# **Évaluation SQL**

Cette évaluation a pour but d’évaluer vos compétences sur le module Sql qui a pour but de vous former à l’utilisation du langage SQL pour gérer une base de données relationnelle.

Vous aurez besoin d’un serveur MySQL en local pour importer la base de données de l’évaluation, vous devrez utiliser vos connaissances en SQL pour répondre aux questions.

Vous répondrez directement aux questions sur le fichier en indiquant les commandes SQL que vous devez rentrer pour répondre aux questions (sauf si question ouverte).

## Q1 : Quelles sont les termes précis à utiliser lorsqu’on travaille sur une base de données relationnelle ? (Donnez-moi les termes ainsi que leur définition)

La base : c’est l’endroit ou les informations sont classées.

Les tables : elles sont contenues dans la base et contiennent des données différentes.

Les champs et les entrées : ce sont les lignes et les colonnes contenus dans les tables

## Q2 : Qu’est-ce qu’un SGBD ? Citez-en au minimum 3

Le système de gestion de base de données est un logiciel qui permet d’interagir et de gérer les bases de données. (Ex : MySQL, Microsoft SQL Server, PostgreSQL)

## Q3 : Combien y a-t-il d’étudiant ?

Il y’a 384 étudiant

SELECT \* FROM etudiant;

## Q4 : Afficher la liste de tous les étudiants avec le nom de la formation qu’ils suivent (affichez seulement le prénom, le nom ainsi que la formation) ?

SELECT e.prenom, e.nom,

 (SELECT f.formation\_nom

 FROM formation f WHERE f.id = e.formation\_id) AS formation

FROM etudiant e;

## Q5 : Combien d’étudiant suivent la formation DWM ?

SELECT COUNT(\*) AS nombre\_etudiants FROM etudiant

WHERE formation\_id = (SELECT id FROM formation WHERE formation\_nom = 'DWM');

## Q6 : Affichez toutes les évaluations de Pierre Bertrand avec le nom du module évalué ?

etudiant.prenom, etudiant.nom, module.module\_nom, evaluation.note, evaluation.date FROM evaluation JOIN etudiant ON evaluation.etudiant\_id = etudiant.id JOIN module ON evaluation.module\_id = module.id WHERE etudiant.prenom = 'Pierre' AND etudiant.nom = 'Bertrand';

## Q7 : Calculez la moyenne générale de Pierre Bertrand ?

SELECT AVG(evaluation.note) AS moyenne\_pierre\_bertrand FROM evaluation

JOIN etudiant ON evaluation.etudiant\_id = etudiant.id

JOIN module ON evaluation.module\_id = module.id WHERE etudiant.prenom = 'Pierre' AND etudiant.nom = 'Bertrand';

## Q8 : Récupérez la liste des formations ainsi que les formateurs ?

SELECT formation.formation\_nom, formateur.formateur\_nom, formateur.formateur\_prenom FROM formation JOIN formation\_formateur ON formation.id = formation\_formateur.formation\_id JOIN formateur ON formation\_formateur.formateur\_id = formateur.id;

## Q9 : Affichez seulement les formations où Chuck Norris intervient ?

SELECT f.\*

FROM formation f

JOIN formation\_formateur ff ON f.id = ff.formation\_id JOIN formateur fo ON ff.formateur\_id = fo.id WHERE fo.formateur\_nom = 'Norris' AND fo.formateur\_prenom = 'Chuck';

## Q11 : Créez une table qui va stocker le type de formation (e-learning, présentiel, hybride). Cette table devra être reliée à la table formation sachant qu’une formation ne peut avoir qu’un type, mais qu’un type de formation peut être attribuer à plusieurs formations ?

CREATE TABLE type\_formation (

 id INT PRIMARY KEY AUTO\_INCREMENT,

 type\_nom VARCHAR(100)

 );

ALTER TABLE formation ADD COLUMN type\_id INT;

ALTER TABLE formation

 ADD CONSTRAINT fk\_type\_formation FOREIGN KEY (type\_id) REFERENCES type\_formation(id);

## Q12 : Affichez maintenant toutes les formations ainsi que le type de formation ?

SELECT formation.formation\_nom, type\_formation.type\_nom

 -> FROM formation

 -> JOIN type\_formation ON formation.type\_id = type\_formation.id;

## Q13 : Afficher les résultats de l’évaluation du module SQL, vous devrez afficher le prénom et le nom de chaque étudiant, leur note ainsi que la date de devoir, pour finir, vous devrez classer les résultats de la note la plus haute à la plus basse ?

SELECT e.prenom, e.nom, ev.note, ev.date

FROM evaluation ev

JOIN etudiant e ON ev.etudiant\_id = e.id

JOIN module m ON ev.module\_id = m.id

WHERE m.module\_nom = 'SQL'

ORDER BY ev.note DESC;

## Q14 : Affichez les moyennes de tous les étudiants en les classant du meilleur au moins bon ?

SELECT e.prenom, e.nom, AVG(ev.note) AS moyenne

FROM etudiant e

JOIN evaluation ev ON e.id = ev.etudiant\_id

GROUP BY e.id, e.prenom, e.nom

ORDER BY moyenne DESC;

## Q15 : Combien il a d’étudiant dans chaque école ?

SELECT e.ecole\_nom, COUNT(et.id) AS nombre\_etudiants

FROM etudiant et

JOIN formation f ON et.formation\_id = f.id

JOIN ecole\_formation ef ON f.id = ef.formation\_id

JOIN ecole e ON ef.ecole\_id = e.id

GROUP BY e.ecole\_nom

ORDER BY nombre\_etudiants DESC;

## Q16 : Affichez seulement les écoles qui ont plus de 130 étudiants ?

SELECT e.ecole\_nom, COUNT(et.id) AS nombre\_etudiants

FROM etudiant et

JOIN formation f ON et.formation\_id = f.id

JOIN ecole\_formation ef ON f.id = ef.formation\_id

JOIN ecole e ON ef.ecole\_id = e.id

GROUP BY e.ecole\_nom

HAVING COUNT(et.id) > 130

ORDER BY nombre\_etudiants DESC;

## Q17 : Récupérez la liste des 5 écoles avec le plus d’étudiants ?

SELECT e.ecole\_nom, COUNT(et.id) AS nombre\_etudiants

FROM etudiant et

JOIN formation f ON et.formation\_id = f.id

JOIN ecole\_formation ef ON f.id = ef.formation\_id

JOIN ecole e ON ef.ecole\_id = e.id

GROUP BY e.ecole\_nom

ORDER BY nombre\_etudiants DESC

LIMIT 5;

## Q18 : Créer une table qui va stocker des certifications pour les écoles (ex : certifiées par l’état, RNCP, CGE, EEPSIG, AASCB etc..) sachant qu’une école peut avoir plusieurs certifications et que les certifications peuvent être attribuer à plusieurs écoles ?

CREATE TABLE certification (

id INT PRIMARY KEY AUTO\_INCREMENT,

certification\_nom VARCHAR(100)

);

Query OK, 0 rows affected (0.02 sec)

CREATE TABLE ecole\_certification (

ecole\_id INT,

 certification\_id INT,

PRIMARY KEY (ecole\_id, certification\_id),

FOREIGN KEY (ecole\_id) REFERENCES ecole(id),

FOREIGN KEY (certification\_id) REFERENCES certification(id)

);

## Q19 : Attribuez maintenant des certifications aux écoles ?

INSERT INTO certification (certification\_nom) VALUES ('Certifiée par l’état');

INSERT INTO certification (certification\_nom) VALUES ('ouais ouais ouais');

INSERT INTO certification (certification\_nom) VALUES ('la télé la télé ');

INSERT INTO certification (certification\_nom) VALUES ('M6 M6');

INSERT INTO certification (certification\_nom) VALUES ('Monstre brozer');

INSERT INTO ecole\_certification (ecole\_id, certification\_id) VALUES (1, 1);

INSERT INTO ecole\_certification (ecole\_id, certification\_id) VALUES (1, 2);

INSERT INTO ecole\_certification (ecole\_id, certification\_id) VALUES (2, 3);

INSERT INTO ecole\_certification (ecole\_id, certification\_id) VALUES (2, 4);

INSERT INTO ecole\_certification (ecole\_id, certification\_id) VALUES (3, 5);

## Q20 : Affichez les écoles avec leurs certifications ?

SELECT ecole\_nom AS ecole\_nom, certification.certification\_nom

 FROM ecole

JOIN ecole\_certification ON ecole.id = ecole\_certification.ecole\_id

JOIN certification ON ecole\_certification.certification\_id = certification.id