

Systèmes d'Information & Base de données

-- SQL--

Abdeslam EN-NOUAARY, PhD , Ingénieur fonctionnel

abdeslam@inpt.ac.ma

Anass RACHDI, Docteur , Ingénieur

anass.rach@gmail.com

Mai 2018

SQL

- **Définition du langage de requête structurée sur les BDR (SQL)**
- **Langage de Description des Données (LDD)**
- **Langage d'Interrogation des Données (LID)**
- **Langage de Manipulation des Données (LMD)**

SQL

- SQL est un langage de requête de base de données et de programmation largement utilisé pour interroger, mettre à jour et gérer des données dans des systèmes de bases de données relationnelles.
- SQL nous permet également de créer des bases de données et des objets de base de données et exécuter d'autres fonctions plus complexes sur les données.
- Avec SQL, nous pouvons également de modifier des paramètres de base de données ou de session et de contrôler les instructions de données et d'accès.

SQL

- La base de données sur la quelle nous nous basons pour illustrer les exemples SQL s'appelle RH. Elle est composée des tables (relations) suivantes:
 - regions (region_id, region_name)
 - countries (country_id, country_name, region_id)
 - locations (location_id, street_address, postal_code , city, state_province, country_id)
 - departments (department_id, department_name, manager_id, location_id)
 - employees (employee_id, first_name, last_name, email, phone_number, hire_date, job_id, salary, commission_pct, manager_id, department_id)
 - jobs (job_id, job_title, min_salary, max_salary)
 - job_history (employee_id, start_date, end_date, job_id, department_id).
- **Le SGBD utilisé est PostgreSQL**

SQL

Langage de Description des Données (LDD)

SQL permet de créer des relations sous forme de tables et de définir lors de la création des contraintes d'intégrité variées sur les attributs. Ainsi, une commande de création permet de spécifier le nom de la table et de définir les éléments de table correspondant aux colonnes ou aux contraintes, selon la syntaxe suivante :

CREATE TABLE <nom de table> (<élément de table>+)

**<ÉLÉMENT DE TABLE> ::= <DÉFINITION DE COLONNE> |
<CONTRAİNTE DE TABLE>**

**<DÉFINITION DE COLONNE> : ::= <NOM DE COLONNE> <TYPE
DE DONNÉES> [<CLAUSE DÉFAUT>] [<CONTRAİNTE DE
COLONNE>]**

SQL

Langage de Description des Données (LDD)

Les contraintes de colonnes permettent de spécifier différentes contraintes d'intégrité portant sur un seul attribut, y compris les contraintes référentielles. Les différentes variantes possibles sont :

- valeur nulle impossible (syntaxe NOT NULL) ;
- unicité de l'attribut (syntaxe UNIQUE ou PRIMARY KEY) ;
- contrainte référentielle - syntaxe REFERENCES <table référencée> [(<colonne référencée>)] -, le nom de la colonne référencée étant optionnel s'il est identique à celui de la colonne référant ;
- contrainte générale (syntaxe CHECK <condition>) ; la condition est une condition pouvant spécifier des plages ou des listes de valeurs possibles (voir condition de recherche ci-dessous).

SQL

Langage de Description des Données (LDD)

Les types de données supportés sont :

- les chaînes de caractères de longueurs fixes - CHAR(<longueur>) -, la valeur par défaut de la longueur étant 1.
- les chaînes de caractères de longueurs variables - VARCHAR (<longueur>)
- les numériques exacts - NUMERIC et DECIMAL avec précision et échelle optionnelles, INTEGER et SMALLINT -,
- les numériques approchés - FLOAT avec précision optionnelle, REAL et DOUBLE PRECISION.
- Le type DATE est spécifié pour un attribut contenant une année, un mois et un jour (par exemple 1992/08/21). Le type TIME correspond à un groupe heures, minutes et secondes (par exemple, 09 :17 :23).

SQL

Langage de Description des Données (LDD)

- Le type `TIMESTAMP` correspond à une concaténation d'une date et d'une heure (`DATE` concaténé à `TIME`).
- SQL offre également un type de données permettant de spécifier un intervalle de temps (`INTERVAL`) avec une précision en mois-année ou en seconde-minute-heure-jour;
- SQL permet de stocker des textes de plus de 255 caractères en utilisant le type `TEXT`, ou un de ses dérivés `TINYTEXT`, `MEDIUMTEXT` ou `LONGTEXT`.

SQL

Langage de Description des Données (LDD)

CREATE TABLE departments

(department_id SERIAL PRIMARY KEY

, department_name VARCHAR(30) NOT NULL

, manager_id INTEGER

, location_id INTEGER references locations (location_id)) ;

Nom de table

Contrainte sur
colonne

Type de données

Nom de colonne

ALTER TABLE DEPARTMENTS ADD CONSTRAINT dept_mgr_fk

FOREIGN KEY (manager_id)

REFERENCES employees (employee_id);

Clé étrangère

Contrainte sur
la table

SQL

Langage de Description des Données (LDD)

```
CREATE TABLE job_history
( employee_id  INTEGER NOT NULL REFERENCES
employees(employee_id)
, start_date   TIMESTAMP NOT NULL
, end_date     TIMESTAMP NOT NULL
, job_id       VARCHAR(10) NOT NULL REFERENCES
jobs(job_id)
, department_id INTEGER REFERENCES
departments(department_id)
, CONSTRAINT  jhist_date_interval CHECK (end_date >
start_date)
, PRIMARY KEY (employee_id, start_date) ) ;
```

SQL

Langage de Description des Données (LDD)

SQL permet de modifier un schéma de table à l'aide de la commande :

ALTER TABLE <nom de table> <altération>

- Différents types d'altérations sont possibles :
 - ajout d'une colonne (ADD COLUMN) ;
 - modification de la définition d'une colonne (ALTER COLUMN) ;
 - suppression d'une colonne (DROP COLUMN) ;
 - ajout d'une contrainte (ADD CONSTRAINT) ;
 - suppression d'une contrainte (DROP CONSTRAINT).

SQL

Langage de Description des Données (LDD)

SQL permet de supprimer une table à l'aide de la commande :

DROP TABLE <nom de table>

SQL permet également de changer le nom d'une table à l'aide de la commande :

RENAME <ancien_nom> TO <nouveau_nom>

SQL

Langage d'Interrogation des Données (LID)

L'interrogation des données, s'exprime à l'aide du langage SQL par la clause :

**SELECT [ALL|DISTINCT] <EXPRESSION DE VALEURS>+ FROM
<NOM DE TABLE> [<NOM DE VARIABLE>]**

- Une expression de valeurs est une expression arithmétique (composée avec les opérateurs binaires +, -, * et /), éventuellement parenthésée, de spécifications de constantes ou de colonnes.
 - Une spécification de constante est une constante
 - Une spécification de colonne désigne le nom d'une colonne précédé d'un désignateur de relation éventuel (nom de table ou variable).

SQL

Langage d'Interrogation des Données (LID)

SELECT

EMP.first_name **Prenom,** **EMP.last_name** **Nom,**
EMP.salary, **EMP.phone_number,** **EMP.hire_date**
from
employees EMP;

Nom de colonne

Nom de variable & désignateur

prenom	nom	salary	phone number	hire date
Alexander	Hunold	9000.00	590.423.4567	03/01/1990 00:00
John	Russell	14000.00	011.44.1344.429268	01/10/1996 00:00
Gerald	Cambrault	11000.00	011.44.1344.619268	15/10/1999 00:00
Ellen	Abel	11000.00	011.44.1644.429267	11/05/1996 00:00
Den	Raphaely	11000.00	515.127.4561	07/12/1994 00:00

SQL

Langage d'Interrogation des Données (LID)

SELECT DISTINCT

EMP.first_name Prenom, EMP.last_name Nom,
EMP.salary*12 Salaire_annuel
from
employees EMP;

Expression de valeur

prenom	nom	salaire annuel
Tayler	Fox	115200.00
Ellen	Abel	132000.00
Pat	Fay	72000.00
Harrison	Bloom	120000.00

SQL

Langage d'Interrogation des Données (LID)

La commande WHERE dans une requête Select permet d'extraire les lignes d'une

base de données qui respectent une condition. Cela permet d'obtenir uniquement les informations désirées.

SELECT [ALL|DISTINCT] {<expression de valeurs>+ | *}

FROM <nom de table> [<nom de variable>]

WHERE <condition de recherche>

- Une condition de recherche élémentaire est appelée prédicat en SQL. Un prédicat de restriction permet de comparer deux expressions de valeurs. la première contenant des spécifications de colonnes est appelée terme.

SQL

Langage d'Interrogation des Données (LID)

Il existe une grande diversité de prédicats en SQL.

- un prédicat de comparaison permettant de comparer un terme à une constante à l'aide des opérateurs $\{=, \neq, <, >, \leq, \geq\}$;
- un prédicat d'intervalle BETWEEN permettant de tester si la valeur d'un terme est comprise entre la valeur de deux constantes ;
- un prédicat de comparaison de texte LIKE permettant de tester si un terme de type chaîne de caractères contient une ou plusieurs sous-chaînes ;
- un prédicat de test de nullité qui permet de tester si un terme a une valeur convenue NULL, signifiant que sa valeur est inconnue ;

SQL

Langage d'Interrogation des Données (LID)

- un prédicat d'appartenance IN qui permet de tester si la valeur d'un terme appartient à une liste de constantes.
- Une requête est construite pour chercher le salaire du directeur général (la colonne manager_id doit être NULL)

SELECT

EMP.first_name Prenom, **EMP.last_name** Nom,

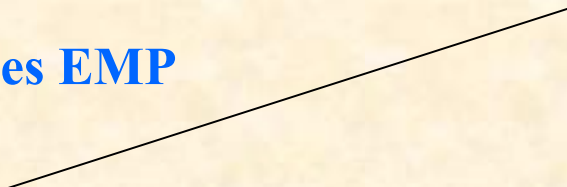
EMP.salary

from employees EMP

WHERE

EMP.manager_id is NULL;

Condition de recherche



prenom	nom	salary
Steven	King	24000.00

SQL

Langage d'Interrogation des Données (LID)

- Une requête est construite pour chercher les noms et les prénoms des employés dont le job_id commence par 'IT' et le salaire dépasse 4500

SELECT

EMP.first_name Prenom, **EMP.last_name** Nom, **EMP.job_id** Poste,
EMP.salary

from employees EMP

WHERE

EMP.job_id LIKE 'IT%' AND EMP.salary > 4500;

prenom	nom	poste	salary
Alexander	Hunold	IT PROG	9000.00
Bruce	Ernst	IT PROG	6000.00
David	Austin	IT PROG	4800.00
Valli	Pataballa	IT PROG	4800.00

SQL

Langage d'Interrogation des Données (LID) FONCTIONS DE CALCULS ET AGRÉGATS

- Au-delà de la sélection habituelle, des possibilités de calcul de fonctions existent. Les fonctions implantées sont :
 - COUNT qui permet de compter le nombre de valeurs d'un ensemble ;
 - SUM qui permet de sommer les valeurs d'un ensemble ;
 - AVG qui permet de calculer la valeur moyenne d'un ensemble ;
 - MAX qui permet de calculer la valeur maximale d'un ensemble ;
 - MIN qui permet de calculer la valeur minimale d'un ensemble.

SQL

Langage d'Interrogation des Données (LID)

- L'agrégat est un partitionnement horizontal d'une table en sous-tables en fonction des valeurs d'un ou de plusieurs attributs de partitionnement, suivi de l'application d'une fonction de calculs à chaque attribut des sous-tables obtenues. Cette fonction est choisie parmi celles indiquées ci-dessus. En SQL, le partitionnement s'exprime par la clause :

GROUP BY <SPÉCIFICATION DE COLONNE>+

- Une restriction peut être appliquée avant calcul de l'agrégat au niveau de la clause WHERE, mais aussi après calcul de l'agrégat sur les résultats de ce dernier. Pour cela, une clause spéciale est ajoutée à la requête SELECT :

HAVING <EXPRESSION DE VALEURS>+

SQL

Langage d'Interrogation des Données (LID)

```
SELECT job_id JOB,  
count(employee_id) nbre_personne_job  
from employees EMP  
group by  
job_id ;
```

job	nbre personne job
AC ACCOUNT	1
ST MAN	5
IT PROG	5
SA MAN	5
AD PRES	1
AC MGR	1
FI MGR	1
AD ASST	1

SQL

Langage d'Interrogation des Données (LID)

```
SELECT job_id JOB,  
count(employee_id) nbre_personne_job  
from employees  
group by job_id  
having count(employee_id) > 3 ;
```

job	nbre personne job
ST MAN	5
IT PROG	5
SA MAN	5
PU CLERK	5
FI ACCOUNT	5
SH CLERK	20
SA REP	30
ST CLERK	20

SQL

Langage d'Interrogation des Données (LID)

La commande ORDER BY permet de trier les lignes dans un résultat d'une requête SQL. Il est possible de trier les données sur une ou plusieurs colonnes, par ordre ascendant ou descendant :

SELECT EMP.first_name Nom, EMP.last_name, EMP.salary salaire from employees EMP order by salaire desc ;

nom	last name	salaire
Steven	King	24000.00
Neena	Kochhar	17000.00
Lex	De Haan	17000.00
John	Russell	14000.00
Karen	Partners	13500.00
Michael	Hartstein	13000.00
Shelley	Higgins	12000.00

SQL

Langage d'Interrogation des Données (LID)

Les jointures en SQL permettent d'associer plusieurs tables dans une même requête. Cela permet d'exploiter la puissance des bases de données relationnelles pour obtenir des résultats qui combinent les données de plusieurs tables de manière efficace.

Les jointures consistent à associer des lignes de 2 tables en associant l'égalité des valeurs d'une colonne d'une première table par rapport à la valeur d'une colonne d'une seconde table.

Nous voulons afficher des informations relatives aux employées ainsi que les informations relatives aux départements aux leur jobs. Nous devons faire une jointure entre trois tables à savoir : EMPLOYEES, DEPARTMENTS, JOBS

SQL

Langage d'Interrogation des Données (LID)

```
SELECT EMP.first_name Prenom, EMP.last_name Nom, EMP.salary  
salaire, DEP.department_name nom_departement, JOB.job_title fonction  
From employees EMP,departments DEP,jobs JOB  
Where EMP.job_id=JOB.job_id and  
EMP.department_id=DEP.department_id;
```

prenom	nom	salaire	nom departement	fonction
Steven	King	24000.00	Executive	President
				Administration
Neena	Kochhar	17000.00	Executive	Vice President
				Administration
Lex	De Haan	17000.00	Executive	Vice President
Alexander	Hunold	9000.00	IT	Programmer
Bruce	Ernst	6000.00	IT	Programmer
David	Austin	4800.00	IT	Programmer

SQL

Langage de Manipulation des Données (LMD)

INSERTION DE TUPLES

L'insertion de tuples dans une relation permet de créer de nouvelles lignes. Elle peut s'effectuer par fourniture directe au terminal d'un tuple à insérer (ou d'une partie de tuple, les valeurs inconnues étant positionnées à NULL),

```
INSERT INTO <NOM DE TABLE> [( <NOM DE COLONNE> +)]  
{VALUES (<CONSTANTE>+) | <COMMANDE DE RECHERCHE>}
```

```
INSERT INTO employees VALUES  
  ( 101, 'Neena' , 'Kochhar', 'NKOCHHAR', '515.123.4568'  
  , TO_DATE('21-SEP-1989', 'dd-MON-yyyy'), 'AD_VP'  
  , 17000, NULL, 100, 90);
```

SQL

Langage de Manipulation des Données (LMD)

MISE A JOUR DE TUPLES

La mise à jour permet de changer des valeurs d'attributs de tuples existants. La mise à jour d'une relation s'effectue par fourniture directe des valeurs à modifier

UPDATE <NOM DE TABLE>

SET {<NOM DE COLONNE> = {<EXPRESSION DE VALEUR> | NULL}}+

WHERE {<CONDITION DE RECHERCHE>

prenom	nom	salaire	nom departement	fonction
Alexander	Hunold	9000.00	IT	Programmer

**UPDATE employees EMP set salary= 10000 Where
EMP.last_name='Hunold';**

prenom	nom	salaire	nom departement	fonction
Alexander	Hunold	10000.00	IT	Programmer

SQL

Langage de Manipulation des Données (LMD)

SUPPRESSION DE TUPLES

L'opération de suppression permet d'enlever d'une relation des tuples existants. Les tuples sont spécifiés à l'aide d'une condition de recherche, à moins que l'on désire supprimer tous les tuples d'une relation. La syntaxe de la commande de suppression est :

**DELETE FROM <NOM DE TABLE>
[WHERE {<CONDITION DE RECHERCHE>}**

Delete from employees EMP where EMP.last_name='Hunold';