2024

A Coffee Shop DB

A UJ STUDENTS PROJECT

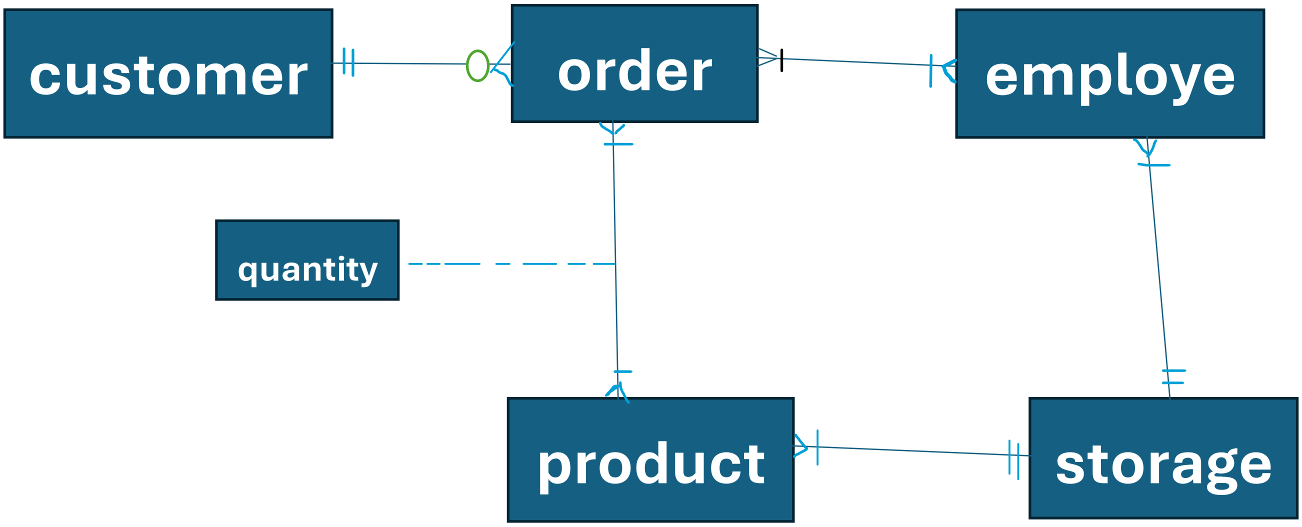
نواف فهد الحربي 2441220
زياد نايف الحربي 2441402
طارق سليمان الغرير 2441172
عمر محمد العنزي 2441263

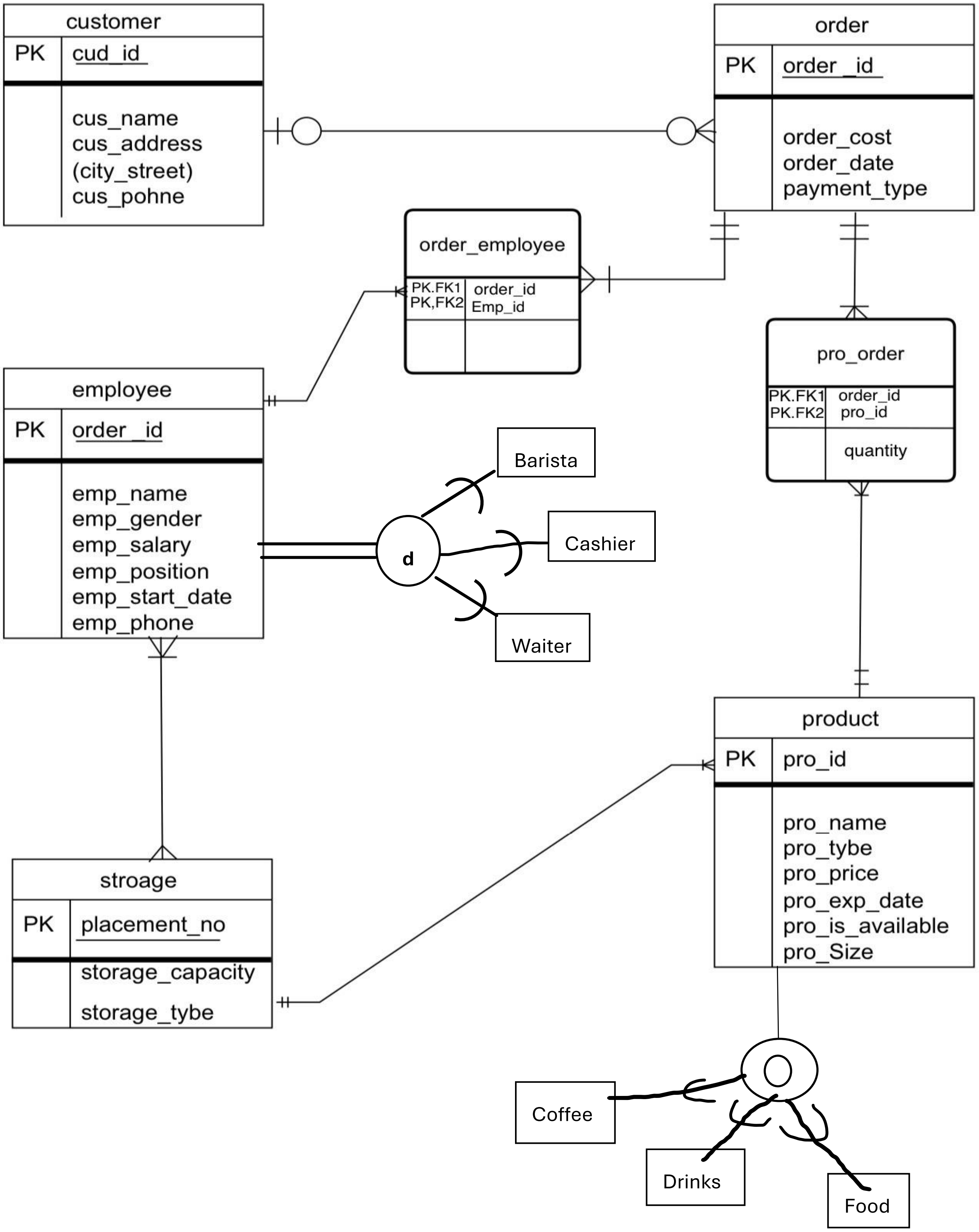
Project Idea: **Making an ER -Entity Relationship- about a coffee shop**.

This project focuses on creating a visual representation of the data relationships within a coffee shop using an Entity-Relationship (ER) Model. This model will help in understanding the data flow, identifying key entities involved in the coffee shop's operations, and establishing a foundation for a future database management system.

Entity	Primary key
order	order_id
customer	cus_id
employee	emp_id
product	pro_id
storage	placement_no
product	pro_id
Foreign key	**Foreign Key's attributes**
order_employee	order_id emp_id
pro_order	pro_id order_id quantity

First Phase:




customer
PK
cud_id
cus_name
cus_address
(city_street)
cus_pohne
order
PK
order _id
order_cost
order_date
payment_type
order_employee
PK.FK1
PK,FK2
order_id
Emp_id
pro_order
PK.FK1
PK.FK2
order_id
pro_id
quantity
employee
PK
order _id
emp_name
emp_gender
emp_salary
emp_position
emp_start_date
emp_phone
d
Barista
Cashier
Waiter
product
PK
pro_id
pro_name
pro_tybe
pro_price
pro_exp_date
pro_is_available
pro_Size
stroage
PK
placement_no
storage_capacity
storage_tybe
O
Coffee
Drinks
Food

- **Customer (cus)**: This entity stores information about the customer who places the coffee order. It includes:
    - `cus_id`: Primary key (PK) for the customer table.
    - `cus_name`: Name of the customer.
    - `cus_address`: Customer's address, including city and street.
    - `cus_phone`: Customer's phone number.
- **Order (order)**: This entity stores information about the coffee order itself. It includes:
    - `order_id`: Primary key (PK) for the order table. It is also a foreign key (FK1) to the `order_employee` table and a foreign key (FK2) to the `pro_order` table.
    - `order_employee`: This field likely refers to a foreign key referencing the `emp_id` in the `employee` table, indicating the employee who took the order.
    - `order_cost`: Total cost of the order.
    - `order_date`: Date the order was placed.
    - `payment_type`: Method of payment for the order.
- **Employee (employee)**: This entity stores information about the coffee shop employees. It includes:
    - `emp_id`: Primary key (PK) for the employee table. It is also a foreign key (FK) to the `order_employee` table.
    - `emp_name`: Name of the employee.
    - `emp_gender`: Gender of the employee.
    - `emp_salary`: Employee's salary.
    - `emp_position`: Employee's position at the coffee shop (e.g., barista, cashier).
    - `emp_start_date`: Date the employee started working at the coffee shop.
    - `emp_phone`: Employee's phone number.
- **Storage (storage)**: This entity stores information about how the coffee shop stores its products. It includes:
    - `placement_no`: Primary key (PK) for the storage table.

    - `storage_capacity`: Storage capacity, possibly referring to the amount of product a certain storage unit can hold.
    - `storage_type`: Type of storage unit (e.g., shelf, bin, refrigerator).
- **Product (pro)**: This entity stores information about the coffee shop's products. It includes:
    - `pro_id`: Primary key (PK) for the product table. It is also a foreign key (FK) to the `pro_order` table.
    - `pro_name`: Name of the coffee shop product (e.g., latte, croissant).
    - `pro_type`: Type of product (e.g., coffee, pastry).
    - `pro_price`: Price of the product.
    - `pro_exp_date`: Expiration date of the product.
    - `pro_is_available`: This field likely indicates whether the product is currently available for purchase (yes/no).
    - `pro_`*`Size:`* **`The size of the pr`oduct**
- **Check (check)**: This entity might be used to record when a customer verifies or confirms their order. It likely links to another table (not shown here) that contains the order details.
- **Order_employee (order_employee)**: This table is not shown explicitly, but it is referenced as having foreign keys from both the `order` and `employee` tables. This table likely stores the relationship between a specific order and the employee who took the order.
- **Pro_order (pro_order)**: This table is also not shown explicitly, but it is referenced as having foreign keys from both the `order` and `product` tables. This table likely stores the relationship between a specific order and the products included in that order.

Second Phase:

Goal:

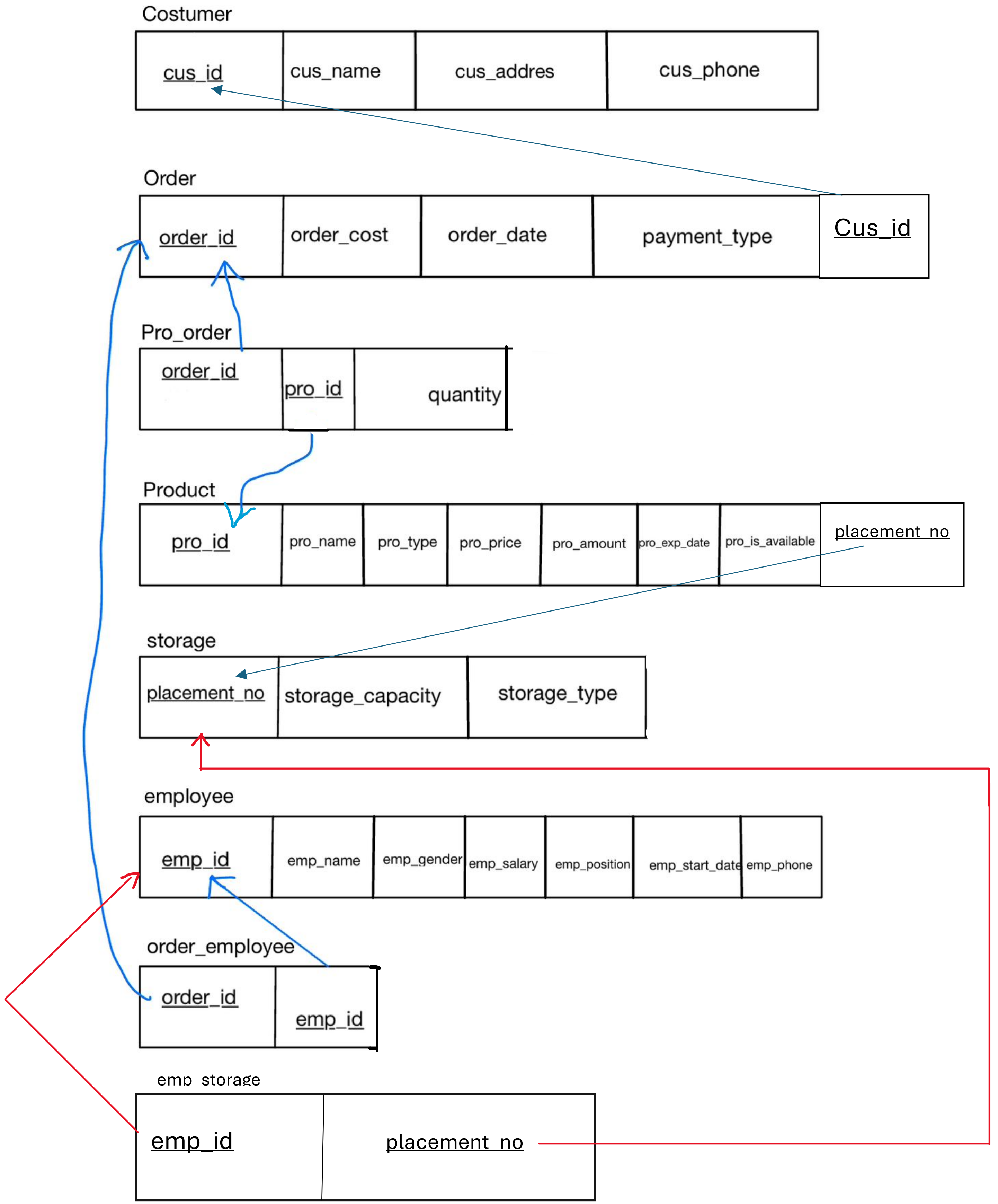
In the second phase of our Coffee Shop Database project, we're diving into the details of database development. Here's what's on the agenda:

1. **ER Diagram Conversion to Relational Schema**: We'll seamlessly transition from our Entity-Relationship (ER) diagram to a relational schema. This step involves translating the conceptual model into a structured format that reflects the relationships between entities as tables and attributes. Every key attribute will be underlined, ensuring clarity and ease of navigation within the database.

2. **Normalization Process**: Normalization is key to ensuring our database is efficient, scalable, and free from data anomalies. Each table will undergo normalization, striving for at least Third Normal Form (3NF). By breaking down data into smaller, logically organized units, we optimize storage and minimize redundancy.

3. **Functional Dependency Specification**: Understanding the dependencies within our data is crucial for maintaining integrity and reliability. In this phase, we'll meticulously outline all functional dependencies for each relation. By mapping out the relationships between attributes, we establish the rules governing data manipulation and retrieval.

Through these meticulous steps, we're not just building a Coffee Shop Database; we're crafting a robust foundation that will support the smooth operation of our coffee business, from inventory management to customer interactions. With each phase, we're one step closer to a seamlessly integrated database system that powers our coffee shop's success.



Costumer

cus_id	cus_name	cus_addres	cus_phone

Order

order_id	order_cost	order_date	payment_type	Cus_id

Pro_order

order_id	pro_id	quantity

Product

pro_id	pro_name	pro_type	pro_price	pro_amount	pro_exp_date	pro_is_available	placement_no

storage

placement_no	storage_capacity	storage_type

employee

emp_id	emp_name	emp_gender	emp_salary	emp_position	emp_start_date	emp_phone

order_employee

order_id	emp_id

emp_storage

emp_id	placement_no

Unnormalized-

costumer, (cus _id, order_id, cus_addres, cus_name, cus_phone)

order(order_ id, order_cost order_date payment_type)

Proudct(Pro_id,Pro_name,Pro_type,pro_price,Pro_amount,Pro_exp_date, pro_is_available)

storage(placement_ no, storage_capacity, storage_type)

Employee(emp_id,emp_name,emp_gender,emp_salary ,emp_position,emp_start_date,emp_phone)

Costumer

cus_id	order_cost	cus_address	cus_name	cus_phone

Order

order_id	order_cost	order_date	payment_type

Product

Pro_id	Pro_name	Pro_type	Pro_exp_date	Pro_is_unavailable

Storage

placement_no	storage_capacity	storage_type

Employee

emp_id	emp_name	emp_gender	emp_salary	emp_position	emp_start_date	emp_phone

NF1

Costumer

(**<u>cus _id</u>**, <u>order_id</u>, cus_addres, cus_name, cus_phone)

Order

(**<u>order _id</u>**, order_cost, order_date, payment_type)

Product

(**<u>Pro_id</u>**,Pro_name,Pro_type,Pro_amount,Pro_exp_date,
pro_is_available,storge)

Storage

(**<u>placement_ no</u>**, storage_capacity, storage_type)

Employee

(**<u>emp_id</u>**,emp_name,emp_gender,emp_salary,

emp_position,emp_start_date,emp_phone)

NF2

Costumer

(**cus _id**, cus_addres, cus_name, cus_phone,#order_id)

Order

(**order _id**, order_cost order_date payment_type)

Product

(**Pro_id**,Pro_name,Pro_type,Pro_amount,Pro_exp_date, pro_is_available,storge)

Storage

(**placement_ no**, storage_capacity, storage_type,# pro_id, #emp_id)

Employee

(**emp_id**,emp_name,emp_gender,emp_salary,emp_position,emp_start_date,emp_phone)

NF3

Costumer

(**cus _id**, , cus_addres, cus_name, cus_phone,#order_id

Order

(**order _id**, order_cost order_date payment_type)

Product

(**Pro_id**,Pro_name,Pro_type,Pro_amount,Pro_exp_date, pro_is_available,storge)

Storage

(**placement_ no**, storage_capacity, storage_type,# pro_id, #emp_id)

Employee

(**emp_id**,emp_name,emp_gender,emp_salary,emp_position,emp_start_date,emp_phone)

Functional Dependencies

cus _id>>>cus_addres, cus_name, cus_phone

order _id>>> order_cost, order_date,payment_type

Pro_id>>>Pro_name,Pro_type,Pro_amount,Pro_exp_date, pro_is_available,storge

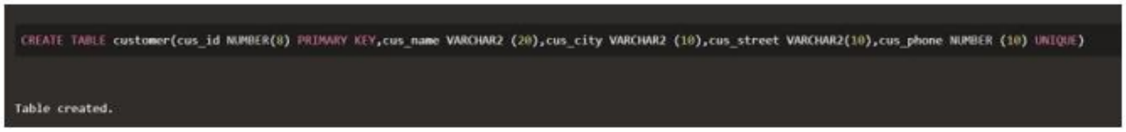
placement_ no>>> storage_capacity, storage_type

emp_id>>>emp_name,emp_gender,emp_salary,emp_position,emp_start_date,emp_phone

Third Phase:

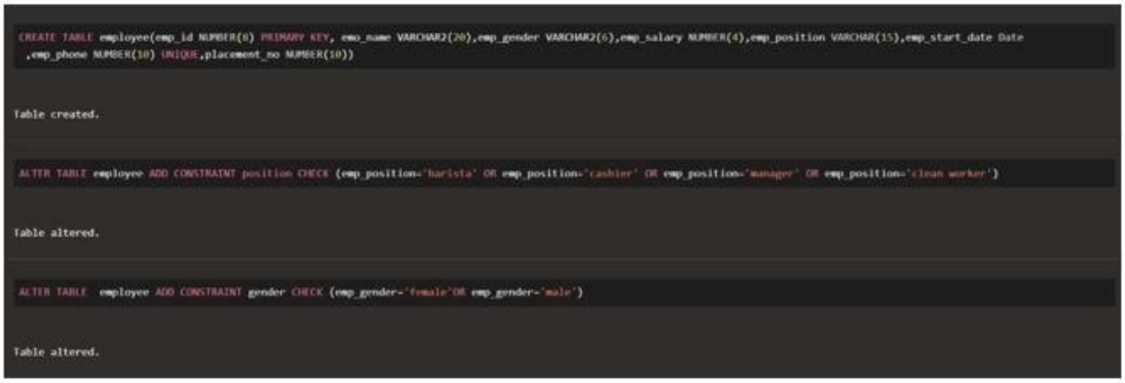
Goal: In the third phase, we'll be crafting SQL code to bring our database schema to life. This involves creating tables based on our relational schema, establishing key constraints for data integrity, indexing columns for performance optimization, populating tables with sample data, and optionally implementing stored procedures, functions, and triggers to automate tasks and enforce business rules.

1- customer table.

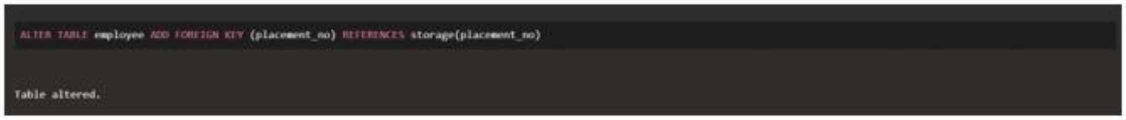


```
CREATE TABLE customer(cus_id NUMBER(8) PRIMARY KEY,cus_name VARCHAR2 (20),cus_city VARCHAR2 (10),cus_street VARCHAR2(10),cus_phone NUMBER (10) UNIQUE)


Table created.
```

2- employee table.

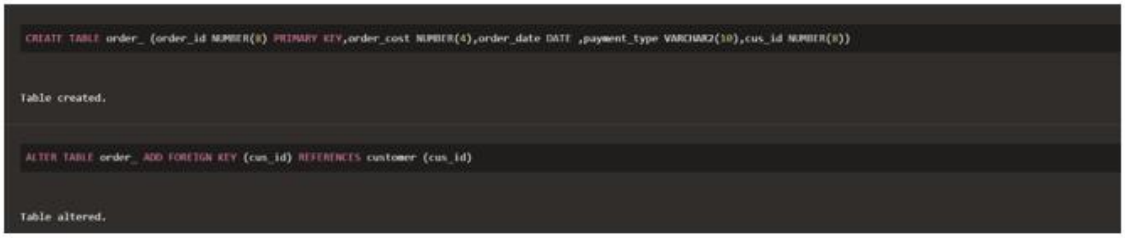


```
CREATE TABLE employee(emp_id NUMBER(8) PRIMARY KEY, emo_name VARCHAR2(20),emp_gender VARCHAR2(6),emp_salary NUMBER(4),emp_position VARCHAR(15),emp_start_date Date
 ,emp_phone NUMBER(10) UNIQUE,placement_no NUMBER(10))


Table created.

ALTER TABLE employee ADD CONSTRAINT position CHECK (emp_position='barista' OR emp_position='cashier' OR emp_position='manager' OR emp_position='clean worker')


Table altered.

ALTER TABLE  employee ADD CONSTRAINT gender CHECK (emp_gender='female'OR emp_gender='male')


Table altered.
```



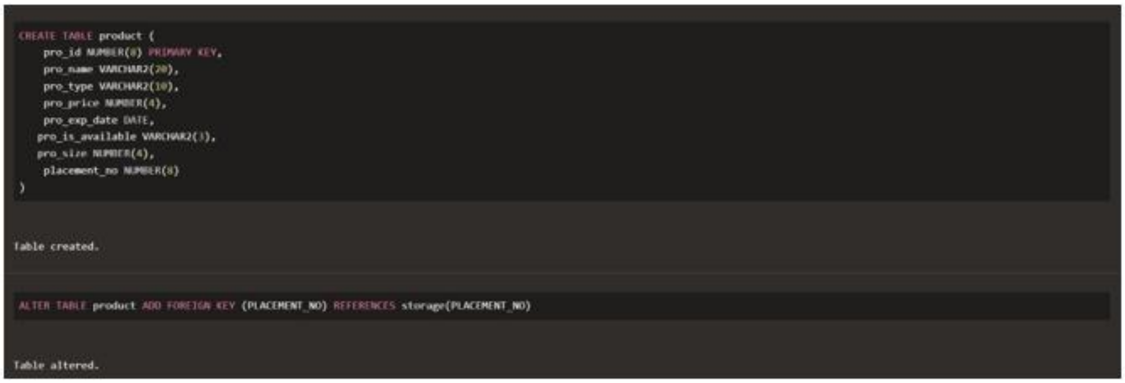
```
ALTER TABLE employee ADD FOREIGN KEY (placement_no) REFERENCES storage(placement_no)


Table altered.
```

3 – order table.



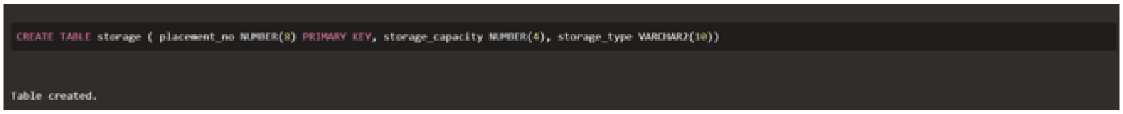
```
CREATE TABLE order_ (order_id NUMBER(8) PRIMARY KEY,order_cost NUMBER(4),order_date DATE ,payment_type VARCHAR2(10),cus_id NUMBER(8))


Table created.

ALTER TABLE order_ ADD FOREIGN KEY (cus_id) REFERENCES customer (cus_id)


Table altered.
```

4- product table.



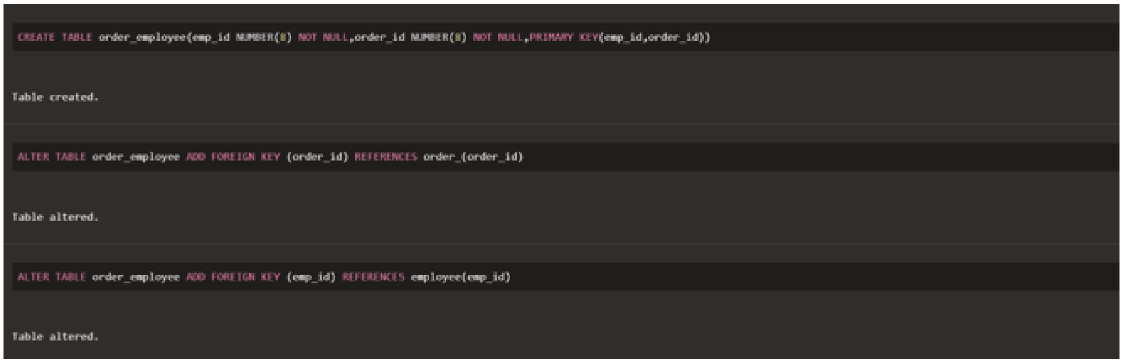
```
CREATE TABLE product (
    pro_id NUMBER(8) PRIMARY KEY,
    pro_name VARCHAR2(20),
    pro_type VARCHAR2(10),
    pro_price NUMBER(4),
    pro_exp_date DATE,
   pro_is_available VARCHAR2(3),
   pro_size NUMBER(4),
    placement_no NUMBER(8)
)


Table created.

ALTER TABLE product ADD FOREIGN KEY (PLACEMENT_NO) REFERENCES storage(PLACEMENT_NO)


Table altered.
```

5- storage table.



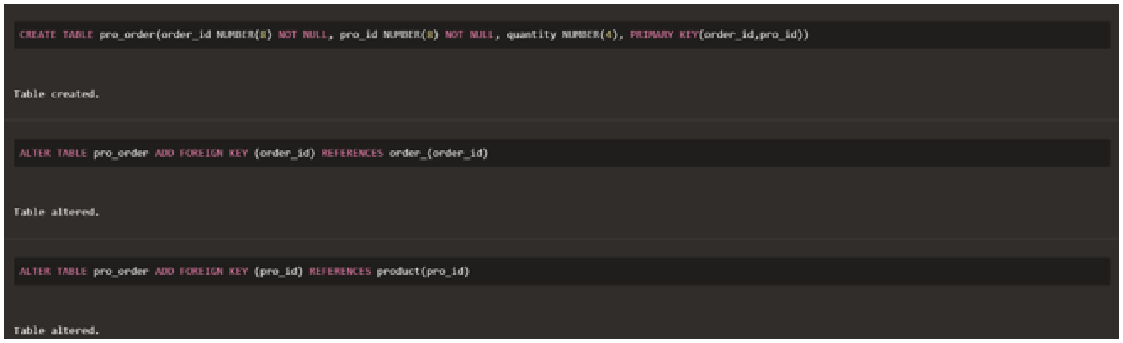
```
CREATE TABLE storage ( placement_no NUMBER(8) PRIMARY KEY, storage_capacity NUMBER(4), storage_type VARCHAR2(10))

Table created.
```

6- order_employee table.



```
CREATE TABLE order_employee(emp_id NUMBER(8) NOT NULL,order_id NUMBER(8) NOT NULL,PRIMARY KEY(emp_id,order_id))

Table created.

ALTER TABLE order_employee ADD FOREIGN KEY (order_id) REFERENCES order_(order_id)

Table altered.

ALTER TABLE order_employee ADD FOREIGN KEY (emp_id) REFERENCES employee(emp_id)

Table altered.
```

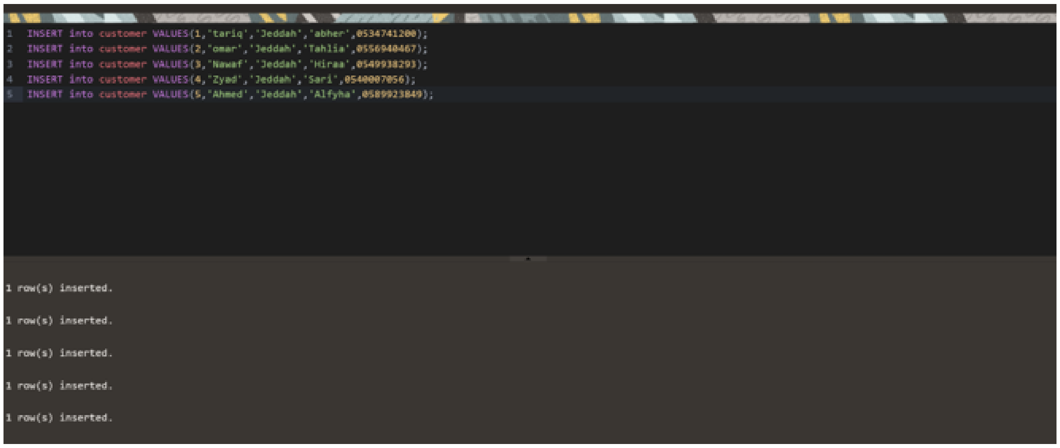
7- pro_order table.



```
CREATE TABLE pro_order(order_id NUMBER(8) NOT NULL, pro_id NUMBER(8) NOT NULL, quantity NUMBER(4), PRIMARY KEY(order_id,pro_id))

Table created.

ALTER TABLE pro_order ADD FOREIGN KEY (order_id) REFERENCES order_(order_id)

Table altered.

ALTER TABLE pro_order ADD FOREIGN KEY (pro_id) REFERENCES product(pro_id)

Table altered.
```

Insert and select

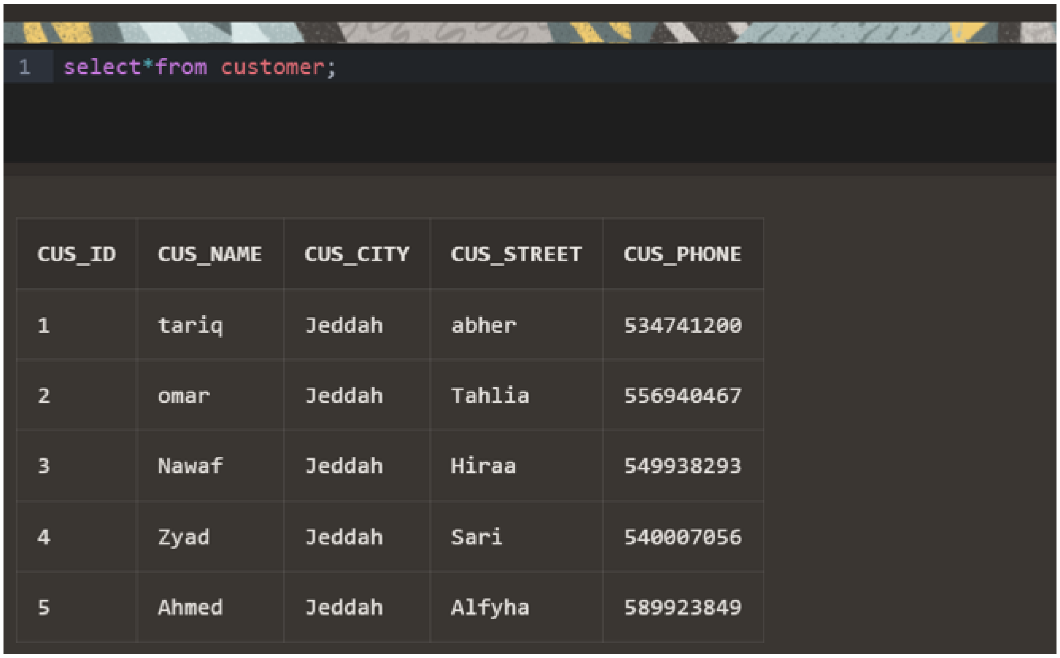
1- insert customer table1



```
INSERT into customer VALUES(1,'tariq','Jeddah','abher',0534741200);
INSERT into customer VALUES(2,'omar','Jeddah','Tahlia',0556940467);
INSERT into customer VALUES(3,'Nawaf','Jeddah','Hiraa',0549938293);
INSERT into customer VALUES(4,'Zyad','Jeddah','Sari',0540007056);
INSERT into customer VALUES(5,'Ahmed','Jeddah','Alfyha',0589923849);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

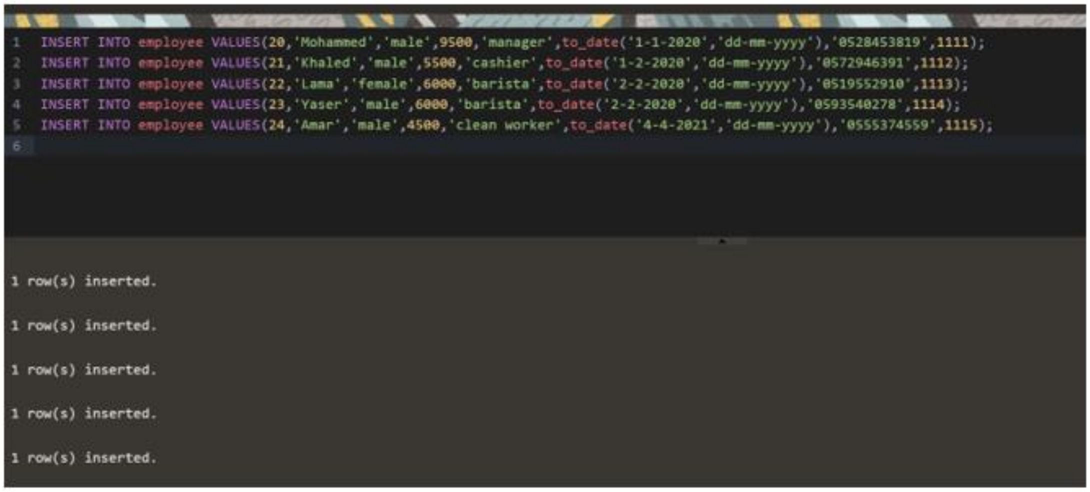
Select customer



```
select*from customer;
```

CUS_ID	CUS_NAME	CUS_CITY	CUS_STREET	CUS_PHONE
1	tariq	Jeddah	abher	534741200
2	omar	Jeddah	Tahlia	556940467
3	Nawaf	Jeddah	Hiraa	549938293
4	Zyad	Jeddah	Sari	540007056
5	Ahmed	Jeddah	Alfyha	589923849

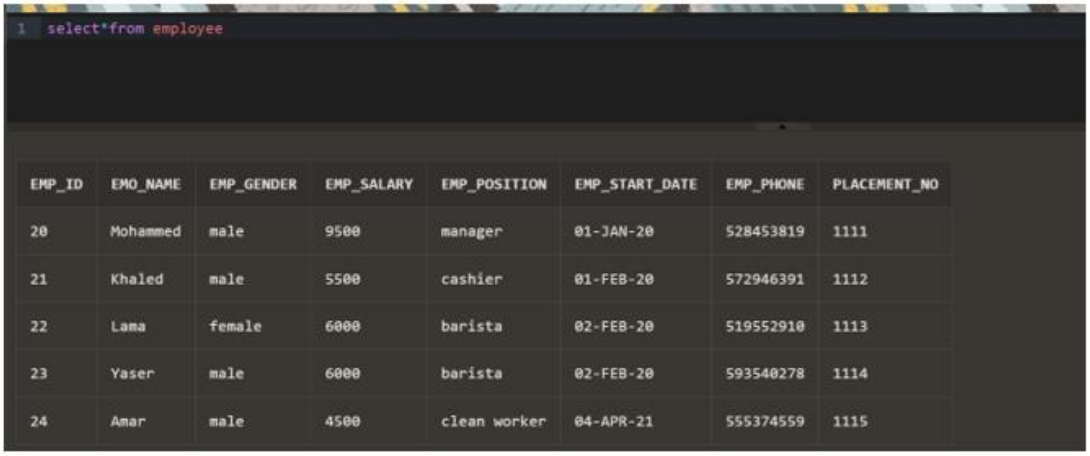
2- insert employee table



```
INSERT INTO employee VALUES(20,'Mohammed','male',9500,'manager',to_date('1-1-2020','dd-mm-yyyy'),'0528453819',1111);
INSERT INTO employee VALUES(21,'Khaled','male',5500,'cashier',to_date('1-2-2020','dd-mm-yyyy'),'0572946391',1112);
INSERT INTO employee VALUES(22,'Lama','female',6000,'barista',to_date('2-2-2020','dd-mm-yyyy'),'0519552910',1113);
INSERT INTO employee VALUES(23,'Yaser','male',6000,'barista',to_date('2-2-2020','dd-mm-yyyy'),'0593540278',1114);
INSERT INTO employee VALUES(24,'Amar','male',4500,'clean worker',to_date('4-4-2021','dd-mm-yyyy'),'0555374559',1115);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

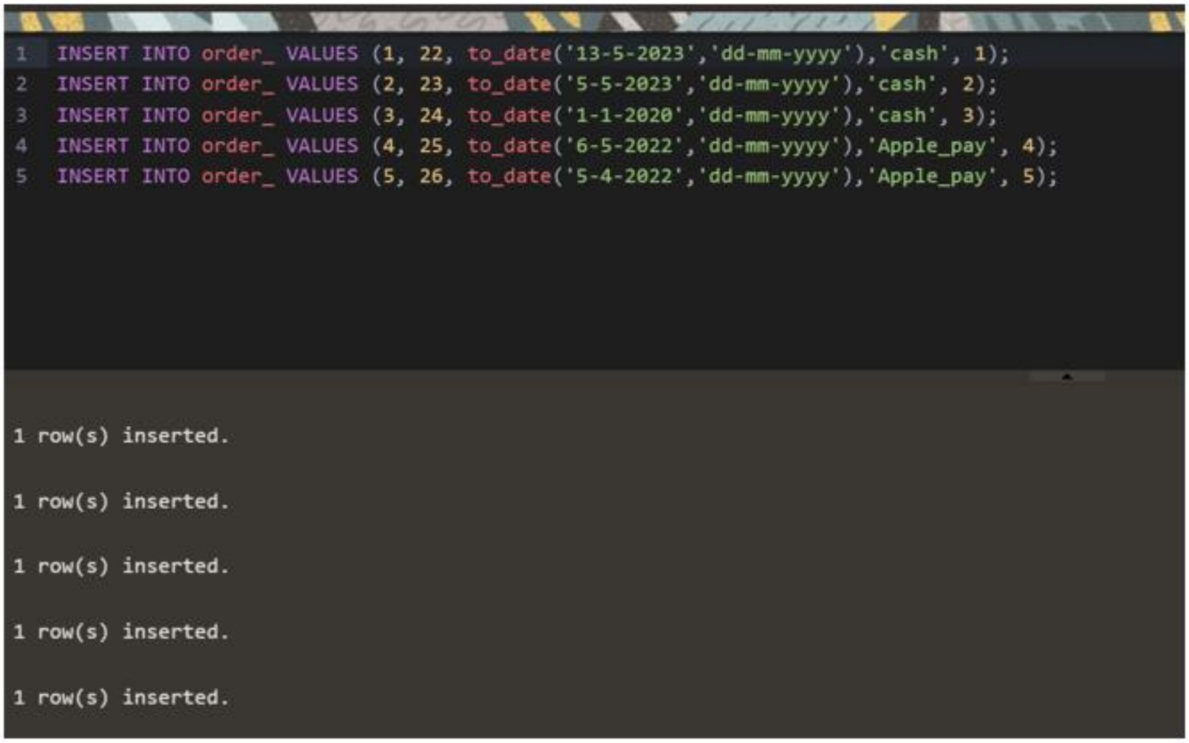
Select employee



```
select*from employee
```

EMP_ID	EMO_NAME	EMP_GENDER	EMP_SALARY	EMP_POSITION	EMP_START_DATE	EMP_PHONE	PLACEMENT_NO
20	Mohammed	male	9500	manager	01-JAN-20	528453819	1111
21	Khaled	male	5500	cashier	01-FEB-20	572946391	1112
22	Lama	female	6000	barista	02-FEB-20	519552910	1113
23	Yaser	male	6000	barista	02-FEB-20	593540278	1114
24	Amar	male	4500	clean worker	04-APR-21	555374559	1115

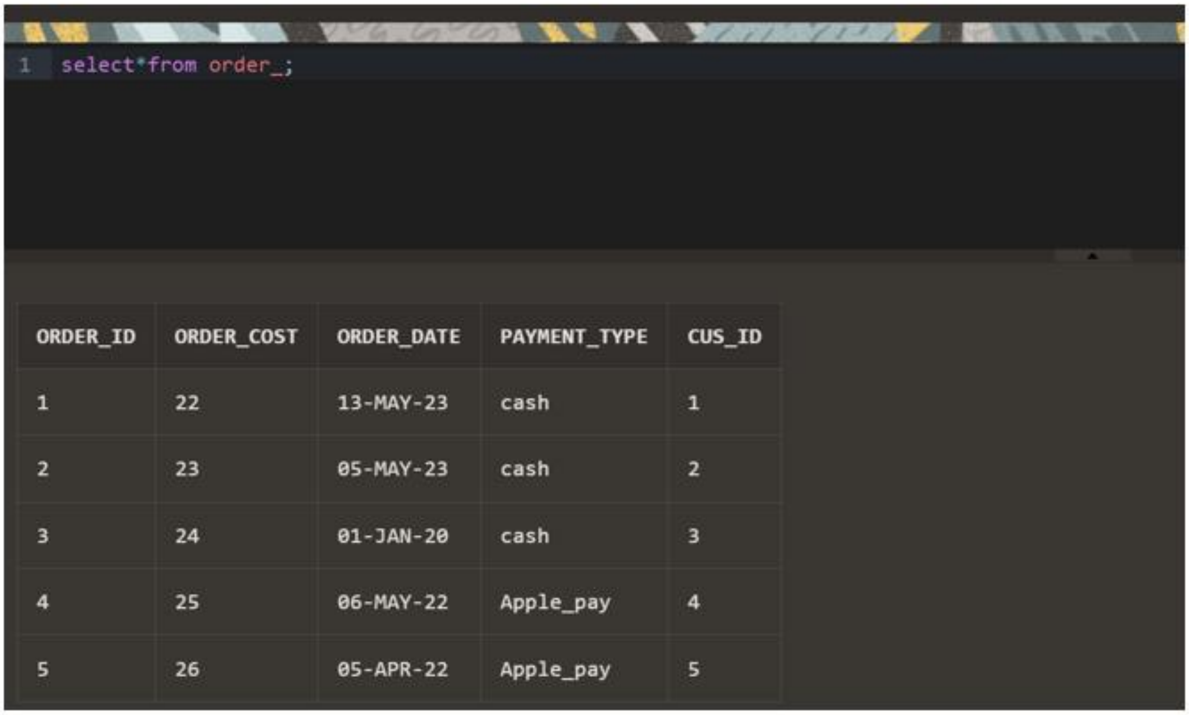
3- insert order table



```
INSERT INTO order_ VALUES (1, 22, to_date('13-5-2023','dd-mm-yyyy'),'cash', 1);
INSERT INTO order_ VALUES (2, 23, to_date('5-5-2023','dd-mm-yyyy'),'cash', 2);
INSERT INTO order_ VALUES (3, 24, to_date('1-1-2020','dd-mm-yyyy'),'cash', 3);
INSERT INTO order_ VALUES (4, 25, to_date('6-5-2022','dd-mm-yyyy'),'Apple_pay', 4);
INSERT INTO order_ VALUES (5, 26, to_date('5-4-2022','dd-mm-yyyy'),'Apple_pay', 5);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

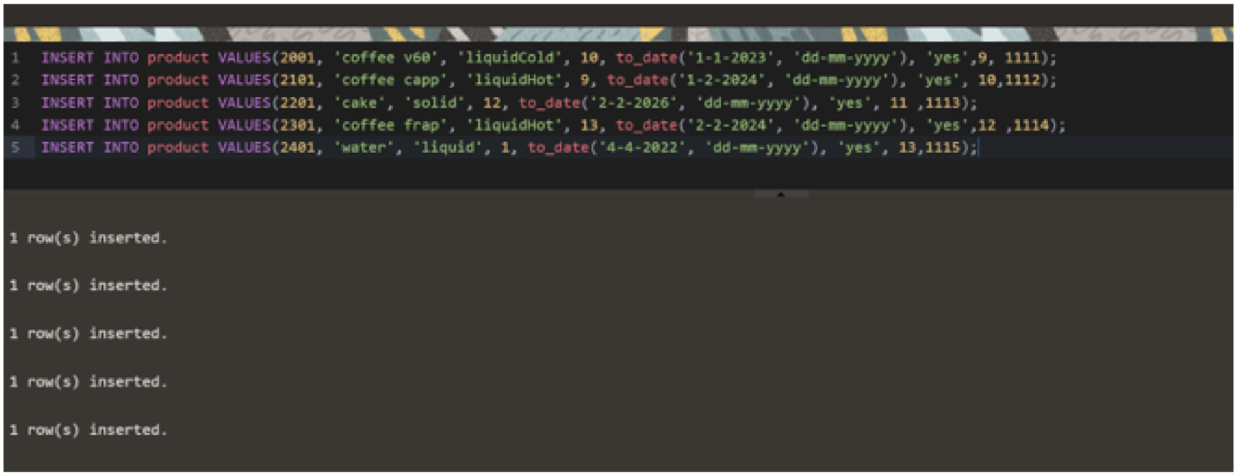
Select order



```
select*from order_;
```

ORDER_ID	ORDER_COST	ORDER_DATE	PAYMENT_TYPE	CUS_ID
1	22	13-MAY-23	cash	1
2	23	05-MAY-23	cash	2
3	24	01-JAN-20	cash	3
4	25	06-MAY-22	Apple_pay	4
5	26	05-APR-22	Apple_pay	5

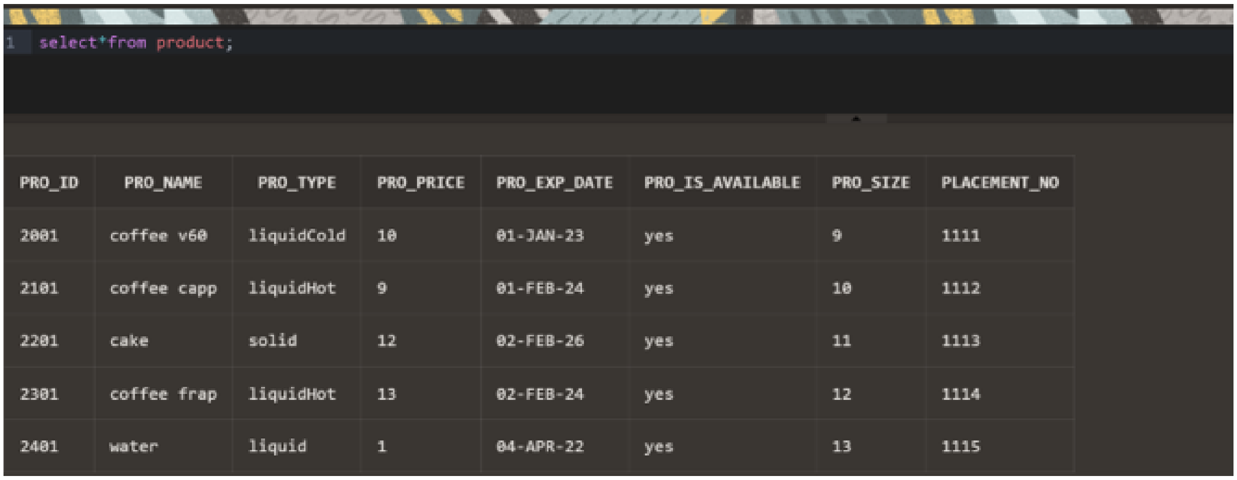
4- insert product table



```
INSERT INTO product VALUES(2001, 'coffee v60', 'liquidCold', 10, to_date('1-1-2023', 'dd-mm-yyyy'), 'yes',9, 1111);
INSERT INTO product VALUES(2101, 'coffee capp', 'liquidHot', 9, to_date('1-2-2024', 'dd-mm-yyyy'), 'yes', 10,1112);
INSERT INTO product VALUES(2201, 'cake', 'solid', 12, to_date('2-2-2026', 'dd-mm-yyyy'), 'yes', 11 ,1113);
INSERT INTO product VALUES(2301, 'coffee frap', 'liquidHot', 13, to_date('2-2-2024', 'dd-mm-yyyy'), 'yes',12 ,1114);
INSERT INTO product VALUES(2401, 'water', 'liquid', 1, to_date('4-4-2022', 'dd-mm-yyyy'), 'yes', 13,1115);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

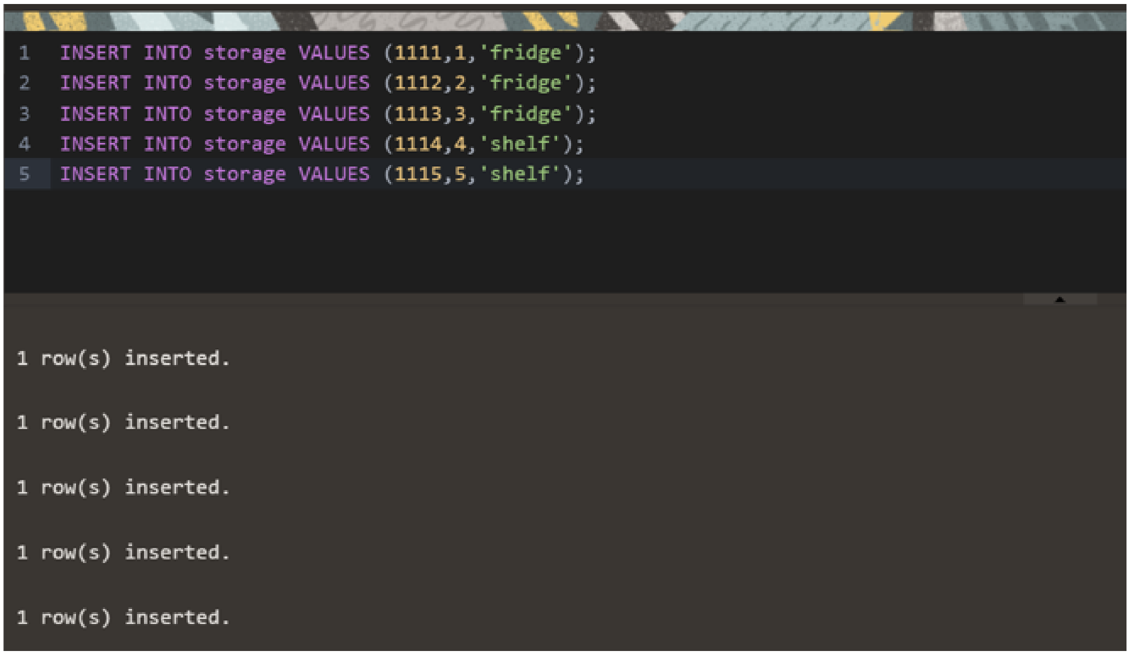
Select product



```
select*from product;
```

PRO_ID	PRO_NAME	PRO_TYPE	PRO_PRICE	PRO_EXP_DATE	PRO_IS_AVAILABLE	PRO_SIZE	PLACEMENT_NO
2001	coffee v60	liquidCold	10	01-JAN-23	yes	9	1111
2101	coffee capp	liquidHot	9	01-FEB-24	yes	10	1112
2201	cake	solid	12	02-FEB-26	yes	11	1113
2301	coffee frap	liquidHot	13	02-FEB-24	yes	12	1114
2401	water	liquid	1	04-APR-22	yes	13	1115

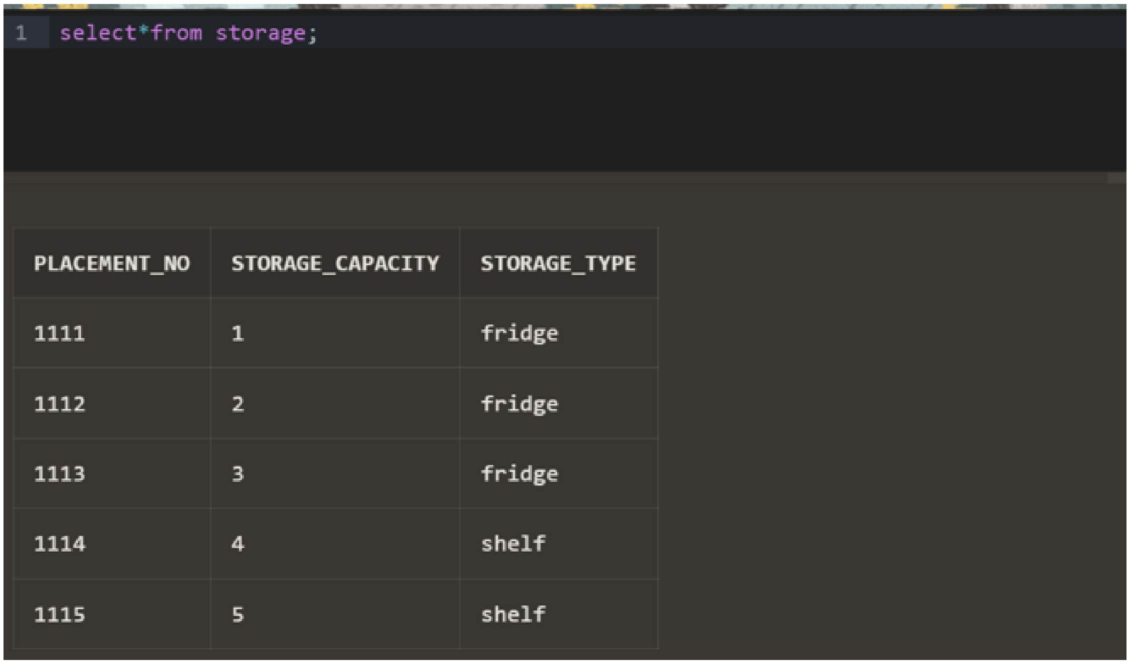
5- insert storage table



```sql
INSERT INTO storage VALUES (1111,1,'fridge');
INSERT INTO storage VALUES (1112,2,'fridge');
INSERT INTO storage VALUES (1113,3,'fridge');
INSERT INTO storage VALUES (1114,4,'shelf');
INSERT INTO storage VALUES (1115,5,'shelf');
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

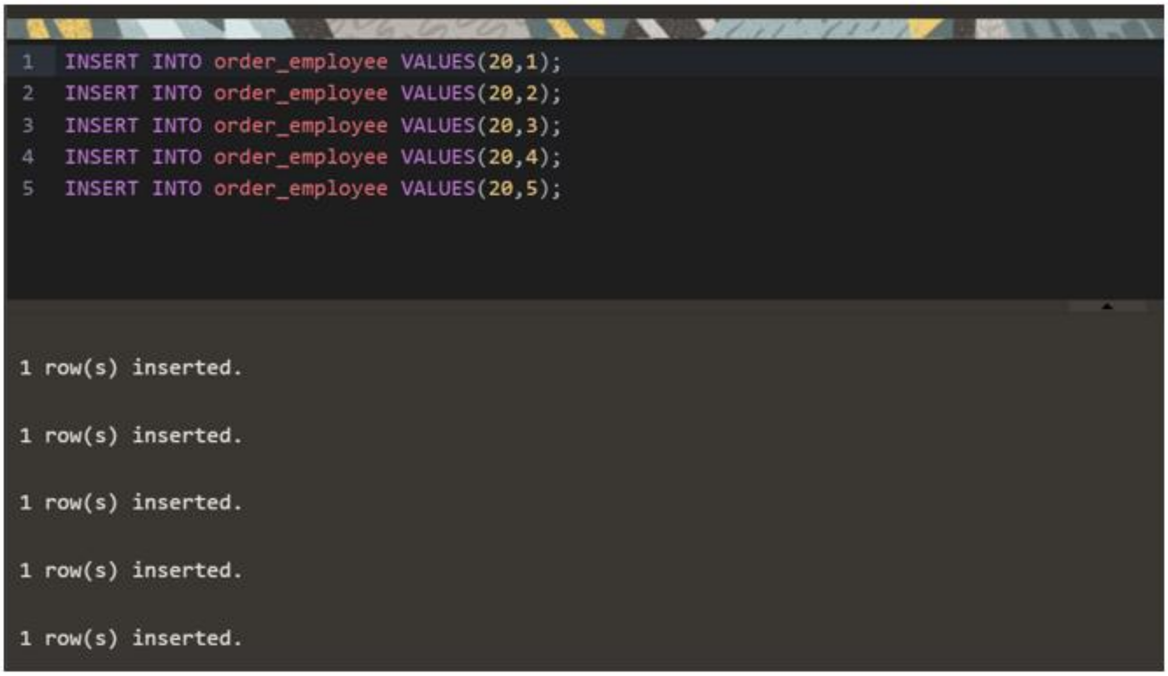
Select storage



```sql
select*from storage;
```

PLACEMENT_NO	STORAGE_CAPACITY	STORAGE_TYPE
1111	1	fridge
1112	2	fridge
1113	3	fridge
1114	4	shelf
1115	5	shelf

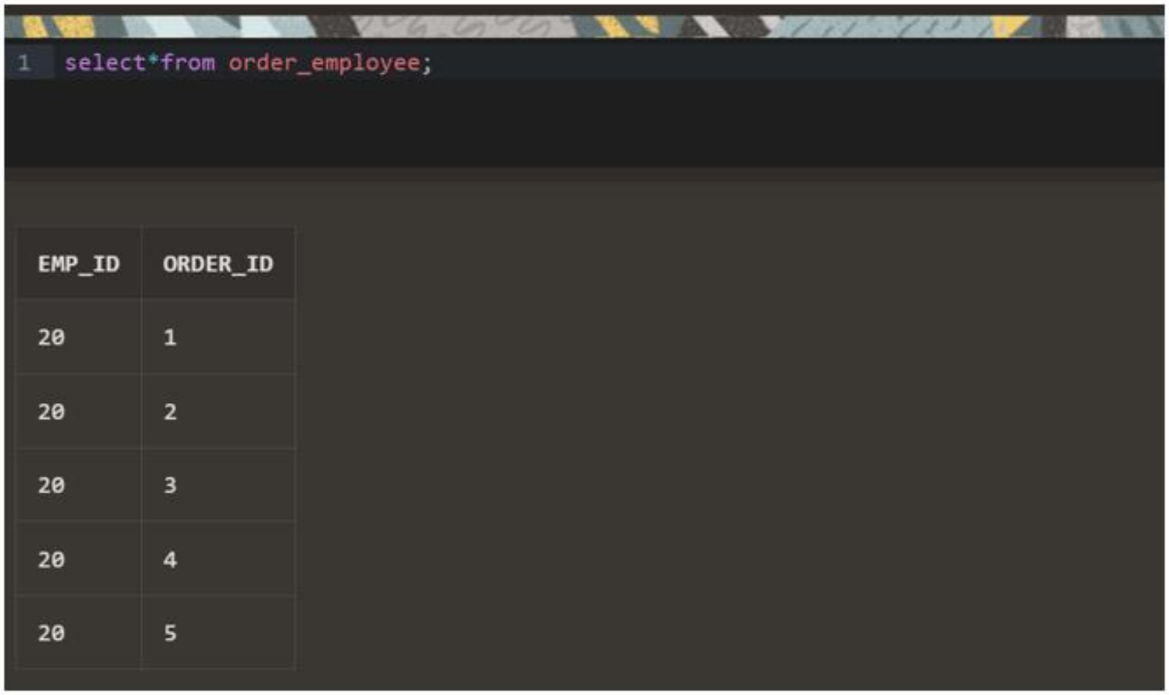
6- order_employee



```sql
INSERT INTO order_employee VALUES(20,1);
INSERT INTO order_employee VALUES(20,2);
INSERT INTO order_employee VALUES(20,3);
INSERT INTO order_employee VALUES(20,4);
INSERT INTO order_employee VALUES(20,5);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

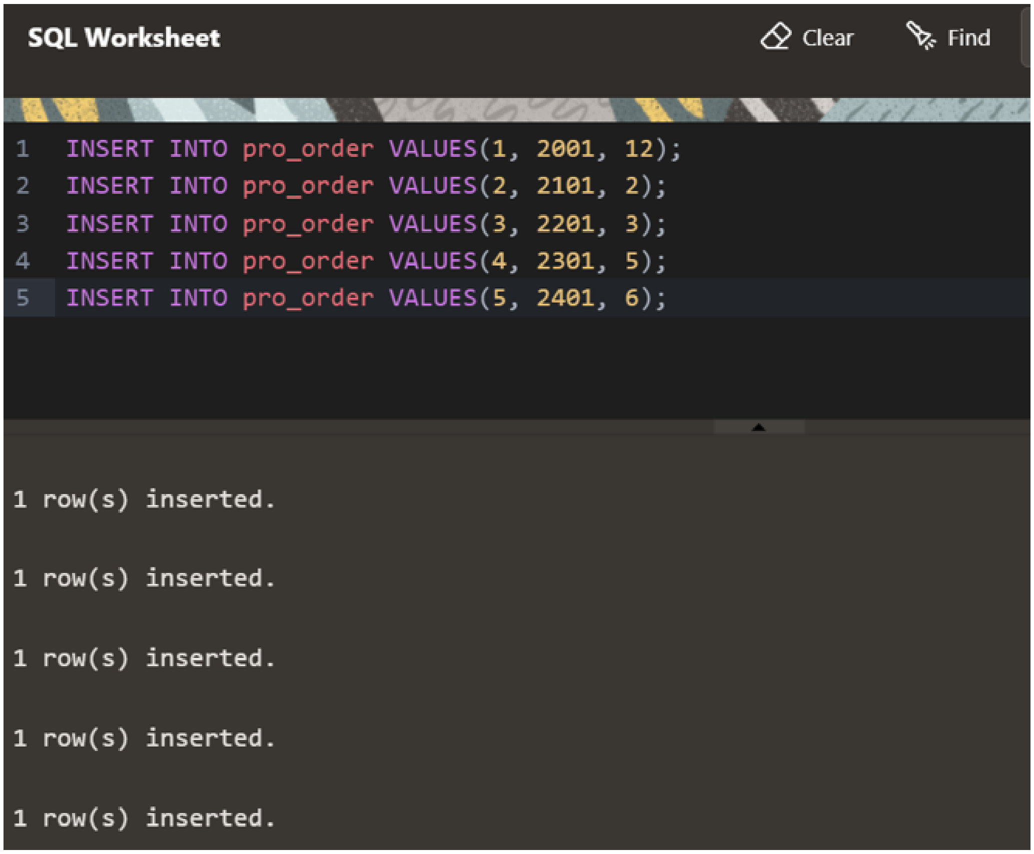
Select order_employee



```sql
select*from order_employee;
```

EMP_ID	ORDER_ID
20	1
20	2
20	3
20	4
20	5

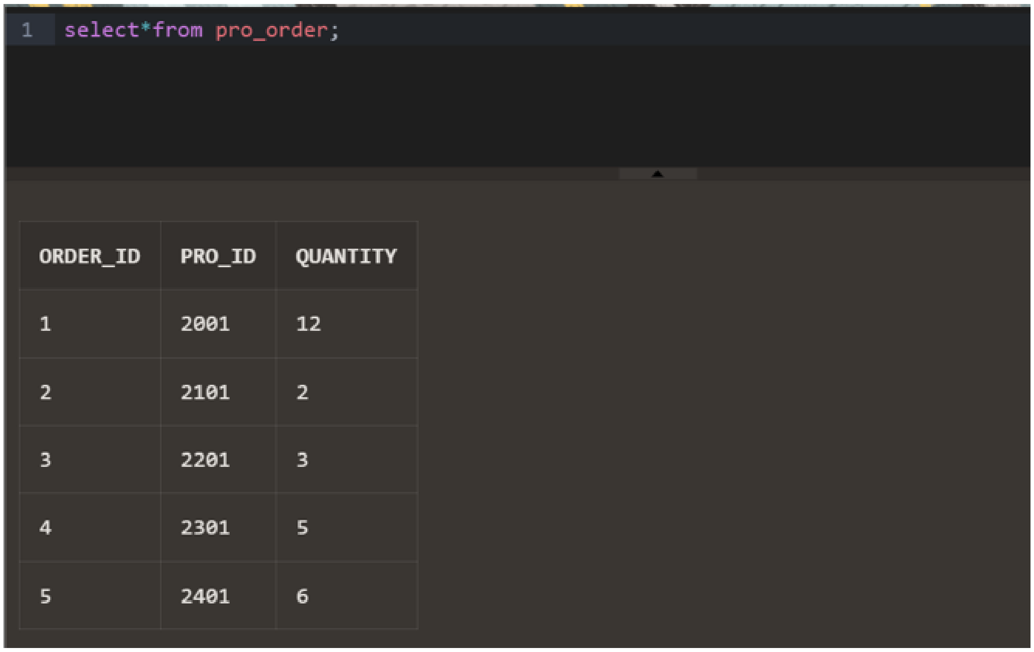
7 – insert pro_order



```sql
INSERT INTO pro_order VALUES(1, 2001, 12);
INSERT INTO pro_order VALUES(2, 2101, 2);
INSERT INTO pro_order VALUES(3, 2201, 3);
INSERT INTO pro_order VALUES(4, 2301, 5);
INSERT INTO pro_order VALUES(5, 2401, 6);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

Select pro_order

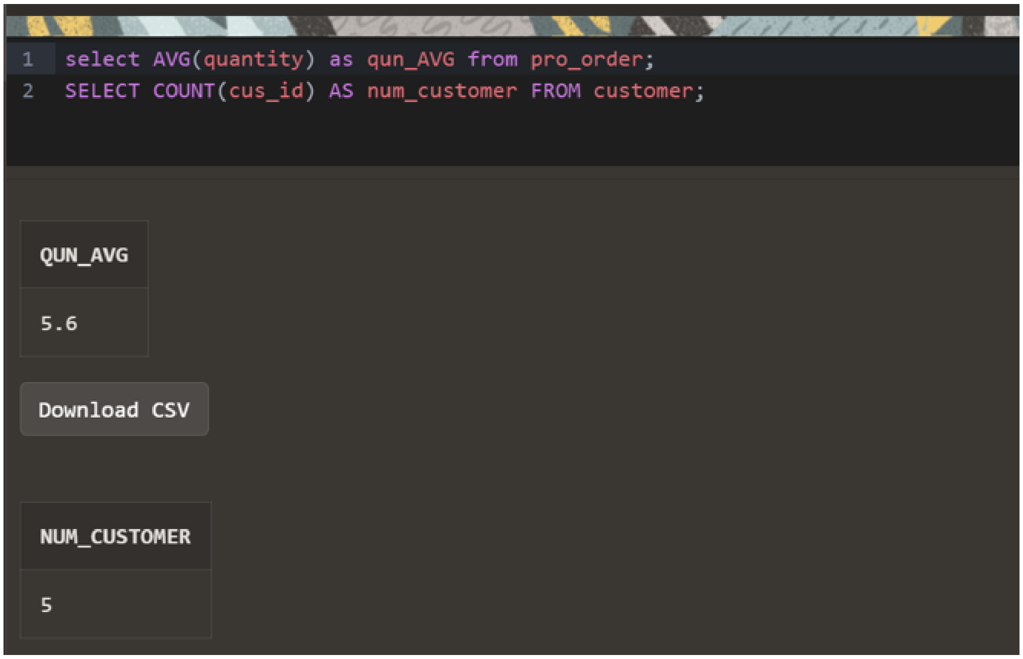


```sql
select*from pro_order;
```

ORDER_ID	PRO_ID	QUANTITY
1	2001	12
2	2101	2
3	2201	3
4	2301	5
5	2401	6

Design quertis :

1-aggregate

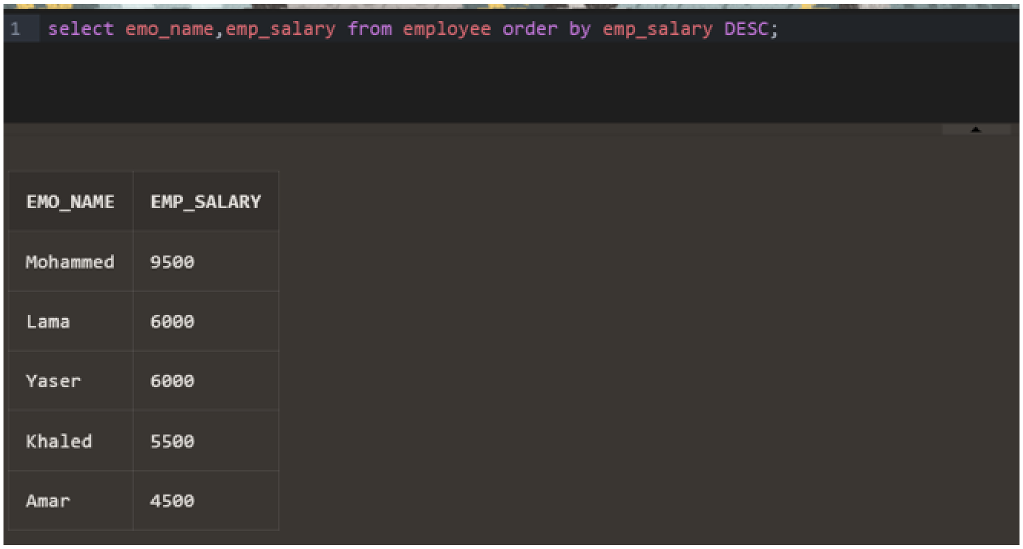


```sql
select AVG(quantity) as qun_AVG from pro_order;
SELECT COUNT(cus_id) AS num_customer FROM customer;
```

QUN_AVG
5.6

NUM_CUSTOMER
5

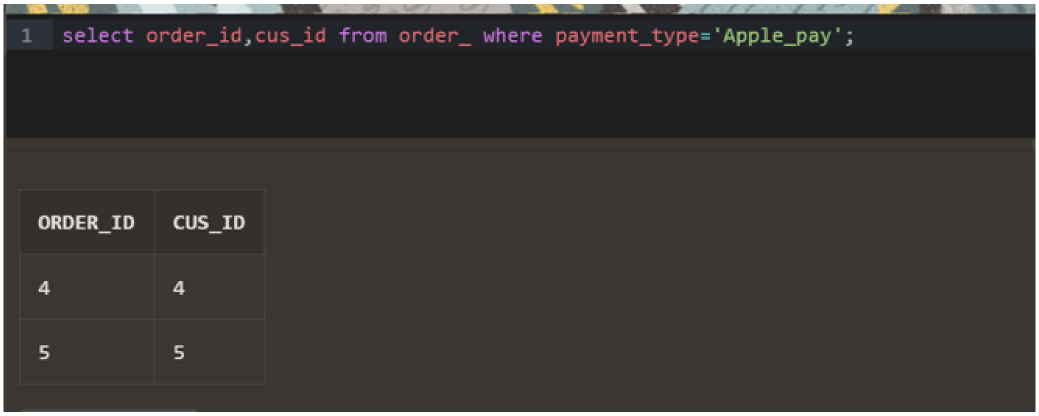
2- Order by



```sql
select emo_name,emp_salary from employee order by emp_salary DESC;
```

EMO_NAME	EMP_SALARY
Mohammed	9500
Lama	6000
Yaser	6000
Khaled	5500
Amar	4500

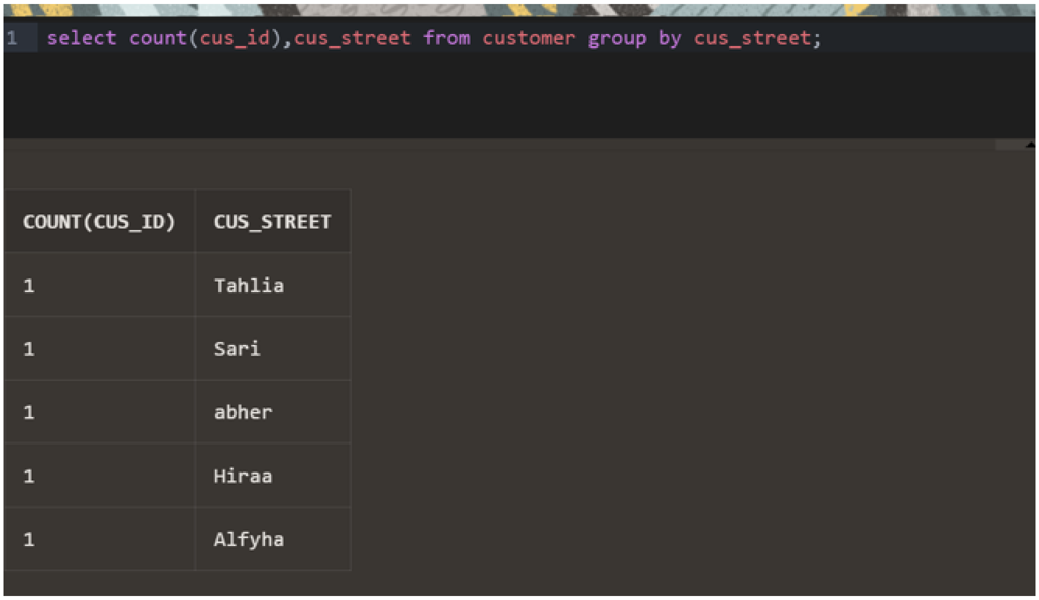
3- Select order and customer responsible fo where the payment is done by Apple_pay



```sql
select order_id,cus_id from order_ where payment_type='Apple_pay';
```

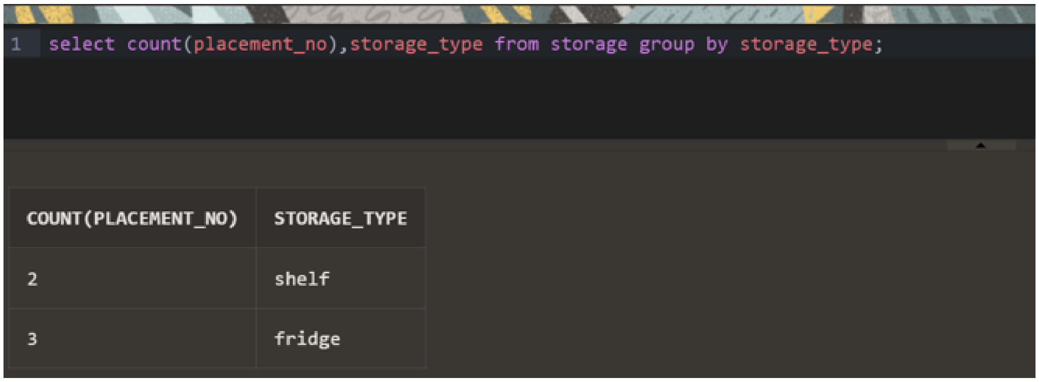
ORDER_ID	CUS_ID
4	4
5	5

4- Group by



```sql
select count(cus_id),cus_street from customer group by cus_street;
```

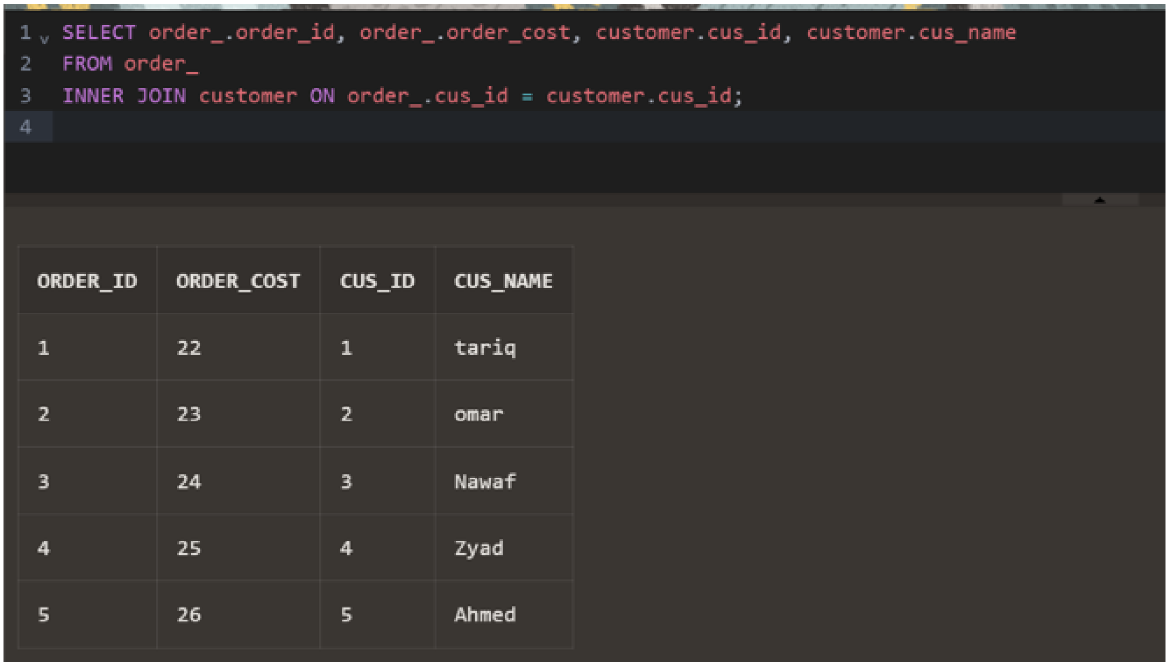
COUNT(CUS_ID)	CUS_STREET
1	Tahlia
1	Sari
1	abher
1	Hiraa
1	Alfyha



```sql
select count(placement_no),storage_type from storage group by storage_type;
```

COUNT(PLACEMENT_NO)	STORAGE_TYPE
2	shelf
3	fridge

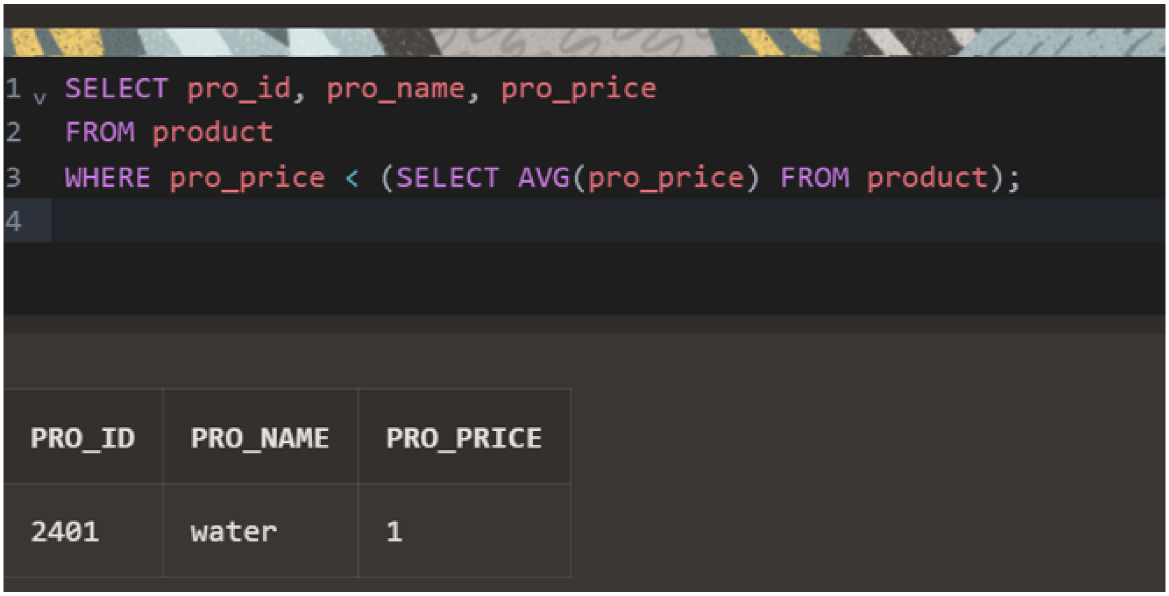
5- Join



```sql
SELECT order_.order_id, order_.order_cost, customer.cus_id, customer.cus_name
FROM order_
INNER JOIN customer ON order_.cus_id = customer.cus_id;
```

ORDER_ID	ORDER_COST	CUS_ID	CUS_NAME
1	22	1	tariq
2	23	2	omar
3	24	3	Nawaf
4	25	4	Zyad
5	26	5	Ahmed

6- sub query



```sql
SELECT pro_id, pro_name, pro_price
FROM product
WHERE pro_price < (SELECT AVG(pro_price) FROM product);
```

PRO_ID	PRO_NAME	PRO_PRICE
2401	water	1

THANK YOU!

for entrusting us with this project! We've worked diligently to fulfill every objective and ensure your satisfaction. From designing the initial ER model to converting it into a relational schema, normalizing the tables, and finally writing the SQL code, we've strived for excellence every step of the way. Our goal has been to create a robust Coffee Shop Database that meets your needs and supports the smooth operation of your business. Should you have any further questions or need additional assistance in the future, please don't hesitate to reach out. We're here to help ensure the success of your coffee shop venture!

This project was made by :

- Nawaf Fahad Al-Harbi
- Tariq Soliman Al-Gharir
- Omar Mohammed Al-Anizi
- Ziyad Naif Al-Harbi

SQL code -if needed-:

Create Table and Alter

```
CREATE TABLE customer(cus_id NUMBER(8) PRIMARY KEY,cus_name VARCHAR2 (20),cus_city VARCHAR2
;(10),cus_street VARCHAR2(10),cus_phone NUMBER (10) UNIQUE)


CREATE TABLE employee(emp_id NUMBER(8) PRIMARY KEY, emo_name VARCHAR2(20),emp_gender
VARCHAR2(6),emp_salary NUMBER(4),emp_position VARCHAR(15),emp_start_date Date

;((emp_phone NUMBER(10) UNIQUE,placement_no NUMBER(10,


ALTER TABLE employee ADD CONSTRAINT position CHECK (emp_position='barista' OR
;emp_position='cashier' OR emp_position='manager' OR emp_position='clean worker')


;ALTER TABLE  employee ADD CONSTRAINT gender CHECK (emp_gender='female'OR emp_gender='male')


CREATE TABLE storage ( placement_no NUMBER(8) PRIMARY KEY, storage_capacity NUMBER(4),
;storage_type VARCHAR2(10))


;ALTER TABLE employee ADD FOREIGN KEY (placement_no) REFERENCES storage(placement_no)


CREATE TABLE order_ (order_id NUMBER(8) PRIMARY KEY,order_cost NUMBER(4),order_date DATE
;,payment_type VARCHAR2(10),cus_id NUMBER(8))


;ALTER TABLE order_ ADD FOREIGN KEY (cus_id) REFERENCES customer (cus_id)


) CREATE TABLE product

,pro_id NUMBER(8) PRIMARY KEY
```

```
,(20)pro_name VARCHAR2

,(10)pro_type VARCHAR2

,(4)pro_price NUMBER

,pro_exp_date DATE

,(3)pro_is_available VARCHAR2

,(4)pro_size NUMBER

(8)placement_no NUMBER

;(

;ALTER TABLE product ADD FOREIGN KEY (PLACEMENT_NO) REFERENCES storage(PLACEMENT_NO)


CREATE TABLE order_employee(emp_id NUMBER(8) NOT NULL,order_id NUMBER(8) NOT NULL,PRIMARY
;KEY(emp_id,order_id))


;ALTER TABLE order_employee ADD FOREIGN KEY (order_id) REFERENCES order_(order_id)


;ALTER TABLE order_employee ADD FOREIGN KEY (emp_id) REFERENCES employee(emp_id)

CREATE TABLE pro_order(order_id NUMBER(8) NOT NULL, pro_id NUMBER(8) NOT NULL, quantity
;NUMBER(4), PRIMARY KEY(order_id,pro_id))

;ALTER TABLE pro_order ADD FOREIGN KEY (order_id) REFERENCES order_(order_id)

ALTER TABLE pro_order ADD FOREIGN KEY (pro_id) REFERENCES

; product(pro_id)
```

Insert and select

```
;INSERT into customer VALUES(1,'tariq','Jeddah','abher',0534741200)

;INSERT into customer VALUES(2,'omar','Jeddah','Tahlia',0556940467)

;INSERT into customer VALUES(3,'Nawaf','Jeddah','Hiraa',0549938293)

;INSERT into customer VALUES(4,'Zyad','Jeddah','Sari',0540007056)
```

```
;INSERT into customer VALUES(5,'Ahmed','Jeddah','Alfyha',0589923849)

;select*from customer

;INSERT INTO storage VALUES (1111,1,'fridge')

;INSERT INTO storage VALUES (1112,2,'fridge')

;INSERT INTO storage VALUES (1113,3,'fridge')

;INSERT INTO storage VALUES (1114,4,'shelf')

;INSERT INTO storage VALUES (1115,5,'shelf')

;select*from storage


INSERT INTO employee VALUES(20,'Mohammed','male',9500,'manager',to_date('1-1-2020','dd-mm-
;yyyy'),'0528453819',1111)

INSERT INTO employee VALUES(21,'Khaled','male',5500,'cashier',to_date('1-2-2020','dd-mm-
;yyyy'),'0572946391',1112)

INSERT INTO employee VALUES(22,'Lama','female',6000,'barista',to_date('2-2-2020','dd-mm-
;yyyy'),'0519552910',1113)

INSERT INTO employee VALUES(23,'Yaser','male',6000,'barista',to_date('2-2-2020','dd-mm-
;yyyy'),'0593540278',1114)

INSERT INTO employee VALUES(24,'Amar','male',4500,'clean worker',to_date('4-4-2021','dd-mm-
;yyyy'),'0555374559',1115)

;select*from employee


;INSERT INTO order_ VALUES (1, 22, to_date('13-5-2023','dd-mm-yyyy'),'cash', 1)

;INSERT INTO order_ VALUES (2, 23, to_date('5-5-2023','dd-mm-yyyy'),'cash', 2)

;INSERT INTO order_ VALUES (3, 24, to_date('1-1-2020','dd-mm-yyyy'),'cash', 3)

;INSERT INTO order_ VALUES (4, 25, to_date('6-5-2022','dd-mm-yyyy'),'Apple_pay', 4)

;INSERT INTO order_ VALUES (5, 26, to_date('5-4-2022','dd-mm-yyyy'),'Apple_pay', 5)

;_select*from order
```

```
INSERT INTO product VALUES(2001, 'coffee v60', 'liquidCold', 10, to_date('1-1-2023', 'dd-mm-yyyy'), 'yes',9,
;1111)

INSERT INTO product VALUES(2101, 'coffee capp', 'liquidHot', 9, to_date('1-2-2024', 'dd-mm-yyyy'), 'yes',
;10,1112)

;INSERT INTO product VALUES(2201, 'cake', 'solid', 12, to_date('2-2-2026', 'dd-mm-yyyy'), 'yes', 11 ,1113)

INSERT INTO product VALUES(2301, 'coffee frap', 'liquidHot', 13, to_date('2-2-2024', 'dd-mm-yyyy'), 'yes',12
;,1114)

;INSERT INTO product VALUES(2401, 'water', 'liquid', 1, to_date('4-4-2022', 'dd-mm-yyyy'), 'yes', 13,1115)

;select*from product


;(20,1)INSERT INTO order_employee VALUES

;(20,2)INSERT INTO order_employee VALUES

;(20,3)INSERT INTO order_employee VALUES

;(20,4)INSERT INTO order_employee VALUES

;(20,5)INSERT INTO order_employee VALUES

;select*from order_employee

;(12 ,2001 ,1)INSERT INTO pro_order VALUES

;(2 ,2101 ,2)INSERT INTO pro_order VALUES

;(3 ,2201 ,3)INSERT INTO pro_order VALUES

;(5 ,2301 ,4)INSERT INTO pro_order VALUES

;(6 ,2401 ,5)INSERT INTO pro_order VALUES

;select*from pro_order

;select AVG(quantity) as gun_AVG from pro_order


;SELECT COUNT(cus_id) AS num_customer FROM customer

;select emo_name,emp_salary from employee order by emp_salary DESC

;'select order_id,cus_id from order_ where payment_type='Apple_pay

;select count(cus_id),cus_street from customer group by cus_street
```

```
select count(placement_no),storage_type from storage group by storage_type

;STORAGE_TYPE            COUNT(PLACEMENT_NO)

SELECT order_.order_id, order_.order_cost, customer.cus_id, customer.cus_name

_FROM order

;INNER JOIN customer ON order_.cus_id = customer.cus_id

SELECT pro_id, pro_name, pro_price

FROM product

WHERE pro_price < (SELECT AVG(pro_price) FROM product)
```