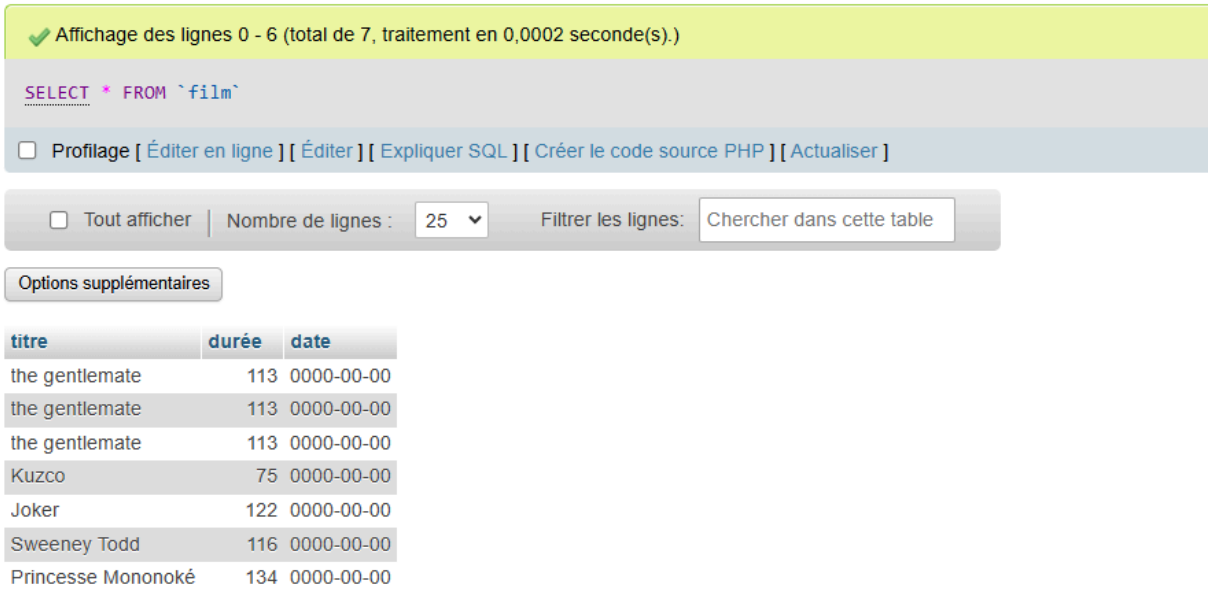
Exercice 1:

créee un base de donnée:



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 requetes sql les films qui sont sortis avant 2010 et qui durent moins de 120 min

```sql
DELETE FROM film
WHERE date < 2010
  AND durée < 120;
```

exercice 2:

1/ voici ma requêtes sql

```sql
SELECT ville_nom_reel, ville_surface
FROM villes_france_free
ORDER BY ville_surface ASC
LIMIT 5;
```

2/ voici ma requêtes sql

```sql
SELECT ville_code_postal, ville_population_2010
FROM villes_france_free
ORDER BY ville_population_2010 DESC
LIMIT 15;
```

3/ voici ma requêtes sql

```sql
SELECT ville_departement, ville_nom
FROM villes_france_free
WHERE ville_nom LIKE 'P%';
```

4/ voici ma requêtes sql

```sql
SELECT ville_departement, SUM(ville_population_2010) AS population_totale
FROM villes_france_free
GROUP BY ville_departement
ORDER BY population_totale DESC;
```

5/ voici ma requêtes sql

```sql
SELECT ville_nom, ville_population_1999
FROM villes_france_free
WHERE ville_population_1999 > 20000
ORDER BY ville_population_1999 DESC;
```

6/ voici ma requêtes sql

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

7/ voici ma requêtes sql

```sql
UPDATE villes_france_free
SET ville_nom = REVERSE(ville_nom)
WHERE ville_departement LIKE '97%';
```

8/ voici ma requêtes sql

```sql
SELECT ville_nom, ville_surface, ville_departement
FROM villes_france_free
ORDER BY ville_surface DESC
LIMIT 5;
```

9/ voici ma requêtes sql

```sql
SELECT ville_departement, SUM(ville_population_2010) AS population_totale
FROM villes_france_free
GROUP BY ville_departement
ORDER BY population_totale DESC
LIMIT 5;
```

exercice 3:

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

```sql
CREATE DATABASE IF NOT EXISTS electromenager;

USE electromenager;

CREATE TABLE IF NOT EXISTS produits (
    id INT AUTO_INCREMENT PRIMARY KEY,
    marque VARCHAR(50) NOT NULL,
    categorie VARCHAR(50) NOT NULL,
    prix INT NOT NULL,
    modele VARCHAR(50) NOT NULL
);

INSERT INTO produits (marque, categorie, prix, modele) VALUES
('Bosch', 'Lave-vaisselle', 699, 'WOP1154A'),
('Brandt', 'Lave-vaisselle', 599, 'WTC1256H'),
('Brandt', 'Lave-vaisselle', 699, 'WTC1356L'),
('Gorenje', 'Lave-linge', 389, 'AD65L425F'),
('Indesit', 'Lave-linge', 359, 'XP89X333G'),
('Indesit', 'Lave-linge', 409, 'XP92X333L'),
('Miele', 'Lave-linge', 1099, 'ME20H252V'),
('Thomson', 'Lave-linge', 549, 'AS192201S'),
('Thomson', 'Lave-linge', 699, 'AS192301S');
```

2/ voici ma requêtes sql

```sql
SELECT modele, marque, categorie
FROM produits;
```

3/ voici ma requêtes sql

```sql
DELETE FROM produits
WHERE marque = 'Indesit';
```

4/ voici ma requêtes sql

```sql
UPDATE produits
SET prix = prix - 50
WHERE prix > 600;
```

Exercice 4:

1/ voici ma requetes pour crée ma base de donnée

```sql
CREATE DATABASE IF NOT EXISTS personnages_metiers;

USE personnages_metiers;

CREATE TABLE IF NOT EXISTS personnages (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nom VARCHAR(50) NOT NULL,
    metier VARCHAR(100) NOT NULL
);

INSERT INTO personnages (nom, metier) VALUES
('Leia', 'Brodeur de nuage'),
('Sherlock', 'Dresseur de limace'),
('Lara', 'Brodeur de nuage'),
('Mario', 'Plieur de rayons'),
('Arya', 'Plieur de rayons'),
('Dexter', 'Plieur de rayons'),
('Neo', 'Plieur de rayons'),
('Katniss', 'Plieur de rayons');
```

2/ voici ma requêtes sql

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

3/ voici ma requêtes sql

```sql
SELECT nom, metier
FROM personnages;
```

4/ voici ma requêtes sql

```sql
SELECT nom
FROM personnages
WHERE metier IS NULL OR metier = '';
```

exercice 5:

1/ creation de la base de donné:

```sql
CREATE DATABASE IF NOT EXISTS site_multimedia;

USE site_multimedia;

CREATE TABLE IF NOT EXISTS utilisateurs (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nom VARCHAR(50) NOT NULL
);

CREATE TABLE IF NOT EXISTS produits (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nom VARCHAR(100) NOT NULL
);

CREATE TABLE IF NOT EXISTS notes (
    id INT AUTO_INCREMENT PRIMARY KEY,
    utilisateur_id INT NOT NULL,
    produit_id INT NOT NULL,
    note INT CHECK (note BETWEEN 1 AND 5),
    FOREIGN KEY (utilisateur_id) REFERENCES utilisateurs(id) ON DELETE CASCADE,
    FOREIGN KEY (produit_id) REFERENCES produits(id) ON DELETE CASCADE
);

INSERT INTO utilisateurs (nom) VALUES
('Alice'), ('Bob'), ('Charlie'), ('David'), ('Eve');

INSERT INTO produits (nom) VALUES
('Télévision HD'), ('Ordinateur Portable'), ('Casque Audio');

INSERT INTO notes (utilisateur_id, produit_id, note) VALUES
(1, 1, 5), (1, 2, 4), (1, 3, 3),
(2, 1, 4), (2, 2, 5), (2, 3, 2),
(3, 1, 3), (3, 2, 3), (3, 3, 4),
(4, 1, 5), (4, 2, 4), (4, 3, 5),
(5, 1, 4), (5, 2, 5), (5, 3, 3);
```

2/ affichage de la moyenne de chaque produit:

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

exercice bonus, pokemon:

0/ creation de la base de donnée, avec des requetes sql,

```sql
CREATE DATABASE PokemonRivals;

USE PokemonRivals;

CREATE TABLE Rivals (
    RivalID INT AUTO_INCREMENT PRIMARY KEY,
    Name VARCHAR(50) NOT NULL,
    Age INT NOT NULL,
    City VARCHAR(50) NOT NULL
);

CREATE TABLE Pokemon (
    PokemonID INT AUTO_INCREMENT PRIMARY KEY,
    Name VARCHAR(50) NOT NULL,
    Power INT NOT NULL,
    Type1 VARCHAR(50) NOT NULL,
    Type2 VARCHAR(50),
    RivalID INT,
    FOREIGN KEY (RivalID) REFERENCES Rivals(RivalID) ON DELETE CASCADE
);
```

insertion des donnée via requête sql:

```sql
INSERT INTO Rivals (Name, Age, City) VALUES ('Victor', 26, 'Kanto');
INSERT INTO Rivals (Name, Age, City) VALUES ('Christy', 18, 'Kalos');
INSERT INTO Rivals (Name, Age, City) VALUES ('Dianthéa', 32, 'Alola');
INSERT INTO Rivals (Name, Age, City) VALUES ('Cathy', 48, 'Kanto');
INSERT INTO Rivals (Name, Age, City) VALUES ('Matis', 16, 'Alola');
INSERT INTO Rivals (Name, Age, City) VALUES ('Flo', 32, 'Sinnoh');

INSERT INTO Pokemon (Name, Power, Type1, Type2, RivalID)
VALUES ('Tentacool', 45, 'Eau', 'Poison', 1);
INSERT INTO Pokemon (Name, Power, Type1, Type2, RivalID)
VALUES ('Galar', 49, 'Poison', 'Fée', 1);

INSERT INTO Pokemon (Name, Power, Type1, Type2, RivalID)
VALUES ('Lucario', 55, 'Combat', 'Acier', 2);
INSERT INTO Pokemon (Name, Power, Type1, Type2, RivalID)
VALUES ('Steelix', 53, 'Sol', 'Acier', 2);
INSERT INTO Pokemon (Name, Power, Type1, Type2, RivalID)
VALUES ('Pikachu', 40, 'Electrique', NULL, 2);

INSERT INTO Pokemon (Name, Power, Type1, Type2, RivalID)
VALUES ('Elektor', 65, 'Vol', 'Electrique', 3);

INSERT INTO Pokemon (Name, Power, Type1, Type2, RivalID)
VALUES ('Pikachu', 40, 'Electrique', NULL, 4);
INSERT INTO Pokemon (Name, Power, Type1, Type2, RivalID)
VALUES ('Tentacool', 45, 'Eau', 'Poison', 4);
INSERT INTO Pokemon (Name, Power, Type1, Type2, RivalID)
VALUES ('Magikarp', 1, 'Eau', NULL, 4);

INSERT INTO Pokemon (Name, Power, Type1, Type2, RivalID)
VALUES ('Papilusion', 35, 'Vol', 'Insecte', 5);
INSERT INTO Pokemon (Name, Power, Type1, Type2, RivalID)
VALUES ('Lucario', 55, 'Combat', 'Acier', 5);

INSERT INTO Pokemon (Name, Power, Type1, Type2, RivalID)
VALUES ('Galar', 49, 'Poison', 'Fée', 6);
INSERT INTO Pokemon (Name, Power, Type1, Type2, RivalID)
VALUES ('Rondoudou', 34, 'Normal', 'Fée', 6);
INSERT INTO Pokemon (Name, Power, Type1, Type2, RivalID)
VALUES ('Togepi', 20, 'Fée', NULL, 6);
```

1/ changé le noms de mathis par sont vrais noms,

```sql
UPDATE Rivals
SET Name = 'Gros Nul'
WHERE Name = 'Matis';
```

2/ affichée tout les pokemon avec leur puissance,

```sql
SELECT Name AS PokemonName, Power AS PokemonPower
FROM Pokemon;
```

3/ afficher les rival entre 18 et 40 ans qui ne vienne pas de alola,

```sql
SELECT Name AS RivalName, Age, City
FROM Rivals
WHERE Age BETWEEN 18 AND 40
  AND City != 'Alola';
```

4/ affiché les pokemon de type fée,

```sql
SELECT Name AS PokemonName, Power, Type1, Type2
FROM Pokemon
WHERE Type1 = 'Fée' OR Type2 = 'Fée';
```

5/ affichée la puissance de chaque equipe

```sql
SELECT Rivals.Name AS RivalName, SUM(Pokemon.Power) AS TotalPower
FROM Rivals
JOIN Pokemon ON Rivals.RivalID = Pokemon.RivalID
GROUP BY Rivals.Name;
```

6/ pour chaque pokémon, la liste des dresseurs qui possèdent se pokémon, et les types d'éléments qu'il possède

```sql
SELECT
    Pokemon.Name AS PokemonName,
    GROUP_CONCAT(DISTINCT Rivals.Name SEPARATOR ', ') AS RivalNames,
    Pokemon.Type1 AS Type1,
    Pokemon.Type2 AS Type2
FROM Pokemon
JOIN Rivals ON Pokemon.RivalID = Rivals.RivalID
GROUP BY Pokemon.Name, Pokemon.Type1, Pokemon.Type2;
```

7/ et pour finir affichée la puissance electrique de chaque dresseur avec +10%

```sql
SELECT
    Rivals.Name AS RivalName,
    ROUND(SUM(Pokemon.Power) * 1.1, 2) AS AdjustedElectricPower
FROM Pokemon
JOIN Rivals ON Pokemon.RivalID = Rivals.RivalID
WHERE Pokemon.Type1 = 'Electrique' OR Pokemon.Type2 = 'Electrique'
GROUP BY Rivals.Name;

```