How to Create a Database?

Schema:

screenShootSchema

Time to Read: 7 min

Physical Store Database:

This project is designed as a smaller integration of the French physical store `Intermarché`, and it manages various aspects of the business such as ***bosses***, ***cash registers***, ***establishments***, ***products***, ***users***, ***shopping carts***, ***payments***, and ***online orders (drive)***.

Database Structure:

1. **User**: Stores the personal information of users.
2. **Establishment**: Stores the details of each ***cash register*** and the associated ***boss***.
3. **CartItem**: Stores the history of ***user*** purchases in the supermarket.
4. **Product**: Contains all relevant details of the ***products*** available in the store.
5. **OrderDetails**: Stores details of each order, including ***payment*** information.
6. **Payment**: Tracks user payments and provides data for bank transactions.
7. **CashRegister**: Contains information on the ***cash registers*** in each ***establishment***.

Links into the database:

1. **user** and **OrderDetail** have one relationship **(1:N)**.
2. **user** and **cardItem** have one relationship **(1:N)**.
3. **user** and **etablishment** have one relasionship **(1:N)**.
4. **user** and **payement** have one relasionship **(1:N)**.
5. **etablishment** and **boss** have one relastionship **(1:N)**.
6. **etablishment** and **cashRegister** have two relationship **(2:N)**.
7. **cardItem** and **product** have one relationship **(1:N)**.

Command executed for create and setup database:

Setup the databse

First for creat the databse run this command:

```sql
CREATE DATABASE myDataBaseName;
```

Creat tables:

`This code is on the main branch (dataBase.sql)`

*Also all the **nonSQL** string are in quote => it's for have no probleme with SQL compilor.*

Creat product TABLE

```
CREATE TABLE "product" (
    "idProduct" INT PRIMARY KEY,
    "dateExpiration" DATE NOT NULL,
    "nameProduct" VARCHAR(20) NOT NULL,
    "price" DECIMAL(10, 2) NOT NULL,
    "stockQuantity" INT NOT NULL DEFAULT 0
);
```

Explanation: Each product has a unique ID (idProduct) defined as a primary key of type INT. Other important fields like expiration date, price, and stock quantity are also included.

Creat product cashRegister

```
CREATE TABLE "product" (
    CREATE TABLE "cashRegister" (
    "idCashRegister" INT PRIMARY KEY,
    "cashInRegister" FLOAT
);
```

Explanation: Every cash register has a unique ID (idCashRegister), and the cashInRegister field tracks the amount of cash present.

Creat product establishment

```
CREATE TABLE "establishment" (
    "idEtablissement" INT PRIMARY KEY,
    "localisation" VARCHAR(20) NOT NULL,
    "dateOpenAndClose" DATE,
    "idRegisterCashManual" INT,
    "idRegisterCashAuto" INT,
    FOREIGN KEY ("idRegisterCashManual") REFERENCES "cashRegister"
("idCashRegister"),
    FOREIGN KEY ("idRegisterCashAuto") REFERENCES "cashRegister"
("idCashRegister")
);
```

Explanation: Establishment stores data about the store location and references both manual and automatic cash registers through foreign keys.

Creat boss

```
CREATE TABLE "boss" (
    "idBoss" INT PRIMARY KEY,
    "locationOfWorkId" INT NOT NULL,
    FOREIGN KEY ("locationOfWorkId") REFERENCES "establishment"
("idEtablissement")
);
```

Explanation: Each boss has a unique ID (idBoss) and is linked to an establishment through a foreign key (locationOfWorkId).

Creat product user

```
CREATE TABLE "user" (
    "IdUser" INT PRIMARY KEY,
    "firstName" VARCHAR(16) NOT NULL,
    "lastName" VARCHAR(16) NOT NULL,
    "email" VARCHAR(50) NOT NULL UNIQUE,
    "phoneNumber" VARCHAR(15),
    "favoriteStoreId" INT,
    FOREIGN KEY ("favoriteStoreId") REFERENCES "establishment"
("idEtablissement")
);
```

Explanation: Each user has a unique ID (IdUser), and fields for first name, last name, email, and phone number. The user's favorite store is linked through a foreign key.

Creat product OrderDetail

```
CREATE TABLE "OrderDetail"(
    "idOrderDetail" INT PRIMARY KEY,
    "nameOrder" VARCHAR(20) NOT NULL UNIQUE,
    "timeToPrepareOrder" TIME NOT NULL,
    "nameOfTheClient" VARCHAR(20),
    "customerId" INT,
    FOREIGN KEY ("customerId") REFERENCES "user"("IdUser")
);
```

Explanation: OrderDetail tracks the preparation time, order name, and is linked to the user placing the order through the customerId foreign key.

Creat product CartItem

```
CREATE TABLE "CartItem"(
    "idCartItem" INT PRIMARY KEY,
    "productId" INT,
    "customerId" INT,
    "quantity" INT NOT NULL,
    FOREIGN KEY ("productId") REFERENCES "product"("idProduct"),
    FOREIGN KEY ("customerId") REFERENCES "user"("IdUser")
);
```

Explanation: Each CartItem records which product and user are involved, along with the quantity, with foreign keys linking to the product and user tables.

Creat product payment

```
CREATE TABLE "payment"(
    "idPayment" INT PRIMARY KEY,
    "nameOfSender" VARCHAR(50),
    "nameOfReceiver" VARCHAR(50),
    "amount" DECIMAL(10, 2) NOT NULL,
    "customerId" INT NOT NULL,
    "orderDetailId" INT NOT NULL,
    FOREIGN KEY ("customerId") REFERENCES "user"("IdUser"),
    FOREIGN KEY ("orderDetailId") REFERENCES "OrderDetail"
("idOrderDetail")
);
```

Explanation: The payment table records who sends and receives the payment, the amount, and links to the user and the OrderDetail through foreign keys.

Some Queries SQL

Let's put some product into products

```
INSERT INTO "product" ("idProduct", "dateExpiration", "nameProduct",
"price", "stockQuantity")
VALUES (1, '2024-12-31', 'Apple', 1.50, 200);
```

Output

```
INSERT 0 1
```

Check:

```
intermarche=# SELECT * FROM "product" WHERE "idProduct" = 1;
```

Output

```
 idProduct | dateExpiration | nameProduct | price | stockQuantity
-----------+----------------+-------------+-------+---------------
         1 | 2024-12-31     | Apple       |  1.50 |           200
(1 ligne)
```

Let's creat a new user

```sql
INSERT INTO "user" ("IdUser", "firstName", "lastName", "email",
"phoneNumber", "favoriteStoreId")
VALUES (5, 'Alice', 'Smith', 'alice@example.com', '1234567890', NULL);
```

Output

```
INSERT 0 1
```

Check

Check 1 for see if Alice's data on data base:

```sql
SELECT * FROM "user" WHERE "IdUser" = 5;
```

Output

```
 IdUser | firstName | lastName |       email       | phoneNumber |
favoriteStoreId
--------+-----------+----------+-------------------+-------------+--------
---------
      5 | Alice     | Smith    | alice@example.com | 1234567890  |
(1 ligne)
```

Check 2 for check user's data:

```sql
SELECT * FROM "user";
```

Output

```
 IdUser | firstName | lastName |           email           | phoneNumber |
favoriteStoreId
--------+-----------+----------+---------------------------+-------------
+-----------------
      1 | Alice     | Durand   | alice.durand@example.com  | 0601234567  |
      2 | Bob       | Martin   | bob.martin@example.com    | 0612345678  |
      3 | Claire    | Dupont   | claire.dupont@example.com | 0623456789  |
      5 | Alice     | Smith    | alice@example.com         | 1234567890  |
(4 lignes)
```

Take order with a client and check the order

```
INSERT INTO "OrderDetail" ("idOrderDetail", "nameOrder",
"timeToPrepareOrder", "nameOfTheClient", "customerId")
VALUES (1, 'AliceOrder1', '00:30:00', 'Alice', 1);
```

Output

```
INSERT 0 1
```

Check command :

```
SELECT * FROM "OrderDetail" WHERE "customerId" = 1;
```

Output

```
 idOrderDetail |  nameOrder  | timeToPrepareOrder | nameOfTheClient |
customerId
---------------+-------------+--------------------+-----------------+------
-------
             1 | AliceOrder1 | 00:30:00           | Alice           |
1
(1 ligne)
```

Add an element in the cart of the user

```
INSERT INTO "CartItem" ("idCartItem", "productId", "customerId",
"quantity")
VALUES (1, 1, 1, 3);
```

Check command:

```
SELECT * FROM "CartItem";
```

or

```
SELECT * FROM "CartItem" WHERE "customerId" = 1;
```

if you want a specific user Output

```
idCartItem | productId | customerId | quantity
------------+-----------+------------+----------
          1 |         1 |          1 |        3
(1 ligne)
```

Add a payement for a command

```
INSERT INTO "payment" ("idPayment", "nameOfSender", "nameOfReceiver",
"amount", "customerId", "orderDetailId")
VALUES (1, 'Alice Smith', 'Intermarché', 4.50, 1, 1);
```

Check command without a specific ID:

```
SELECT * FROM "payement";
```

and check with a specifique ID:

```
SELECT * FROM "payement" WHERE "customerId" = 1;
```

Output:

```
 idPayment | nameOfSender | nameOfReceiver | amount | customerId |
orderDetailId
-----------+--------------+----------------+--------+------------+--------
-------
         1 | Alice Smith  | Intermarché    |   4.50 |          1 |
1
(1 ligne)
```

Same output because I have do not add other payement (only one on the dataBase for now).