

## Exercice 1:

crée une base de données:

✓ Affichage des lignes 0 - 6 (total de 7, traitement en 0,0002 seconde(s).)

```
SELECT * FROM `film`
```

☐ Profilage [ [Éditer en ligne](#) ] [ [Éditer](#) ] [ [Expliquer SQL](#) ] [ [Créer le code source PHP](#) ] [ [Actualiser](#) ]

☐ Tout afficher | Nombre de lignes : 25 ▼ | Filtrer les lignes:

Options supplémentaires

titre	durée	date
the gentlemate	113	0000-00-00
the gentlemate	113	0000-00-00
the gentlemate	113	0000-00-00
Kuzco	75	0000-00-00
Joker	122	0000-00-00
Sweeney Todd	116	0000-00-00
Princesse Mononoké	134	0000-00-00

supprimer avec une requête sql les films qui sont sortis avant 2010 et qui durent moins de 120 min

```
1 DELETE FROM film
2 WHERE date < 2010
3     AND durée < 120;
```

## exercice 2:

1/ voici ma requête sql

```
1 SELECT ville_nom_reel, ville_surface
2 FROM villes_france_free
3 ORDER BY ville_surface ASC
4 LIMIT 5;
```

2/ voici ma requête sql

```
1 SELECT ville_code_postal, ville_population_2010
2 FROM villes_france_free
3 ORDER BY ville_population_2010 DESC
4 LIMIT 15;
```

3/ voici ma requêtes sql

```
1 SELECT ville_departement, ville_nom
2 FROM villes_france_free
3 WHERE ville_nom LIKE 'P%';|
```

4/ voici ma requêtes sql

```
1 SELECT ville_departement, SUM(ville_population_2010) AS population_totale
2 FROM villes_france_free
3 GROUP BY ville_departement
4 ORDER BY population_totale DESC;|
```

5/ voici ma requêtes sql

```
1 SELECT ville_nom, ville_population_1999
2 FROM villes_france_free
3 WHERE ville_population_1999 > 20000
4 ORDER BY ville_population_1999 DESC;|
```

6/ voici ma requêtes sql

```
1 UPDATE villes_france_free
2 SET ville_longitude_dms = REPLACE(ville_longitude_dms, '+', '*');
```

7/ voici ma requêtes sql

```
1 UPDATE villes_france_free
2 SET ville_nom = REVERSE(ville_nom)
3 WHERE ville_departement LIKE '97%';|
```

8/ voici ma requêtes sql

```
1 SELECT ville_nom, ville_surface, ville_departement
2 FROM villes_france_free
3 ORDER BY ville_surface DESC
4 LIMIT 5;
```

9/ voici ma requêtes sql

```
1 SELECT ville_departement, SUM(ville_population_2010) AS population_totale
2 FROM villes_france_free
3 GROUP BY ville_departement
4 ORDER BY population_totale DESC
5 LIMIT 5;
```

exercice 3:

1/ crée la base de donnée via une requete sql:

```
1 CREATE DATABASE IF NOT EXISTS electromenager;
2
3 USE electromenager;
4
5 CREATE TABLE IF NOT EXISTS produits (
6     id INT AUTO_INCREMENT PRIMARY KEY,
7     marque VARCHAR(50) NOT NULL,
8     categorie VARCHAR(50) NOT NULL,
9     prix INT NOT NULL,
10    modele VARCHAR(50) NOT NULL
11 );
12
13 INSERT INTO produits (marque, categorie, prix, modele) VALUES
14 ('Bosch', 'Lave-vaisselle', 699, 'WOP1154A'),
15 ('Brandt', 'Lave-vaisselle', 599, 'WTC1256H'),
16 ('Brandt', 'Lave-vaisselle', 699, 'WTC1356L'),
17 ('Gorenje', 'Lave-linge', 389, 'AD65L425F'),
18 ('Indesit', 'Lave-linge', 359, 'XP89X333G'),
19 ('Indesit', 'Lave-linge', 409, 'XP92X333L'),
20 ('Miele', 'Lave-linge', 1099, 'ME20H252V'),
21 ('Thomson', 'Lave-linge', 549, 'AS192201S'),
22 ('Thomson', 'Lave-linge', 699, 'AS192301S');
```

2/ voici ma requêtes sql

```
1 SELECT modele, marque, categorie
2 FROM produits;
```

3/ voici ma requêtes sql

```
1 DELETE FROM produits
2 WHERE marque = 'Indesit';|
```

4/ voici ma requêtes sql

```
1 UPDATE produits
2 SET prix = prix - 50
3 WHERE prix > 600;
```

Exercice 4:

1/ voici ma requetes pour crée ma base de donnée

```
1 CREATE DATABASE IF NOT EXISTS personnages_metiers;
2
3 USE personnages_metiers;
4
5 CREATE TABLE IF NOT EXISTS personnages (
6     id INT AUTO_INCREMENT PRIMARY KEY,
7     nom VARCHAR(50) NOT NULL,
8     metier VARCHAR(100) NOT NULL
9 );
10
11 INSERT INTO personnages (nom, metier) VALUES
12 ('Leia', 'Brodeur de nuage'),
13 ('Sherlock', 'Dresseur de limace'),
14 ('Lara', 'Brodeur de nuage'),
15 ('Mario', 'Plieur de rayons'),
16 ('Arya', 'Plieur de rayons'),
17 ('Dexter', 'Plieur de rayons'),
18 ('Neo', 'Plieur de rayons'),
19 ('Katniss', 'Plieur de rayons');
```

2/ voici ma requêtes sql

```
1 SELECT nom, metier
2 FROM personnages
3 WHERE metier IS NOT NULL AND metier <> '';
```

3/ voici ma requêtes sql

```
1 SELECT nom, metier
2 FROM personnages;
3 |
```

4/ voici ma requêtes sql

```
1 SELECT nom
2 FROM personnages
3 WHERE metier IS NULL OR metier = '';
```

exercice 5:

1/ creation de la base de donné:

```
1 CREATE DATABASE IF NOT EXISTS site_multimedia;
2
3 USE site_multimedia;
4
5 CREATE TABLE IF NOT EXISTS utilisateurs (
6     id INT AUTO_INCREMENT PRIMARY KEY,
7     nom VARCHAR(50) NOT NULL
8 );
9
10 CREATE TABLE IF NOT EXISTS produits (
11     id INT AUTO_INCREMENT PRIMARY KEY,
12     nom VARCHAR(100) NOT NULL
13 );
14
15 CREATE TABLE IF NOT EXISTS notes (
16     id INT AUTO_INCREMENT PRIMARY KEY,
17     utilisateur_id INT NOT NULL,
18     produit_id INT NOT NULL,
19     note INT CHECK (note BETWEEN 1 AND 5),
20     FOREIGN KEY (utilisateur_id) REFERENCES utilisateurs(id) ON DELETE CASCADE,
21     FOREIGN KEY (produit_id) REFERENCES produits(id) ON DELETE CASCADE
22 );
23
24 INSERT INTO utilisateurs (nom) VALUES
25 ('Alice'), ('Bob'), ('Charlie'), ('David'), ('Eve');
26
27 INSERT INTO produits (nom) VALUES
28 ('Télévision HD'), ('Ordinateur Portable'), ('Casque Audio');
29
30 INSERT INTO notes (utilisateur_id, produit_id, note) VALUES
31 (1, 1, 5), (1, 2, 4), (1, 3, 3),
32 (2, 1, 4), (2, 2, 5), (2, 3, 2),
33 (3, 1, 3), (3, 2, 3), (3, 3, 4),
34 (4, 1, 5), (4, 2, 4), (4, 3, 5),
35 (5, 1, 4), (5, 2, 5), (5, 3, 3);
```

2/ affichage de la moyenne de chaque produit:

```
1 SELECT p.nom AS produit, AVG(n.note) AS moyenne_note
2 FROM produits p
3 JOIN notes n ON p.id = n.produit_id
4 GROUP BY p.id
5 ORDER BY moyenne_note DESC;
```