréhension du langage.

1.1.1 Définition de SQL

SQL (*Structured Query Language*, langage de requête structuré) est le langage standard utilisé pour communiquer avec une base de données relationnelle. Le prototype d'origine a été développé par IBM à partir de l'article du Dr. E.F. Codd, *A Relational Model of Data for Large Shared Data Banks*. En 1979, peu de temps après la sortie du prototype d'IBM, le premier produit SQL, Oracle, a été mis sur le marché par Relational Software Incorporated (rebaptisé plus tard Oracle Corporation). Oracle est désormais devenu l'un des principaux acteurs du marché des technologies de base de données relationnelle.

Si vous visitez un pays étranger, il vous sera sans doute nécessaire de connaître la langue de ce pays. Par exemple, vous aurez peut-être quelques difficultés à commander un menu dans votre langue maternelle si le maître d'hôtel ne parle que la langue du pays. Considérez une base de données comme un pays étranger dans lequel vous recherchez des informations. La langue « étrangère » avec laquelle vous exprimez vos besoins auprès d'une base de données prend la forme d'une requête en langage SQL.

1.1.2 SQL à la norme ANSI

L'ANSI (*American National Standards Institute*) est un organisme qui approuve certains standards de l'industrie américaine. SQL a été défini comme langage standard de communication des bases de données relationnelles, depuis 1986, à partir de l'implémentation faite par IBM. En 1987, le standard SQL ANSI a été accepté au niveau international par l'ISO (*International Standards Organization*). Le standard a subi une révision en 1992 (nommé SQL-92) et une nouvelle fois en 1999 (nommé SQL-99). Le standard le plus récent se nomme SQL-2008 et a été adopté officiellement en juillet 2008.

1.1.3 SQL-2008 : le nouveau standard

SQL-2008 possède neuf documents (numérotés 1, 2, 3, 4, 9, 10, 11, 13, 14) reliés entre eux et susceptibles d'être complétés ultérieurement en fonction de l'évolution des technologies :

- Partie 1 – *SQL/Framework* (cadre) : spécifie les principales conditions de conformité et les concepts fondamentaux de SQL.
- Partie 2 – *SQL/Foundation* (fondements) : définit la syntaxe et le fonctionnement de SQL.
- Partie 3 – *SQL/Call-Level Interface* (interface d'appel) : définit l'interface de programmation d'applications SQL.
- Partie 4 – *SQL/Persistent Stored Modules* (modules persistants) : définit les structures de contrôle qui spécifient les routines SQL. La partie 4 définit également des modules contenant les routines SQL.
- Partie 9 – *Management of External Data* (gestion des données externes) : définit les extensions apportées au langage SQL pour la gestion des données externes par l'utilisation de « data-wrapper » (encapsulation) et du type de données « data-link » (liens vers les données).

- Partie 10 : *Object Language Bindings* (liens vers les langages à objets) : définit les extensions apportées au langage SQL pour supporter l'intégration de requêtes SQL dans les programmes écrits en Java.
- Partie 11 : *Information and Definition Schemas* (schémas d'information et de définition) : définit les spécifications pour les schémas d'information et de définition qui fournissent des indications sur la structure et la sécurité des données SQL.
- Partie 13 : *Routines and Types Using the Java Programming Language* (routines et types qui utilisent le langage Java) : définit la possibilité d'utiliser des routines et classes d'objets Java comme des routines SQL.
- Partie 14 : *XML-Related Specifications* (spécifications liées à XML) : définit la manière d'utiliser SQL avec XML.

Le nouveau standard ANSI (SQL-2008) comprend deux niveaux de compatibilité minimale pouvant être revendiqués par un SGBD : *Core SQL Support* et *Enhanced SQL Support*, autrement dit le support du SQL de base et le support optimisé de SQL.

Tout standard s'accompagne d'avantages nombreux et évidents, avec quelques inconvénients. Principalement, un standard oriente les constructeurs dans la direction industrielle propice au développement. À ce titre, le langage SQL propose une structure de prérequis fondamentaux qui, au final, contribuent à préserver une certaine logique entre différentes implémentations et à engendrer une portabilité accrue (non seulement pour les programmes de bases de données, mais pour les bases de données en général et les personnes qui les gèrent).

Certains diront qu'il n'est pas bon d'instaurer un standard, dans la mesure où celui-ci n'est pas flexible et où il limite les fonctionnalités d'une implémentation particulière. La plupart des constructeurs, cependant, se conforment au standard tout en y ajoutant des améliorations propriétaires pour pallier ses points faibles.

La qualité d'un standard se mesure à ses avantages et inconvénients. Ce que l'on attend d'un standard est de décrire les fonctionnalités qui doivent être présentes dans une implémentation de SQL, mais aussi de souligner les concepts fondamentaux qui forcent à respecter une certaine cohérence entre les différentes implémentations de SQL et contribuent à valoriser les compétences des programmeurs SQL.

Une *implémentation SQL* est le produit SQL d'un éditeur de logiciels particulier. Comme on le rappellera tout au long de l'ouvrage, les implémentations de SQL varient dans de larges mesures. Il n'existe pas d'implémentations qui respectent totalement le standard SQL, même si la plupart peuvent justifier d'une « compatibilité ANSI ». Comme le nombre de fonctionnalités nécessaires pour se conformer au standard n'a guère changé durant ces dernières années, la plupart des SGBD actuels sont désormais compatibles.

1.1.4 Définition d'une base de données

Pour formuler les choses simplement, disons qu'une *base de données* est un ensemble de données. Certains considèrent une base de données comme un mécanisme organisé ayant la capacité de stocker des informations et permettant à un utilisateur de retrouver efficacement des informations stockées.

On exploite quotidiennement des bases de données sans même s'en rendre compte. Un annuaire est une base de données. Les données sont les noms de personnes, les adresses et les numéros de téléphone. Les listes sont classées par ordre alphabétique ou indexées. Ainsi, un

utilisateur peut facilement retrouver un habitant particulier. Enfin, ces données sont stockées quelque part dans une base de données sur un ordinateur, étant entendu que chaque page d'un annuaire n'est pas saisie manuellement à chaque nouvelle édition.

La base de données doit être tenue à jour. À chaque changement de situation, il est nécessaire d'ajouter, de modifier ou de supprimer les entrées d'un annuaire : changement de nom, déménagement, etc. La figure 1.1 illustre une base de données simple.

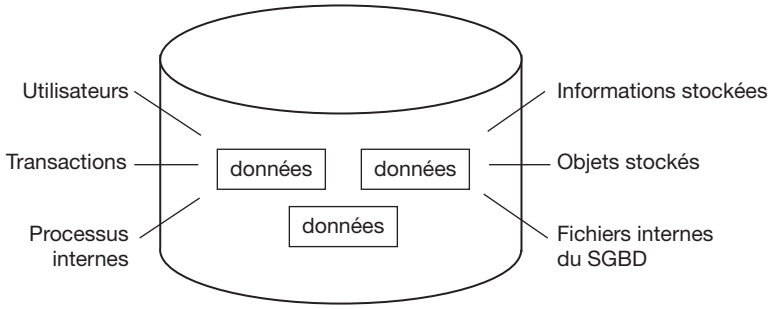

Figure 1.1 • La base de données.

1.1.5 Introduction aux bases de données relationnelles

Une *base de données relationnelle* est une base de données divisée en unités logiques appelées *tables*, en relation les unes avec les autres au sein de la base. Une base de données relationnelle permet de diviser les données en unités logiques plus petites et plus simples à gérer. Il en résulte une maintenance simplifiée et une amélioration des performances en fonction du niveau d'organisation. À la figure 1.2, vous pouvez voir que les tables sont en relation les unes avec les autres par le biais d'une clé commune.

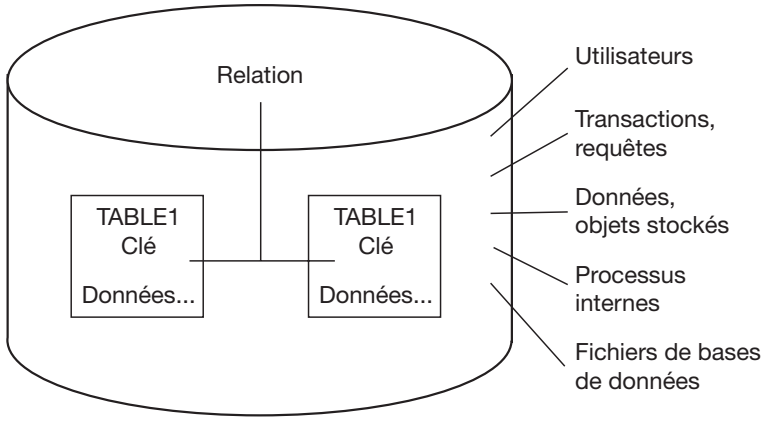

Figure 1.2 • La base de données relationnelle.

Ces relations entre les tables permettent de retrouver les données adéquates par simple requête, sachant que les données recherchées peuvent se trouver dans plusieurs tables. Avec des clés ou champs communs à plusieurs tables d'une base de données relationnelle, les données des différentes tables peuvent être assemblées pour former un résultat unique. Plus vous avancerez dans la lecture de ce livre, plus vous trouverez d'avantages aux bases de données relationnelles, notamment en matière de performances globales et de simplification de l'accès aux données.

1.1.6 Introduction à la technologie client-serveur

Par le passé, l'industrie informatique était principalement dominée par les ordinateurs mainframe, des systèmes puissants disposant de capacités considérables de stockage et de traitement des données. Les utilisateurs communiquaient avec le mainframe par de simples terminaux dépourvus de capacité propre, mais totalement dépendants de celles du mainframe en termes de stockage, mémoire et traitement. Chaque terminal disposait d'une ligne de données reliée au mainframe. L'environnement mainframe remplit toujours parfaitement son rôle aujourd'hui, malgré l'avènement d'une technologie plus performante : le modèle client-serveur.

Dans le *système client-serveur*, l'ordinateur principal, appelé s*erveur*, est accessible par un réseau, généralement local (LAN pour *Local Area Network*), mais également distant (WAN pour *Wide Area Network*). On accède au serveur depuis son ordinateur personnel (PC) ou par le biais d'autres serveurs en lieu et place des anciens terminaux. Chaque PC, appelé *client*, dispose d'un accès au réseau, permettant la communication entre le client et le serveur, ce qui explique l'expression « client-serveur ». La principale différence entre les environnements client-serveur et mainframe réside dans le fait que, dans l'environnement client-serveur, le PC de l'utilisateur est doté de capacités propres grâce auxquelles il peut exécuter ses propres processus en utilisant son processeur tout en disposant d'un accès en lecture à un ordinateur serveur *via* le réseau. Dans bien des situations, le système client-serveur offre une plus grande souplesse au regard des besoins actuels et est à même de remplacer le mainframe.

Les systèmes de bases de données relationnelles sont installés sur divers plates-formes et systèmes d'exploitation. Les systèmes d'exploitation les plus répandus sont Windows, Linux et d'autres systèmes utilisant les lignes de commande comme la famille Unix. Actuellement, la plupart des systèmes de bases de données reposent sur le modèle client-serveur ou des environnements Web.

Le manque de formation et d'expérience sont les principales raisons des implémentations ratées dans le domaine des bases de données. Quoi qu'il en soit, une bonne compréhension du modèle client-serveur et de l'architecture basée sur le Web est impérative face à l'accroissement parfois irraisonné des besoins des entreprises ainsi qu'au développement des technologies de l'Internet et des réseaux informatiques. La figure 1.3 illustre le concept de la technologie client-serveur.

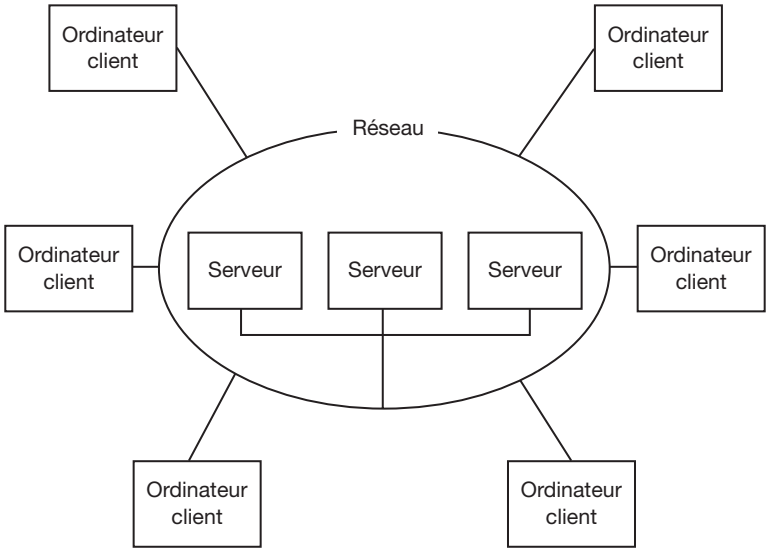

Figure 1.3 • Le modèle client-serveur.

1.1.7 Les systèmes de bases de données sur le Web

Les systèmes d'information professionnels s'orientent rapidement vers une intégration Web. Les bases de données sont rendues accessibles au travers de l'Internet, ce qui signifie que l'accès aux informations s'effectue par des navigateurs tels que Internet Explorer ou Firefox. On donne la possibilité aux clients de commander des marchandises, de vérifier les stocks, de consulter l'état d'une commande, de modifier les informations de leur compte, de transférer de l'argent d'un compte à un autre, et ainsi de suite.

Le client lance simplement un navigateur et accède au site Web de l'entreprise, se connecte si nécessaire, et peut utiliser l'application développée sur le site pour accéder aux données. La plupart des sites requièrent que l'on s'identifie et fournissent un « compte/mot de passe » à cet effet.

De nombreuses opérations ont lieu en arrière-plan lorsque l'on accède à une base de données *via* un navigateur Web. On peut faire exécuter des requêtes en langage SQL par les applications Web. Cette requête SQL permet d'accéder au système d'information de l'entreprise, de retourner le résultat au serveur Web qui à son tour le retransmet au navigateur du client.

La structure fondamentale d'un système de base de données sur le Web est comparable, du point de vue de l'utilisateur, à celle d'un client-serveur (voir figure 1.3). Chaque utilisateur utilise une machine cliente munie d'un navigateur et dotée d'une connexion à l'Internet. Dans le cas d'un système d'information accessible par le Web, le réseau de la figure 1.3 représente l'Internet par opposition à un réseau local. Dans la majorité des cas, le client accède aux données en interrogeant un serveur. Peu importe que le serveur soit situé dans un autre pays. L'essentiel pour un système de base de données sur le Web réside en la possibilité d'étendre le nombre d'utilisateurs potentiels d'un système d'information sans tenir compte des limitations physiques. Ainsi, on peut augmenter la disponibilité des données et son nombre d'utilisateurs.

1.1.8 Principaux éditeurs de bases de données relationnelles

Parmi les éditeurs de bases de données prédominants, on trouve Oracle, Microsoft, Informix, Sybase et IBM. Ils distribuent différentes versions payantes de système de base de données relationnelle et sont généralement de type « source-closed » (code source non publié). D'autres éditeurs proposent des versions « open source » (code source publié) de SGBD SQL, comme MySQL, PostgreSql et SAP. Il existe bien d'autres éditeurs que ceux mentionnés ici. Vous les connaissez pour les avoir vus dans un livre, un journal, des magazines, à la Bourse ou sur le Web.

Les implémentations SQL spécifiques à chaque éditeur sont uniques dans leurs fonctionnalités et leur nature. Un serveur de base de données est un produit, comme tout autre produit sur le marché, développé par un large éventail d'éditeurs. Pour des raisons de portabilité et de commodité pour les utilisateurs, il est dans l'intérêt du constructeur d'assurer que son implémentation est compatible avec le standard ANSI actuel. Par exemple, si une entreprise migre d'un serveur de base de données à un autre, il pourrait être décourageant pour les utilisateurs d'avoir à apprendre un nouveau langage pour maintenir le même niveau de fonctionnalité du nouveau système.

Avec l'implémentation SQL de chaque constructeur, cependant, vous noterez des améliorations répondant à l'objectif de chaque serveur de base de données. Ces améliorations, ou extensions, sont des commandes et des options additionnelles qui viennent en supplément du langage SQL standard et qui sont disponibles avec une implémentation spécifique.

1.2 Initialiser une session SQL

Une *session SQL* est un moment pendant lequel un utilisateur interagit avec une base de données relationnelle *via* les commandes SQL. Lorsqu'un utilisateur se connecte à la base de données, il établit une session. Pendant une session SQL, des commandes SQL peuvent être saisies pour interroger la base de données, manipuler les données qui s'y trouvent et définir les structures de données, comme les tables.

1.2.1 CONNECT

Lorsque l'utilisateur se connecte à la base de données, la session SQL est initialisée. La commande `CONNECT` sert à établir la connexion avec la base de données. Cette commande permet d'invoquer une connexion ou de modifier les connexions à la base de données. Par exemple, si vous êtes connecté en tant que `UTILISATEUR1`, vous pouvez faire appel à la commande `CONNECT` pour vous connecter à la base de données en tant que `UTILISATEUR2`. Dans ce cas, la session SQL de `UTILISATEUR1` est implicitement déconnectée.

```
CONNECT utilisateur@base_de_données
```

Lorsque vous essayez de vous connecter à une base de données, il vous est automatiquement demandé un mot de passe correspondant à votre nom d'utilisateur actif.

1.2.2 DISCONNECT et EXIT

Lorsqu'un utilisateur se déconnecte d'une base de données, la session SQL prend fin. La commande `DISCONNECT` sert à déconnecter l'utilisateur de la base de données. Quand vous vous déconnectez, vous vous trouvez toujours dans l'outil du SGBD permettant de communiquer avec une base de données, mais vous n'êtes plus connecté. Si vous utilisez `EXIT` pour quitter la base de données, votre session SQL se termine et l'outil que vous employez pour accéder à la base de données est normalement fermé.

```
DISCONNECT
```

1.3 Types de commandes SQL

Les sections suivantes traitent des principales catégories de commandes SQL permettant d'accomplir différentes fonctions : construire des objets de base de données, manipuler des objets, entrer des données dans les tables de la base de données, actualiser des données existantes dans les tables, supprimer des données, lancer des requêtes, contrôler l'accès à la base de données et toute autre fonction d'administration.

Voici les principales catégories :

- DDL (*Data Definition Language*, langage de définition de données) ;
- DML (*Data Manipulation Language*, langage de manipulation de données) ;
- DQL (*Data Query Language*, langage de requête de données) ;
- DCL (*Data Control Language*, langage de contrôle de données) ;
- commandes d'administration des données ;
- commandes de contrôle transactionnel.

1.3.1 Définir les structures des objets de la base de données (DDL)

Le DDL (*Data Definition Language*) est la partie de SQL permettant à l'utilisateur de la base de données de créer et de restructurer les objets de la base, comme la création ou la suppression d'une table.

Voici les principales commandes DDL traitées dans les prochains chapitres :

```
CREATE TABLE
ALTER TABLE
DROP TABLE
CREATE INDEX
ALTER INDEX
DROP INDEX
CREATE VIEW
DROP VIEW
```

Ces commandes sont traitées en détail aux chapitres 3, « Gestion des objets de base de données », 17, « Optimisation des performances » et 20, « Vues et synonymes ».

1.3.2 Manipuler les données (DML)

Le DML (*Data Manipulation Language*) est la partie de SQL utilisée pour manipuler les données présentes dans les objets d'une base de données relationnelle.

Voici trois commandes DML de base :

```
INSERT
UPDATE
DELETE
```

Ces commandes sont traitées en détail au chapitre 5, « Manipulation des données ».

1.3.3 Sélectionner des données (DQL)

Bien que ne comprenant qu'une seule commande, le DQL (*Data Query Language*) est le plus utile pour l'utilisateur d'une base de données relationnelle. Voici cette commande :

```
SELECT
```

Cette commande, accompagnée de nombreuses options et clauses, sert à composer des requêtes avec une base de données relationnelle. Les requêtes ainsi créées vont de la plus simple à la plus complexe et de la plus générale à la plus spécifique. Une *requête* est une interrogation d'une base de données.

Voir chapitres 7 à 16 pour tous les détails de la commande `SELECT`.

1.3.4 Langage de contrôle des données (DCL)

Dans SQL, les commandes de contrôle des données permettent de contrôler l'accès aux données de la base de données. Ces commandes DCL (*Data Control Language*) servent principalement à créer des objets relatifs à l'accès utilisateur ainsi qu'au contrôle de la distribution des privilèges des utilisateurs. Voici quelques-unes des commandes de contrôle des données :

```
ALTER PASSWORD
GRANT
REVOKE
CREATE SYNONYM
```

Ces commandes sont souvent associées à d'autres commandes et sont reprises dans plusieurs chapitres.

1.3.5 Commandes d'administration de données

Les commandes d'administration de données permettent à l'utilisateur de réaliser des audits et des analyses d'opérations dans la base de données. Elles peuvent également servir à surveiller les performances du système. Voici les deux principales commandes d'administration des données :

```
START AUDIT
STOP AUDIT
```

Ne confondez pas l'administration des données et l'administration de la base de données. Cette dernière est l'administration globale de la base de données, comprenant tous les niveaux de commande. L'administration de données est en général spécifique à l'implémentation de SQL considérée, et les commandes ne font pas partie des fonctions de base de SQL.

1.3.6 Commandes de contrôle transactionnel

Outre les catégories de commandes que nous venons d'introduire, il existe des commandes grâce auxquelles l'utilisateur peut gérer les transactions de la base de données.

- `COMMIT` : enregistre les transactions de la base de données.
- `ROLLBACK` : annule les transactions de la base de données.
- `SAVEPOINT` : crée des points dans les groupes de transactions dans lesquels on fait appel à `ROLLBACK`.
- `SET TRANSACTION` : donne un nom à une transaction.

Les commandes transactionnelles sont traitées au chapitre 6, « Transactions de base de données ».

1.4 Présentation de la base de données utilisée dans cet ouvrage

Avant de poursuivre votre voyage dans les principes fondamentaux de SQL, nous allons vous présenter les tables et les données que vous utiliserez au cours des chapitres de ce livre. Les deux prochaines sections proposent un aperçu des tables spécifiques (la base de données) utilisées, leurs relations, leur structure et quelques exemples de données.

1.4.1 Diagramme des tables de ce livre

La figure 1.4 illustre les relations entre les tables utilisées dans les exemples de ce livre. Chaque table et chaque champ résidant dans une table est identifié par un nom. Suivez les liaisons pour comparer les relations entre les tables *via* un champ commun, que l'on nomme généralement *clé primaire*.

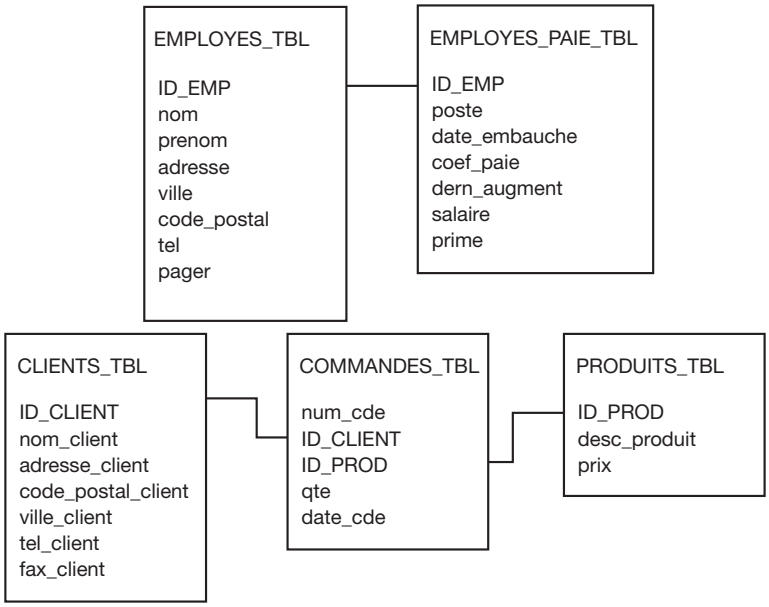

Figure 1.4 • Relations entre tables de ce livre.

Voir chapitre 3, « Gestion des objets de base de données ».

1.4.2 Standards de désignation des tables

Les standards de désignation des tables, à l'instar des standards dans les activités commerciales, sont cruciaux pour maîtriser le contrôle des informations. Après avoir étudié les tables et les données des sections précédentes, vous avez probablement noté que le suffixe de chaque table est `_TBL`. Il s'agit d'un standard de désignation exploité sur de nombreux sites. `_TBL` indique simplement que l'objet est une table. Il existe un grand nombre de types d'objets différents dans une base de données relationnelle. Par exemple, le suffixe `_INX` sert à identifier les index dans les tables des prochains chapitres. Les standards de désignation existent dans presque toutes les entreprises et simplifient énormément l'administration des bases de données relationnelles. Sachez toutefois qu'il n'est pas obligatoire d'utiliser ces suffixes particuliers pour la désignation des objets d'une base. Vous pouvez choisir librement le standard qui vous convient.

Info

Nommer les standards

Ne vous contentez pas de suivre la syntaxe de désignation d'objet des implémentations SQL. Prenez également en compte les règles internes à l'entreprise et créez des noms descriptifs en relation avec les groupes de données de l'entreprise.

1.4.3 Les données

Cette section expose les données exploitées dans ce livre. Prenez quelques minutes pour les étudier, ainsi que les différences et les relations entre les tables et les données elles-mêmes. Notez que certains champs n'ont pas de données, ce qui est spécifié lors de la création de la table concernée.

EMPLOYES_TBL

ID_EMP	NOM	PRENOM	ADRESSE	VILLE	CODE_POSTAL	TEL	PAGER
442346889	DESMARTIN	JEAN	53 RUE SAINT CHARLES	CRONENBOURG	67200	0388254012	NULL
313268956	SUGIER	KEVIN	300 AVE DE STALINGRAD	PARIS	75014	0140425698	NULL
213764555	STEPANIAN	KARINE	480 BLD GAMBETTA	MONTPELLIER	34000	0467025789	0456234595
313782439	CHASSEUR	DAVID	39 RUE DES VIOLETTES	PARIS	75010	0466568743	0322453412
220984332	CHRISTOPHE	SYLVIE	422 RUE PRINCIPALE	ALES	30100	0466589851	NULL
443679012	LEBIHEN	MAUD	31 AVE DU GAL DE GAULLE	PARIS	75005	0493568452	NULL

EMPLOYES_PAIE_TBL

ID_EMP	POSTE	DATE_EMBAUCHE	COEF_PAIE	DERN_AUGMENT	SALAIRE	PRIME
311549902	MARKETING	1989-05-23	0.00	1999-05-01	9999.99	NULL
442346889	CHEF EQUIPE	1990-06-17	14.75	1999-06-01	NULL	NULL
213764555	DIRECTEUR VENTE	1994-08-14	NULL	1999-08-14	9999.99	9999.99
313782439	COMMERCIAL	1997-06-28	NULL	NULL	9999.99	6000.00
220984332	EXPEDITEUR	1996-06-22	11.00	1999-07-01	NULL	NULL
443679012	EXPEDITEUR	1991-01-14	15.00	1999-01-01	NULL	NULL

CLIENTS_TBL

ID_CLIENT	NOM_CLIENT	ADRESSE_CLIENT	CODE_POSTAL_CLIENT	VILLE_CLIENT	TEL_CLIENT	FAX_CLIENT
232	BRASSERIE DU PECHEUR	62 RUE DU 23 NOVEMBR	67200	STRASBOURG	0388254012	NULL
109	CONSULTANTS REDACTION	23 RUE DU CHATEAU	75014	PARIS	0140425698	NULL
345	LES GRANDS MECHANTS LIVRES	125 AVE DES VOSGES	34000	MONTPELLIER	0467025789	NULL
090	SOLUTIONS INFORMATIQUES	43 RUE DES CHARMES	30000	NIMES	0466568743	NULL
12	ACADEMIE CEVENOLE DE DANSE	2 GRAND RUE	30100	ALES	0466589851	0466589852
432	LA MAIN TENDUE	31 AVE DU GAL CASTEL	06000	NICE	0493568452	0493568455
333	ACADEMIE BALLARD	231 RUE DE LA SOMME	67000	STRASBOURG	0388548591	0388655925
21	CONFISERIE MORGAN	45 BLD J.F. KENNEDY	94270	LE KREMLIN BICE	0140454698	NULL
43	LE FIL DU RASOIR	10 RUE MOZART	66000	PERPIGNAN	0468552363	NULL
287	POUPEES DECORATION	472 AVE DE L OCEAN	17110	ST SAINT GEORGES	0546025879	0546023564
288	CABINET JARDIN ET DURAND	23 ALLEE DES ORCHIDE	64000	PAU	0585649215	NULL

```
590     LES SPECIALISTES DU DESIGN      33 BLD ALFRED NOBEL     44600   SAINT NAZAIRE   0240705894      NULL
610     BOUTIQUE DE CADEAUX MARIE       83 PLACE SAINT SULPI    44000   NANTES  0250246980      0250253986
560     BAUGER BIOTECHNIQUE     56 RUE F. BUISSON       38000   GRENOBLE        0436598742      NULL
221     AMEUBLEMENT PIERRE      27 RUE ST SAINT CHRISTOPHE    25300   DIJON   0381560439      NULL


COMMANDES_TBL

NUM_CDE ID_CLIENT       ID_PROD QTE     DATE_CDE
56A901  232     11235   1       1999-10-22
56A917  12      907     100     1999-09-30
32A132  43      222     25      1999-10-10
16C17   090     222     2       1999-10-17
18D778  287     90      10      1999-10-17
23E934  432     13      20      1999-10-15

PRODUITS_TBL

ID_PROD DESC_PRODUIT    PRIX
11235   COSTUME SORCIERE        179.90
222     POUPEE PLASTIQUE 18 CM  46.50
13      FAUSSES DENTS PARAFFINE 6.60
907     LAMPION 87.00
15      COSTUMES ASSORTIS       60.00
9       POP-CORN CARAMEL        8.25
6       BONBONS POTIRON 9.45
87      ARAIGNEE PLASTIQUE      6.50
119     ASSORTIMENT DE MASQUES  29.90
```

1.4.4 Composition d'une table

Le stockage et la maintenance des données importantes sont la principale raison de l'existence des bases de données. Vous venez de voir les données utilisées pour expliquer les concepts SQL de ce livre. Les prochaines sections examinent plus en détail les éléments d'une table. Rappelez-vous qu'une table est la forme la plus courante et la plus simple du stockage de données dans une base de données relationnelle.

Un champ

Chaque table est divisée en plusieurs entités plus petites appelées *champs*. Les champs de la table `PRODUITS_TBL` sont `ID_PROD`, `DESC_PROD` et `PRIX`. Ces champs catégorisent les informations spécifiques contenues dans la table. Un *champ* est une colonne conçue pour héberger des informations spécifiques à chaque enregistrement de la table.

Un enregistrement ou ligne de données

Un *enregistrement*, également appelé *ligne de données*, correspond à chaque entrée existant dans la table. Observons le premier enregistrement de la dernière table, `PRODUITS_TBL` :

```
11235     COSTUME SORCIERE     179.90
```

Cet enregistrement est visiblement composé d'une identification du produit, d'une description et d'un coût unitaire. Pour chaque produit distinct, on doit trouver un enregistrement correspondant dans la table `PRODUITS_TBL`. Un enregistrement est une entité horizontale de la table.

Une *ligne de données* est un enregistrement complet dans une table de la base de données relationnelle.

Une colonne

Une *colonne* est une entité verticale de la table contenant toutes les informations associées à un champ spécifique de cette table. Voici, par exemple, la colonne de la table `PRODUITS_TBL` contenant la description des produits :

```
COSTUME SORCIERE
POUPEE PLASTIQUE 18 CM
FAUSSES DENTS PARAFFINE
LAMPION
COSTUMES ASSORTIS
POP-CORN CARAMEL
BONBONS POTIRON
ARAIGNEE PLASTIQUE
ASSORTIMENT DE MASQUES
```

Cette colonne est basée sur le champ `DESC_PROD`, la description du produit. Une colonne extrait l'ensemble des données d'un certain champ de tous les enregistrements de la table.

La clé primaire

La *clé primaire* est une colonne qui fait de chaque ligne de données dans la table une ligne unique dans la base de données relationnelle. La clé primaire de la table `PRODUITS_TBL` est `ID_PROD`. Elle est initialisée durant le processus de création de la table. Par nature, la clé primaire assure l'unicité de toutes les identifications, de manière que chaque enregistrement de la table `PRODUITS_TBL` dispose de sa propre identification `ID_PROD`. Les clés primaires diminuent les risques de doublons dans une table et sont également exploitées à d'autres fins, comme vous le verrez au chapitre 3, « Gestion des objets de base de données ».

Une valeur NULL

NULL est le terme utilisé pour représenter une valeur absente. Dans une table, NULL est la valeur d'un champ vide. Un champ contenant la valeur NULL ne contient pas de valeur. Il est très important de comprendre que la valeur NULL est différente de la valeur zéro ou d'un champ contenant des espaces. Un champ prenant la valeur NULL aura été laissé vide lors de la création de l'enregistrement. Notez que, dans la table EMPLOYES_PAIE_TBL, tous les employés n'ont pas de coefficient de paie. Les enregistrements des employés qui n'ont pas d'entrée pour le coefficient de paie prennent la valeur NULL dans ce champ.

Dans les prochains chapitres, vous aurez l'occasion de découvrir d'autres éléments composant les tables.

Exercices

1. Dans quelles catégories entrent les commandes SQL suivantes ?
   - **A.** CREATE TABLE
   - **B.** DELETE
   - **C.** SELECT
   - **D.** INSERT
   - **E.** ALTER TABLE
   - **F.** UPDATE

2. Étudiez les tables suivantes et sélectionnez la colonne la plus indiquée pour faire office de clé primaire :

EMPLOYES_TBL	INVENTAIRE_TBL	EQUIPEMENT_TBL
Nom	Produit	Modèle
Téléphone	Description	Année
Date début	Quantité	Numéro de série
Adresse	Numéro produit	Numéro équipement
Numéro employé	Emplacement	Affecté à