| **Mise en place de la base de données** |
| --- |

Visualiser la vidéo ci-dessous pour télécharger le logiciel **mysql workbench de 0 à 6:29**

[**https://www.youtube.com/watch?v=isik2eA-eEw**](https://www.youtube.com/watch?v=isik2eA-eEw)

1. **Importer la base de données sur lien ci-dessous**

[**https://drive.google.com/file/d/1NcrfRPvSSKpNknSvDAcxPQ3R1kghLfzz/view?usp=sharing**](https://drive.google.com/file/d/1NcrfRPvSSKpNknSvDAcxPQ3R1kghLfzz/view?usp=sharing)

1. **Exécuter le fichier**
2. **Aller sur Schémas**
3. **Rafraîchir la base**
4. **Visualiser les tables de la base de données sql\_store.**

|  |
| --- |

**Q1. citer les différentes tables sur la base de données sql\_store**

| **USE,SELECT,FROM, DISTINCT** |
| --- |

**Q2. commenter les lignes en bleu.**

| **USE sql\_store;**  **Utilise la base de donnée**  **SELECT \* FROM customers**  **Affiche les données de customers** | **SELECT \* FROM sql\_store.customers**  **Affiche les données de customers depuis sql\_store** |
| --- | --- |

**ressources**

[**https://sql.sh/cours/select**](https://sql.sh/cours/select)

| **USE** | **Utilise la base de donnée** |
| --- | --- |
| **SELECT** | **Selectionne la colonne d’une bdd** |
| **FROM** | **Nom de la bdd** |

**Q3. Copier coller et comparer les deux requêtes précédentes .**

**1- Zone écriture code**

**2- Exécuter**

**3- Visualiser**

|  |
| --- |

**Q4. l’étoile (\*) permet de sélectionner toutes les colonnes du tableau.**

**On souhaite maintenant sélectionné le nom est le prénom ainsi que l’adresse des clients (customers)**

**réaliser le code sql qui permet de faire cette tâche.**

| **SELECT first\_name,last\_name,address FROM customers** |
| --- |

**Q5. Il est possible d’utiliser des opérations mathématiques dans l'instruction SELECT afin d’afficher le résultat de ce calcule, tester l’exemple ci-dessous:**

| **USE sql\_store;**  **SELECT last\_name,**  **first\_name,**  **points,points\*100**  **FROM customers** |
| --- |

**On souhaite maintenant changer le nom de la colonne “point\*100” par "résultat”.**

**voir la colonne avant et après.**

|  |  |
| --- | --- |

**Documentation:**

[**https://sql.sh/cours/alias**](https://sql.sh/cours/alias)

**USE sql\_store;**

**SELECT last\_name,**

**first\_name,**

**points,points\*100 AS résultats**

**FROM customers**

**Q6. Utilise la table product et afficher les noms des produit ,les prix, le nouveau prix = (prix\*1.1) avec l’attribution des noms comme ci-dessous.**

|  |
| --- |

**USE sql\_store;**

**SELECT name AS nom\_produit,unit\_price AS ancien\_prix ,unit\_price\*1.1 AS nouveau\_prix FROM products**

**Q7. Sélectionner le tableau customers en cliquant sur l'icône ci-dessous:**

|  |
| --- |

**Ensuite modifier la valeur state de la ligne 1 par VA au lieu de MA et cliquer sur Apply**

|  |
| --- |

**Écrire la requête qui permet de sélectionner la colonne state dans la table customers.**

| **SELECT state FROM sql\_store.customers;** |
| --- |

**Vous aurez le résultat suivant avec deux clients du même département :**

|  |
| --- |

**Q8. Ecrire une requête qui permet de ne pas sélectionner les lignes en double (sans redondance)**

**SELECT DISTINCT state FROM sql\_store.customers;**

**Documentation:**

[**https://sql.sh/cours/distinct**](https://sql.sh/cours/distinct)

| **WHERE** |
| --- |

**Q9.Exécuter la requête suivante.**

| **USE sql\_store;**  **SELECT \* FROM customers**  **WHERE first\_name ='Ines'** |
| --- |

**Que fait l’instruction WHERE ?**

**Elle filtre un élément**

**Q10 Sélectionner tous les clients avec un identifiant inférieur à 10 avec**

**l’instruction WHERE et un opérateur de comparaison.**

**USE sql\_store;**

**SELECT \* FROM customers**

**WHERE customer\_id<10**

**Documentation:**

[**https://sql.sh/cours/where**](https://sql.sh/cours/where)

**Q11. Quels sont les opérateurs de comparaison utiles avec l’instruction WHERE.**

| Opérateur | Description |
| --- | --- |
| = | Égale |
| <> | Pas égale |
| != | Pas égale |
| > | Supérieur à |
| < | Inférieur à |
| >= | Supérieur ou égale à |
| <= | Inférieur ou égale à |
| IN | Liste de plusieurs valeurs possibles |
| BETWEEN | Valeur comprise dans un intervalle donnée (utile pour les nombres ou dates) |
| LIKE | Recherche en spécifiant le début, milieu ou fin d'un mot. |
| IS NULL | Valeur est nulle |
| IS NOT NULL | Valeur n'est pas nulle |

**Q12. Tester les deux requêtes ci-dessous et commenter.**

| **SELECT \***  **FROM Customers**  **WHERE birth\_date > '1990-01-01'** | **SELECT \***  **FROM Customers**  **WHERE birth\_date > '01-01-1990'** |
| --- | --- |

**Les 2 requêtes son semblable mais la 2ème ne marchera pas car le format de la date n’est pas bon**

**Q13. Dans la table orders(commandes) présente dans la base de données sql\_store, afficher toutes les commandes (ordres) faites après 2019.**

**SELECT \* FROM sql\_store.orders**

**WHERE order\_date > "2019-01-01"**

| **Opérateurs AND, OR , NOT** |
| --- |

**Q14.** Comparer et expliquer les deux requêtes ci-dessous.

| 1 | 2 |
| --- | --- |
| SELECT \*  FROM Customers  WHERE birth\_date > '1990-01-01' AND points > 1000 | SELECT \*  FROM Customers  WHERE birth\_date > '1990-01-01' OR points > 1000 |

L’une combine les 2 filtre pour en faire un unique l’autre additione chaque filtre

**Documentation**

[**https://sql.sh/cours/where/and-or**](https://sql.sh/cours/where/and-or)

**Q15. On peut également associer les opérateurs logiques.**

**Commenter la requête ci-dessous en vous basant sur le résultat obtenu.**

|  |
| --- |

**Q16. Afin de comprendre bien la question suivante, essayer de démontrer l’équation mathématique ci-dessous.**

**Sachant que \* correspond à AND**

**et + correspond à OR**

|  |
| --- |

| **A** | **B** | **/(A+B)** | **(/A \* /B)** |
| --- | --- | --- | --- |
| **0** | **0** | **0** | **0** |
| **0** | **1** | **1** | **0** |
| **1** | **0** | **1** | **0** |
| **1** | **1** | **2** | **1** |

**Exemple avec SQL a tester.**

| **/(A+B)** | **(/A\*/B)** |
| --- | --- |
|  |  |

**Maintenant donner le code SQL de la requête ci-dessous sans l’opérateur NOT**

| **SELECT \***  **FROM Customers**  **WHERE NOT (birth\_date > '1990-01-01' AND points > 1000 )** |
| --- |

**SELECT \***

**FROM Customers**

**WHERE (birth\_date < '1990-01-01' OR points < 1000 )**

**Q17. On utilisera la base de données order\_items,**

**Sélectionner les commandes(orders) avec l'identifiant 6 et total prix supérieur à 30**

**SELECT \***

**from order\_items**

**where order\_id=6 and unit\_price\*quantity > 30**

**Remarque: total prix =quantité\*prix unitaire**

| **IN** |
| --- |

**L’opérateur IN est équivalent à plusieurs opérateurs logiques ‘OU’.**

**Comparer les deux exemples ci-dessous pour comprendre :**

| **SELECT \***  **FROM sql\_store.customers**  **WHERE state = 'VA' OR state = 'GA' OR state ='FL'** | **SELECT \***  **FROM sql\_store.customers**  **WHERE state IN ('VA','FL','GA')** |
| --- | --- |

**Q18.Utiliser la table ‘products’ et afficher les produits qui ont une quantité de stock équivalente à 49 ou 38 ou bien 72 avec l’opérateur IN.**

SELECT \*

FROM sql\_store.products

WHERE quantity\_in\_stock IN(49,38,72);

| **BETWEEN** |
| --- |

**L’opérateur BETWEEN est utilisé dans une requête SQL pour sélectionner un intervalle de données dans une requête utilisant WHERE. L’intervalle peut être constitué de chaînes de caractères, de nombres ou de dates**

**voir exemple :**

| **SELECT \* FROM sql\_store.customers**  **WHERE points >=1000 AND points <=3000** | **SELECT \* FROM sql\_store.customers**  **WHERE points BETWEEN 1000 AND 3000** |
| --- | --- |

**Q19. Utiliser la table customers ‘clients’ et sélectionner les clients nées entre 1/1/1990 et 1/1/2000**

**Remarque: il y a une manière pour écrire les dates sur une requête SQL.**

**SELECT \* FROM sql\_store.customers**

**WHERE birth\_date BETWEEN "1990-1-1" and "2000-1-1"**

| **LIKE** |
| --- |

**L’opérateur LIKE est utilisé dans la clause WHERE des requêtes SQL. Ce mot-clé permet d’effectuer une recherche sur un modèle particulier.**

**Voir exemple.**

| **SELECT \* FROM sql\_store.customers**  **WHERE last\_name LIKE 'brushfield'** |
| --- |

**On peut également sélectionner les noms qui commencent par ‘b’ de la manière suivante.**

| **SELECT \* FROM sql\_store.customers**  **WHERE last\_name LIKE 'b%'** |
| --- |

**Ou bien les noms qui se terminent par ‘y’**

| **SELECT \* FROM sql\_store.customers**  **WHERE last\_name LIKE '%y'** |
| --- |

**Q20. Dans la table customers (clients), afficher les clients avec une adresse contenant “TRAIL” ou bien “AVENUE”.**

**Remarque: cette opérateur est très puissant si on utilise les expressions régulières**

**mais on ne va pas aborder cela dans cette séance.**

**documentation:**

[**https://www.lucaswillems.com/fr/articles/25/tutoriel-pour-maitriser-les-expressions-regulieres**](https://www.lucaswillems.com/fr/articles/25/tutoriel-pour-maitriser-les-expressions-regulieres)

| **ORDER BY** |
| --- |

**La commande ORDER BY permet de trier les lignes dans le résultat d’une requête SQL. Il est possible de trier les données par ordre ascendant ou descendant.**

**Tester la requête ci-dessous.**

| **SELECT \***  **FROM sql\_store.customers**  **ORDER BY first\_name DESC** |
| --- |

**Q21. Ensuite enlever DESC, Que se passe-t-il ?**

**Avec DESC sa inverse l’ordre des de la liste en fonction du prénom**

**Donc sans DESC c’est de A à Z en fonction du prénom**

**Q22.Expliquer le mode de tri ci-dessous.**

**Elle trie dans l’ordre en priorité les états puis les prénoms**

| **SELECT \***  **FROM customers**  **ORDER BY state,first\_name** |
| --- |

**Q23.Nous allons utiliser la table (order\_items), sélectionner toutes les commandes(orders) numéro 2 ensuite trier ces commandes par ordre ascendant de leurs (quantité\*prix).**

**SELECT \***

**FROM sql\_store.order\_items**

**WHERE order\_id LIKE '2'**

**ORDER BY quantity\*unit\_price;**

| **TRAVAUX PRATIQUES** |
| --- |

**Importer la base de données sur le lien ci-dessous dans votre logiciel Workbench.**

[**https://drive.google.com/file/d/1jTPslZeTF7XYMmqqu6el9Z\_DWS9\_1uZ\_/view?usp=sharing**](https://drive.google.com/file/d/1jTPslZeTF7XYMmqqu6el9Z_DWS9_1uZ_/view?usp=sharing)

1. **sélectionner le titre des films**

**SELECT titre FROM films.film;**

1. **sélectionner les réalisateurs sans redondance.**

**SELECT distinct prenom\_realisateur,nom\_realisateur FROM films.film;**

1. **sélectionner tous les films dont les recettes sont supérieurs à 1000**

**SELECT \***

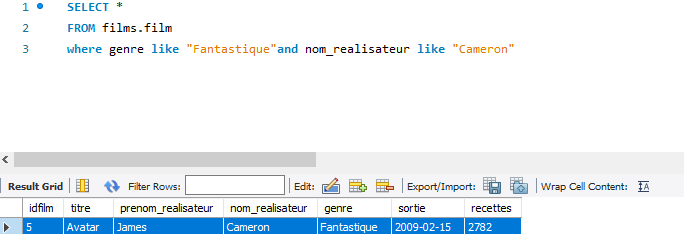
**FROM films.film**

**where recettes>1000**

1. **Quels sont les films fantastiques .**

****

1. **Quels sont les films fantastiques réalisés par James Cameron .**

****

1. **Afficher les films Fantastique par ordre croissant des recettes.**

**SELECT \***

**FROM films.film**

**where genre like "Fantastique"**

**order by recettes**

1. **Afficher tous les films fantastiques réalisés entre 2000 et 2010.**

**SELECT \***

**FROM films.film**

**where genre like "Fantastique"and sortie BETWEEN "2000-1-1" and "2010-1-1"**

1. **Afficher tous les films réalisés entre janvier 2010 et décembre 2010.**

**SELECT \***

**FROM films.film**

**where sortie BETWEEN "2000-1-1" and "2010-1-1"**

**order by recettes**

1. **Afficher tous les films dont le titre commence par “P”.**

**SELECT \***

**FROM films.film**

**WHERE titre like"P%"**

1. **Afficher tous les films dont le titre comporte le mot «Monde».**

**SELECT \***

**FROM films.film**

**WHERE titre like"%monde%"**

1. **Afficher les différentes catégories de films.**

**SELECT distinct genre**

**FROM films.film**

