Mise en place d'une base de données relationnelle

Sommaire

Méthode MERISE

Methode d'**E**tude et de **R**ealisation **I**nformatique pour les **S**ystemes d'**E**ntreprise

MCD

MLD

MPD

Base de données

SGBD

Bases de données

SQL

MySQL

Installation de MySQL Workbench : https://dev.mysql.com/downloads/workbench/

SQL

Langage de définition de données

Langage de manipulation de données

Langage de contrôle de données

Langage de contrôle des transactions

Création, modification et suppression de tables

CREATE

DROP

RENAME

ALTER

clé primaire, clé étrangère

```
CREATE TABLE article (
    article_id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    article_name VARCHAR(255) NOT NULL,
    unit_price DECIMAL(10, 2) NOT NULL
);
```

```
CREATE TABLE invoice (
    invoice_id INT NOT NULL PRIMARY KEY,
    date DATE NOT NULL,
    time TIME NOT NULL,
    total DECIMAL(10, 2) NOT NULL
);
```

```
CREATE TABLE invoice_item (
    invoice_id INT,
    article_id INT,
    quantity INT NOT NULL,
    PRIMARY KEY (invoice_id, article_id),
    FOREIGN KEY (invoice_id) REFERENCES invoice(invoice_id),
    FOREIGN KEY (article_id) REFERENCES article(article_id)
);
```

Insertion, modification et suppression de données

INSERT INTO

UPDATE

DELETE

```
INSERT INTO article (article_name, unit_price) VALUES ('BAGUETTE', 1.00);
INSERT INTO article (article_name, unit_price) VALUES ('BAGUETTE TRADITION',
1.20);
INSERT INTO article (article_name, unit_price) VALUES ('BAGUETTE CEREALES', 1.30);
INSERT INTO article (article_name, unit_price) VALUES ('CAMPAGNE', 2.00);
INSERT INTO article (article_name, unit_price) VALUES ('COMPLET', 1.80);
INSERT INTO article (article_name, unit_price) VALUES ('SEIGLE', 2.00);
INSERT INTO article (article_name, unit_price) VALUES ('BANETTE', 1.20);
INSERT INTO article (article_name, unit_price) VALUES ('PAIN', 1.10);
INSERT INTO article (article_name, unit_price) VALUES ('PAIN SPECIAL', 2.50);
INSERT INTO article (article_name, unit_price) VALUES ('BOULE 200G', 1.10);
INSERT INTO article (article_name, unit_price) VALUES ('BOULE 400G', 1.60);
INSERT INTO article (article_name, unit_price) VALUES ('CROISSANT', 1.20);
INSERT INTO article (article_name, unit_price) VALUES ('PAIN AU CHOCOLAT', 1.30);
INSERT INTO article (article_name, unit_price) VALUES ('PAIN AUX RAISINS', 1.80);
INSERT INTO article (article_name, unit_price) VALUES ('CHAUSSON AUX POMMES',
2.20);
INSERT INTO article (article_name, unit_price) VALUES ('BRIOCHE', 4.00);
INSERT INTO article (article_name, unit_price) VALUES ('GAL FRANGIPANE 4P', 8.00);
INSERT INTO article (article_name, unit_price) VALUES ('GAL FRANGIPANE 6P',
12.00);
INSERT INTO article (article_name, unit_price) VALUES ('COOKIE', 1.20);
INSERT INTO article (article_name, unit_price) VALUES ('CHOUQUETTE', 0.30);
```

```
INSERT INTO invoice VALUES (160020, '2024-01-02', '08:32', 5.10);
INSERT INTO invoice VALUES (160021, '2024-01-02', '08:38', 4.60);
INSERT INTO invoice VALUES (160022, '2024-01-02', '09:12', 6.00);
INSERT INTO invoice VALUES (160023, '2024-01-02', '09:25', 5.60);
INSERT INTO invoice VALUES (160024, '2024-01-02', '09:27', 1.20);
INSERT INTO invoice VALUES (160025, '2024-01-02', '09:30', 10.80);
INSERT INTO invoice VALUES (160026, '2024-01-02', '09:37', 20.80);
INSERT INTO invoice VALUES (160027, '2024-01-02', '09:41', 3.60);
INSERT INTO invoice VALUES (160028, '2024-01-02', '09:46', 3.60);
INSERT INTO invoice VALUES (160029, '2024-01-02', '09:57', 3.50);
INSERT INTO invoice VALUES (160030, '2024-01-02', '09:58', 2.50);
INSERT INTO invoice VALUES (160031, '2024-01-02', '10:03', 4.80);
INSERT INTO invoice VALUES (160032, '2024-01-02', '10:05', 10.90);
INSERT INTO invoice VALUES (160033, '2024-01-02', '10:12', 13.60);
INSERT INTO invoice VALUES (160034, '2024-01-02', '10:17', 6.40);
INSERT INTO invoice VALUES (160035, '2024-01-02', '10:25', 8.60);
```

```
INSERT INTO invoice_item VALUES (160020, 2, 1);
INSERT INTO invoice_item VALUES (160020, 13, 3);
INSERT INTO invoice_item VALUES (160021, 13, 2);
INSERT INTO invoice_item VALUES (160021, 4, 1);
INSERT INTO invoice_item VALUES (160022, 2, 5);
INSERT INTO invoice_item VALUES (160023, 1, 2);
INSERT INTO invoice_item VALUES (160023, 12, 3);
INSERT INTO invoice_item VALUES (160024, 7, 1);
INSERT INTO invoice_item VALUES (160025, 2, 3);
INSERT INTO invoice_item VALUES (160025, 12, 6);
INSERT INTO invoice_item VALUES (160026, 13, 4);
INSERT INTO invoice_item VALUES (160026, 12, 4);
INSERT INTO invoice_item VALUES (160026, 5, 6);
INSERT INTO invoice_item VALUES (160027, 12, 3);
INSERT INTO invoice_item VALUES (160028, 14, 2);
INSERT INTO invoice_item VALUES (160029, 2, 2);
INSERT INTO invoice_item VALUES (160029, 8, 1);
INSERT INTO invoice_item VALUES (160030, 9, 1);
INSERT INTO invoice_item VALUES (160031, 2, 4);
INSERT INTO invoice_item VALUES (160032, 12, 5);
INSERT INTO invoice_item VALUES (160032, 2, 2);
INSERT INTO invoice_item VALUES (160032, 9, 1);
INSERT INTO invoice_item VALUES (160033, 11, 1);
INSERT INTO invoice_item VALUES (160033, 18, 1);
INSERT INTO invoice_item VALUES (160034, 2, 2);
INSERT INTO invoice_item VALUES (160034, 16, 1);
INSERT INTO invoice_item VALUES (160035, 20, 12);
INSERT INTO invoice_item VALUES (160035, 3, 2);
INSERT INTO invoice_item VALUES (160035, 2, 2);
```

Sélection de données

Sélectionner toutes les colonnes :

```
SELECT *
FROM invoice;
```

```
SELECT *
FROM article;
```

Sélectionner quelques colonnes :

```
SELECT invoice_id, total
FROM invoice;
```

```
SELECT article_name, unit_price
FROM article;
```

Sélectionner des valeurs uniques d'une colonne avec `DISTINCT` :

```
SELECT DISTINCT invoice_id
FROM invoice_item;
```

```
SELECT DISTINCT article_id
FROM invoice_item;
```

`LIMIT` :

```
SELECT *
FROM invoice
LIMIT 5;
```

Fonctions d'agrégation :

- `COUNT()` compter le nombre de lignes
- `SUM()` calculer la somme sur un ensemble de lignes
- `AVG()` calculer la moyenne sur un ensemble de lignes
- `MIN()` obtenir la valeur minimum d'une colonne
- `MAX()` obtenir la valeur maximum d'une colonne

```
SELECT COUNT(*)
FROM invoice;
```

```
SELECT SUM(total)
FROM invoice;
```

```
SELECT AVG(total)
FROM invoice;
```

```
SELECT MIN(total)
FROM invoice;
```

```
SELECT MAX(total)
FROM invoice;
```

La clause `WHERE` :

```
SELECT *
FROM invoice
WHERE invoice_id = 160025;
```

Des opérateurs courants pour la clause `WHERE` :

```
=
>
```

```
<
>=
<=
<>
BETWEEN ... AND ...
IN
LIKE
NOT IN
NOT LIKE
AND
OR
```

```sql
SELECT *
FROM invoice
WHERE total >= 10;
```

```sql
SELECT *
FROM invoice
WHERE time BETWEEN '9:00' AND '9:30';
```

```sql
SELECT *
FROM article
WHERE article_id IN (12, 13, 14, 15);
```

```sql
SELECT *
FROM article
WHERE article_name LIKE 'BAGUETTE%';
```

```sql
SELECT *
FROM article
WHERE article_name NOT LIKE 'BAGUETTE%';
```

```sql
SELECT *
FROM invoice
WHERE time BETWEEN '9:00' AND '9:30'
AND total > 5;
```

Il est possible faire une sous-requête dans la clause `WHERE` :

```
SELECT *
FROM invoice_item
WHERE article_id =
    (SELECT
     article_id
     FROM article
     WHERE article_name = 'CROISSANT');
```

Les tris avec `ORDER BY` :

```
SELECT *
FROM invoice
ORDER BY total;
```

```
SELECT *
FROM invoice
ORDER BY total DESC;
```

Les regroupements avec `GROUP BY` :

```
SELECT article_id, SUM(quantity) AS quantite_vendue_par_article
FROM invoice_item
GROUP BY article_id;
```

On utilise `AS` pour donner un alias à une colonne ou à une table dans une requête.

La clause `HAVING` pour les données agrégées :

```
SELECT article_id, SUM(quantity) AS quantite_vendue_par_article
FROM invoice_item
GROUP BY article_id
HAVING quantite_vendue_par_article >= 5;
```

Les jointures : https://fr.wikipedia.org/wiki/Jointure_(informatique)

```
SELECT *
FROM invoice_item
INNER JOIN article ON invoice_item.article_id = article.article_id;
```

```sql
SELECT *
FROM invoice_item
LEFT JOIN article ON invoice_item.article_id = article.article_id;
```

```sql
SELECT *
FROM invoice_item
RIGHT JOIN article ON invoice_item.article_id = article.article_id;
```

En MySQL :

- `JOIN` = `INNER JOIN` = `CROSS JOIN` (https://dev.mysql.com/doc/refman/9.1/en/join.html)
- `LEFT JOIN` = `LEFT OUTER JOIN`
- `RIGHT JOIN` = `RIGHT OUTER JOIN`

Le `FULL JOIN` ou `FULL OUTER JOIN` n'est pas supporté par MySQL.

On peut utiliser des alias pour avoir des noms plus courts :

```sql
SELECT item.invoice_id, a.article_name, item.quantity
FROM invoice_item AS item
INNER JOIN article AS a ON item.article_id = a.article_id
ORDER BY item.invoice_id;
```

On peut joindre les trois tables :

```sql
SELECT i.invoice_id, i.date, i.time, a.article_name, a.unit_price, item.quantity,
i.total
FROM invoice_item AS item
INNER JOIN invoice AS i ON item.invoice_id = i.invoice_id
INNER JOIN article AS a ON item.article_id = a.article_id
ORDER BY i.invoice_id;
```

Compléments

Quelques mots sur

- Transactions ACID
- Sécurité
- Bases de données NoSQL
- Aspect réglementaire : RGPD, CNIL