

Module 106

Interroger, traiter et assurer la maintenance des bases de données

Karl Legrand

Qui suis-je ?

Activités



Formations

- ✓ CIE 105 - Langage SQL – VD & GE
- ✓ CIE 130 – Réseau – VD & GE
- ✓ CIE 340 – Virtualisation – VD
- ✓ CIE 106 - Langage SQL et maintenance DB – VD
- ✓ 12Harmos – Culture générale informatique



Développement Full-stack

- ✓ CRM
- ✓ Base de données SQL
- ✓ Outils DotNet spécifiques



Support

- ✓ Pour nos applications, Sage, etc.



CIE 106 : Déroulement



4.5 jours : cours présentiel (sauf ordre DGEP) / participation obligatoire



- ✓ Théorie : 0.5 jour
- ✓ Exercice et correction : 4 jours
- ✓ Pause matin, midi et après-midi

0.5 jour = examen individuel en ligne sur Moodle



- ✓ Avec document
 - QCM
 - Requête SQL à écrire à partir de MPD
 - Questions ouvertes

DML - Ordre SELECT

- ✓ L'instruction SELECT permet de sélectionner des données dans une base de données.
- ✓ Il peut extraire des données d'une ou de plusieurs tables, tables temporaires ou vues.
- ✓ La sélection est stockée dans une table de résultats, appelée jeu de résultats.
- ✓ SELECT est la commande de langage de manipulation de données (DML) la plus utilisée
- ✓ SELECT Ne modifie pas les données
- ✓ Si la syntaxe est juste SELECT retourne des résultats, qui ne correspondent peut être pas à ce qu'on imaginait

SELECT nom_colonne, nom_colonne2, etc.

FROM nom_table;

Exemple:

SELECT * FROM Etudiants;

Renvoi toutes les colonnes de la table Etudiants

Table Etudiants

ID	Nom	Prenom	Ville	Note
1	Dupin	Paul	Lausanne	5
2	Dupuis	Pierre	Lausanne	5.5
3	Dupin	Sylvie	Lausanne	4.5

SELECT - Clauses

Les clauses du SELECT sont obligatoirement dans cet ordre.

1. SELECT
2. FROM
3. WHERE
4. GROUP BY
5. HAVING
6. ORDER BY

SELECT – Distinct

- ✓ Certains champs d'une table peuvent contenir des valeurs en double.
- ✓ Le fait d'utiliser DISTINCT dans le SELECT permet d'affiner le jeu de résultat. Les doublons sont ignorés (dans le jeu de résultat).
- ✓ DISTINCT ne renvoie que des valeurs distinctes (uniques) en se basant sur toutes les colonnes spécifiées dans le SELECT

SELECT nom_colonne
FROM nom_table;

Exemple:

SELECT DISTINCT Nom, Ville
FROM Etudiants;

Jeu de résultats

Nom	Ville
Dupin	Lausanne
Dupuis	Lausanne

Table Etudiants

ID	Nom	Prenom	Ville	Note
1	Dupin	Paul	Lausanne	5
2	Dupuis	Pierre	Lausanne	5.5
3	Dupin	Sylvie	Lausanne	4.5

Renvoie les colonnes Nom et Ville de la table Etudiants et supprime les doublons pour le même Nom de la même Ville

SELECT - WHERE

- ✓ La clause WHERE est utilisée pour extraire uniquement les enregistrements qui répondent à un ensemble de critères spécifié.
- ✓ WHERE permet de spécifier les lignes à récupérer.

SELECT column_name(s)
FROM table_name
WHERE column_name = variable;



La variable doit correspondre au type de données de la colonne (Numérique, texte, date, etc.)

Les colonnes utilisées dans le WHERE n'ont pas obligation d'être spécifié dans le SELECT

Exemple:

SELECT Nom, Prenom, Note
FROM Etudiants
WHERE Note > 5 AND Ville = 'Lausanne';

Renvoie les colonnes Nom, Prenom et Note
de la table Etudiants où la note est plus grande que 5 et dont la ville est Lausanne

Table Etudiants

ID	Nom	Prenom	Ville	Note
1	Dupin	Paul	Lausanne	5
2	Dupuis	Pierre	Lausanne	5.5
3	Dupin	Sylvie	Lausanne	4.5

Jeu de résultats

Nom	Prenom	Note
Dupuis	Pierre	5.5

SELECT – WHERE - Opérateur logique AND/OR

Opérateurs booléens

- ✓ Conçu pour fonctionner avec des valeurs booléennes de vrai ou faux.
- ✓ Les opérateurs booléens SQL sont AND (conjonction logique), OR (inclusion logique) et NOT (négation logique).

SELECT column_name(s)
FROM table_name
WHERE column_name = variable OR column_name = variable2;



Le AND est prioritaire sur le OR => niveau de parenthèse

Exemple:

SELECT Nom, Prenom, Note
FROM Etudiants
WHERE Note > 5 AND Ville = 'Lausanne';

Table Etudiants

ID	Nom	Prenom	Ville	Note
1	Dupin	Paul	Lausanne	5
2	Dupuis	Pierre	Lausanne	5.5
3	Dupin	Sylvie	Lausanne	4.5

Jeu de résultats

Nom	Prenom	Note
Dupuis	Pierre	5.5

SELECT – WHERE – Opération

Opération

✓ =	Egale
✓ >=	Plus grand ou égale que
✓ <=	Plus petit ou égale que
✓ <>	Différent de
✓ !=	Différent de
✓ BETWEEN	Entre 2 valeurs incluses
✓ IN	Dans une liste de valeur
✓ LIKE	Comme (avec caractère générique % et _)
✓ IS NULL	est nulle



Toutes ces opérations peuvent être combinées et être nié par un **NOT**

Exemples:

SELECT Nom, Prenom FROM Etudiants WHERE Note > 5 AND Ville IN ('Lausanne','Yverdon');

SELECT Nom FROM Etudiants WHERE Note != 4;

SELECT Nom FROM Etudiants WHERE Nom Like 'Dup%';

SELECT – ORDER BY

- ✓ Trie le jeu de résultats en fonction d'une colonne spécifiée par ordre croissant (ASC) (par défaut) ou décroissant (DESC).

```
SELECT column_name(s)
FROM table_name
ORDER BY column_name;
```



Plusieurs ordre de trie peuvent être donnée dans ORDER BY. L'ordre de priorité va de gauche vers la droite

Exemple:

```
SELECT Nom, Prenom, Note
FROM Etudiants
ORDER BY Note;
```

Renvoie les colonnes Nom, Prenom et Note
de la table Etudiants ou la note est plus grande que 5 et dont la ville est Lausanne

Jeu de résultats

Nom	Prenom	Note
Dupin	Sylvie	4.5
Dupin	Paul	5
Dupuis	Pierre	5.5

Table Etudiants

ID	Nom	Prenom	Ville	Note
1	Dupin	Paul	Lausanne	5
2	Dupuis	Pierre	Lausanne	5.5
3	Dupin	Sylvie	Lausanne	4.5

SELECT – Fonctions d'agrégation SQL

- ✓ Les fonctions SQL sont standardisés dans le DML
- ✓ Elle sont au nombre de 5 :
 - ✓ **COUNT** Compter
 - ✓ **SUM** Totaliser
 - ✓ **AVG** Faire une moyenne
 - ✓ **MIN** Trouver le données la plus petite
 - ✓ **MAX** Trouver le données la plus grande



SELECT SUM(column_name)
FROM table_name

Exemple:

SELECT SUM(Note) AS TotalNote
FROM Etudiants;

Renvoi le total des notes de tous les étudiants

Table Etudiants

ID	Nom	Prenom	Ville	Note
1	Dupin	Paul	Lausanne	5
2	Dupuis	Pierre	Lausanne	5.5
3	Dupin	Sylvie	Lausanne	4.5

Jeu de résultats

TotalNote
15