Formation Développeur Expert Java

SQL avec MySQL

Module SQL

1. **Structure d'une base**
2. Installation de MySQL
3. Interrogation de données
4. Requêtes multi-tables
5. Requêtes complexes
6. Manipulation de données
7. Indexes
8. Transactions

- Qu'est-ce qu'une base de données
- Les objets d'une base de données
- Les tables
- Notion d'intégrité de données
- MySQL / MariaDB
- Les moteurs de MariaDB
- Les types de données

1. Structure d'une base

Module SQL

Objectif :

➔ Décrire les composants d'un SGBDR
- Qu'est-ce qu'une base de données relationnelle
- Les objets d'une base de données relationnelle
- Les tables et leurs caractéristiques
- La notion d'intégrité de données

Module SQL

Qu'est-ce qu'une base de données ?

- ➔ Permet de stocker la description des objets (par exemple les tables)
- ➔ Permet de stocker des données dans une ou plusieurs tables
- ➔ Permet de gérer l'accès aux données
- ➔ Permet d'assurer l'intégrité des données

Module SQL

Les objets d'une base de données ?

- Tables
- Index
- Vues
- Séquences
- Fonctions & procédures
- Triggers

Module SQL

Les tables et leurs caractéristiques

- ➔ Permettent de stocker les données
- ➔ Les tables possèdent un ensemble de colonnes
- ➔ Chaque colonne comporte un nom et un type de données

Module SQL

MySQL / MariaDB

- ➔ MySQL a été racheté par Oracle en 2009
- ➔ MariaDB est un "fork" de MySQL suite à ce rachat

Module SQL

Les moteurs de MySQL et MariaDB

MySQL possède plusieurs moteurs de base de données pour gérer les table.

- MyISAM
  - Très rapide
  - Inconvénient : pas de transactions
  - Inconvénient : Pas de clés étrangères
- InnoDB
  - Moteur par défaut sur MariaDB
  - Gère les transactions
  - Gère les clés étrangères et les contraintes d'intégrité
  - Système de récupération lors d'un crash par relecture de log
  - Plus lent

Le moteur est spécifié pour chaque table

Module SQL

Type de données

chaîne de caractères	varchar(n) text
Entier	tinyint (1 octet : -127 à 127) smallint (2 octets : -32768 à 32768) mediumint (3 octets -8388608 à 8388608) int (4 octets : -2 milliards à 2 milliards) bigint (8 octets)
Numérique	decimal (nb_digit, nb_decimal) float (4 octets) : attention, stockage de valeur approximative double (8 octets) : attention, stockage de valeur approximative
Date & time	date time datetime timestamp

Module SQL

Notion d'intégrité de données

- ➔ Une colonne peut être NULL / NOT NULL
- ➔ Une colonne est définie par un type de données et une taille

Module SQL

Ordre de tri et comparaison (Collation)

- MySql gère plusieurs jeux de caractère
  - latin1 : européen (1 caractère = 1 octet)
  - utf8 : international
- Mysql gère plusieurs ordre de tri
  - ci : case insensitive (pas de différences entre majuscules / minuscules / accents)
  - cs : case sensitive
  - bin : différence stricte

- Ces 2 éléments forment la collation :
  - latin1_general_ci
  - latin1_general_cs
  - utf8_general_ci
  - utf8_bin

Module SQL

1. Structure d'une base
2. **Installation de MySQL**
3. Interrogation de données
4. Requêtes multi-tables
5. Requêtes complexes
6. Manipulation de données
7. Indexes
8. Transactions

- Installation de MariaDB
- Installation de MySQLWorkbench
- Création d'une base

2. Installation de MySQL

Module SQL

Objectif :

- ➔ Savoir installer MariaDB
- ➔ Savoir se connecter au serveur
- ➔ Savoir créer une base
- ➔ Savoir y importer des données

Module SQL

Télécharger MariaDB sur le site de la fondation

- ➔ https://downloads.mariadb.org/
- ➔ Prendre la dernière version winx64.msi

Le package comprend le moteur MariaDB et l'outil HeidiSQL (permettant d'effectuer des requêtes

Module SQL

Exercice 1

- ➔ Installer MariaDB
  - Télécharger l'installer
  - Installer MariaDB
  - Laisser tous les paramètres par défaut

Module SQL

Exercice 2

➔ Se connecter avec HeidiSQL
- Nom de session : localhost
- Hôte : 127.0.0.1
- Utilisateur : root
- Mot de passe :
- Port :3306

Module SQL

Exercice 3

- ➔ Créer une base et y importer les données pour les TPs
  - Créer une base formation
    - *Collation : utf8_bin*
  - Importer la structure de la base
    - *Fichier -> Exécuter un fichier SQL -> formation.sql*
  - Importer les données
    - *Fichier -> Exécuter un fichier SQL -> formation_data.sql*

Module SQL

1. Structure d'une base
2. Installation de MySQL
3. **Interrogation de données**
4. Requêtes multi-tables
5. Requêtes complexes
6. Manipulation de données
7. Indexes
8. Transactions

- Select
- Opérateurs arithmétiques
- Concaténation
- Clause where
- Agrégats
  - Group By
  - Fonctions d'agrégat
- Tri
- Fonctions de chaîne
- Conversion de type
- Fonctions de date
- Fonctions mathématiques
- Expression case

3. Interrogation de données

Module SQL

Objectif :

- ➔ Connaître la syntaxe basique de l'ordre SELECT
- ➔ Réaliser des extractions mono-table
- ➔ Se servir des fonctions de base

Module SQL

L'ordre SELECT

- Syntaxe de l'ordre SELECT
  - SELECT col1, col2, … FROM table
- Toutes les colonnes : *
- Suppression des lignes identiques
  - SELECT DISTINCT col1, col2, … FROM table
- Alias
  - SELECT col1 AS alias1 FROM table

Module SQL

Tris

- ➔ Les tris sont définis par ORDER BY
- ➔ Syntaxe
  - SELECT col1, col2 FROM table ORDER BY col1 [ASC|DESC], col2 [ASC|DESC]
  - SELECT col1, col2 FROM table ORDER BY n1 [ASC|DESC], n2 [ASC|DESC]
- ➔ Exemple
  - SELECT nom, prenom FROM abonne ORDER BY nom ASC
  - SELECT nom, prenom FROM abonne ORDER BY 1 ASC

Module SQL

Clause WHERE

- ➔ Permet de conditionner la sélection des lignes
- ➔ Syntaxe
  - ◆ SELECT col1,col2 FROM table WHERE condition

Clauses LIMIT et OFFSET

- ➔ Permet de limiter le nombre de lignes renvoyées
- ➔ Syntaxe
  - ◆ SELECT col1,col2 FROM table LIMIT n OFFSET n

Module SQL

Opérateurs logiques

Egal	=
Différent	<> ou !=
Inférieur	<
Supérieur	>
Inférieur ou égal	<=
Supérieur ou égal	>=
Faisant parti d'une liste de valeur	IN (val1, val2, ..)
Comprise entre 2 valeurs	BETWEEN val1 AND val2
Nulle	IS NULL (et pas = NULL)
Recherche sur une partie d'un mot (joker % et _)	LIKE

Module SQL

Négation et combinaisons

- Négation : NOT
- Combinaisons
  - Et : AND
  - Ou : OR
- Utilisation des parenthèses pour prioritiser les combinaisons

Module SQL

Exercice 4

- ➔ Récupérer tous les abonnés habitant à MONTPELLIER et les trier par ordre alphabétique de nom et prénom.
- ➔ Afficher les colonnes nom, prenom et ville

Module SQL

Exercice 5

➔ Lister tous les prénoms différents qui commencent par la lettre L

Module SQL

Exercice 6

➔ Lister tous les prénoms différents qui commencent par la lettre L ou la lettre M

Module SQL

Exercice 7

- ➔ Récupérer tous les abonnés habitant à MONTPELLIER dont le prénom commence par J et les trier par ordre alphabétique de nom et prénom.
- ➔ Afficher les colonnes nom, prenom et ville

Module SQL

Opérateurs arithmétiques

- ➔ Addition : +
- ➔ Soustraction : -
- ➔ Multiplication : *
- ➔ Division : /

Module SQL

Opérateurs de concaténation de chaîne

- concat(val1, val2)
- concat(val1, val2, val3, valn)

Module SQL

Fonctions de chaîne

Longueur de chaîne	length(chaine)
Conversion en minuscule	lower(chaine)
Conversion en majuscule	upper(chaine)
Partie de chaîne	substring(chaine, debut, nombre)
Début d'une chaîne	left(chaine, nombre)
Fin d'une chaîne	right(chaine, nombre)
Suppression des espaces de début et fin	trim(chaine)
Position d'une sous-chaîne	strpos(chaine, sous-chaine)
Remplacement d'une sous-chaine	replace(chaine, sous-chaine, nouv_sous-chaine)

Module SQL

Exercice 8

- ➔ Récupérer tous les abonnés habitant à Montpellier sans tenir compte des majuscules et minuscules et les trier par ordre alphabétique.
- ➔ Afficher le résultat avec une seule colonne avec sous la forme “NOM Prénom”

Module SQL

Fonctions de date

Date & heure système	now() : date/heure courante curdate() : date courante current_date / current_date() : synonyme de curdate() curtime() : heure courante current_time / current_time() : synonyme de curtime()
Partie d'une date	date(datetime) : extrait la date d'un datetime time(datetime) : extrait l'heure dun datetime extract (partie from date) ex : extract(year from now()) year(datetime) : année de la date month(datetime) : mois de la date day(datetime) : jour de la date weekofyear(datetime) : semaine de la date

Module SQL

Exercice 9

➔ Récupérer tous les abonnés dont l'abonnement est valide (date_fin_abo supérieure ou égale à la date courante).

Module SQL

Conversion de date

Date vers numérique	to_days(date) : converti en nombre de jour depuis l'an 0 to_seconds(date) : converti en nombre de secondes depuis l'an 0 unix_timestamp(date) : converti la date en timestamp
Numérique vers date	from_days(valeur) : converti un nombre de jour depusi l'an 0 en date from_unixtime(valeur) : converti un timestamp en date

Module SQL

Calculs sur les dates

- ➔ MySql ne permet pas de faire des calculs directement sur les dates
- ➔ On peut par contre faire des calculs en convertissant la date en jour, secondes ou timestamp
- ➔ Exemple :
  - SELECT to_days(current_date) + 5
    - *Retourne la date courante + 5 jours en nombre de jour depuis l'an 0*
  - SELECT from_days(to_days(current_date) + 5)
    - *Retourne la date courante + 5 jours*
  - SELECT to_days(current_date) - to_days('2018-11-01')
    - *Retourne le nombre de jour depuis le 01/11/2018*

Module SQL

Fonctions de calculs sur des dates

Ajout / suppression à une date	date_add(date, interval valeur unité) date_sub(date, interval valeur unité) ex : date_add(now(), interval 10 day)
Différence entre 2 dates	datediff(date1, date2) : nombre de jours entre les 2 dates timestampdiff(unit, date1, date2) nombre de unit entre les 2 dates ex : timestampdiff(month, date1, now())

Module SQL

Exercice 10

➔ Récupérer tous les abonnés ayant moins de 20 ans

Module SQL

Exercice 11

➔ Récupérer tous les abonnés dont l'anniversaire tombe le mois courant

Module SQL

Exercice 12

➔ Récupérer tous les abonnés dont l'anniversaire tombe demain

Module SQL

Exercice 13

➔ Récupérer tous les abonnés qui se sont abonnés la semaine dernière (de lundi de la semaine dernière à dimanche dernier)

Module SQL

Agrégats

- ➔ Utilisation de GROUP BY pour regrouper les données
- ➔ Syntaxe
  - ◆ SELECT col1, fonction(col2) FROM table GROUP BY col1
- ➔ Explication
  - ◆ Retourne le résultat de fonction sur la colonne 2 pour toutes les valeurs distinctes de col1

Module SQL

Fonctions d'agrégat

- Moyenne : AVG
- Nombre d'enregistrements : COUNT
- Valeur la plus grande : MAX
- Valeur la plus petite : MIN
- Total : SUM
- Exemple
  - select count(*) from abonne
  - select ville, count(id) from abonne group by ville
  - Renvoi la liste des villes avec pour chacune le nombre d'abonné

Module SQL

Condition Having

- ➔ Permet de filtrer des résultats en appliquant un filtre sur un agrégat
- ➔ Exemple
  - ◆ select ville, count(id) from abonne group by ville having count(id) > 10
  - ◆ Renvoi la liste des villes ayant plus de 10 abonné avec pour chacune le nombre d'abonné

Module SQL

Exercice 14

➔ Compter le nombre d’abonnés

Module SQL

Exercice 15

➔ Compter le nombre d'abonnés entre 30 et 40 ans

Module SQL

Exercice 16

➔ Afficher le nombre d’abonné pour chaque ville et trier par ordre descendantdu nombre d'abonnés (classement des villes par nombre d'abonnés)

Module SQL

Exercice 17

➔ Limiter le résultat précédant aux villes ayant au moins 20 abonnés

Module SQL

Conversion de type

- MySQL effectue de conversion implicite quand il le peut
  - exemple select concat(1, ' mot') retournera une chaîne
- Fonction CAST
- Syntaxe
  - cast(valeur as type)
- Exemples
  - SELECT cast('2015-01-15' as date)

Module SQL

Fonctions de formatage

Date vers texte	date_format(date, format)
Numérique vers texte	format(number, nb_decimal)
Texte vers date	str_to_date(text, format)
Vers un timestamp	timestamp(text, format) timestamp(number)

- ➔ format est une chaîne contenant un format d'affichage
- ➔ Exemple :
  - ◆ str_to_date('01/12/2015', '%d/%m/%Y')
  - ◆ date_format(now(), '%H:%i')

Module SQL

Exercice 18

➔ Récupérer les abonnés dont le nom commence par A et afficher leur statut d'abonnement sous la forme "abonné jusqu'au dd/mm/yyyy" ou "expiré"

Module SQL

Exercice 19

➔ Compter le nombre de membre de chaque famille (même nom)

Module SQL

Exercice 20

➔ Compter le nombre de membre de chaque famille (même nom) et retourner pour chacune la date de naissance du plus jeune et du plus vieux

Module SQL

Exercice 21

➔ Compter le nombre de membre de chaque famille (même nom) et retourner pour chacune l'âge du plus jeune et du plus vieux en années

Module SQL

Traitement de la valeur nulle

- Fonction COALESCE
- Syntaxe
  - COALESCE(val1, val2, val3, ...)
  - Renvoi la 1ere valeur non nulle
- Exemple
  - SELECT COALESCE(libelle_long, libelle, 'vide') FROM table1

Module SQL

Fonctions mathématiques

Modulo	% mod mod()
Valeur absolue	abs()
Division entière	div
Arrondi	round(valeur, precision) ex : round(42.9876, 2) : 42.99
Troncature	truncate(valeur, precision) ex : truncate(42.9876, 2) : 42.98
Valeur aléatoire	random()
Signe	sign(valeur) ex : sign(-12345) : -1 sign(123345) : 1

Module SQL

Exercice 22

➔ Calculer le nombre d'abonné par tranche d'âge (10-20 ans, 20-30 ans, ...)

Module SQL

1. Structure d'une base
2. Installation de MySQL
3. Interrogation de données
4. **Requêtes multi-tables**
5. Requêtes complexes
6. Manipulation de données
7. Indexes
8. Transactions

- Notion de jointure
- Produit cartésien
- Jointure
- Jointure externe

4. Requêtes multi-tables

Module SQL

Objectif :

- ➔ Comprendre la notion de jointure
- ➔ Réaliser des extractions sur plusieurs tables

Module SQL

Relation entre les tables

➔ Relation 1 à plusieurs

table : edition	
noEdition	clé primaire
edition	
adresse	
telephone	

relation
1 à n

- une maison d'édition peut avoir plusieurs livre
- un livre à qu'une seule maison d'édition

table : livre	
no	clé primaire
titre	
sujet	
auteur	
pages	
noEdition	clé étrangère

Module SQL

Relation entre les tables

table : eleve	
noEleve	clé primaire
nom	
prenom	
annee	

relation
m à n

- un élève a plusieurs cours
- un cours a plusieurs élèves

table : cours	
cote	clé primaire
titreCours	
description	

table : eleve	
noEleve	clé primaire
nom	
prenom	
annee	

2 relations 1 à n

1 à n **n à 1**

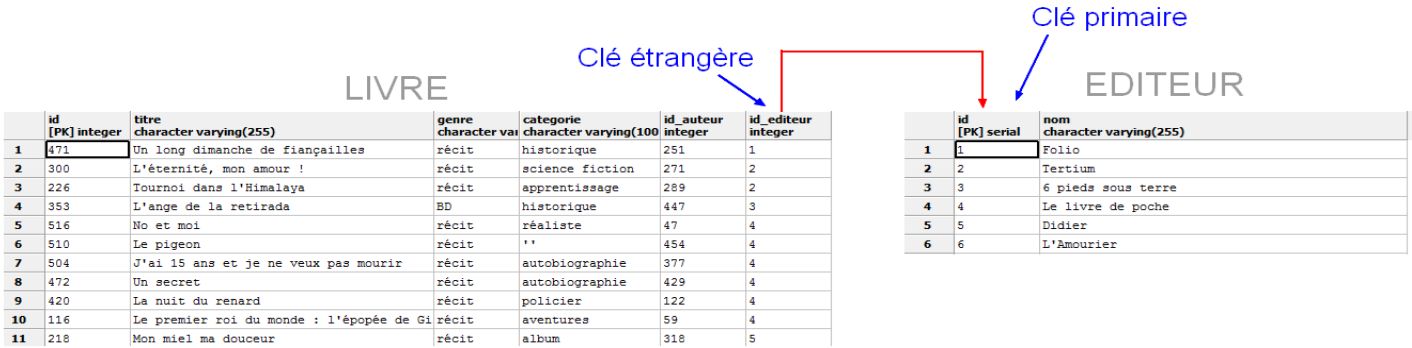
table : suitCours	
noEleve	clé primaire
cote	clé primaire

table : cours	
cote	clé primaire
titreCours	
description	

Module SQL

Notion de jointure

➔ Une jointure permet de “lier” 2 tables



	id [PK] integer	titre character varying(255)	genre character var	categorie character varying(100	id_auteur integer	id_editeur integer
1	471	Un long dimanche de fiançailles	récit	historique	251	1
2	300	L'éternité, mon amour !	récit	science fiction	271	2
3	226	Tournoi dans l'Himalaya	récit	apprentissage	289	2
4	353	L'ange de la retirada	BD	historique	447	3
5	516	No et moi	récit	réaliste	47	4
6	510	Le pigeon	récit	''	454	4
7	504	J'ai 15 ans et je ne veux pas mourir	récit	autobiographie	377	4
8	472	Un secret	récit	autobiographie	429	4
9	420	La nuit du renard	récit	policier	122	4
10	116	Le premier roi du monde : l'épopée de Gi	récit	aventures	59	4
11	218	Mon miel ma douceur	récit	album	318	5

	id [PK] serial	nom character varying(255)
1	1	Folio
2	2	Tertium
3	3	6 pieds sous terre
4	4	Le livre de poche
5	5	Didier
6	6	L'Amourier

➔Tous les éditeurs sont stockées dans la table “editeur”

➔Chaque livre est lié à un éditeur

Module SQL

Produit cartésien

- ➔ Il s'agit d'une jointure sans condition
- ➔ Syntaxe :
  - ◆ SELECT champ1, champ2
    FROM table1 AS alias1, table2 AS alias2, ...
- ➔ le résultat est la fusion des 2 tables :
  - ◆ chaque ligne de table1 est répétée autant de fois qu'il y a de ligne dans table2

Module SQL

Jointure avec condition

- ➔ Syntaxe non normalisée :
  - SELECT champ1, champ2
    FROM table1, table2
    WHERE table1.colonne = table2.colonne ;
- ➔ Syntaxe normalisée :
  - SELECT champ1, champ2
    FROM table1 JOIN table2 ON table1.colonne = table2.colonne

Module SQL

Jointure externe

- ➔ Cela permet d'afficher les lignes d'une table qui ne correspondent pas à la condition de jointure.
- ➔ Par exemple quand le champ de jointure est vide
- ➔ Syntaxe
  - ◆ SELECT champ1, champ2
    FROM table1 LEFT OUTER JOIN table2 ON table1.colonne = table2.colonne

Module SQL

Exercice 23

➔ Lister tous les livres avec leur auteur

Module SQL

Exercice 24

➔ Lister tous les livres avec leur auteur et leur éditeur

Module SQL

Exercice 25

➔ Afficher la liste des éditeurs avec le nombre de livre de chacun

Module SQL

Exercice 26

➔ Lister tous les livres actuellement empruntés avec le nom de l'abonné l'ayant emprunté

Module SQL

Exercice 27

➔ Trouver les abonnés ayant emprunté un livre depuis plus de 2 mois et ne l'ayant pas rendu

Module SQL

Exercice 28

➔ Lister tous les livres dont la dernière date d'emprunt date de plus de 2 ans

Module SQL

Exercice 29

➔ Trouver les 10 abonnés ayant emprunté le plus de livres (rendus et non rendus)

Module SQL

Exercice 30

➔ Lister les catégories les plus lues

Module SQL

Exercice 31

➔ Trouver les abonnés ayant plusieurs livres en leur possession

Module SQL

Exercice 32

➔ Trouver le nombre de livre empruntés pour chaque année

Module SQL

Exercice 33

➔ Trouver l'âge moyen des abonnés

Module SQL

Exercice 34

➔ Trouver la ville ayant la moyenne d'âge la plus petite

Module SQL

Exercice 35

➔ Lister tous les abonnés qui ont des homonyme (même nom et prénom)

Module SQL

Exercice 36

➔ Trouver l'âge moyen des lecteurs de chaque catégorie

Module SQL

1. Structure d'une base
2. Installation de MySQL
3. Interrogation de données
4. Requêtes multi-tables
5. **Requêtes complexes**
6. Manipulation de données
7. Indexes
8. Transactions

- Opérateurs ensemblistes
- Sous-requêtes mono-ligne
- Sous-requêtes multi-lignes
- Sous-requêtes dans la clause from

5. Requêtes complexes

Module SQL

Objectif :

- ➔ Comprendre la syntaxe des sous-requêtes
- ➔ Écrire des requêtes utilisant des sous-requêtes mono-lignes
- ➔ Écrire des requêtes utilisant des sous-requêtes multi-lignes
- ➔ Écrire des requêtes utilisant des sous-requêtes dans la clause FROM

Module SQL

Sous requête mono-ligne

- ➔ Utilisation d'un Select renvoyant une ligne unique
- ➔ Peut être utilisé dans
  - ◆ Comme champ dans la clause SELECT
  - ◆ Comme valeur de condition dans la clause WHERE
- ➔ Syntaxe :
  - ◆ SELECT champ1, champ2, (sous-requête)
    FROM table
  - ◆ SELECT champ1, champ2
    FROM table
    WHERE champ opérateur (sous-requête)

Module SQL

Exercice 39

➔ Lister les 10 livres les plus empruntés (avec le nombre d'emprunt)

Module SQL

Exercice 40

➔ Lister tous les abonnés avec le dernier livre qu'ils ont empruntés même s'ils n'ont jamais emprunté de livre (sous select dans le SELECT)

Module SQL

Sous requête multi-lignes

- ➔ Utilisation d'un Select renvoyant plusieurs ligne dans la clause WHERE
- ➔ Utilisation des opérateur
  - IN
  - NOT IN
  - EXISTS
  - NOT EXISTS
- ➔ Syntaxe :
  - SELECT champ1, champ2
    FROM table
    WHERE champ [NOT] IN (sous-requête)

Module SQL

Exercice 41

➔ Lister les livres qui n'ont jamais été empruntés

Module SQL

1. Structure d'une base
2. Installation de MySQL
3. Interrogation de données
4. Requêtes multi-tables
5. Requêtes complexes
6. **Manipulation de données**
7. Indexes
8. Transactions

- La commande INSERT
- La commande UPDATE
- La commande DELETE

6. Manipulation de données

Module SQL

Objectif :

- ➔ Savoir insérer des données dans la base
- ➔ Savoir modifier des données dans la base
- ➔ Savoir supprimer des données de la base

Module SQL

La commande INSERT

- ➔ Permet d'insérer des données dans la base
- ➔ Toutes les colonnes
  - ◆ INSERT INTO table VALUES(valeur1, valeur2, …)
- ➔ Colonne spécifiques
  - ◆ INSERT INTO table (col1, col2, …) VALUES(valeur1, valeur2, …)
  - ◆ Les colonnes non alimentées ne doivent pas être obligatoire (NOT NULL)
- ➔ Plusieurs lignes
  - ◆ INSERT INTO table (col1, col2, …) VALUES
    (valeur1, valeur2, …),
    (valeur1b, valeur2b, …)

Module SQL

Insertion à partir d'une requête

- ➔ Permet d'insérer des lignes dans une table à partir de données provenant d'une requête
- ➔ Syntaxe :
  - ◆ INSERT INTO table (col1, col2, ...) SELECT col1b, col2b, ... FROM table2 ;

Module SQL

Exercice 47

➔ Ajouter une ligne dans la table abonné avec votre nom

Module SQL

Exercice 48

➔ Ajouter 3 personnes dans la table abonné en une seule requête

Module SQL

La commande UPDATE

- ➔ Permet de modifier les données d'une table
- ➔ Syntaxe :
  - ◆ UPDATE table SET col1 = valeur1, col2 = valeur2, …
    WHERE condition ;
- ➔ Les valeurs peuvent être des valeurs fixes ou des valeurs issues de sous requêtes

Module SQL

Exercice 49

➔ Mettre à jour la date de fin de votre abonnement à la date du jour + 1 an

Module SQL

La commande DELETE

- ➔ Permet de supprimer des lignes d'une table
- ➔ Syntaxe :
  - ◆ DELETE FROM table
    WHERE condition ;

Module SQL

Exercice 50

➔ Supprimer la ligne correspondant à votre nom dans la table abonné

Module SQL

La commande TRUNCATE

- ➔ Permet de supprimer TOUTES les lignes d'une table
- ➔ Syntaxe :
  - ◆ TRUNCATE table ;

Module SQL

Exercice 51

➔ Essayer d'insérer une ligne dans la table livre avec un id_auteur qui n'existe pas dans la table auteur

Module SQL

Exercice 52

➔ Remplir la table genre à partir des différents genre présents dans la table livre

Module SQL

Exercice 53

➔ Mettre à jour la colonne id_genre de la table livre en fonction de la valeur contenue dans la colonne genre

Module SQL

Exercice 54

➔ Essayer de supprimer une ligne de la table genre

Module SQL

1. Structure d'une base
2. Installation de MySQL
3. Interrogation de données
4. Requêtes multi-tables
5. Requêtes complexes
6. Manipulation de données
7. **Indexes**
8. Transactions

- La commande INSERT
- La commande UPDATE
- La commande DELETE

7. Indexes

Module SQL

Les index

- ➔ S'applique à une ou plusieurs colonnes
- ➔ Permettent un accès plus rapide aux données.
- ➔ Permettent de forcer l'unicité
- ➔ Syntaxe :
  - ◆ CREATE INDEX nom_index
    ON table (col1, ...) ;

  - ◆ CREATE UNIQUE INDEX nom_index
    ON table (col1, ...) ;

Module SQL

Démonstration

La base formation0 contient 3,3 M de lignes

- ➔ Faire une recherche sur un abonné par son nom et noter le temps d'exécution
  - ◆ 1,6 seconde
- ➔ Créer un index sur la colonne nom de la table abonne
  - ◆ Cela va prendre environ 10 s
- ➔ Refaire la recherche et comparer le temps d'exécution
  - ◆ 0 s

Module SQL

Démonstration

- ➔ Créer un index unique sur le nom, prenom, date_naissance de la table abonne
- ➔ Essayer d'ajouter une donnée déjà existante

Module SQL

1. Structure d'une base
2. Installation de MySQL
3. Interrogation de données
4. Requêtes multi-tables
5. Requêtes complexes
6. Manipulation de données
7. Indexes
8. **Transactions**

- Qu'est qu'une transaction
- Gestion du rollback
- Verrous

8. Transactions

Module SQL

Objectif :

- ➔ Connaître le principe des transactions
- ➔ Savoir les gérer
- ➔ Connaître l'impact sur les verrous

Module SQL

Notion de transaction

- ➔ Une transaction permet de regrouper plusieurs ordre sql de mise à jour en un ensemble cohérent
- ➔ Toutes les mises à jour effectuées sont validées ensemble à la fin de la transaction (Commit)
- ➔ Si la transaction est annulée les actions déjà effectué sont annulées (Rollback)

Module SQL

Utilisation

➔ Syntaxe :

- BEGIN ;
  <sql>
  <sql>
  COMMIT ; ou ROLLBACK ;

Module SQL

Verrouillage

- ➔ Lorsqu'un enregistrement est mis à jour, il est verrouillé (Lock) jusqu'à ce que la transaction soit terminée
- ➔ Les transactions doivent être courtes

Module SQL

Démonstration

➔ Effectuer plusieurs mises à jour de la table abonne après avoir démarré une transaction puis effectuer un rollback

Restons en contact.

DIGINAMIC

Lionel Cabon, Directeur
contact@diginamic.fr

IOCEAN
Les projets informatiques dont les clients parlent avec plaisir

Nos coordonnées : 04 34 09 04 60
contact@iocean.fr - www.iocean.fr

N° Déclaration OF : 91 34 08867 34
Nantes, Paris, Montpellier
www.diginamic.fr