DATABASE

Final Project

INFORMATION STORAGE AND RETIEVAL SYSTEM

TABLES

EER PLOT

SQL

Narges BabaAhmadi

610398102

SCHOOL OF MATHEMATICS, STATISTICS AND COMPUTER SCIENCE

Abstract:

In this project I examined a few online shops and gathered some information about the data they require to save in their database. The next step was to have a brief explanation of the environment I am creating a DBMS for. Then, I extracted the entities and relations from that environment description and created an EER diagram. After finding the tables from the diagram and normalizing them up to BCNF normal form, I created the tables in *DataGrip* using *mysql* and added data to each table. I have also created some views, triggers, functions and stored procedures in this project.

Part 1:

Choosing the environment:

ONLINE SHOP

Part 2:

Explaining the environment:

1. This online shop registers various users. The information to be held on each user includes a unique ID, first name, last name, password, Phone number, gender ,and age. We also want to keep user's address information which includes one or two addresses, city, country, fax ,and postal code. Each user has one address information but there can be several users with the same address information.
   Each user can order from this website as many times as he/she wants to.
2. There are two type of users, premium user and normal user. Each premium user has a credit card number that should be saved.
   The list of all premium users and all normal users is being checked every month.
3. There are also so many people working here and the information stored on each employee includes a unique ID , first name, last name, email, phone number, age, hired date, gender, salary, address information(just like users), a photo ,and one job title. Each employee has one address information but there can be several employees with the same address information.
   There is also a manager who checks all job titles monthly, and adds to them if needed.
4. Some of the employees are also managers. All managers have a meeting each month and the number of meetings each manager has attended is being checked by the CEO.
5. This online shop has several products. The information stored on each product includes a unique ID, product's name, a description of the product, its price, an image of the product, product's weight ,and its stock.
   Each product can have a few comments that has been written by various users.
6. Each product is in one category and different categories are being edited and checked every month and there is a possibility that the manager adds to the list of categories sometimes.
7. Each product has supplier and the information stored on each supplier is a unique id, phone number, company name, website URL ,and address information. Each product can have multiple suppliers and each supplier can have multiple products.
8. When a user orders a product, there information than needs to be saved is a unique order ID, supplier ID, order's date, shipping date, products' cost, delivery cost, total discount ,and total fee.

Each order can contain multiple products.

9. Each user can have several orders but each order is for one user.
10. Each order has a delivery method and a payment method.
11. Each delivery method has a unique ID, name, company name, company website ,and description.
12. Each payment method has a unique ID, name, provider and description.
13. Users can have a discount code and discount information includes a unique discount ID, name, expiration date and description.
14. Only one discount code can be used for each order ,and each discount code can be used once.
15. There is also a support option on orders and if any user had any problem related to his/her order, he/she can call the support number and the problem's description and the employee's ID who has talked to the user should also be kept.

Part 3:

Finding entities and attributes:

1. Entity: users
   Attributes: user ID, first name, last name, password, Phone number, gender ,and age.
2. As it is said in the description, the list of all premium users and all normal users is **being checked every month**, so I needed to have two entities for them. Although this will **use more memory** but my system **runs faster** in this way.
   All users are either normal or premium.
   Both, premium and normal, should be **weak entity** because if I delete a user from users' table, considering whether that user was normal or premium, it should also get deleted from normal or premium table.
   Entity: Premium
   Attributes: card_num.
3. Entity: normal users

* I could turn these three tables into one table and add a columns to that table which includes Boolean values and show whether each user is premium or normal in that single table but due to **two reasons** I made three tables although in this way I will have redundancy:

1- Making one table and adding that Boolean column will result in having so many NULL values and I want to prevent that.

2- The list of premium and normal users will be checked each month so making three tables will expedite my system.

4. As the address information is **common** in users, employees and suppliers, I made an entity out of that so than less storage is being used.
   Entity: Addresses
   Attributes: address ID, one or two addresses, city, country, fax and postal code.
5. Entity: Employees
   Attributes: emp ID, first name, last name, email, phone number, age, hired date, gender, salary and a photo path.

6. Managers are being **checked too often** and the number of meeting they have attended should also be stored ,so to make my system **faster**(in finding the list of managers) and to use **less storage**(to store the number of meeting only for managers) I made an entity for that.
   Managers are a small part of employees so this should be an **IS-A relation**.
   Entity: managers
   Attributes: meeting.No
   This entity is also a **weak entity** because it depends on employee entity and if we delete an employee from employees' table, it will also get deleted from managers' table.
7. Job titles are being **checked monthly** and add to if needed ,that's why I needed an entity for jobs.
   Entity: Jobs
   Attribute: job ID, job title.
8. Entity: Products
   Attributes: product ID, product's name, description, price, image path, weight, stock.
9. Entity: comments
   Attributes : comment
   This is also a **weak entity** because we delete a product, the comments of that product will also get deleted.
10. The list of all categories should be able to be seen and add to if needed so its better to have an entity for categories.
    Entity: Categories
    Attributes: Category ID, name and description.
11. Each supplier has some attributes and adding all of that to each order is **waste of memory**, that's why I made an entity for suppliers.
    Entity: Suppliers
    Attributes: supplier ID, phone number, website and company name.
12. It is better to make an entity for orders ,instead of relation. Because orders has discount, payment methods ,and delivery method and each of them has their own attributes so making orders and entity both **expedites** the system and uses **less storage** than other ways.
    Entity: Orders
    Attributes: order ID, order's date, shipping date, products' cost, delivery cost, total discount and total fee.
13. Delivery methods has some attributes as well and because of the reason that is mentioned in entity number 11, it is better to have an entity for delivery methods.
    Entity: Delivery methods
    Attributes: Delivery ID, name, company name, company website and description.
14. Just like delivery methods, having an entity for payments methods will use **less storage**.
    Entity: Payment methods
    Attributes: payment ID, name, provider and description.
15. Entity: Discounts
    Attributes: Discount ID, name, EXP.Date and description.

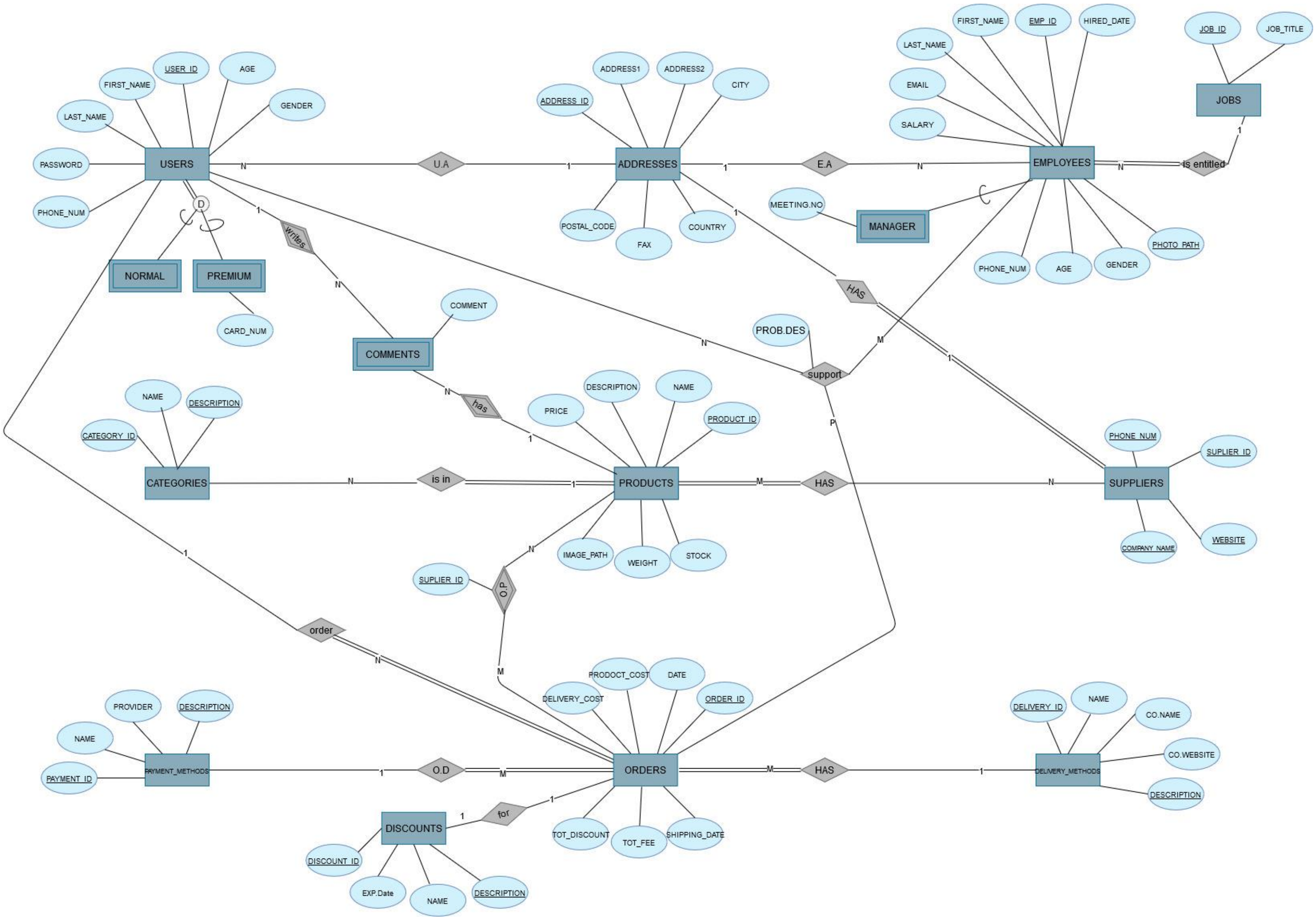
Finding relations:

1. Users and Address : a N:1 relation which shows the address information of each user.

2. Users and Normal: an “is a” relation.
3. User and Premium: an “is a” relation.
4. Users and comments: a 1:N relation because a user can write N comments but each comment is for one user.
5. Employee and Address: a N:1 relation that shows the address information of each employee.
6. Employee and Job: a N:1 relation that indicates the job of each employee.
7. Employee and managers: it is an “is a “ relation.
8. Products and Category: a N:1 relation that shows the category of a product.
9. Products and Supplier: a N:M relation that shows the supplier of each product.
10. Comments and products: a N:1 relation because each comment is for one product but each product can have N comments.
11. Products and Orders: a M:N relation ship that shows the products of each order and the orders that each product is in and the supplier of the ordered product.

    P.O is a **weak relation**, because if we delete an order, all the data about that order in O.P should get deleted.
12. Orders and Delivery methods: a N:1 relation that shows the delivery method of each order.
13. Orders and Payment methods: a N:1 relation that shows the payment method of each order.
14. Orders and discount: a 1:1 relation.
15. Orders and Users: a N:1 relation because each user can order from this website N times.
16. User and Employee: we have a N to M relation here. We make a table through this relation to be able to keep the desired data.
17. Order, User and Employee: a N:M:P relation that shows which user has called for support, which products has that needed a support for and which employee has answered the user’s call.

Part 4:



Part 5 and Part 6:

Turning the diagram to tables:

Let's , first, demonstrate my road map:

* If I had a **M:N** relation(Rl) between en1 and en2 I will make 3 tables:

En1(attr1, …)

En2(attr'1, …)

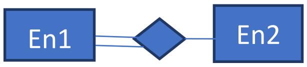
Rl(attr1, attr'1, attr''1)

* If I had a **1:N** relation between en1 and en2 I will make 2 tables:

En1(attr1, …)

En2(attr’1, attr1, …)

* If I had a **1:1** mandatory relationship between en1 and en2 I will make two tables:



En1(attr1, attr’1, …)

En2(attr’1, …)

* If I had an **IS-A relation** I make a table for the father with its primary key and attributes and we make another table for the child with father’s PK and child’s attributes.

* If I had a weak entity and a weak relation I will add the strong entity’s PK as an FK and a part of PK to the weak entity’s table.

Now it’s time to make my tables:

1. **USERS**(USER_ID, FIRST_NAME, LAST_NAME, PASSWORD, PHONE_NUM, AGE, GENDER, ADDRESS_ID)

   ADDRESS_ID is FK from ADDRESSES.

2. **PREMIUM**(USER_ID, CARD_NUM)

   I have an IS-A relation here, so the PK is equal to the PK of the USERS table.

   Premium is also a weak entity but using USER_ID as PK in enough and unique and there is no need to add another attribute to this table.

3. **NORMAL**(USER_ID)

   I have an IS-A relation here, so the PK is equal to the PK of the USERS table.

   NORMAL is also a weak entity but using USER_ID as PK in enough and unique and there is no need to add another attribute to this table.

4. **ADDRESSES**(ADDRESS_ID, ADDRESS1, ADDRESS2, CITY, COUNTRY, FAX, POSTAL_CODE)

5. **EMPLOYEES**(EMP_ID, FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUM, AGE, SALARY, PHOTO_PATH, HIRED_DATE, GENDER, ADDRESS_ID, JOB_ID)

   I have N:1 relation between employee and addresses and also employee and jobs, so that I have two foreign keys here. One from the addresses table and one from the jobs table.

   ADDRESS_ID is **FK** from ADDRESSES.

   JOB_ID is **FK** from JOBS.

6. **MANAGERS**(EMP_ID, MEETING.NO)

   I have an IS-A relation here, so the PK is equal to the PK of the EMPLOYEES table.

MANAGERS is also a weak entity but using EMP_ID as PK in enough and unique and there is no need to add another attribute to this table.

7. **JOBS**(JOB_ID, JOB_TITLE)

8. **PRODUCTS**(PRODUCT_ID, NAME, DESCRIPTION, PRICE, IMAGE_PATH, WEIGHT, STOCK, CATEGORI_ID, SUPPLIER_ID)

   Here I also have two N:1 relations and that is why I have two foreign keys.

   CATEGORY_ID is **FK** from CATEGORIES.

   SUPPLIER_ID is **FK** from SUPPLIERS.

9. **CATEGORIES**(CATEGORY_ID, NAME, DESCRIPTION)

10. **SUPPLIERS**(SUPPLIER_ID, PHONE_NUM, WEBSITE, COMPANY_NAME, ADDRESS_ID)

    The reason of the FK of this table is the N:1 relation between suppliers and addresses.

    ADDRESS_ID is **FK** from ADDRESSES.

11. **COMMENTS**(COMMENT_ID, USER_ID, PRODUCT_ID, COMMENT)

    USER_ID is **FK** from USERS.(we have an N:1 relation between COMMENTS and USERS)

    PRODUCT_ID is **FK** from PRODUCTS. (we have an N:1 relation between COMMENTS and PRODUCTS)

12. **ORDERS**(ORDER_ID, DATE, PRODUCT_COST, DELIVERY_COST, TOT_DISCOUNT, TOT_FEE, SHIPPING_DATE, USER_ID, DISCOUNT_ID, PAYMENT_ID, DELIVERY_ID)

    four N:1 relations -> four FKS

    USER_ID is FK from USERS.

    DISCOUNT_ID is **FK** from DISCOUNTS.

    PAYMENT_ID is **FK** from PAMENT_METHODS.

    DELIVERY_ID is **FK** from DELIVERY_METHODS.

13. **DELIVERY_METHODS**(DELIVERY_ID, NAME, CO.NAME, CO.WEBSITE, DESCRIPTION)

14. **PAYMENT_METHODS**(PAYMENT_ID, NAME, PROVIDER, DESCRIPTION)

15. **DISCOUNTS**(DISCOUNT_ID, EXP.DATE, NAME, DESCRIPTION)

16. **OP**(OP_ID, PRODUCT_ID, ORDER_ID, SUPPLIER_ID)
17. **SUPPORT**(SUPPORT_ID, USER_ID, EMPLOYEE_ID, ORDER_ID, PROB.DES)

    I use op_ID for the primary key , in this way my system will be **faster**.

Part 7:

Normalizing tables:

1. **USERS**(USER_ID, FIRST_NAME, LAST_NAME, PASSWORD, PHONE_NUM, AGE, GENDER, ADDRESS_ID)
   It is in **1NF** because **none** of the attributes contain **multiple values**.
   The primary key is USER_ID , as it only **contains one attribute** this relation is in **2NF**.
   Other than the FDs between the keys and other attributes, **there isn't any FD** in this relation which makes it **3NF**.
   This relation has two CKs, USE_ID and ADDRESS_ID, but these two **don't have** any **common subset** so this relation is **BCNF**.

2. **PREMIUM**(USER_ID, CARD_NUM)
   It only **has one FD** from USER_ID to CARD_NUM so this relation is in **all normal forms**.

3. **NORMAL**(USER_ID)
   This relation has **one attribute** so it is in **all normal forms**.

4. **ADDRESSES**(ADDRESS_ID, MAIN_ADDRESS, ADDRESS2, CITY, COUNTRY, FAX, POSTAL_CODE)
   It is **1NF** because **none** of the attributes contain **multiple values**.
   ADDRESS_ID only **contains one attribute** this relation is **2NF**.
   Besides the FDs from PK to other attributes, there isn't **any other FD** so it is **3NF** and **BCNF**.

5. **EMPLOYEES**(EMP_ID, FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUM, AGE, PHOTO_PATH, HIRED_DATE, GENDER, SALARY, ADDRESS_ID, JOB_ID)
   Just because of all the reasons that table number 4 was in BCNF this table is also in BCNF.
   This relation is also **BCNF**.

6. **MANAGERS**(EMP_ID)
   This relation has **one attribute** so it is in **all normal forms**.

7. **JOBS**(JOB_ID, JOB_TITLE)
   It only has **one FD** from JOB_ID to JOB_TITLE so this relation is in **all normal forms**.

8. **PRODUCTS**(PRODUCT_ID, NAME, DESCRIPTION, PRICE, IMAGE_PATH, WEIGHT, STOCK, CATEGORI_ID, SUPPLIER_ID)
   It is **1NF** because **none** of the attributes contain **multiple values**.
   PRODUCT_ID contains **one attribute** which makes this relation **2NF**.
   Other than the FDs between the keys and other attributes, there **isn't any FD** in this relation which makes it be **3NF**.
   This relation is also **BCNF**.
9. **CATEGORIES**(CATEGORY_ID, NAME, DESCRIPTION)
   It is **1NF** because **none** of the attributes contain **multiple values**.
   The **PK** contains **one attribute** this relation is **2NF**.

Other than the keys, there is only **one attribute** in this relation, so that it is in **3NF** and **BCNF**.

10. **SUPPLIERS**(SUPPLIER_ID, PHONE_NUM, WEBSITE, COMPANY_NAME, ADDRESS_ID)
    Just like table number 8, this one is also **BCNF** because of having those features.

11. **COMMENTS**(COMMENT_ID, COMMENT, USER_ID, PRODUCT_ID)
    None of the attributes contain **multiple values**, a single attribute is the PK and there **isn't** any other **FDs** other than the FDs between the PK and other attributes, so this relation is **BCNF**.

12. **ORDERS**(ORDER_ID, DATE, PRODUCT_COST, DELIVERY_COST, TOT_DISCOUNT, TOT_FEE, SHIPPING_DATE, USER_ID, DISCOUNT_ID, PAYMENT_ID, DELIVERY_ID)
    It is **1NF** and **2NF** and **3NF** because of the reasons I mentioned before.
    This relation has several CKs, ORDER_ID, USER_ID, DISCOUNT_ID, PAYMENT_ID, DELIVERY_ID, but **these don't have any common subset** so this relation is **BCNF**.

13. **DELIVERY_METHODS**(DELIVERY_ID, NAME, DESCRIPTION, CO.NAME, CO.WEBSITE)
    This is also in BCNF(I have mentioned the reasons before).

14. **PAYMENT_METHODS**(PAYMENT_ID, NAME, DESCRIPTION, PROVIDER)
    This is also **BCNF** due to past instructions.

15. **DISCOUNTS**(DISCOUNT_ID, NAME, EXP.DATE, DESCRIPTION)
    This table is also **BCNF** because it's features is just like other BCNF tables.

16. **OP**(OP_ID, PRODUCT_ID, ORDER_ID, SUPPLIER_ID)
    There **isn't** any attribute with **multiple values**. The **PK** contains **one attribute** and all FDs are between the keys and other attributes.
    So this relation is **BCNF**.

17. **SUPPORT**(SUPPORT_ID, USER_ID, EMPLOYEE_ID, ORDER_ID, PROB.DES)
    Just like the previous relations this one is also **BCNF**.

Part 8:

View 1: In this view I made a table for all users who live in paris.

```
CREATE VIEW paris_users AS
SELECT U.FIRST_NAME, U.LAST_NAME, U.PHONE_NUM, U.GENDER
FROM USERS U
INNER JOIN ADDRESSES A
ON U.ADDRESS_ID = A.ADDRESS_ID
WHERE A.CITY= 'paris' ;
```

View 2: This view includes all adults.

```
CREATE VIEW adult_user AS
SELECT FIRST_NAME, LAST_NAME, AGE
```

```
FROM USERS
WHERE AGE>18;
```

View 3: I also wanted to have a list of all available products.

```
CREATE VIEW stored_product AS
SELECT PRODUCT_ID, DESCRIPTION, STOCK,PRICE
FROM PRODUCTS
WHERE STOCK>0;
```

View 4: This view includes all delivery methods that Maerks Line presents.

```
CREATE VIEW royal_mail_info AS
SELECT CO_WEBSITE, NAME, DESCRIPTION
FROM DELIVERY_METHODS
WHERE CO_NAME= 'Maersk Line';
```

View 5: In this view I made a table of all datascientists of this company.

```
CREATE VIEW datascientists AS
SELECT E.FIRST_NAME, E.LAST_NAME, E.EMAIL
FROM EMPLOYEES E
INNER JOIN JOBS J
ON E.JOB_ID = J.JOB_ID
WHERE J.JOB_TITLE = 'datascientist' ;
```

PART 9:

making some select and insert queries:

1. Selecting all products in groceries category.

```
SELECT P.NAME, P.DESCRIPTION, P.PRICE
FROM PRODUCTS P
INNER JOIN CATEGORIES C
ON P.CATEGORY_ID = C.CATEGORY_ID
WHERE C.NAME = 'groceries';
```

2. Selecting first name and last name of all premium users.

```
SELECT U.USER_ID, U.FIRST_NAME, U.LAST_NAME
FROM USERS U
INNER JOIN PREMIUM P
ON U.USER_ID = P.USER_ID;
```

3. Selecting ID and EXP date of all 15% OFF discounts.

```
SELECT DISCOUNT_ID, EXP_DATE
FROM DISCOUNTS
WHERE DESCRIPTION = '15% OFF';
```

4. Selecting all employees who are older than 40.

```
SELECT FIRST_NAME, LAST_NAME, SALARY
FROM EMPLOYEES
WHERE AGE > 40;
```

5. Selecting all products that are more expensive than 150$.

```
SELECT ORDER_ID, SHIPPING_DATE
FROM ORDERS
WHERE TOT_FEE > 150;
```

6. Selecting the image and description and supplier’s website of all products with the stock of 1 and more.

```
SELECT P.IMAGE_PATH, P.DESCRIPTION,  S.WEBSITE
FROM PRODUCTS P
INNER JOIN SUPPLIERS S
ON P.SUPPLIER_ID = S.SUPPLIER_ID
WHERE P.STOCK > 0;
```

7.

```
INSERT INTO DISCOUNTS
VALUES ('11D', '21-3-10', 'legendary discount', '15% OFF');
```

8.

```
INSERT INTO USERS (USER_ID, FIRST_NAME, LAST_NAME, PASSWORD, PHONE_NUM,
ADDRESS_ID)
VALUES ('NARGES__', 'narges', 'baba ahmadi', '1234567890', '091123', '1A');
```

9. Selecting all addresses in which the city either starts or ends with ‘a’

```
SELECT *
FROM ADDRESSES
WHERE CITY LIKE 'a%' OR CITY LIKE '%a';
```

10. Selecting the users with the age of older than 20 and younger than 40.

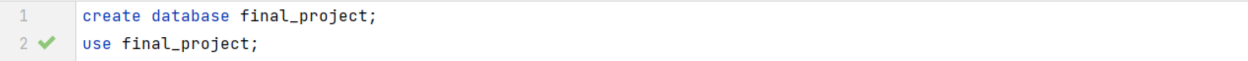
```
SELECT *
FROM USERS
WHERE AGE > 20 AND AGE < 40;
```

Part 11:

I used **mysql** for this project.

Part 12:

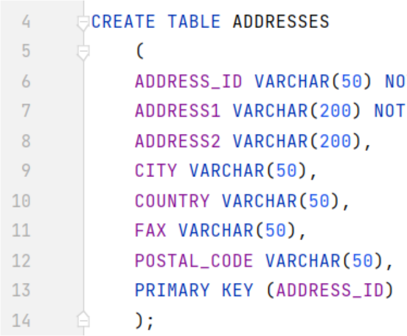
I ,first, have to **create a database** and then use it to write my queries and create tables.



```sql
create database final_project;
use final_project;
```

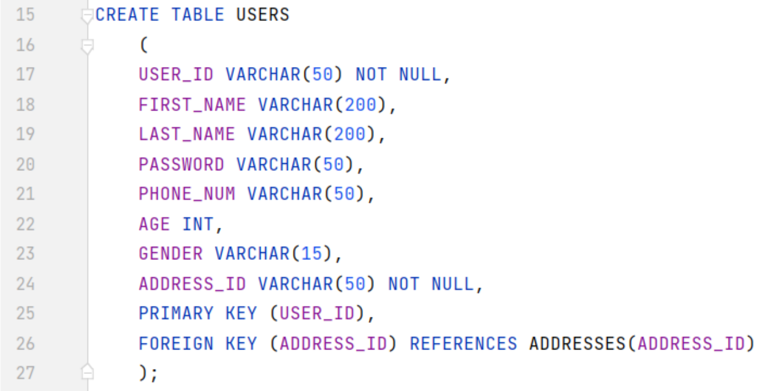
Now I begin creating the tables:

1. Creating Addresses



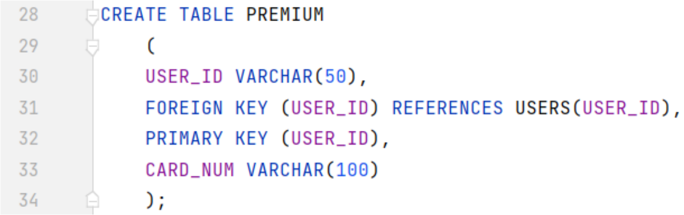
```sql
CREATE TABLE ADDRESSES
    (
    ADDRESS_ID VARCHAR(50) NOT NULL ,
    ADDRESS1 VARCHAR(200) NOT NULL,
    ADDRESS2 VARCHAR(200),
    CITY VARCHAR(50),
    COUNTRY VARCHAR(50),
    FAX VARCHAR(50),
    POSTAL_CODE VARCHAR(50),
    PRIMARY KEY (ADDRESS_ID)
    );
```

2. Creating users



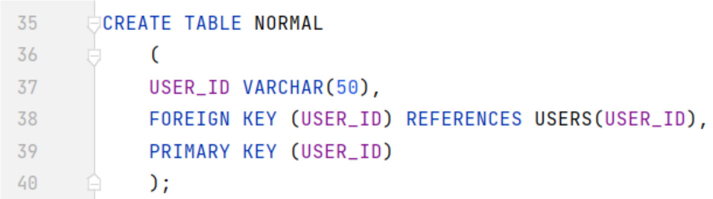
```sql
CREATE TABLE USERS
    (
    USER_ID VARCHAR(50) NOT NULL,
    FIRST_NAME VARCHAR(200),
    LAST_NAME VARCHAR(200),
    PASSWORD VARCHAR(50),
    PHONE_NUM VARCHAR(50),
    AGE INT,
    GENDER VARCHAR(15),
    ADDRESS_ID VARCHAR(50) NOT NULL,
    PRIMARY KEY (USER_ID),
    FOREIGN KEY (ADDRESS_ID) REFERENCES ADDRESSES(ADDRESS_ID)
    );
```

3. Creating premium



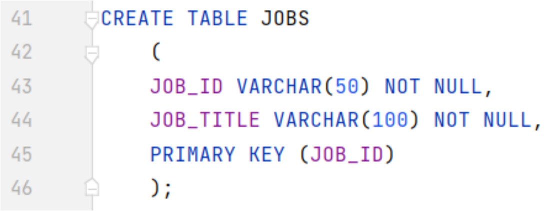
```sql
CREATE TABLE PREMIUM
    (
    USER_ID VARCHAR(50),
    FOREIGN KEY (USER_ID) REFERENCES USERS(USER_ID),
    PRIMARY KEY (USER_ID),
    CARD_NUM VARCHAR(100)
    );
```

4. Creating normal



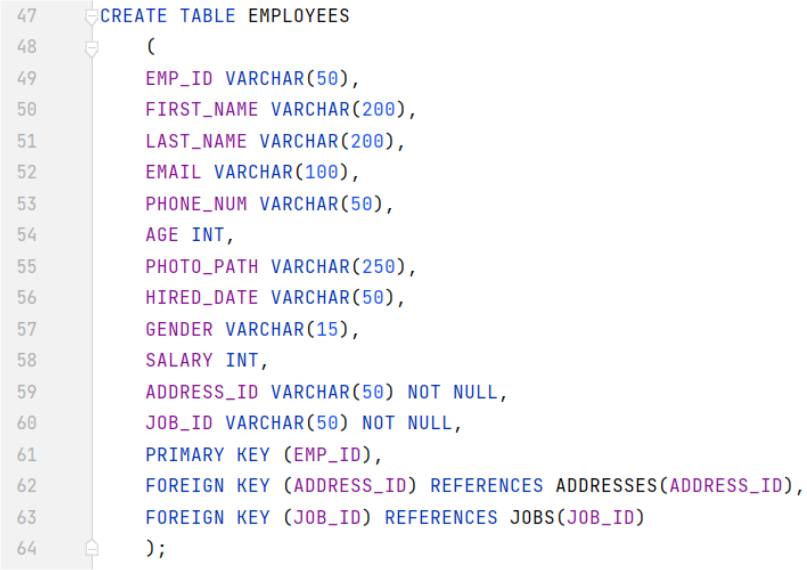
```sql
CREATE TABLE NORMAL
    (
    USER_ID VARCHAR(50),
    FOREIGN KEY (USER_ID) REFERENCES USERS(USER_ID),
    PRIMARY KEY (USER_ID)
    );
```

5. Creating jobs



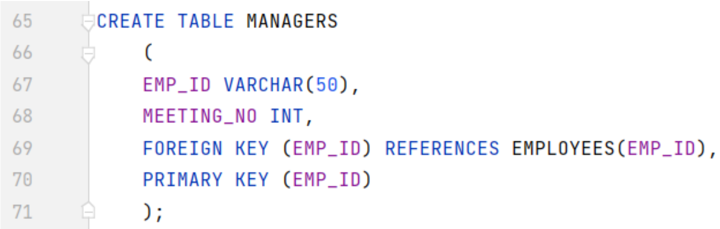
```sql
CREATE TABLE JOBS
    (
    JOB_ID VARCHAR(50) NOT NULL,
    JOB_TITLE VARCHAR(100) NOT NULL,
    PRIMARY KEY (JOB_ID)
    );
```

6. Creating employee



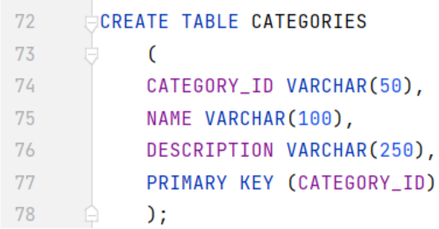
```sql
CREATE TABLE EMPLOYEES
    (
    EMP_ID VARCHAR(50),
    FIRST_NAME VARCHAR(200),
    LAST_NAME VARCHAR(200),
    EMAIL VARCHAR(100),
    PHONE_NUM VARCHAR(50),
    AGE INT,
    PHOTO_PATH VARCHAR(250),
    HIRED_DATE VARCHAR(50),
    GENDER VARCHAR(15),
    SALARY INT,
    ADDRESS_ID VARCHAR(50) NOT NULL,
    JOB_ID VARCHAR(50) NOT NULL,
    PRIMARY KEY (EMP_ID),
    FOREIGN KEY (ADDRESS_ID) REFERENCES ADDRESSES(ADDRESS_ID),
    FOREIGN KEY (JOB_ID) REFERENCES JOBS(JOB_ID)
    );
```

7. Creating managers



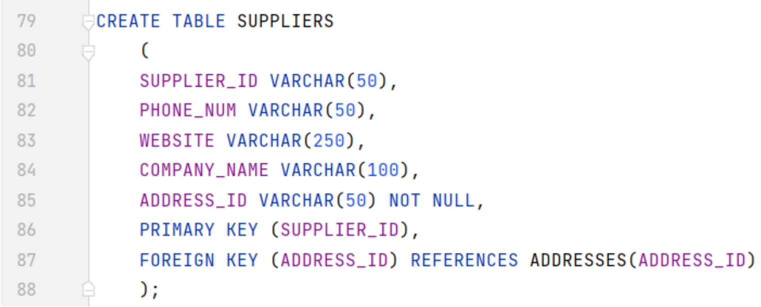
```sql
CREATE TABLE MANAGERS
    (
    EMP_ID VARCHAR(50),
    MEETING_NO INT,
    FOREIGN KEY (EMP_ID) REFERENCES EMPLOYEES(EMP_ID),
    PRIMARY KEY (EMP_ID)
    );
```

8. Creating categories



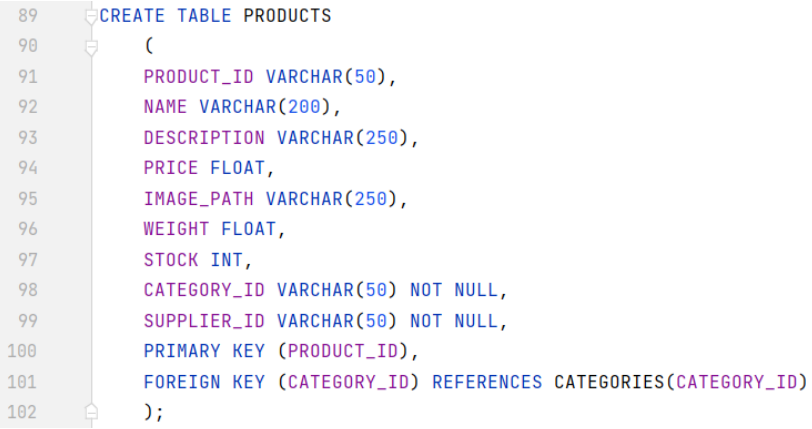
```
CREATE TABLE CATEGORIES
    (
    CATEGORY_ID VARCHAR(50),
    NAME VARCHAR(100),
    DESCRIPTION VARCHAR(250),
    PRIMARY KEY (CATEGORY_ID)
    );
```

9. Creating suppliers



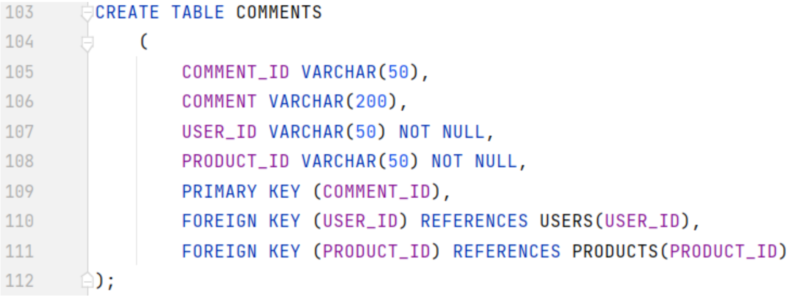
```
CREATE TABLE SUPPLIERS
    (
    SUPPLIER_ID VARCHAR(50),
    PHONE_NUM VARCHAR(50),
    WEBSITE VARCHAR(250),
    COMPANY_NAME VARCHAR(100),
    ADDRESS_ID VARCHAR(50) NOT NULL,
    PRIMARY KEY (SUPPLIER_ID),
    FOREIGN KEY (ADDRESS_ID) REFERENCES ADDRESSES(ADDRESS_ID)
    );
```

10. Creating products



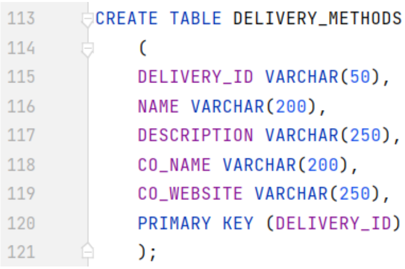
```
CREATE TABLE PRODUCTS
    (
    PRODUCT_ID VARCHAR(50),
    NAME VARCHAR(200),
    DESCRIPTION VARCHAR(250),
    PRICE FLOAT,
    IMAGE_PATH VARCHAR(250),
    WEIGHT FLOAT,
    STOCK INT,
    CATEGORY_ID VARCHAR(50) NOT NULL,
    SUPPLIER_ID VARCHAR(50) NOT NULL,
    PRIMARY KEY (PRODUCT_ID),
    FOREIGN KEY (CATEGORY_ID) REFERENCES CATEGORIES(CATEGORY_ID)
    );
```

11. Creating comments



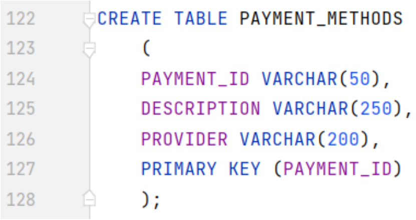
```
CREATE TABLE COMMENTS
    (
        COMMENT_ID VARCHAR(50),
        COMMENT VARCHAR(200),
        USER_ID VARCHAR(50) NOT NULL,
        PRODUCT_ID VARCHAR(50) NOT NULL,
        PRIMARY KEY (COMMENT_ID),
        FOREIGN KEY (USER_ID) REFERENCES USERS(USER_ID),
        FOREIGN KEY (PRODUCT_ID) REFERENCES PRODUCTS(PRODUCT_ID)
);
```

12. Creating delivery_methods



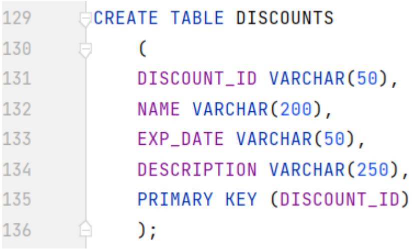
```
CREATE TABLE DELIVERY_METHODS
    (
    DELIVERY_ID VARCHAR(50),
    NAME VARCHAR(200),
    DESCRIPTION VARCHAR(250),
    CO_NAME VARCHAR(200),
    CO_WEBSITE VARCHAR(250),
    PRIMARY KEY (DELIVERY_ID)
    );
```

13. Creating payment_methods



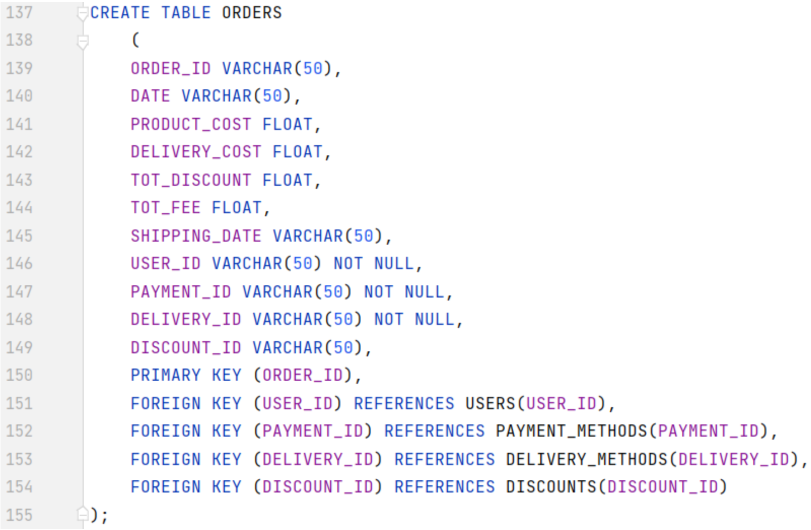
```
CREATE TABLE PAYMENT_METHODS
    (
    PAYMENT_ID VARCHAR(50),
    DESCRIPTION VARCHAR(250),
    PROVIDER VARCHAR(200),
    PRIMARY KEY (PAYMENT_ID)
    );
```

14. Creating discount



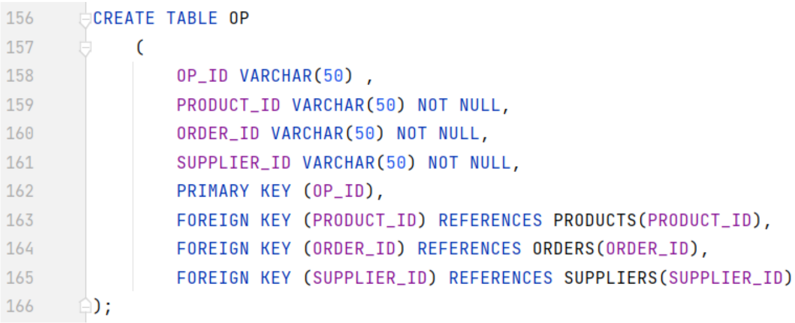
```
CREATE TABLE DISCOUNTS
    (
    DISCOUNT_ID VARCHAR(50),
    NAME VARCHAR(200),
    EXP_DATE VARCHAR(50),
    DESCRIPTION VARCHAR(250),
    PRIMARY KEY (DISCOUNT_ID)
    );
```

15. Creating orders



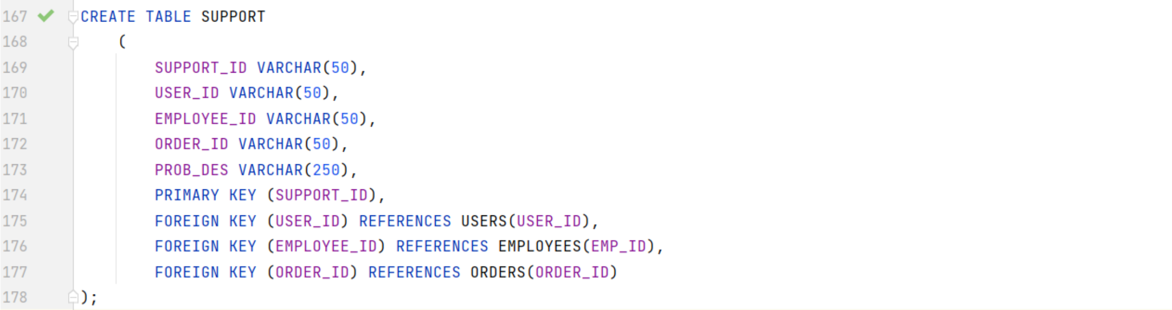
```
CREATE TABLE ORDERS
    (
    ORDER_ID VARCHAR(50),
    DATE VARCHAR(50),
    PRODUCT_COST FLOAT,
    DELIVERY_COST FLOAT,
    TOT_DISCOUNT FLOAT,
    TOT_FEE FLOAT,
    SHIPPING_DATE VARCHAR(50),
    USER_ID VARCHAR(50) NOT NULL,
    PAYMENT_ID VARCHAR(50) NOT NULL,
    DELIVERY_ID VARCHAR(50) NOT NULL,
    DISCOUNT_ID VARCHAR(50),
    PRIMARY KEY (ORDER_ID),
    FOREIGN KEY (USER_ID) REFERENCES USERS(USER_ID),
    FOREIGN KEY (PAYMENT_ID) REFERENCES PAYMENT_METHODS(PAYMENT_ID),
    FOREIGN KEY (DELIVERY_ID) REFERENCES DELIVERY_METHODS(DELIVERY_ID),
    FOREIGN KEY (DISCOUNT_ID) REFERENCES DISCOUNTS(DISCOUNT_ID)
);
```

16. Creating op



```sql
CREATE TABLE OP
    (
        OP_ID VARCHAR(50) ,
        PRODUCT_ID VARCHAR(50) NOT NULL,
        ORDER_ID VARCHAR(50) NOT NULL,
        SUPPLIER_ID VARCHAR(50) NOT NULL,
        PRIMARY KEY (OP_ID),
        FOREIGN KEY (PRODUCT_ID) REFERENCES PRODUCTS(PRODUCT_ID),
        FOREIGN KEY (ORDER_ID) REFERENCES ORDERS(ORDER_ID),
        FOREIGN KEY (SUPPLIER_ID) REFERENCES SUPPLIERS(SUPPLIER_ID)
);
```

17. Creating support

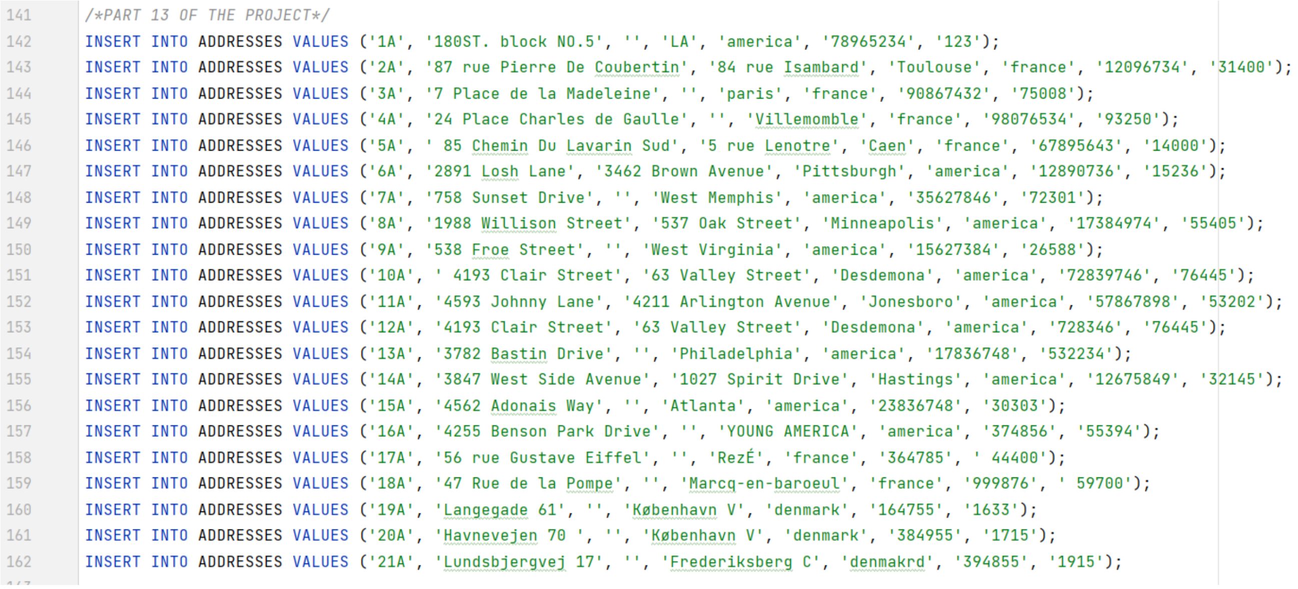


```sql
CREATE TABLE SUPPORT
    (
        SUPPORT_ID VARCHAR(50),
        USER_ID VARCHAR(50),
        EMPLOYEE_ID VARCHAR(50),
        ORDER_ID VARCHAR(50),
        PROB_DES VARCHAR(250),
        PRIMARY KEY (SUPPORT_ID),
        FOREIGN KEY (USER_ID) REFERENCES USERS(USER_ID),
        FOREIGN KEY (EMPLOYEE_ID) REFERENCES EMPLOYEES(EMP_ID),
        FOREIGN KEY (ORDER_ID) REFERENCES ORDERS(ORDER_ID)
);
```

Part 13:

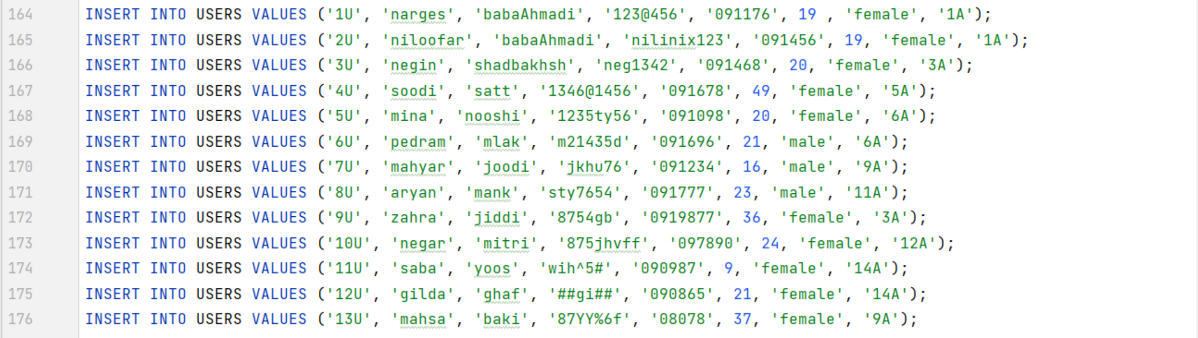
Inserting values into tables:

1. Inserting into addresses



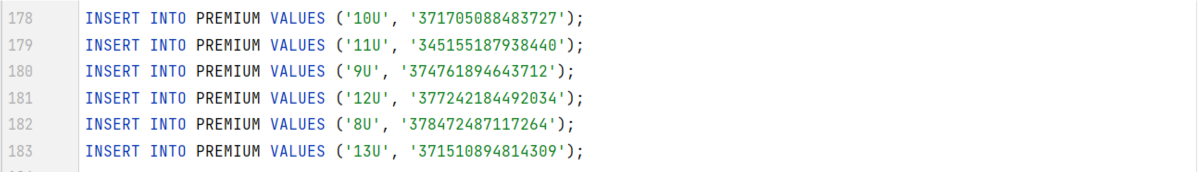
```sql
/*PART 13 OF THE PROJECT*/
INSERT INTO ADDRESSES VALUES ('1A', '180ST. block NO.5', '', 'LA', 'america', '78965234', '123');
INSERT INTO ADDRESSES VALUES ('2A', '87 rue Pierre De Coubertin', '84 rue Isambard', 'Toulouse', 'france', '12096734', '31400');
INSERT INTO ADDRESSES VALUES ('3A', '7 Place de la Madeleine', '', 'paris', 'france', '90867432', '75008');
INSERT INTO ADDRESSES VALUES ('4A', '24 Place Charles de Gaulle', '', 'Villemomble', 'france', '98076534', '93250');
INSERT INTO ADDRESSES VALUES ('5A', ' 85 Chemin Du Lavarin Sud', '5 rue Lenotre', 'Caen', 'france', '67895643', '14000');
INSERT INTO ADDRESSES VALUES ('6A', '2891 Losh Lane', '3462 Brown Avenue', 'Pittsburgh', 'america', '12890736', '15236');
INSERT INTO ADDRESSES VALUES ('7A', '758 Sunset Drive', '', 'West Memphis', 'america', '35627846', '72301');
INSERT INTO ADDRESSES VALUES ('8A', '1988 Willison Street', '537 Oak Street', 'Minneapolis', 'america', '17384974', '55405');
INSERT INTO ADDRESSES VALUES ('9A', '538 Froe Street', '', 'West Virginia', 'america', '15627384', '26588');
INSERT INTO ADDRESSES VALUES ('10A', ' 4193 Clair Street', '63 Valley Street', 'Desdemona', 'america', '72839746', '76445');
INSERT INTO ADDRESSES VALUES ('11A', '4593 Johnny Lane', '4211 Arlington Avenue', 'Jonesboro', 'america', '57867898', '53202');
INSERT INTO ADDRESSES VALUES ('12A', '4193 Clair Street', '63 Valley Street', 'Desdemona', 'america', '728346', '76445');
INSERT INTO ADDRESSES VALUES ('13A', '3782 Bastin Drive', '', 'Philadelphia', 'america', '17836748', '532234');
INSERT INTO ADDRESSES VALUES ('14A', '3847 West Side Avenue', '1027 Spirit Drive', 'Hastings', 'america', '12675849', '32145');
INSERT INTO ADDRESSES VALUES ('15A', '4562 Adonais Way', '', 'Atlanta', 'america', '23836748', '30303');
INSERT INTO ADDRESSES VALUES ('16A', '4255 Benson Park Drive', '', 'YOUNG AMERICA', 'america', '374856', '55394');
INSERT INTO ADDRESSES VALUES ('17A', '56 rue Gustave Eiffel', '', 'RezÉ', 'france', '364785', ' 44400');
INSERT INTO ADDRESSES VALUES ('18A', '47 Rue de la Pompe', '', 'Marcq-en-baroeul', 'france', '999876', ' 59700');
INSERT INTO ADDRESSES VALUES ('19A', 'Langegade 61', '', 'København V', 'denmark', '164755', '1633');
INSERT INTO ADDRESSES VALUES ('20A', 'Havnevejen 70 ', '', 'København V', 'denmark', '384955', '1715');
INSERT INTO ADDRESSES VALUES ('21A', 'Lundsbjergvej 17', '', 'Frederiksberg C', 'denmakrd', '394855', '1915');
```

2. Inserting into users



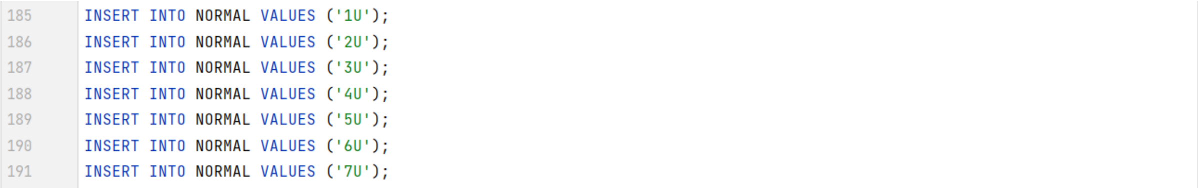
```
INSERT INTO USERS VALUES ('1U', 'narges', 'babaAhmadi', '123@456', '091176', 19 , 'female', '1A');
INSERT INTO USERS VALUES ('2U', 'niloofar', 'babaAhmadi', 'nilinix123', '091456', 19, 'female', '1A');
INSERT INTO USERS VALUES ('3U', 'negin', 'shadbakhsh', 'neg1342', '091468', 20, 'female', '3A');
INSERT INTO USERS VALUES ('4U', 'soodi', 'satt', '1346@1456', '091678', 49, 'female', '5A');
INSERT INTO USERS VALUES ('5U', 'mina', 'nooshi', '1235ty56', '091098', 20, 'female', '6A');
INSERT INTO USERS VALUES ('6U', 'pedram', 'mlak', 'm21435d', '091696', 21, 'male', '6A');
INSERT INTO USERS VALUES ('7U', 'mahyar', 'joodi', 'jkhu76', '091234', 16, 'male', '9A');
INSERT INTO USERS VALUES ('8U', 'aryan', 'mank', 'sty7654', '091777', 23, 'male', '11A');
INSERT INTO USERS VALUES ('9U', 'zahra', 'jiddi', '8754gb', '0919877', 36, 'female', '3A');
INSERT INTO USERS VALUES ('10U', 'negar', 'mitri', '875jhvff', '097890', 24, 'female', '12A');
INSERT INTO USERS VALUES ('11U', 'saba', 'yoos', 'wih^5#', '090987', 9, 'female', '14A');
INSERT INTO USERS VALUES ('12U', 'gilda', 'ghaf', '##gi##', '090865', 21, 'female', '14A');
INSERT INTO USERS VALUES ('13U', 'mahsa', 'baki', '87YY%6f', '08078', 37, 'female', '9A');
```

3.Inserting into premium



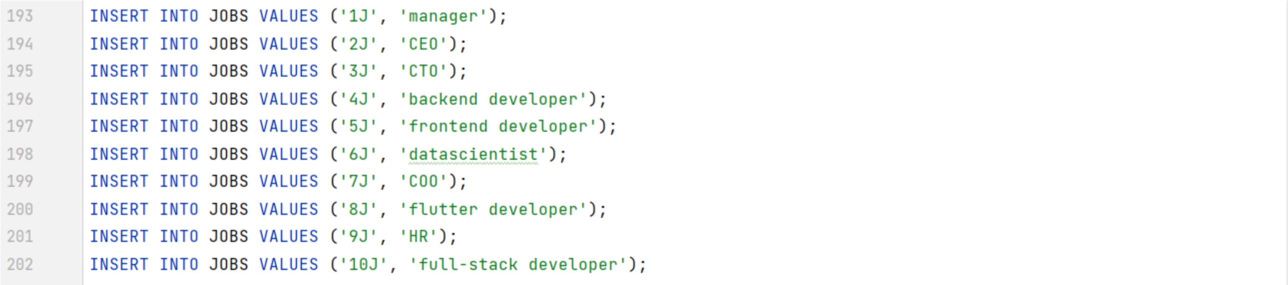
```
INSERT INTO PREMIUM VALUES ('10U', '371705088483727');
INSERT INTO PREMIUM VALUES ('11U', '345155187938440');
INSERT INTO PREMIUM VALUES ('9U', '374761894643712');
INSERT INTO PREMIUM VALUES ('12U', '377242184492034');
INSERT INTO PREMIUM VALUES ('8U', '378472487117264');
INSERT INTO PREMIUM VALUES ('13U', '371510894814309');
```

4. Inserting into normal



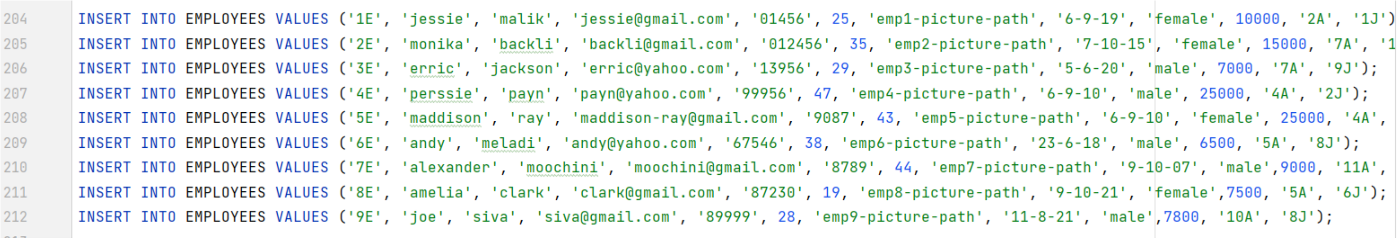
```
INSERT INTO NORMAL VALUES ('1U');
INSERT INTO NORMAL VALUES ('2U');
INSERT INTO NORMAL VALUES ('3U');
INSERT INTO NORMAL VALUES ('4U');
INSERT INTO NORMAL VALUES ('5U');
INSERT INTO NORMAL VALUES ('6U');
INSERT INTO NORMAL VALUES ('7U');
```

5. Inserting into jobs



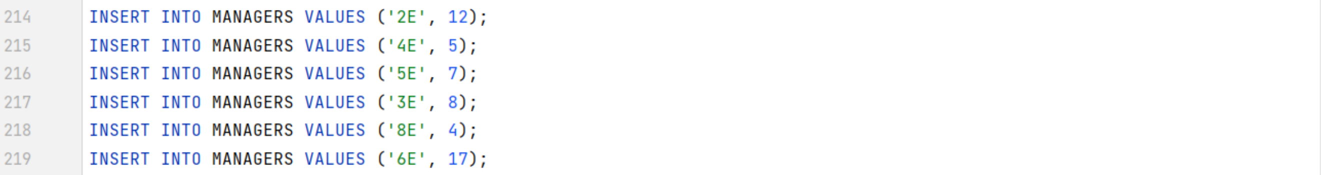
```
INSERT INTO JOBS VALUES ('1J', 'manager');
INSERT INTO JOBS VALUES ('2J', 'CEO');
INSERT INTO JOBS VALUES ('3J', 'CTO');
INSERT INTO JOBS VALUES ('4J', 'backend developer');
INSERT INTO JOBS VALUES ('5J', 'frontend developer');
INSERT INTO JOBS VALUES ('6J', 'datascientist');
INSERT INTO JOBS VALUES ('7J', 'COO');
INSERT INTO JOBS VALUES ('8J', 'flutter developer');
INSERT INTO JOBS VALUES ('9J', 'HR');
INSERT INTO JOBS VALUES ('10J', 'full-stack developer');
```

6. Inserting into employees



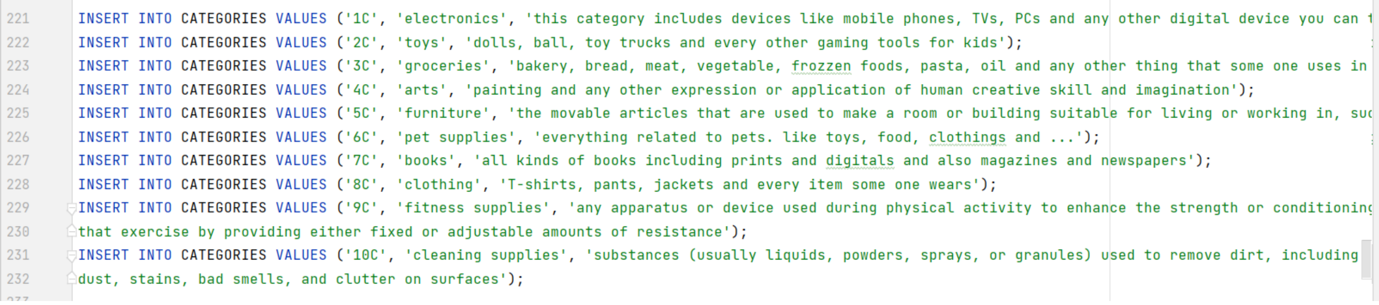
```
INSERT INTO EMPLOYEES VALUES ('1E', 'jessie', 'malik', 'jessie@gmail.com', '01456', 25, 'emp1-picture-path', '6-9-19', 'female', 10000, '2A', '1J')
INSERT INTO EMPLOYEES VALUES ('2E', 'monika', 'backli', 'backli@gmail.com', '012456', 35, 'emp2-picture-path', '7-10-15', 'female', 15000, '7A', '1
INSERT INTO EMPLOYEES VALUES ('3E', 'erric', 'jackson', 'erric@yahoo.com', '13956', 29, 'emp3-picture-path', '5-6-20', 'male', 7000, '7A', '9J');
INSERT INTO EMPLOYEES VALUES ('4E', 'perssie', 'payn', 'payn@yahoo.com', '99956', 47, 'emp4-picture-path', '6-9-10', 'male', 25000, '4A', '2J');
INSERT INTO EMPLOYEES VALUES ('5E', 'maddison', 'ray', 'maddison-ray@gmail.com', '9087', 43, 'emp5-picture-path', '6-9-10', 'female', 25000, '4A',
INSERT INTO EMPLOYEES VALUES ('6E', 'andy', 'meladi', 'andy@yahoo.com', '67546', 38, 'emp6-picture-path', '23-6-18', 'male', 6500, '5A', '8J');
INSERT INTO EMPLOYEES VALUES ('7E', 'alexander', 'moochini', 'moochini@gmail.com', '8789', 44, 'emp7-picture-path', '9-10-07', 'male',9000, '11A',
INSERT INTO EMPLOYEES VALUES ('8E', 'amelia', 'clark', 'clark@gmail.com', '87230', 19, 'emp8-picture-path', '9-10-21', 'female',7500, '5A', '6J');
INSERT INTO EMPLOYEES VALUES ('9E', 'joe', 'siva', 'siva@gmail.com', '89999', 28, 'emp9-picture-path', '11-8-21', 'male',7800, '10A', '8J');
```

7. Inserting into managers



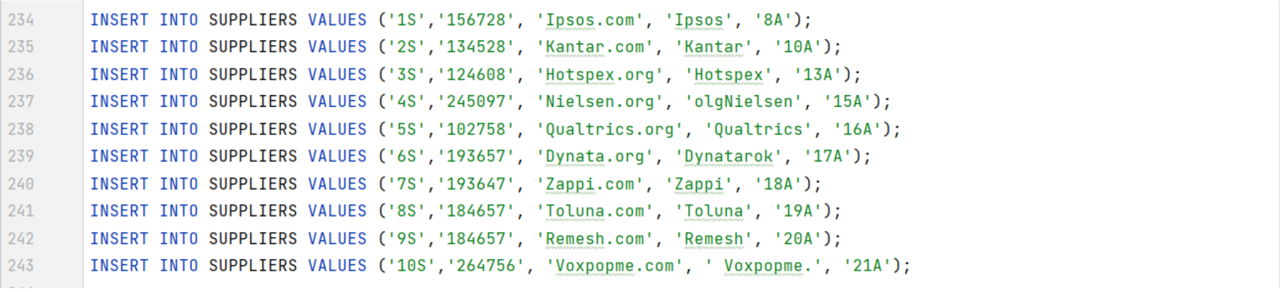
```
INSERT INTO MANAGERS VALUES ('2E', 12);
INSERT INTO MANAGERS VALUES ('4E', 5);
INSERT INTO MANAGERS VALUES ('5E', 7);
INSERT INTO MANAGERS VALUES ('3E', 8);
INSERT INTO MANAGERS VALUES ('8E', 4);
INSERT INTO MANAGERS VALUES ('6E', 17);
```

8. Inserting into categories



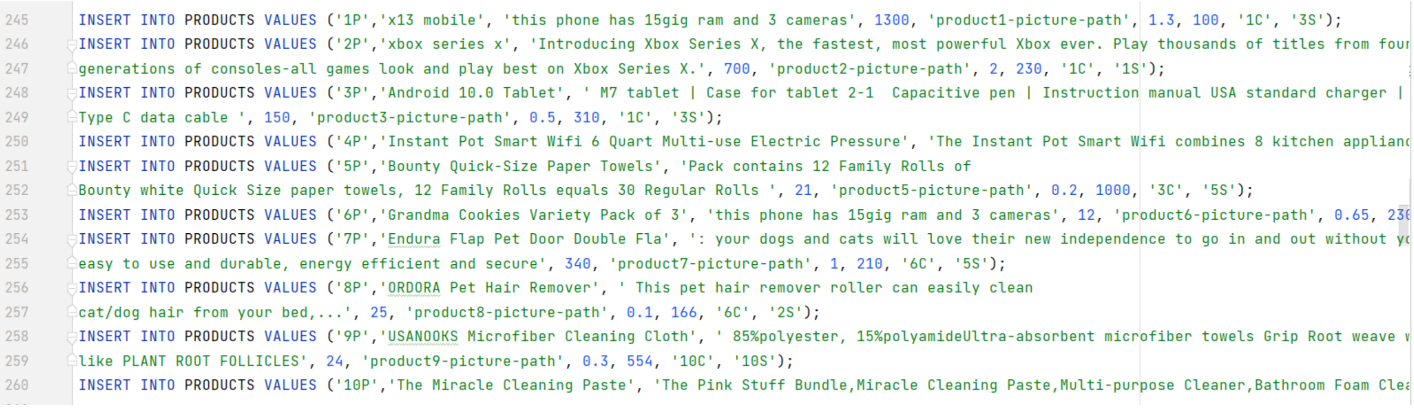
```
INSERT INTO CATEGORIES VALUES ('1C', 'electronics', 'this category includes devices like mobile phones, TVs, PCs and any other digital device you can t
INSERT INTO CATEGORIES VALUES ('2C', 'toys', 'dolls, ball, toy trucks and every other gaming tools for kids');
INSERT INTO CATEGORIES VALUES ('3C', 'groceries', 'bakery, bread, meat, vegetable, frozzen foods, pasta, oil and any other thing that some one uses in
INSERT INTO CATEGORIES VALUES ('4C', 'arts', 'painting and any other expression or application of human creative skill and imagination');
INSERT INTO CATEGORIES VALUES ('5C', 'furniture', 'the movable articles that are used to make a room or building suitable for living or working in, suc
INSERT INTO CATEGORIES VALUES ('6C', 'pet supplies', 'everything related to pets. like toys, food, clothings and ...');
INSERT INTO CATEGORIES VALUES ('7C', 'books', 'all kinds of books including prints and digitals and also magazines and newspapers');
INSERT INTO CATEGORIES VALUES ('8C', 'clothing', 'T-shirts, pants, jackets and every item some one wears');
INSERT INTO CATEGORIES VALUES ('9C', 'fitness supplies', 'any apparatus or device used during physical activity to enhance the strength or conditioning
that exercise by providing either fixed or adjustable amounts of resistance');
INSERT INTO CATEGORIES VALUES ('10C', 'cleaning supplies', 'substances (usually liquids, powders, sprays, or granules) used to remove dirt, including
dust, stains, bad smells, and clutter on surfaces');
```

9. Inserting into suppliers



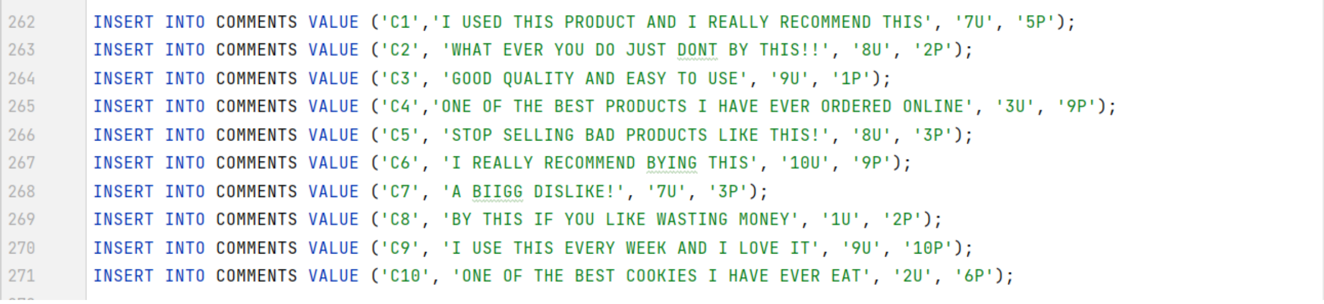
```
INSERT INTO SUPPLIERS VALUES ('1S','156728', 'Ipsos.com', 'Ipsos', '8A');
INSERT INTO SUPPLIERS VALUES ('2S','134528', 'Kantar.com', 'Kantar', '10A');
INSERT INTO SUPPLIERS VALUES ('3S','124608', 'Hotspex.org', 'Hotspex', '13A');
INSERT INTO SUPPLIERS VALUES ('4S','245097', 'Nielsen.org', 'olgNielsen', '15A');
INSERT INTO SUPPLIERS VALUES ('5S','102758', 'Qualtrics.org', 'Qualtrics', '16A');
INSERT INTO SUPPLIERS VALUES ('6S','193657', 'Dynata.org', 'Dynatarok', '17A');
INSERT INTO SUPPLIERS VALUES ('7S','193647', 'Zappi.com', 'Zappi', '18A');
INSERT INTO SUPPLIERS VALUES ('8S','184657', 'Toluna.com', 'Toluna', '19A');
INSERT INTO SUPPLIERS VALUES ('9S','184657', 'Remesh.com', 'Remesh', '20A');
INSERT INTO SUPPLIERS VALUES ('10S','264756', 'Voxpopme.com', ' Voxpopme.', '21A');
```

10. Inserting into products



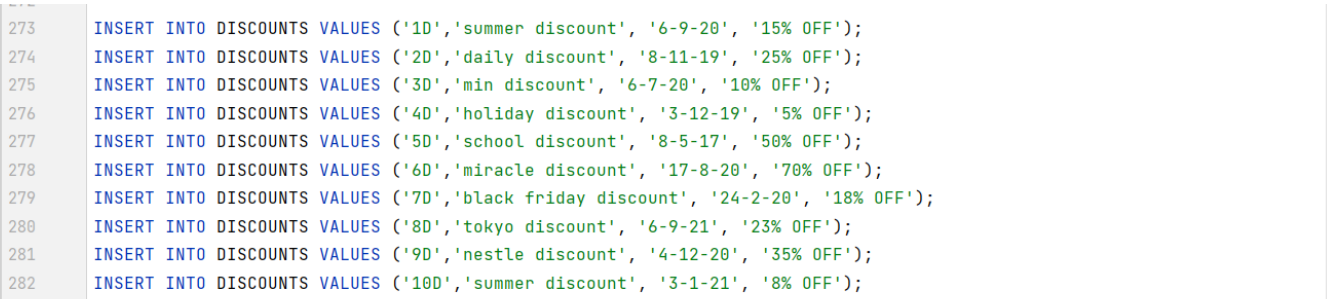
```
INSERT INTO PRODUCTS VALUES ('1P','x13 mobile', 'this phone has 15gig ram and 3 cameras', 1300, 'product1-picture-path', 1.3, 100, '1C', '3S');
INSERT INTO PRODUCTS VALUES ('2P','xbox series x', 'Introducing Xbox Series X, the fastest, most powerful Xbox ever. Play thousands of titles from four
generations of consoles-all games look and play best on Xbox Series X.', 700, 'product2-picture-path', 2, 230, '1C', '1S');
INSERT INTO PRODUCTS VALUES ('3P','Android 10.0 Tablet', ' M7 tablet | Case for tablet 2-1  Capacitive pen | Instruction manual USA standard charger |
Type C data cable ', 150, 'product3-picture-path', 0.5, 310, '1C', '3S');
INSERT INTO PRODUCTS VALUES ('4P','Instant Pot Smart Wifi 6 Quart Multi-use Electric Pressure', 'The Instant Pot Smart Wifi combines 8 kitchen applianc
INSERT INTO PRODUCTS VALUES ('5P','Bounty Quick-Size Paper Towels', 'Pack contains 12 Family Rolls of
Bounty white Quick Size paper towels, 12 Family Rolls equals 30 Regular Rolls ', 21, 'product5-picture-path', 0.2, 1000, '3C', '5S');
INSERT INTO PRODUCTS VALUES ('6P','Grandma Cookies Variety Pack of 3', 'this phone has 15gig ram and 3 cameras', 12, 'product6-picture-path', 0.65, 23
INSERT INTO PRODUCTS VALUES ('7P','Endura Flap Pet Door Double Fla', ': your dogs and cats will love their new independence to go in and out without y
easy to use and durable, energy efficient and secure', 340, 'product7-picture-path', 1, 210, '6C', '5S');
INSERT INTO PRODUCTS VALUES ('8P','ORDORA Pet Hair Remover', ' This pet hair remover roller can easily clean
cat/dog hair from your bed,...', 25, 'product8-picture-path', 0.1, 166, '6C', '2S');
INSERT INTO PRODUCTS VALUES ('9P','USANOOKS Microfiber Cleaning Cloth', ' 85%polyester, 15%polyamideUltra-absorbent microfiber towels Grip Root weave w
like PLANT ROOT FOLLICLES', 24, 'product9-picture-path', 0.3, 554, '10C', '10S');
INSERT INTO PRODUCTS VALUES ('10P','The Miracle Cleaning Paste', 'The Pink Stuff Bundle,Miracle Cleaning Paste,Multi-purpose Cleaner,Bathroom Foam Clea
```

11. Inserting into comments



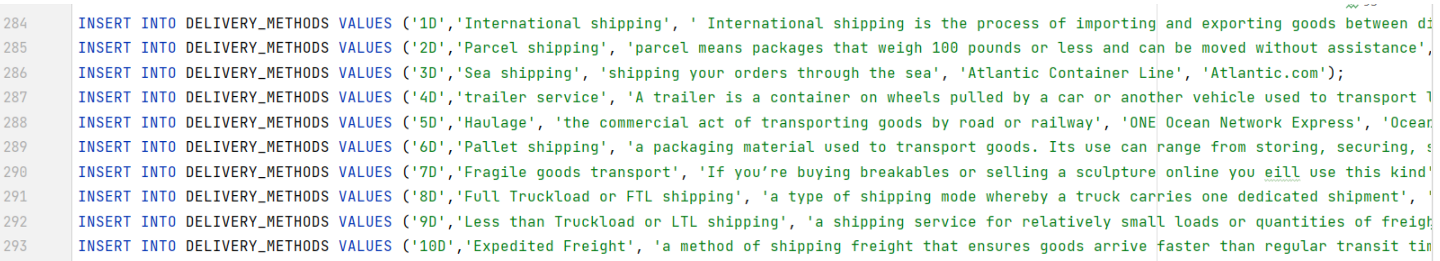
```sql
INSERT INTO COMMENTS VALUE ('C1','I USED THIS PRODUCT AND I REALLY RECOMMEND THIS', '7U', '5P');
INSERT INTO COMMENTS VALUE ('C2', 'WHAT EVER YOU DO JUST DONT BY THIS!!', '8U', '2P');
INSERT INTO COMMENTS VALUE ('C3', 'GOOD QUALITY AND EASY TO USE', '9U', '1P');
INSERT INTO COMMENTS VALUE ('C4','ONE OF THE BEST PRODUCTS I HAVE EVER ORDERED ONLINE', '3U', '9P');
INSERT INTO COMMENTS VALUE ('C5', 'STOP SELLING BAD PRODUCTS LIKE THIS!', '8U', '3P');
INSERT INTO COMMENTS VALUE ('C6', 'I REALLY RECOMMEND BYING THIS', '10U', '9P');
INSERT INTO COMMENTS VALUE ('C7', 'A BIIGG DISLIKE!', '7U', '3P');
INSERT INTO COMMENTS VALUE ('C8', 'BY THIS IF YOU LIKE WASTING MONEY', '1U', '2P');
INSERT INTO COMMENTS VALUE ('C9', 'I USE THIS EVERY WEEK AND I LOVE IT', '9U', '10P');
INSERT INTO COMMENTS VALUE ('C10', 'ONE OF THE BEST COOKIES I HAVE EVER EAT', '2U', '6P');
```

12. Inserting into discounts



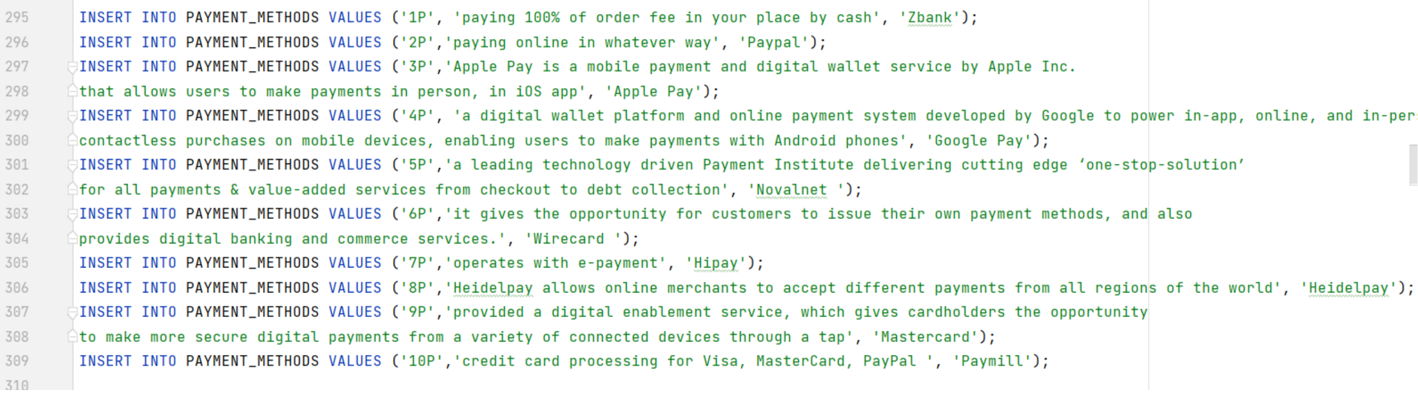
```sql
INSERT INTO DISCOUNTS VALUES ('1D','summer discount', '6-9-20', '15% OFF');
INSERT INTO DISCOUNTS VALUES ('2D','daily discount', '8-11-19', '25% OFF');
INSERT INTO DISCOUNTS VALUES ('3D','min discount', '6-7-20', '10% OFF');
INSERT INTO DISCOUNTS VALUES ('4D','holiday discount', '3-12-19', '5% OFF');
INSERT INTO DISCOUNTS VALUES ('5D','school discount', '8-5-17', '50% OFF');
INSERT INTO DISCOUNTS VALUES ('6D','miracle discount', '17-8-20', '70% OFF');
INSERT INTO DISCOUNTS VALUES ('7D','black friday discount', '24-2-20', '18% OFF');
INSERT INTO DISCOUNTS VALUES ('8D','tokyo discount', '6-9-21', '23% OFF');
INSERT INTO DISCOUNTS VALUES ('9D','nestle discount', '4-12-20', '35% OFF');
INSERT INTO DISCOUNTS VALUES ('10D','summer discount', '3-1-21', '8% OFF');
```

13. Inserting into delivery_methods



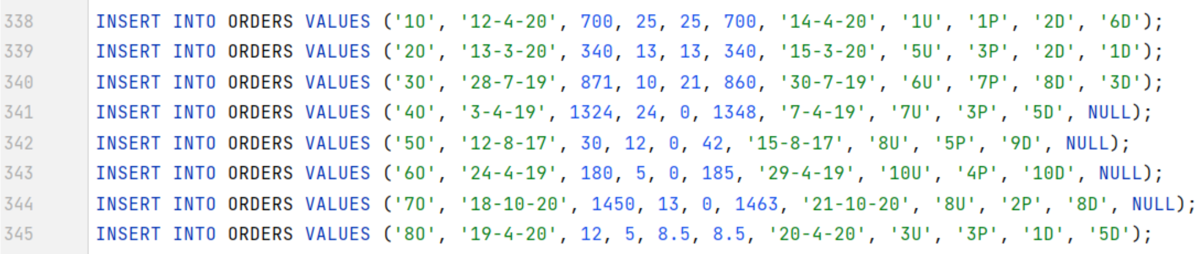
```sql
INSERT INTO DELIVERY_METHODS VALUES ('1D','International shipping', ' International shipping is the process of importing and exporting goods between di
INSERT INTO DELIVERY_METHODS VALUES ('2D','Parcel shipping', 'parcel means packages that weigh 100 pounds or less and can be moved without assistance',
INSERT INTO DELIVERY_METHODS VALUES ('3D','Sea shipping', 'shipping your orders through the sea', 'Atlantic Container Line', 'Atlantic.com');
INSERT INTO DELIVERY_METHODS VALUES ('4D','trailer service', 'A trailer is a container on wheels pulled by a car or another vehicle used to transport l
INSERT INTO DELIVERY_METHODS VALUES ('5D','Haulage', 'the commercial act of transporting goods by road or railway', 'ONE Ocean Network Express', 'Ocean
INSERT INTO DELIVERY_METHODS VALUES ('6D','Pallet shipping', 'a packaging material used to transport goods. Its use can range from storing, securing, s
INSERT INTO DELIVERY_METHODS VALUES ('7D','Fragile goods transport', 'If you’re buying breakables or selling a sculpture online you eill use this kind'
INSERT INTO DELIVERY_METHODS VALUES ('8D','Full Truckload or FTL shipping', 'a type of shipping mode whereby a truck carries one dedicated shipment', '
INSERT INTO DELIVERY_METHODS VALUES ('9D','Less than Truckload or LTL shipping', 'a shipping service for relatively small loads or quantities of freigh
INSERT INTO DELIVERY_METHODS VALUES ('10D','Expedited Freight', 'a method of shipping freight that ensures goods arrive faster than regular transit tim
```

14. Inserting into payment_methods



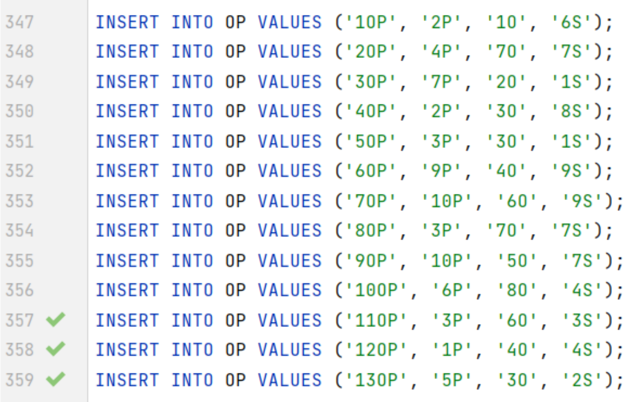
```sql
INSERT INTO PAYMENT_METHODS VALUES ('1P', 'paying 100% of order fee in your place by cash', 'Zbank');
INSERT INTO PAYMENT_METHODS VALUES ('2P','paying online in whatever way', 'Paypal');
INSERT INTO PAYMENT_METHODS VALUES ('3P','Apple Pay is a mobile payment and digital wallet service by Apple Inc.
that allows users to make payments in person, in iOS app', 'Apple Pay');
INSERT INTO PAYMENT_METHODS VALUES ('4P', 'a digital wallet platform and online payment system developed by Google to power in-app, online, and in-per
contactless purchases on mobile devices, enabling users to make payments with Android phones', 'Google Pay');
INSERT INTO PAYMENT_METHODS VALUES ('5P','a leading technology driven Payment Institute delivering cutting edge ‘one-stop-solution’
for all payments & value-added services from checkout to debt collection', 'Novalnet ');
INSERT INTO PAYMENT_METHODS VALUES ('6P','it gives the opportunity for customers to issue their own payment methods, and also
provides digital banking and commerce services.', 'Wirecard ');
INSERT INTO PAYMENT_METHODS VALUES ('7P','operates with e-payment', 'Hipay');
INSERT INTO PAYMENT_METHODS VALUES ('8P','Heidelpay allows online merchants to accept different payments from all regions of the world', 'Heidelpay');
INSERT INTO PAYMENT_METHODS VALUES ('9P','provided a digital enablement service, which gives cardholders the opportunity
to make more secure digital payments from a variety of connected devices through a tap', 'Mastercard');
INSERT INTO PAYMENT_METHODS VALUES ('10P','credit card processing for Visa, MasterCard, PayPal ', 'Paymill');
```

15. Inserting into orders



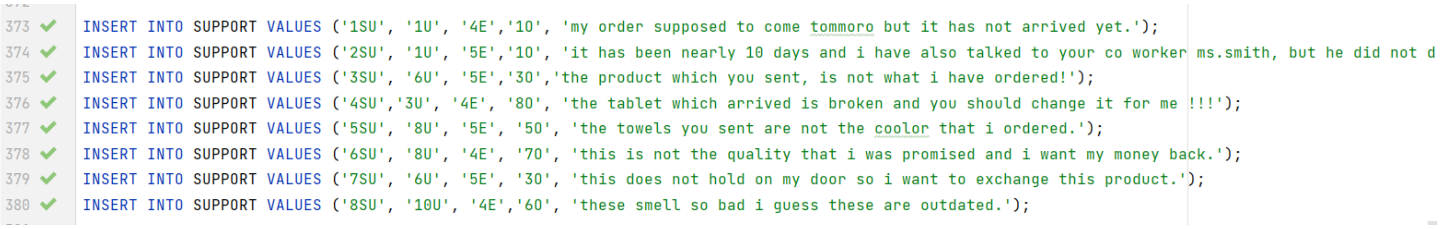
```
INSERT INTO ORDERS VALUES ('1O', '12-4-20', 700, 25, 25, 700, '14-4-20', '1U', '1P', '2D', '6D');
INSERT INTO ORDERS VALUES ('2O', '13-3-20', 340, 13, 13, 340, '15-3-20', '5U', '3P', '2D', '1D');
INSERT INTO ORDERS VALUES ('3O', '28-7-19', 871, 10, 21, 860, '30-7-19', '6U', '7P', '8D', '3D');
INSERT INTO ORDERS VALUES ('4O', '3-4-19', 1324, 24, 0, 1348, '7-4-19', '7U', '3P', '5D', NULL);
INSERT INTO ORDERS VALUES ('5O', '12-8-17', 30, 12, 0, 42, '15-8-17', '8U', '5P', '9D', NULL);
INSERT INTO ORDERS VALUES ('6O', '24-4-19', 180, 5, 0, 185, '29-4-19', '10U', '4P', '10D', NULL);
INSERT INTO ORDERS VALUES ('7O', '18-10-20', 1450, 13, 0, 1463, '21-10-20', '8U', '2P', '8D', NULL);
INSERT INTO ORDERS VALUES ('8O', '19-4-20', 12, 5, 8.5, 8.5, '20-4-20', '3U', '3P', '1D', '5D');
```

16. Inserting into op



```
INSERT INTO OP VALUES ('1OP', '2P', '1O', '6S');
INSERT INTO OP VALUES ('2OP', '4P', '7O', '7S');
INSERT INTO OP VALUES ('3OP', '7P', '2O', '1S');
INSERT INTO OP VALUES ('4OP', '2P', '3O', '8S');
INSERT INTO OP VALUES ('5OP', '3P', '3O', '1S');
INSERT INTO OP VALUES ('6OP', '9P', '4O', '9S');
INSERT INTO OP VALUES ('7OP', '10P', '6O', '9S');
INSERT INTO OP VALUES ('8OP', '3P', '7O', '7S');
INSERT INTO OP VALUES ('9OP', '10P', '5O', '7S');
INSERT INTO OP VALUES ('10OP', '6P', '8O', '4S');
INSERT INTO OP VALUES ('11OP', '3P', '6O', '3S');
INSERT INTO OP VALUES ('12OP', '1P', '4O', '4S');
INSERT INTO OP VALUES ('13OP', '5P', '3O', '2S');
```

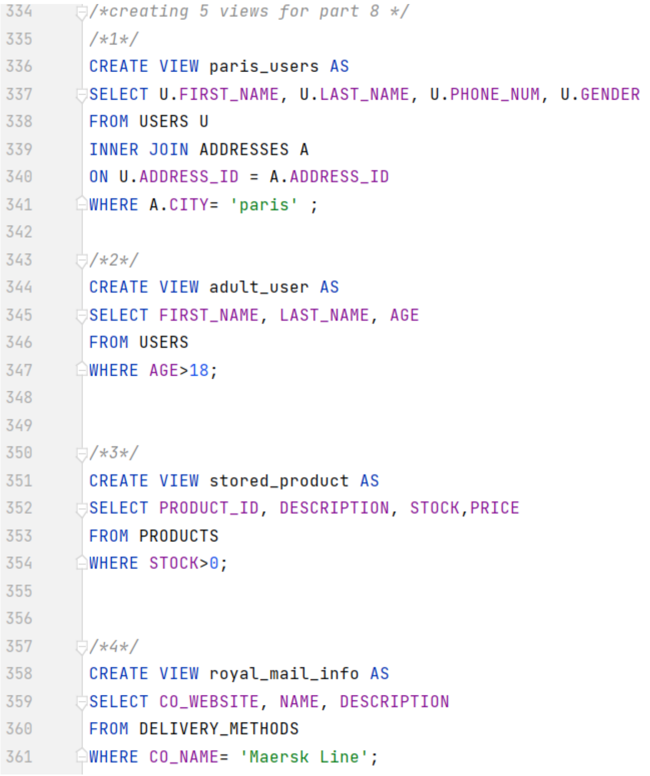
17. Inserting into supports



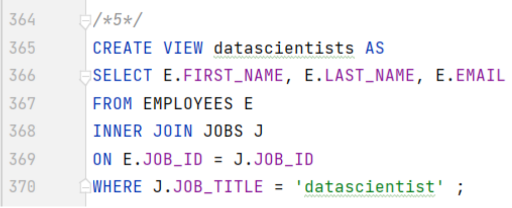
```
INSERT INTO SUPPORT VALUES ('1SU', '1U', '4E','1O', 'my order supposed to come tommoro but it has not arrived yet.');
INSERT INTO SUPPORT VALUES ('2SU', '1U', '5E','1O', 'it has been nearly 10 days and i have also talked to your co worker ms.smith, but he did not d
INSERT INTO SUPPORT VALUES ('3SU', '6U', '5E','3O','the product which you sent, is not what i have ordered!');
INSERT INTO SUPPORT VALUES ('4SU','3U', '4E', '8O', 'the tablet which arrived is broken and you should change it for me !!!');
INSERT INTO SUPPORT VALUES ('5SU', '8U', '5E', '5O', 'the towels you sent are not the coolor that i ordered.');
INSERT INTO SUPPORT VALUES ('6SU', '8U', '4E', '7O', 'this is not the quality that i was promised and i want my money back.');
INSERT INTO SUPPORT VALUES ('7SU', '6U', '5E', '3O', 'this does not hold on my door so i want to exchange this product.');
INSERT INTO SUPPORT VALUES ('8SU', '10U', '4E','6O', 'these smell so bad i guess these are outdated.');
```

Part 14 and Part 15:

Creating the views from part 8:



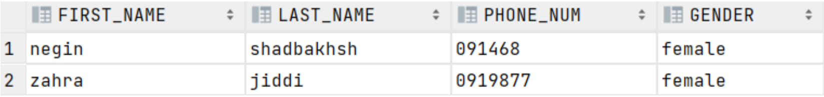
```
/*creating 5 views for part 8 */
/*1*/
CREATE VIEW paris_users AS
SELECT U.FIRST_NAME, U.LAST_NAME, U.PHONE_NUM, U.GENDER
FROM USERS U
INNER JOIN ADDRESSES A
ON U.ADDRESS_ID = A.ADDRESS_ID
WHERE A.CITY= 'paris' ;

/*2*/
CREATE VIEW adult_user AS
SELECT FIRST_NAME, LAST_NAME, AGE
FROM USERS
WHERE AGE>18;


/*3*/
CREATE VIEW stored_product AS
SELECT PRODUCT_ID, DESCRIPTION, STOCK,PRICE
FROM PRODUCTS
WHERE STOCK>0;


/*4*/
CREATE VIEW royal_mail_info AS
SELECT CO_WEBSITE, NAME, DESCRIPTION
FROM DELIVERY_METHODS
WHERE CO_NAME= 'Maersk Line';
```



```
/*5*/
CREATE VIEW datascientists AS
SELECT E.FIRST_NAME, E.LAST_NAME, E.EMAIL
FROM EMPLOYEES E
INNER JOIN JOBS J
ON E.JOB_ID = J.JOB_ID
WHERE J.JOB_TITLE = 'datascientist' ;
```

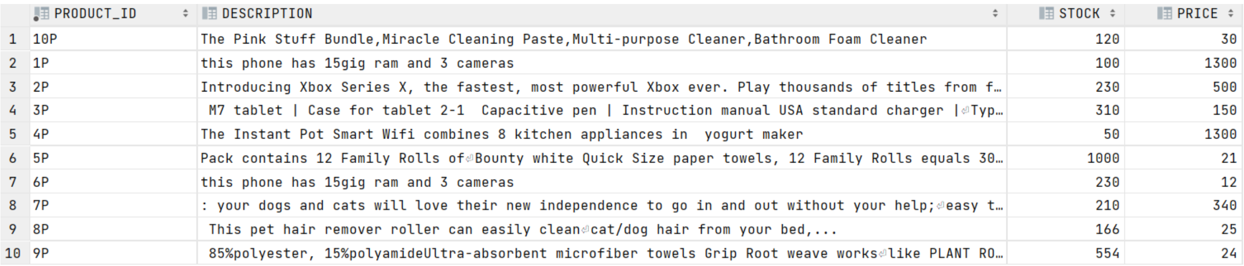
The outputs of views:

1. paris_users:



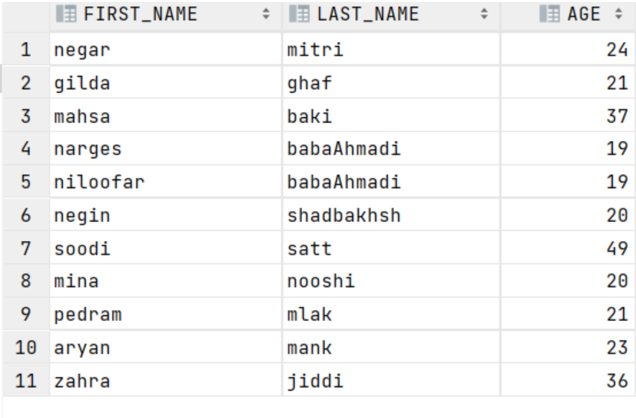
	FIRST_NAME	LAST_NAME	PHONE_NUM	GENDER
1	negin	shadbakhsh	091468	female
2	zahra	jiddi	0919877	female

2. stored_products:



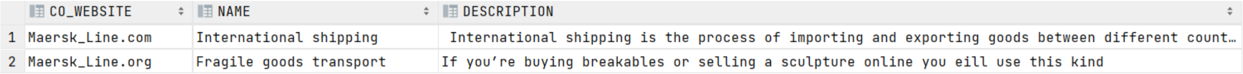
	PRODUCT_ID	DESCRIPTION	STOCK	PRICE
1	10P	The Pink Stuff Bundle,Miracle Cleaning Paste,Multi-purpose Cleaner,Bathroom Foam Cleaner	120	30
2	1P	this phone has 15gig ram and 3 cameras	100	1300
3	2P	Introducing Xbox Series X, the fastest, most powerful Xbox ever. Play thousands of titles from f…	230	500
4	3P	M7 tablet | Case for tablet 2-1 Capacitive pen | Instruction manual USA standard charger |⏎Typ…	310	150
5	4P	The Instant Pot Smart Wifi combines 8 kitchen appliances in yogurt maker	50	1300
6	5P	Pack contains 12 Family Rolls of⏎Bounty white Quick Size paper towels, 12 Family Rolls equals 30…	1000	21
7	6P	this phone has 15gig ram and 3 cameras	230	12
8	7P	: your dogs and cats will love their new independence to go in and out without your help;⏎easy t…	210	340
9	8P	This pet hair remover roller can easily clean⏎cat/dog hair from your bed,...	166	25
10	9P	85%polyester, 15%polyamideUltra-absorbent microfiber towels Grip Root weave works⏎like PLANT RO…	554	24

3. adault_user:



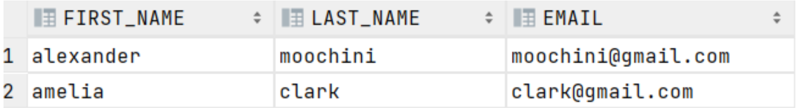
	FIRST_NAME	LAST_NAME	AGE
1	negar	mitri	24
2	gilda	ghaf	21
3	mahsa	baki	37
4	narges	babaAhmadi	19
5	niloofar	babaAhmadi	19
6	negin	shadbakhsh	20
7	soodi	satt	49
8	mina	nooshi	20
9	pedram	mlak	21
10	aryan	mank	23
11	zahra	jiddi	36

4. royal_mail_info:



	CO_WEBSITE	NAME	DESCRIPTION
1	Maersk_Line.com	International shipping	International shipping is the process of importing and exporting goods between different count…
2	Maersk_Line.org	Fragile goods transport	If you're buying breakables or selling a sculpture online you eill use this kind

5. datascientists:

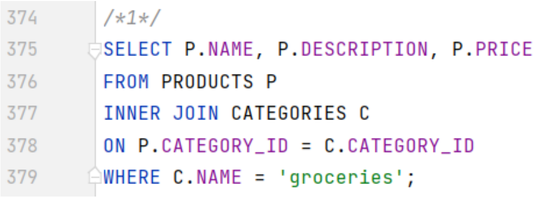


	FIRST_NAME	LAST_NAME	EMAIL
1	alexander	moochini	moochini@gmail.com
2	amelia	clark	clark@gmail.com

Selecting and inserting from part 9:

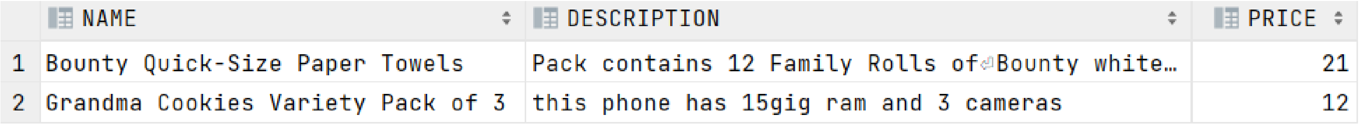
1.

Code:



```
/*1*/
SELECT P.NAME, P.DESCRIPTION, P.PRICE
FROM PRODUCTS P
INNER JOIN CATEGORIES C
ON P.CATEGORY_ID = C.CATEGORY_ID
WHERE C.NAME = 'groceries';
```

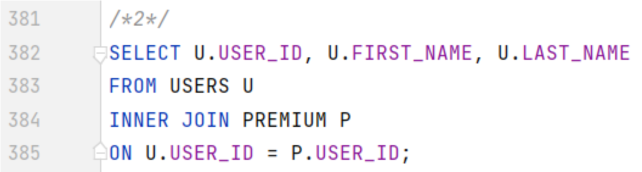
Output:



	NAME	DESCRIPTION	PRICE
1	Bounty Quick-Size Paper Towels	Pack contains 12 Family Rolls of⏎Bounty white…	21
2	Grandma Cookies Variety Pack of 3	this phone has 15gig ram and 3 cameras	12

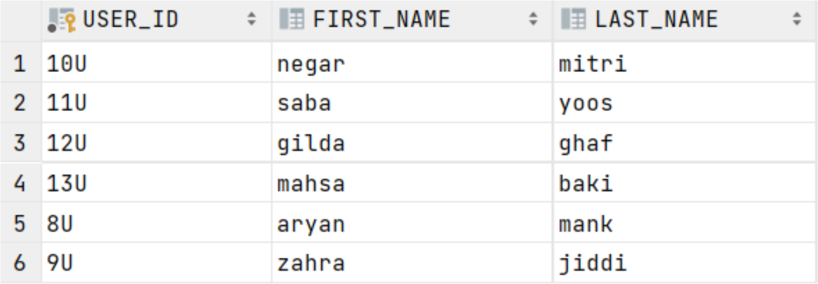
2.

Code:



```
/*2*/
SELECT U.USER_ID, U.FIRST_NAME, U.LAST_NAME
FROM USERS U
INNER JOIN PREMIUM P
ON U.USER_ID = P.USER_ID;
```

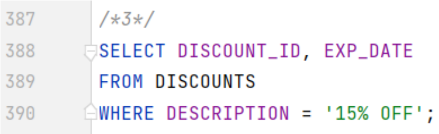
Output:



	USER_ID	FIRST_NAME	LAST_NAME
1	10U	negar	mitri
2	11U	saba	yoos
3	12U	gilda	ghaf
4	13U	mahsa	baki
5	8U	aryan	mank
6	9U	zahra	jiddi

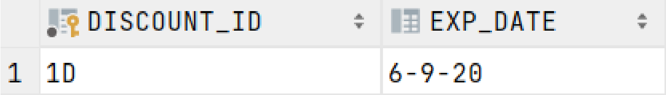
3.

Code:



```
/*3*/
SELECT DISCOUNT_ID, EXP_DATE
FROM DISCOUNTS
WHERE DESCRIPTION = '15% OFF';
```

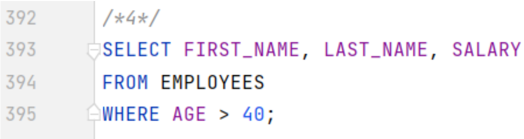
Output:



	DISCOUNT_ID	EXP_DATE
1	1D	6-9-20

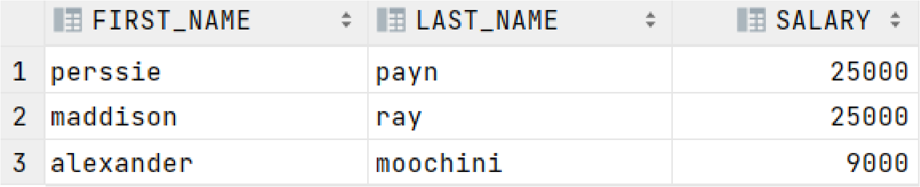
4.

Code:



```
/*4*/
SELECT FIRST_NAME, LAST_NAME, SALARY
FROM EMPLOYEES
WHERE AGE > 40;
```

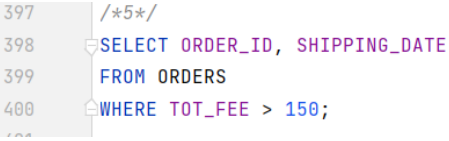
Output:



	FIRST_NAME	LAST_NAME	SALARY
1	perssie	payn	25000
2	maddison	ray	25000
3	alexander	moochini	9000

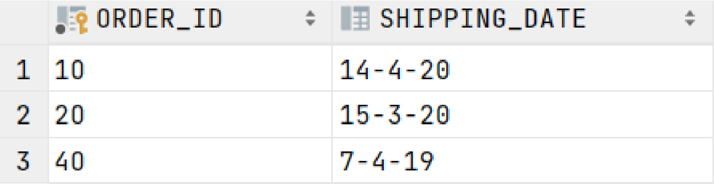
5.

Code:



```
/*5*/
SELECT ORDER_ID, SHIPPING_DATE
FROM ORDERS
WHERE TOT_FEE > 150;
```

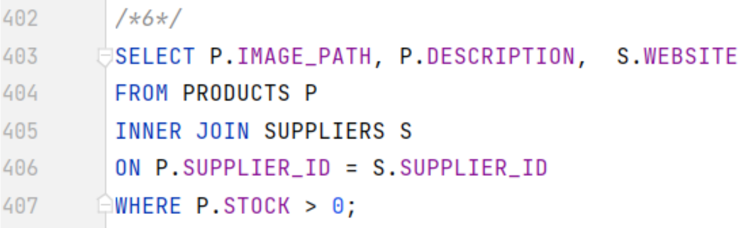
Output:



	ORDER_ID	SHIPPING_DATE
1	10	14-4-20
2	20	15-3-20
3	40	7-4-19

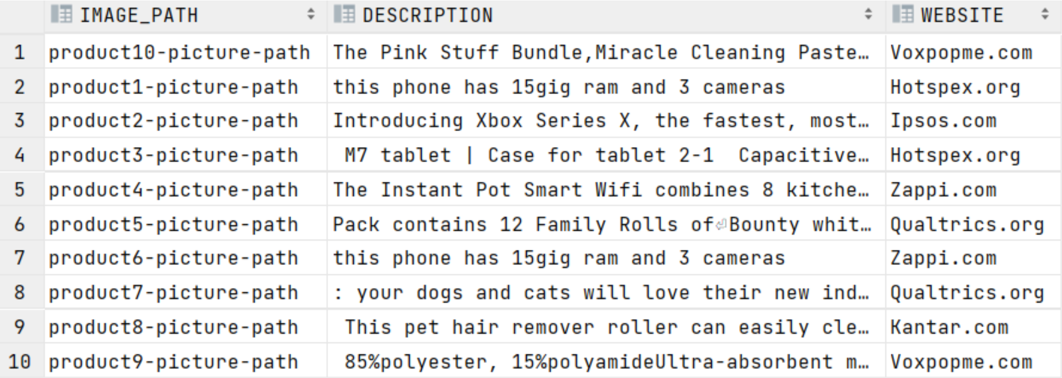
6.

Code:



```
/*6*/
SELECT P.IMAGE_PATH, P.DESCRIPTION,  S.WEBSITE
FROM PRODUCTS P
INNER JOIN SUPPLIERS S
ON P.SUPPLIER_ID = S.SUPPLIER_ID
WHERE P.STOCK > 0;
```

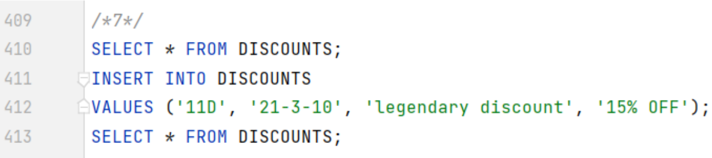
Output:



	IMAGE_PATH	DESCRIPTION	WEBSITE
1	product10-picture-path	The Pink Stuff Bundle,Miracle Cleaning Paste…	Voxpopme.com
2	product1-picture-path	this phone has 15gig ram and 3 cameras	Hotspex.org
3	product2-picture-path	Introducing Xbox Series X, the fastest, most…	Ipsos.com
4	product3-picture-path	M7 tablet | Case for tablet 2-1 Capacitive…	Hotspex.org
5	product4-picture-path	The Instant Pot Smart Wifi combines 8 kitche…	Zappi.com
6	product5-picture-path	Pack contains 12 Family Rolls of⏎Bounty whit…	Qualtrics.org
7	product6-picture-path	this phone has 15gig ram and 3 cameras	Zappi.com
8	product7-picture-path	: your dogs and cats will love their new ind…	Qualtrics.org
9	product8-picture-path	This pet hair remover roller can easily cle…	Kantar.com
10	product9-picture-path	85%polyester, 15%polyamideUltra-absorbent m…	Voxpopme.com

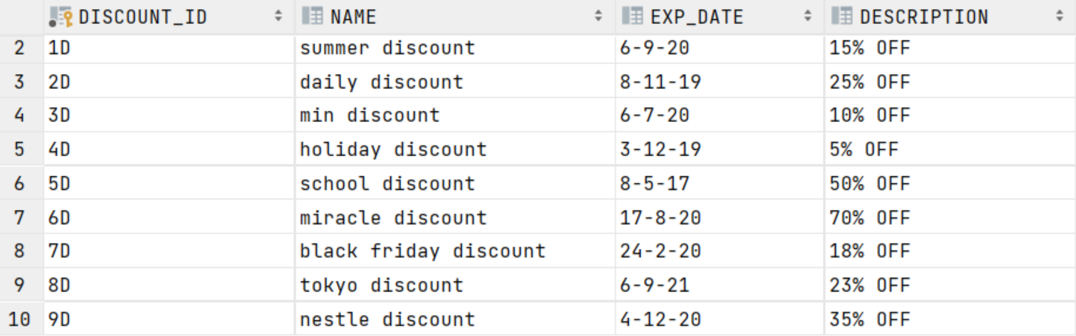
7.

Code:



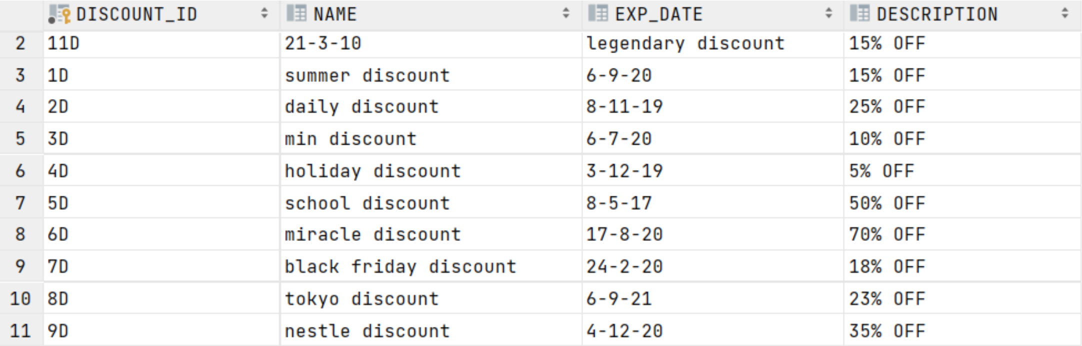
```
/*7*/
SELECT * FROM DISCOUNTS;
INSERT INTO DISCOUNTS
VALUES ('11D', '21-3-10', 'legendary discount', '15% OFF');
SELECT * FROM DISCOUNTS;
```

Before running code:



	DISCOUNT_ID	NAME	EXP_DATE	DESCRIPTION
2	1D	summer discount	6-9-20	15% OFF
3	2D	daily discount	8-11-19	25% OFF
4	3D	min discount	6-7-20	10% OFF
5	4D	holiday discount	3-12-19	5% OFF
6	5D	school discount	8-5-17	50% OFF
7	6D	miracle discount	17-8-20	70% OFF
8	7D	black friday discount	24-2-20	18% OFF
9	8D	tokyo discount	6-9-21	23% OFF
10	9D	nestle discount	4-12-20	35% OFF

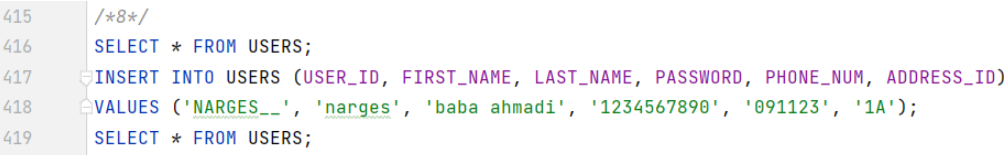
After running code:



	DISCOUNT_ID	NAME	EXP_DATE	DESCRIPTION
2	11D	21-3-10	legendary discount	15% OFF
3	1D	summer discount	6-9-20	15% OFF
4	2D	daily discount	8-11-19	25% OFF
5	3D	min discount	6-7-20	10% OFF
6	4D	holiday discount	3-12-19	5% OFF
7	5D	school discount	8-5-17	50% OFF
8	6D	miracle discount	17-8-20	70% OFF
9	7D	black friday discount	24-2-20	18% OFF
10	8D	tokyo discount	6-9-21	23% OFF
11	9D	nestle discount	4-12-20	35% OFF

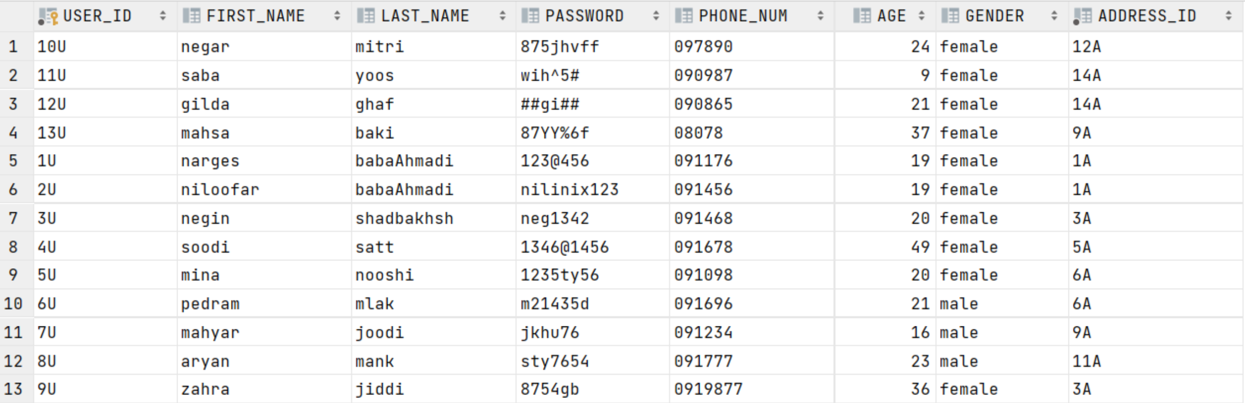
8.

Code:



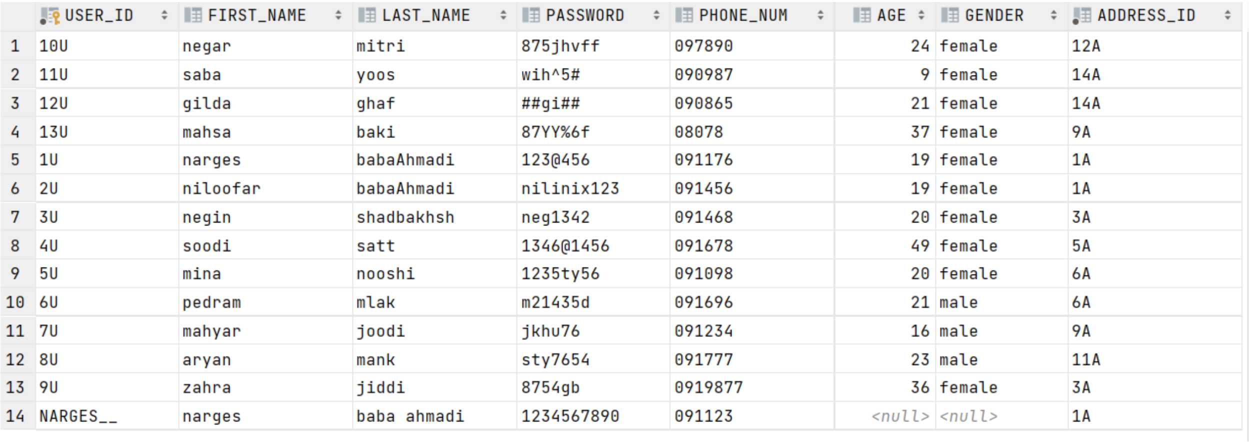
```
/*8*/
SELECT * FROM USERS;
INSERT INTO USERS (USER_ID, FIRST_NAME, LAST_NAME, PASSWORD, PHONE_NUM, ADDRESS_ID)
VALUES ('NARGES__', 'narges', 'baba ahmadi', '1234567890', '091123', '1A');
SELECT * FROM USERS;
```

Before running code:



	USER_ID	FIRST_NAME	LAST_NAME	PASSWORD	PHONE_NUM	AGE	GENDER	ADDRESS_ID
1	10U	negar	mitri	875jhvff	097890	24	female	12A
2	11U	saba	yoos	wih^5#	090987	9	female	14A
3	12U	gilda	ghaf	##gi##	090865	21	female	14A
4	13U	mahsa	baki	87YY%6f	08078	37	female	9A
5	1U	narges	babaAhmadi	123@456	091176	19	female	1A
6	2U	niloofar	babaAhmadi	nilinix123	091456	19	female	1A
7	3U	negin	shadbakhsh	neg1342	091468	20	female	3A
8	4U	soodi	satt	1346@1456	091678	49	female	5A
9	5U	mina	nooshi	1235ty56	091098	20	female	6A
10	6U	pedram	mlak	m21435d	091696	21	male	6A
11	7U	mahyar	joodi	jkhu76	091234	16	male	9A
12	8U	aryan	mank	sty7654	091777	23	male	11A
13	9U	zahra	jiddi	8754gb	0919877	36	female	3A

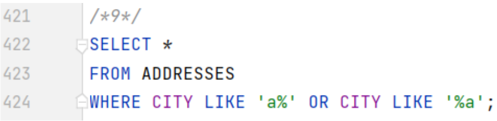
After running code:



	USER_ID	FIRST_NAME	LAST_NAME	PASSWORD	PHONE_NUM	AGE	GENDER	ADDRESS_ID
1	10U	negar	mitri	875jhvff	097890	24	female	12A
2	11U	saba	yoos	wih^5#	090987	9	female	14A
3	12U	gilda	ghaf	##gi##	090865	21	female	14A
4	13U	mahsa	baki	87YY%6f	08078	37	female	9A
5	1U	narges	babaAhmadi	123@456	091176	19	female	1A
6	2U	niloofar	babaAhmadi	nilinix123	091456	19	female	1A
7	3U	negin	shadbakhsh	neg1342	091468	20	female	3A
8	4U	soodi	satt	1346@1456	091678	49	female	5A
9	5U	mina	nooshi	1235ty56	091098	20	female	6A
10	6U	pedram	mlak	m21435d	091696	21	male	6A
11	7U	mahyar	joodi	jkhu76	091234	16	male	9A
12	8U	aryan	mank	sty7654	091777	23	male	11A
13	9U	zahra	jiddi	8754gb	0919877	36	female	3A
14	NARGES__	narges	baba ahmadi	1234567890	091123	<null>	<null>	1A

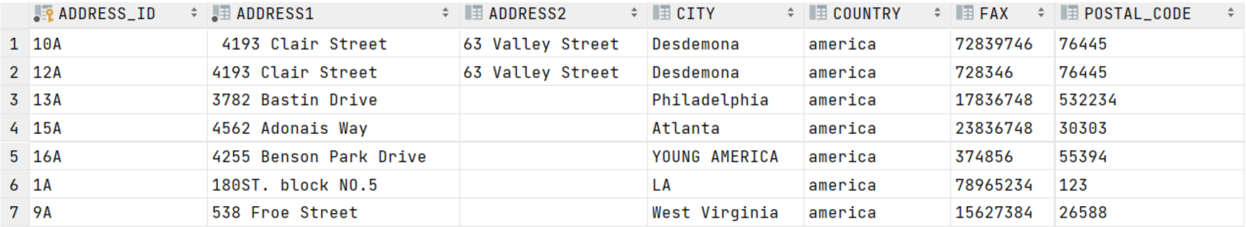
9.

Code:



```
/*9*/
SELECT *
FROM ADDRESSES
WHERE CITY LIKE 'a%' OR CITY LIKE '%a';
```

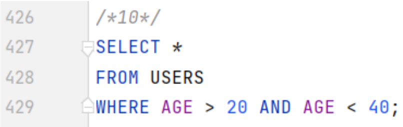
Output:



	ADDRESS_ID	ADDRESS1	ADDRESS2	CITY	COUNTRY	FAX	POSTAL_CODE
1	10A	4193 Clair Street	63 Valley Street	Desdemona	america	72839746	76445
2	12A	4193 Clair Street	63 Valley Street	Desdemona	america	728346	76445
3	13A	3782 Bastin Drive		Philadelphia	america	17836748	532234
4	15A	4562 Adonais Way		Atlanta	america	23836748	30303
5	16A	4255 Benson Park Drive		YOUNG AMERICA	america	374856	55394
6	1A	180ST. block NO.5		LA	america	78965234	123
7	9A	538 Froe Street		West Virginia	america	15627384	26588

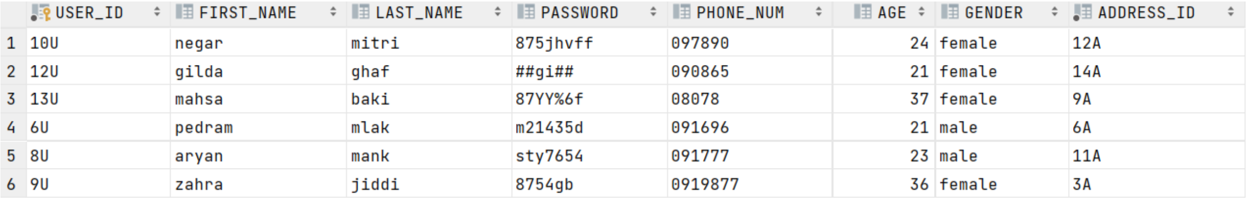
10.

Code:



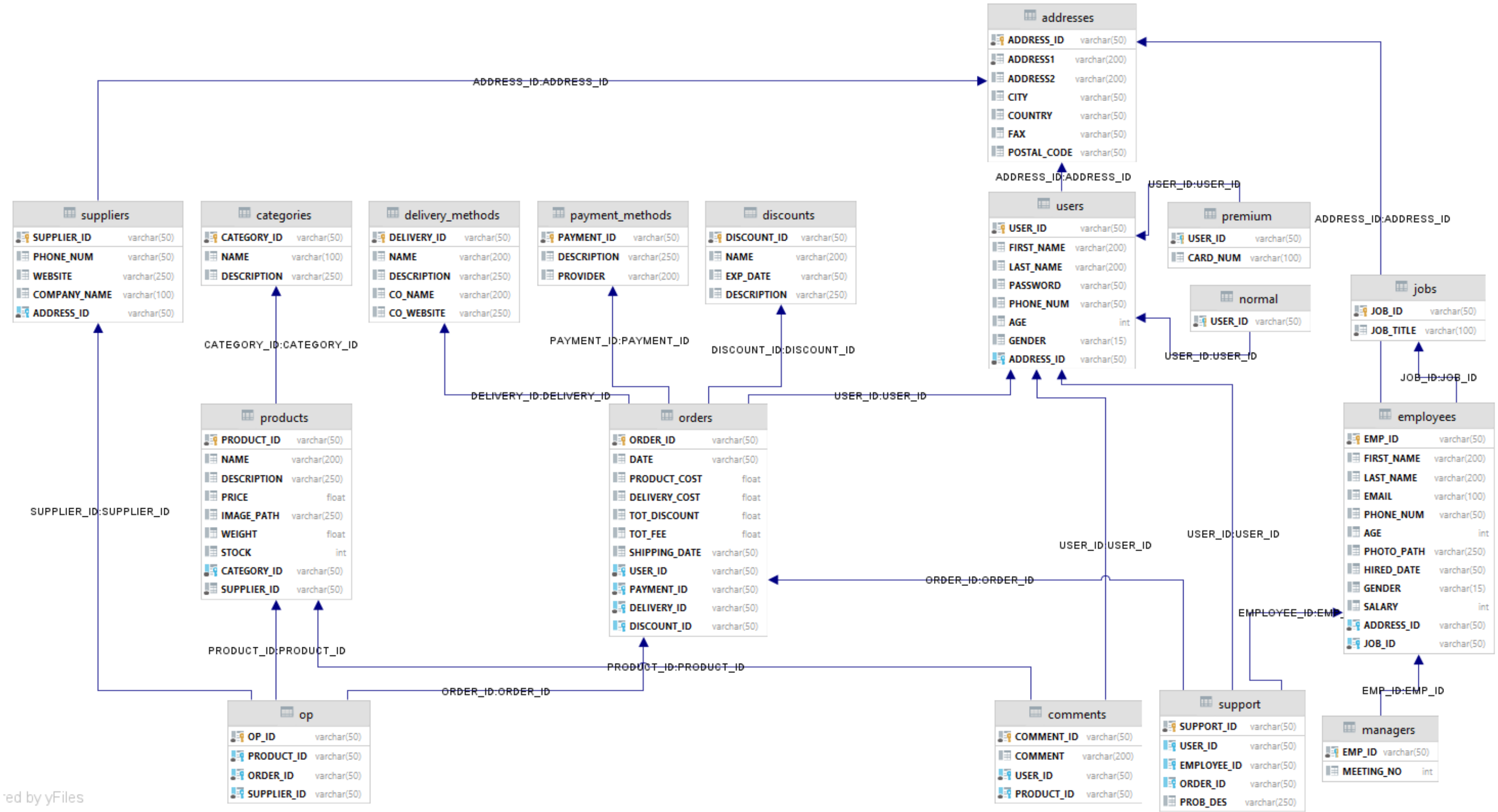
```
/*10*/
SELECT *
FROM USERS
WHERE AGE > 20 AND AGE < 40;
```

Output:



	USER_ID	FIRST_NAME	LAST_NAME	PASSWORD	PHONE_NUM	AGE	GENDER	ADDRESS_ID
1	10U	negar	mitri	875jhvff	097890	24	female	12A
2	12U	gilda	ghaf	##gi##	090865	21	female	14A
3	13U	mahsa	baki	87YY%6f	08078	37	female	9A
4	6U	pedram	mlak	m21435d	091696	21	male	6A
5	8U	aryan	mank	sty7654	091777	23	male	11A
6	9U	zahra	jiddi	8754gb	0919877	36	female	3A

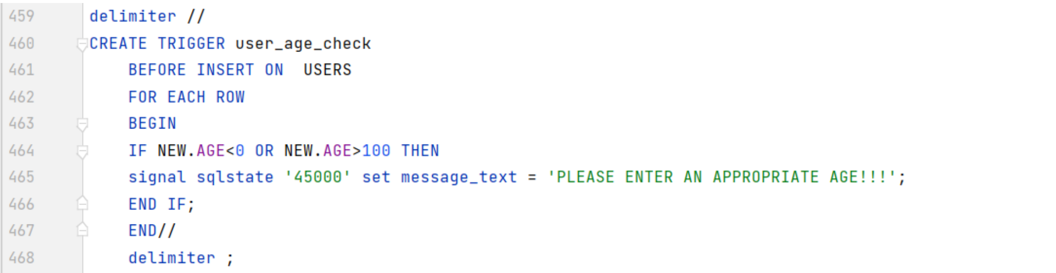
PART 16:



PART 17:

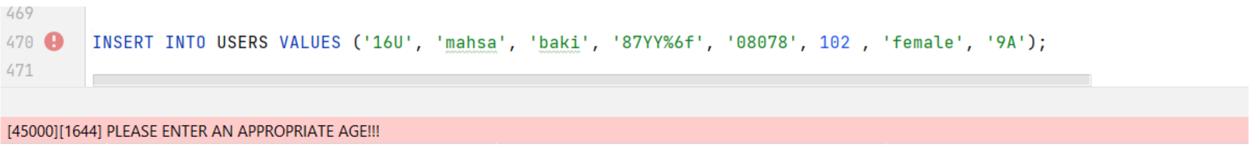
Trigger 1: in this trigger I limited the age of the users(which can not be under 0 or above 100)

Code:



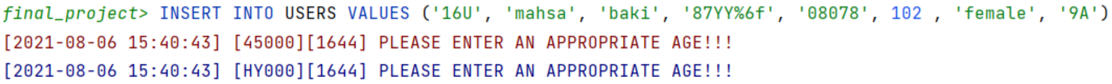
```
delimiter //
CREATE TRIGGER user_age_check
    BEFORE INSERT ON  USERS
    FOR EACH ROW
    BEGIN
    IF NEW.AGE<0 OR NEW.AGE>100 THEN
    signal sqlstate '45000' set message_text = 'PLEASE ENTER AN APPROPRIATE AGE!!!';
    END IF;
    END//
    delimiter ;
```

So when I try to insert someone with the age of 102, I get the error below:



```
INSERT INTO USERS VALUES ('16U', 'mahsa', 'baki', '87YY%6f', '08078', 102 , 'female', '9A');
```

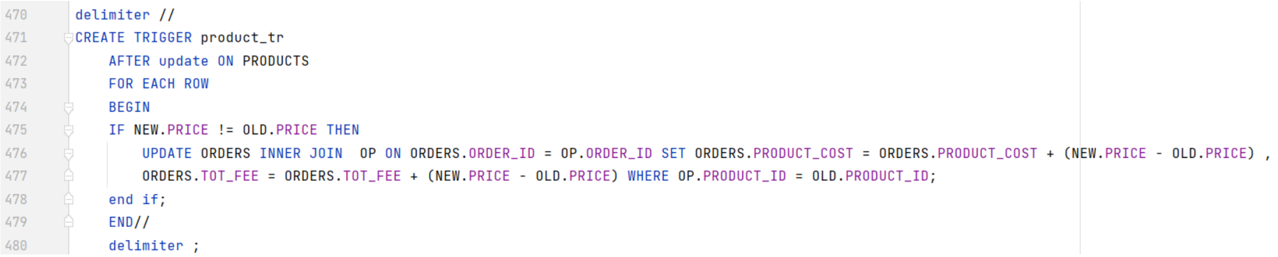
```
[45000][1644] PLEASE ENTER AN APPROPRIATE AGE!!!
```



```
final_project> INSERT INTO USERS VALUES ('16U', 'mahsa', 'baki', '87YY%6f', '08078', 102 , 'female', '9A')
[2021-08-06 15:40:43] [45000][1644] PLEASE ENTER AN APPROPRIATE AGE!!!
[2021-08-06 15:40:43] [HY000][1644] PLEASE ENTER AN APPROPRIATE AGE!!!
```

Trigger 2: in this trigger I wanted to make sure that if a product's price changes, the orders table will also change due to that.

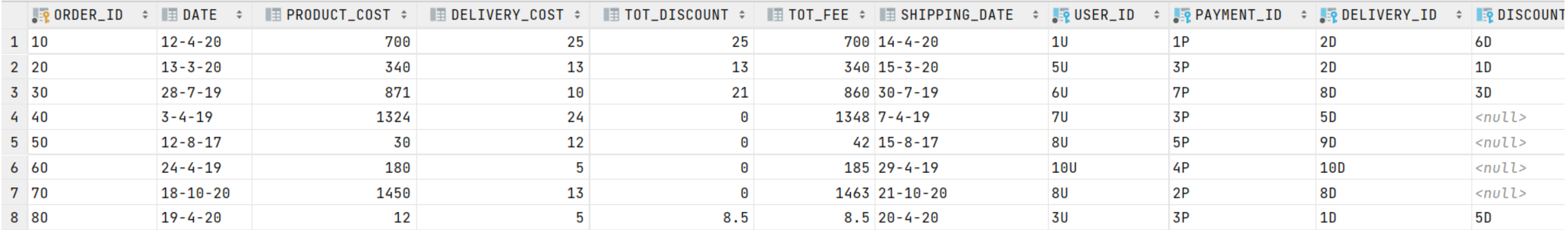
Code:



```
delimiter //
CREATE TRIGGER product_tr
    AFTER update ON PRODUCTS
    FOR EACH ROW
    BEGIN
    IF NEW.PRICE != OLD.PRICE THEN
        UPDATE ORDERS INNER JOIN  OP ON ORDERS.ORDER_ID = OP.ORDER_ID SET ORDERS.PRODUCT_COST = ORDERS.PRODUCT_COST + (NEW.PRICE - OLD.PRICE) ,
        ORDERS.TOT_FEE = ORDERS.TOT_FEE + (NEW.PRICE - OLD.PRICE) WHERE OP.PRODUCT_ID = OLD.PRODUCT_ID;
    end if;
    END//
    delimiter ;
```

Checking if my trigger works:

Orders table before updating products:



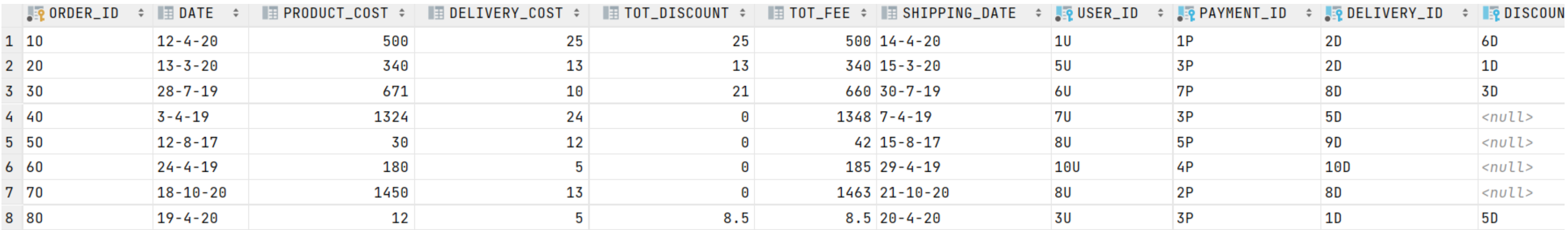
	ORDER_ID	DATE	PRODUCT_COST	DELIVERY_COST	TOT_DISCOUNT	TOT_FEE	SHIPPING_DATE	USER_ID	PAYMENT_ID	DELIVERY_ID	DISCOUNT
1	10	12-4-20	700	25	25	700	14-4-20	1U	1P	2D	6D
2	20	13-3-20	340	13	13	340	15-3-20	5U	3P	2D	1D
3	30	28-7-19	871	10	21	860	30-7-19	6U	7P	8D	3D
4	40	3-4-19	1324	24	0	1348	7-4-19	7U	3P	5D	<null>
5	50	12-8-17	30	12	0	42	15-8-17	8U	5P	9D	<null>
6	60	24-4-19	180	5	0	185	29-4-19	10U	4P	10D	<null>
7	70	18-10-20	1450	13	0	1463	21-10-20	8U	2P	8D	<null>
8	80	19-4-20	12	5	8.5	8.5	20-4-20	3U	3P	1D	5D

Here I updated the price of the product labeled '2p'.

This product costed 700$ initially and I wanted to change it to 500$.

Orders '1O' and '3O' should change.

Orders table after updating products:

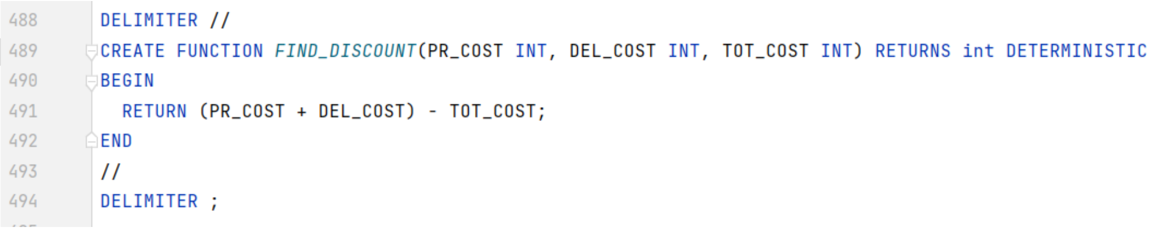


	ORDER_ID	DATE	PRODUCT_COST	DELIVERY_COST	TOT_DISCOUNT	TOT_FEE	SHIPPING_DATE	USER_ID	PAYMENT_ID	DELIVERY_ID	DISCOUN
1	10	12-4-20	500	25	25	500	14-4-20	1U	1P	2D	6D
2	20	13-3-20	340	13	13	340	15-3-20	5U	3P	2D	1D
3	30	28-7-19	671	10	21	660	30-7-19	6U	7P	8D	3D
4	40	3-4-19	1324	24	0	1348	7-4-19	7U	3P	5D	<null>
5	50	12-8-17	30	12	0	42	15-8-17	8U	5P	9D	<null>
6	60	24-4-19	180	5	0	185	29-4-19	10U	4P	10D	<null>
7	70	18-10-20	1450	13	0	1463	21-10-20	8U	2P	8D	<null>
8	80	19-4-20	12	5	8.5	8.5	20-4-20	3U	3P	1D	5D

As you can see they have changed , so my trigger worked properly.

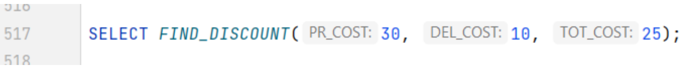
Function 1: This function gets the cost of the product and the delivery cost and the total fee, and return the amount of discount of this order.

Code:



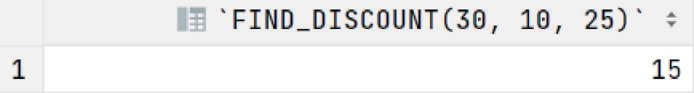
```
DELIMITER //
CREATE FUNCTION FIND_DISCOUNT(PR_COST INT, DEL_COST INT, TOT_COST INT) RETURNS int DETERMINISTIC
BEGIN
  RETURN (PR_COST + DEL_COST) - TOT_COST;
END
//
DELIMITER ;
```

Calling the function:



```
SELECT FIND_DISCOUNT( PR_COST: 30,  DEL_COST: 10,  TOT_COST: 25);
```

Output:

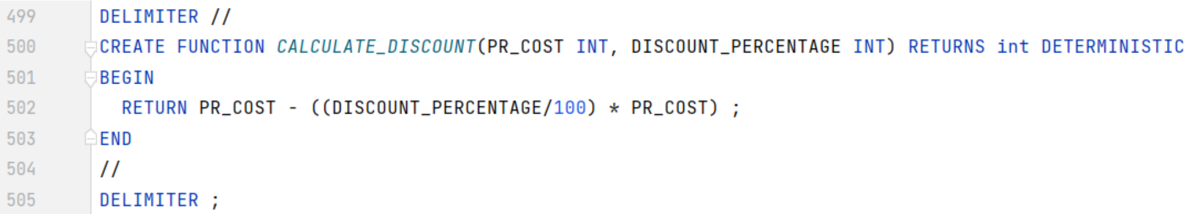


```
     `FIND_DISCOUNT(30, 10, 25)`
1                             15
```

As you can see, I got the right answer.

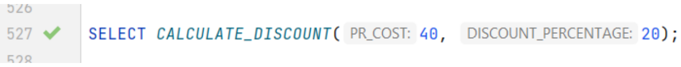
Function 2: This function gets the product's cost and the percentage of discount, and return the amount of discount in dollar.

Code:



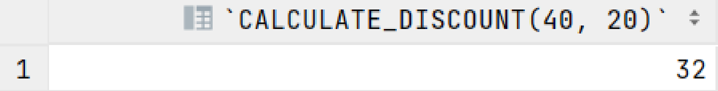
```
DELIMITER //
CREATE FUNCTION CALCULATE_DISCOUNT(PR_COST INT, DISCOUNT_PERCENTAGE INT) RETURNS int DETERMINISTIC
BEGIN
  RETURN PR_COST - ((DISCOUNT_PERCENTAGE/100) * PR_COST) ;
END
//
DELIMITER ;
```

Calling the function:



```
SELECT CALCULATE_DISCOUNT( PR_COST: 40,  DISCOUNT_PERCENTAGE: 20);
```

Output:

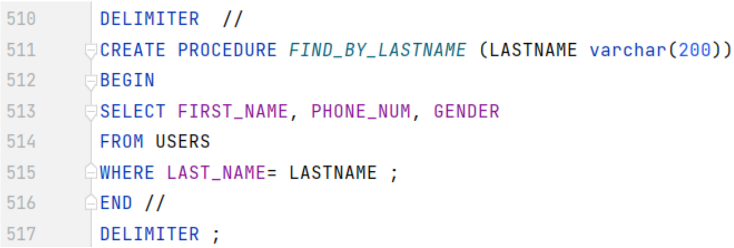


```
     `CALCULATE_DISCOUNT(40, 20)`
1                              32
```

And this is the right answer.

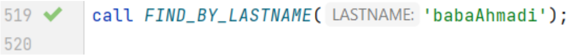
Stored procedure 1: This stored procedure gets the last name and find the first name, phone number and gender of the users with that last name.

Code:

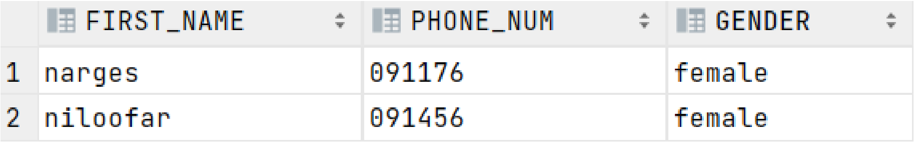


```
DELIMITER  //
CREATE PROCEDURE FIND_BY_LASTNAME (LASTNAME varchar(200))
BEGIN
SELECT FIRST_NAME, PHONE_NUM, GENDER
FROM USERS
WHERE LAST_NAME= LASTNAME ;
END //
DELIMITER ;
```

Checking the procedure:



```
call FIND_BY_LASTNAME( LASTNAME: 'babaAhmadi');
```

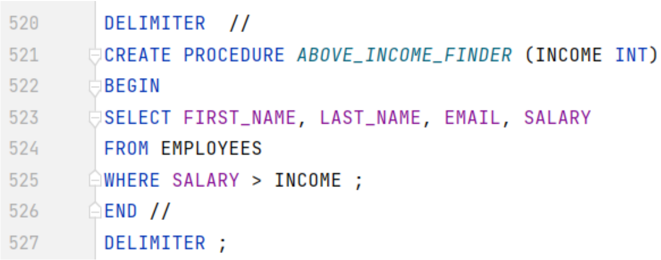


	FIRST_NAME	PHONE_NUM	GENDER
1	narges	091176	female
2	niloofar	091456	female

As you can see this procedure works properly.

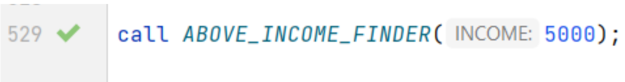
Stored procedure 2: This procedure gets a number and returns the first name, last name , email and salary of the employees who earns more than that amount.

Code:

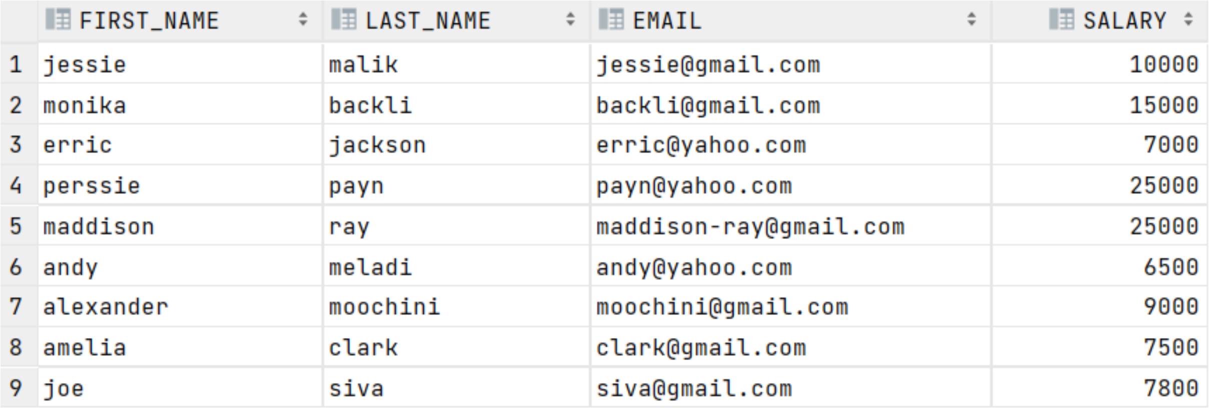


```
DELIMITER  //
CREATE PROCEDURE ABOVE_INCOME_FINDER (INCOME INT)
BEGIN
SELECT FIRST_NAME, LAST_NAME, EMAIL, SALARY
FROM EMPLOYEES
WHERE SALARY > INCOME ;
END //
DELIMITER ;
```

Checking the procedure:



```
call ABOVE_INCOME_FINDER( INCOME: 5000);
```



	FIRST_NAME	LAST_NAME	EMAIL	SALARY
1	jessie	malik	jessie@gmail.com	10000
2	monika	backli	backli@gmail.com	15000
3	erric	jackson	erric@yahoo.com	7000
4	perssie	payn	payn@yahoo.com	25000
5	maddison	ray	maddison-ray@gmail.com	25000
6	andy	meladi	andy@yahoo.com	6500
7	alexander	moochini	moochini@gmail.com	9000
8	amelia	clark	clark@gmail.com	7500
9	joe	siva	siva@gmail.com	7800

As you can see this procedure works properly.

Overall, a database is a system for storing and taking care of data and designing the database wisely can help us manage the data easier and faster.

References:

https://www.jetbrains.com/help/datagrip/meet-the-product.html

https://www.w3schools.com/MySQL/default.asp

https://www.guru99.com/database-normalization.html

https://www.guru99.com/database-design.html