

**Connolly et Begg - Chapitres 6 et 7 Définition et manipulation de données**

Marc Philippe Parent Database Systems 6e de Connolly et Begg

# Base de données

**IFT-2004**

**Département d’informatique et de génie logiciel**

# Chapitres 6 et 7 - Objectifs

* Utilité et importance de SQL
* Types de données supportés par le standard SQL
* Utilité de la propriété d’amélioration d’intégrité
* Définir des contraintes d’intégrité avec SQL
* Gestion d’intégrité dans create et alter table
* MAJ une BD avec insert, update et delete
* Récupérer des données d’une BD avec select et
  + Utiliser des conditions where composées
  + Trier des résultats de requête avec order by
  + Utiliser des fonctions d’agrégation.

Connolly et Begg Database systems 6th 2

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Objectifs de SQL

* Langage de BD doit avec minimum \_\_\_effort\_\_\_
  + Créer \_\_\_\_structure\_\_\_\_\_ de BD + tables
  + Insertions, modifications, suppressions
  + Requêtes \_\_\_\_simples\_\_\_ ou complexes
* Structure et syntaxe relativement facile
* Suffisamment portable

|8:39| : travailler sur plusieurs SGBD de marque différentes.

* SQL
  + Langage orienté-\_\_\_\_transformation\_\_\_\_\_\_\_\_\_\_

|8:40| : cet a dire c’Est un langage qui a la base n’est pas la pour dire quoi faire, le langage SQL c’est plutot un langage pour dire: je veux ce résultat là. On dit le quoi mais on ne dit pas le comment.

* + 2 composants majeurs
    - \_\_DDL –»\_\_\_ Définir la structure de la BD

|8:43| : ma table client c’est : un attribut avec cette contrainte : DDL

* + - \_\_DML -»\_\_\_\_Récupérer et MAJ les données
    - |8:43| : Requete pour avoir un résultat. (manipuler les données)

Connolly et Begg Database systems 6th 3

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Objectifs de SQL

* Jusqu’à SQL3
  + SQL = pas de contrôle de \_\_flot\_\_ d’instructions, implémenté par
    - \_\_\_Autres\_\_\_ langages de programmation
    - Interactivement en fonction des décisions de l’utilisateur
* SQL relativement facile à apprendre
  + Non \_\_\_procédural\_\_\_\_\_\_\_ – Vouloir vs Comment
  + Essentiellement sans \_\_\_format\_\_\_
  + Consiste en des mots communs en anglais

select NUM\_PERSONNEL, NOM\_PER, SALAIRE\_PER

from PERSONNEL

where SALAIRE\_PER > 10000;

Connolly et Begg Database systems 6th 4

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Objectifs de SQL

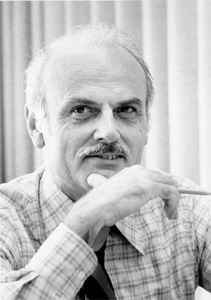
* Peut être utilisé par différents \_\_types\_\_\_ d’utilisateur
  + DBA
  + Gestionnaires
  + Développeurs d’application
  + Autres

|8:52| : C’est un standard ISO, Il est reconnu, Tous les fabricant qui font de relationnel utilise le SQL.

Connolly et Begg Database systems 6th 5

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Histoire de SQL



* 1970 1974

en 70 Frank Coyle

|8:54| : 1974: Bernard Cherbelling il travaillait au IBM, et invente un langage en se basant sur l’algebre relationnel de Coyle, et invente le langage SEQEL. Structured english langage

* 1976
  + Version révisée, SEQUEL/2
  + Nom changé pour SQL pour raisons \_\_\_légales\_\_\_\_
* En anglais
  + Encore prononcé ‘\_\_si-quel\_\_\_\_\_’
  + Prononciation officielle est ‘S-Q-L’
* IBM a ensuite produit un prototype de SGBD appelé System R, basé sur SEQUEL/2.

Connolly et Begg Database systems 6th 6

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Histoire de SQL

* 1979 -> \_\_Oracle\_\_\_\_ 1er SGBDr commercial basé sur SQL
* 1987 -> \_ANSI\_\_\_ et ISO publient un standard initial
* 1989 -> ISO: une addendum -> fonction amélioration \_intégrité\_\_\_
* 1992 -> SQL2 | SQL/92: 1ère révision majeure ISO
* 1999 -> SQL3 support pour la gestion de données OO
* 2003 -> SQL2003 \_\_XML\_ et auto-incrément
* 2006 -> SQL2006 plus de XML et \_\_\_Xquery\_\_\_
* 2008 -> SQL2008 détails sur \_\_\_trigger\_\_\_\_ et truncate
* \_\_2009\_\_ -> NoSQL = Not Only SQL

Connolly et Begg Database systems 6th 7

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Écriture de commandes SQL

* Requête SQL
  + Mots réservés
  + Mots définis par l’utilisateur
  + \_\_\_\_\_Insensibles\_\_\_\_\_\_ à la casse
  + |9:3| : Cet a dire l’interpreteur SQL ne verra pas la différence entre Majuscules et minuscules.
* Plus facile?
* |9:3| : Les attributs on va mettre en majuscule et en minuscule pour les mots clés.
* Voir la norme du cours sur le site web.

Connolly et Begg Database systems 6th 8

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Valeurs littérales

* \_\_\_\_Constantes\_\_\_\_\_\_ utilisées dans requêtes SQL
* Non numériques
  + Encadrés \_\_\_\_guilletets\_\_\_\_\_\_ simples
  + Par exemple: 'Londres'
* Numériques
  + Pas \_\_\_encadrés\_\_\_\_\_
  + Par exemple: 650.00.

|9:12| : Les date on devra mettre ToDATE devant.

Connolly et Begg Database systems 6th 9

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Définition de données

* DDL SQL
  + Permet aux objets de BD
  + D’être créés, modifiés et \_\_\_détruits\_\_\_\_\_
  + Principales requêtes

create/alter \_\_table\_\_\_ drop table create view drop view\_

Connolly et Begg Database systems 6th 10

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Types de données SQL ORACLE

# |9:14| : on prend nvarchar2

# |9:15| : number2 = (entier de 2 char max)

# |9:15| : number3 = (entier de 3 char max)

# |9:15| : number5.2 = (nombre avec 5 char donc 2 apres le .

# |9:16| : char6 : ex:

# |9:17| : Revoir avec les braquettes et parenthese.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TYPE** | **DESCRIPTION** | **Minimum** | **Maximum** | **Exemples de valeurs** |
| varchar2(taille) | Chaîne de caractères de | 1 car. | 2000 car. | 'A' |
| nvarchar2 (taille) | longueur variable Chaîne de caractères de |  |  | 'Bonjour DD' |
|  | longueur variable utilisant le |  |  |  |
|  | jeu de car. national |  |  |  |
| number[(taille[,precision])] | Numérique. (prec<=38, exposant max -84 +127) | 10 exp -84 | 10 exp 127 | 10.9999 |
| long(taille) | Chaîne de caractères de longueur variable. | 1 car. | 2 giga car. | 'AAAHHHHHH...HHH' |
| date | Date (du siècle à la seconde) | 01/01/-4712 | 31/12/9999 | '10-FEB-04' |
|  |  | (avant J.C) |  | '10/02/04'  (dépend du format |
|  |  |  |  | d'affichage ou du |
|  |  |  |  | paramétrage local) |
| char(taille) | Chaîne de caractères de | 1 car. | 255 car. | 'AZERTY' |
| nchar (taille) | longueur fixe  Chaîne de caractères de |  |  | 'W' |
|  | longueur fixe utilisant le jeu |  |  |  |
|  | de car. national |  |  |  |

Extrait de <http://didier.deleglise.free.fr/sql/type_donnees.htm>

Connolly et Begg Database systems 6th 11

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Exercices

* Lequel n’est pas un objectif d’SQL?

A-Définir données B-Manipuler données C-Format fort

D-Utilisable par tous E-Standard F-Aucune de ces réponses

|9:19| : Le format est plutot leger, (sans format)

* SQL 2003 a apporté?

A-fonction amélioration intégrité B-Support gestion données OO C-XML et auto-incrément D-XML et Xquery

E- détails sur trigger et truncate F-Aucune de ces réponses

* Laquelle de ces valeurs est de type number(5,2)?

A-524.36 B-524,36 C-13524.35 D-113524.36 E-13524,35

F-Aucune de ces réponses G-Toutes ces réponses

Connolly et Begg Database systems 6th 12

Département d’informatique et de génie logiciel – Marc Philippe Parent

# CREATE TABLE

create table NOM\_TABLE ({NOM\_COLONNE TypeDonnee [not null]

|9:35| : On doit spécifier not null pour une valeur qui ne peut etre nul.

[default ValeurParDefaut] [,…]}

|9:35| : Veut dire que tout ce qui précédé peut être répète.

[constraint NOM\_CONTRAINTE primary key (LISTE\_COLONNES),]

{constraint NOM\_CONTRAINTE [unique(LISTE\_COLONNES),] […,]}

{constraint NOM\_CONTRAINTE [foreign key(LISTE\_COLONNES) references NOM\_TABLE\_PARENT [(LISTE\_COLONNES)]

[on delete ActionReferentielle ]] [,…]}

|9:37| : Optionnel pour le langage mais nous on va toujours mettre un nom de table.

{constraint NOM\_CONTRAINTE [check(Condition)] [,…] });

* not \_\_\_null\_\_\_ Rejette valeur nulle dans colonne
* \_\_\_\_Default\_\_\_\_\_ Valeur insérée par défaut
* foreign key  Clés étrangères et actions \_\_\_\_\_référentielle\_\_\_\_\_\_\_\_\_
* Exemple 7.1 dans SQL developer.

Connolly et Begg Database systems 6th 13

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Propriété d’amélioration d’intégrité

* Donnée \_\_\_requise\_\_\_\_ --» FONCTION varchar (10) not null
* \_\_\_\_Domaine --»\_\_\_\_\_ constraint CT\_SEXE check (SEXE in ('H','F'))

|9:45| : Je veux que le champ sexe Soit H ou F.

* Intégrité d’entité
  + \_\_PK --»\_\_ Valeur unique non-nulle pour chaque rangée

constraint PK\_PERSONNEl primary key(NUM\_PERSONNEL)

constraint PK\_BAIL primary key (NUM\_CLIENT, NUM\_PROPRIETE)

|9:47| : On ne peut pas créer deux primary key sur la meme table.

|9:49| : On donne des noms aux contraintes, PK\_BAIL n’est pas le nom d’une table.

* + 1 clause / \_\_table\_\_\_
* Clé \_\_\_\_alternative\_\_\_\_\_\_\_ par unique

constraint AK\_PER\_NUM\_TELEPHONE unique(NUM\_TELEPHONE)

|9:51| : Faite attention de pas fusionner vos clé alternative ensemble.

Connolly et Begg Database systems 6th 14

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Intégrité référentielle

* \_FK\_
  + Colonne(s) reliant chaque rangée de la table enfant contenant la FK à la rangée de la table parent contenant la PK
* Intégrité référentielle
  + Si FK contient valeur
    - Valeur doit référer à une \_\_\_existante\_\_\_\_\_\_ du parent
* \_ISO\_\_
  + foreign key dans create et alter table
  + |9:53| : on peut crée des foreign key autant en créant la table quand la modifiant.

constraint FK\_PR\_NUM\_FILIALE foreign key (NUM\_FILIALE) references FILIALE(NUM\_FILIALE) on delete set null

|9:54| : on delete set null, met les valeurs à nul si le parent est supprimé. Si la table le permet.

|9:56| : Mais ce n’est pas loption pas défaut

|9:56| : Par defaut si ta une relation entre 2 tables et on veux supprimer cest refusé.

Connolly et Begg Database systems 6th 15

Département d’informatique et de génie logiciel – Marc Philippe Parent

# PAI – Intégrité référentielle

* Tout insertion/modification de FK dans la table enfant sans

\_\_\_\_\_correspondance\_\_\_\_\_\_\_\_\_ de valeur de PK dans le parent est rejeté

* Action référentielle
  + Si effacement de PK du parent avec FK existante de l’enfant
    - Action spécifiée par la clause on delete

-cascade

|9:58| : les enfant seront delete aussi.

-set null

|9:58| : Si j’efface, la relation correspondance sera mis a null si la table le permet.

-set default

|9:59| : Mettre la valeur par défaut sur les champs correspondant.

-no action.

Connolly et Begg Database systems 6th 16

Département d’informatique et de génie logiciel – Marc Philippe Parent

# PAI – Intégrité référentielle

constraint FK\_PR\_NUM\_PERSONNEl foreign key (NUM\_PERSONNEL) references PERSONNEL(NUM\_PERSONNEL) on

delete set null,

constraint FK\_PR\_NUM\_PROPRIETAIRE foreign key (NUM\_PROPRIETAIRE) references PROPRIETAIRE(NUM\_PROPRIETAIRE) on delete cascade,

Connolly et Begg Database systems 6th 17

Département d’informatique et de génie logiciel – Marc Philippe Parent

# alter table

# |10:2| : permet de modifier la structure la table.

# |10:2| : soit ajouter ou effacer des colonnes

# |10:3| : ajouter une colonne on met pas column on marque juste ADD

# |10:3| : pour enlever une colonne on mearque drop column

# |10:3| : Pour ajouter une contrainte on dois marquer ad constraint.

* Ajoute|efface \_\_\_\_colonne\_\_\_ à une table
* Ajoute|efface \_\_\_\_\_contrainte\_\_\_\_\_ à une table

alter table NOM\_TABLE

add NOM\_COLONNE DEFINITION\_COLONNE;

alter table NOM\_TABLE

add ({NOM\_COLONNE DEFINITION\_COLONNE[,…]});

alter table NOM\_TABLE

drop column NOM\_COLONNE; alter table NOM\_TABLE

|10:6| : toujours mettre vos; apres une requete.

add constraint NOM\_CONTRAINTE DEFINITION\_CONTRAINTE;

Connolly et Begg Database systems 6th 18

Département d’informatique et de génie logiciel – Marc Philippe Parent

# alter et drop table

* drop table

drop table NOM\_TABLE[restrict| cascade]

* Efface table + tout son contenu.
* |10:11| : Par defaut si il la table a une clé étrangère ou correspond a une relation qui a untre clé étrangère, c’est refusé.

Connolly et Begg Database systems 6th 19

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Exercices

* Qu’est-ce que l’intégrité d’entité?

A-Une clé étrangère ne peut avoir une valeur nulle

B-Une clé étrangère ne peut avoir une valeur inexistante dans la table parent

C-Une clé primaire ne peut être nulle

D-Un attribut ne peut être nul E-Aucune de ces réponses

* Quelle requête valide permet d’ajouter une colonne à une table?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A-alter | table | CLIENT | add | column NB\_PREFERE number(2); |
| B-alter | table | CLIENT | add | constraint NB\_PREFERE number(2); |
| C-alter | table | CLIENT | add | NB\_PREFERE; |
| D-alter | table | CLIENT | add | NB\_PREFERE number(2); |
| E-alter | table | CLIENT | add | column NB\_PREFERE; |

Connolly et Begg Database systems 6th 20

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Requête select

select [distinct| all] {\* | [EXPRESSION\_COL [as NOUVEAU\_NOM]][,...] } from NOM\_TABLE [ALIAS] [, ...]

[where CONDITION]

[group by LISTE\_COLONNE[having CONDITION]] [order by LISTE\_COLONNE];

* \_\_\_\_\_ des clauses ne peut être changé
* Seuls select et from \_\_\_\_\_\_\_\_\_\_\_\_
* Exemples.

Connolly et Begg Database systems 6th 21

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Opérateurs de comparaison

* \_ égal
* \_\_ pas égal à (norme ISO)
* ! = pas égal à (autorisé dans certains dialectes)
* \_ plus petit, \_\_ inférieur ou égal à
* \_ plus grand, \_\_ supérieur ou égal à
* s’évalue de gauche à \_\_\_\_\_\_
* Expressions entre \_\_\_\_\_\_\_\_\_\_\_ évaluées en 1er
* \_\_\_ évalué avant and et or
* \_\_\_ évalué avant or
* \_\_\_\_\_\_\_\_\_\_\_ toujours recommandé Éliminer ambiguïté
* Exemples.

Connolly et Begg Database systems 6th 22

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Exercices

* Quelle requête donne les clients de Montréal et de Québec?

A-select \* from CLIENT where VILLE\_CLI = 'Montréal' and FONCTION\_CLI='Québec';

B-select \* from CLIENT where VILLE\_CLI = 'Montréal' or VILLE\_CLI ='Québec';

C-select \* from CLIENT where VILLE\_CLI = 'Montréal' and VILLE\_CLI ='Québec';

D-select \* from CLIENT where VILLE\_CLI = 'Montréal' or 'Québec';

E-Aucune de ces réponses F-Toutes ces réponses

Connolly et Begg Database systems 6th 23

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Appartenance à un ensemble

* in (…)
* Négation (\_\_\_ in)
* pas plus expressif
* in est \_\_\_\_\_\_\_\_ pour
  + Ensemble contient +2 valeurs.

Connolly et Begg Database systems 6th 24

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Like et null

* Correspondance à un masque
  + 2 symboles de correspondance à un masque
    - %: séquence de 0+ \_\_\_\_\_\_\_\_\_\_\_\_
    - \_ (sous-ligné): un \_\_\_\_ caractère, n’importe lequel
  + like ‘%Montréal%’ signifie une séquence de n’importe quelle

\_\_\_\_\_\_\_\_ contenant ‘Montréal’

* Absence de valeur
  + Condition de recherche de null
  + Tester null explicitement avec is null
  + Négation (is \_\_\_ null)
    - Tester valeurs non nulles
* Exemples.

Connolly et Begg Database systems 6th 25

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Tris

* Afficher en ordre
  + Alphabétique
  + \_\_\_\_\_\_\_\_\_
  + Croissant ou décroissant
  + Par \_\_ colonnes
* Exemples

Connolly et Begg Database systems 6th 26

Département d’informatique et de génie logiciel – Marc Philippe Parent

# insert

insert into NOM\_TABLE[(LISTE\_COLONNES)] values(listeValeurs);

* LISTE\_COLONNES \_\_\_\_\_\_\_\_\_
  + Omise  tous les champs ordre de création
  + Colonne omise  null ou default
* Nombre d’items \_\_\_\_ dans 2 listes
* Correspondance directe \_\_\_\_\_\_\_\_
* Types de données de ListeValeurs
  + Compatibles avec colonnes correspondantes
* Exemples

Connolly et Begg Database systems 6th 27

Département d’informatique et de génie logiciel – Marc Philippe Parent

# update

update NOM\_TABLE

\_\_\_ NOM\_COLONNE1 = Valeur1

[, NOM\_COLONNE2 = Valeur2...]

[where CONDITION]

* Exemples

Connolly et Begg Database systems 6th 28

Département d’informatique et de génie logiciel – Marc Philippe Parent

# delete

delete \_\_\_\_ NOM\_TABLE [where CONDITION]

* Exemples.

Connolly et Begg Database systems 6th 29

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Requête select - Agrégation

* ISO, 5 fonctions d’agrégation
  + \_\_\_\_\_ nombre de valeurs
  + \_\_\_ somme des valeurs
  + \_\_\_ moyenne des valeurs
  + \_\_\_ la plus petite valeur
  + \_\_\_ la plus grande valeur
* Une seule colonne et retourne \_\_\_ valeur
* count, min, et max  Champs numériques et non numériques
* sum et avg  Champs \_\_\_\_\_\_\_\_\_\_ seulement
* Outre count(\*) Éliminent \_\_\_\_\_\_ en premier
  + S’exécute sur valeurs restantes.

Connolly et Begg Database systems 6th 30

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Requête select - Agrégation

* count(\*)
  + Compte toutes les \_\_\_\_\_\_\_
  + sans égard aux nulles et doublons
* \_\_\_\_\_\_\_\_
  + Éliminer doublons
  + Pas \_\_\_\_\_\_\_ sur min/max,
  + Peut en avoir sur sum/avg,
* Utilisées seulement dans
  + Liste champs d’un select
  + Dans clause \_\_\_\_\_\_
* Si liste select avec fonction agrégation et pas group by -> liste select que des fonctions.

Connolly et Begg Database systems 6th 31

Département d’informatique et de génie logiciel – Marc Philippe Parent

**Agrégation**

* \_\_\_\_\_\_\_\_

select NUM\_PERSONNEL, count(SALAIRE) from PERSONNEL ;

* Exemples

Connolly et Begg Database systems 6th 32

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Exemple 6.37 insert … select

* Seconde forme d’insert
  + plusieurs rangées \_\_\_\_\_\_\_ d’une ou plusieurs tables vers une autre

insert into NOM\_TABLE[ (LISTE\_COLONNES) ]

\_\_\_\_\_\_ ...

* PERSO\_COMPTE\_PROPRIETE contient numéro des employés et nombre des propriétés qu’ils \_\_\_\_\_\_ PERSO\_COMPTE\_PROPRIETE(NUM\_PERSONNEL,NB\_PROPRIETE\_GEREES)
* \_\_\_\_\_\_\_ PERSO\_COMPTE\_PROPRIETE en utilisant les tables PERSONNEL et PROPRIETE\_A\_LOUER
* Exemple.

Connolly et Begg Database systems 6th 33

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Exercices

* Que trouvera cette clause? …where TYPE\_PRODUIT like '%tion\_'

A-Les lignes terminant par tion B-Les lignes débutant par tion C-Les lignes contenant tion

D-Les lignes terminant par tion suivi d'un et un seul caractère

E-Les lignes débutant par 1 caractère suivi de tion puis des autres caractères

F-Aucune de ces réponses

Connolly et Begg Database systems 6th 34

Département d’informatique et de génie logiciel – Marc Philippe Parent

# Références

* Logo ISO https://[www.isocertificateonline.in/iso-logo-usage-after-certification/](http://www.isocertificateonline.in/iso-logo-usage-after-certification/)
* Image Edgar F. Codd https://en.wikipedia.org/wiki/Edgar\_F.\_Codd
* Image Donald Chamberlin page 6 <http://www.ithistory.org/honor-roll/mr-> donald-d-chamberlin

Connolly et Begg Database systems 6th 35

Département d’informatique et de génie logiciel – Marc Philippe Parent