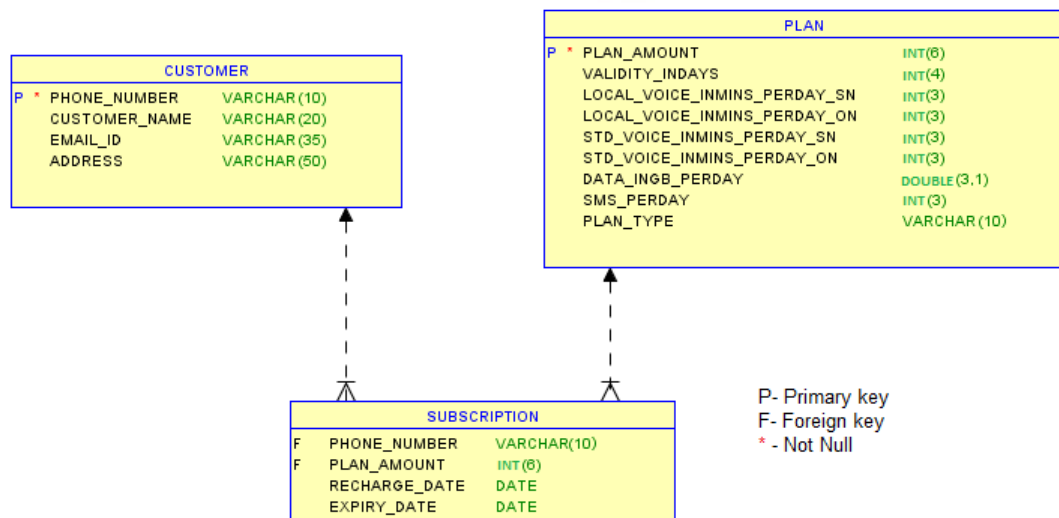1.Write a query to display the details like the customer name, phone number, plan amount, and recharge date of all the customers whose plan expired in January, 2000.Sort the results based on the phone number in ascending order.

## Mobile Prepaid System

| CUSTOMER | | |
|---|---|---|
| P * PHONE_NUMBER | VARCHAR (10) | |
| CUSTOMER_NAME | VARCHAR (20) | |
| EMAIL_ID | VARCHAR (35) | |
| ADDRESS | VARCHAR (50) | |

| PLAN | | |
|---|---|---|
| P * PLAN_AMOUNT | INT(6) | |
| VALIDITY_INDAYS | INT(4) | |
| LOCAL_VOICE_INMINS_PERDAY_SN | INT(3) | |
| LOCAL_VOICE_INMINS_PERDAY_ON | INT(3) | |
| STD_VOICE_INMINS_PERDAY_SN | INT(3) | |
| STD_VOICE_INMINS_PERDAY_ON | INT(3) | |
| DATA_INGB_PERDAY | DOUBLE (3.1) | |
| SMS_PERDAY | INT (3) | |
| PLAN_TYPE | VARCHAR (10) | |

| SUBSCRIPTION | | |
|---|---|---|
| F PHONE_NUMBER | VARCHAR(10) | |
| F PLAN_AMOUNT | INT (6) | |
| RECHARGE_DATE | DATE | |
| EXPIRY_DATE | DATE | |

P- Primary key
F- Foreign key
* - Not Null

2. Write a query to display the patient id, first name, city and contact number of the patients who took more than a day to pay their bill.

Sort the results based on the bill number in descending order.
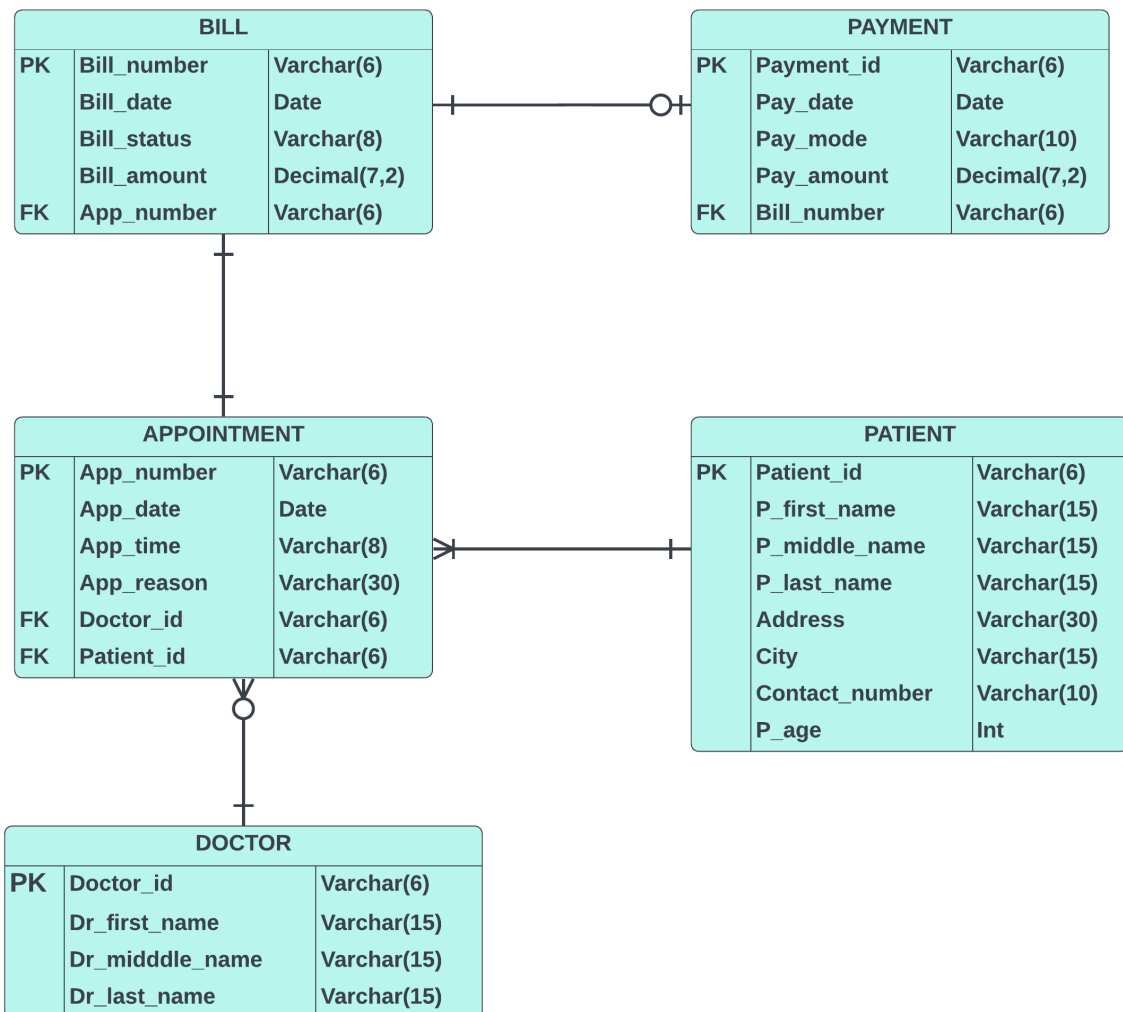(Hint: Use bill date, pay date.)
**Sample Data**

| Patient_id | p_first_name | city | bill_date | pay_date |
|---|---|---|---|---|
| P1 | David | Miami | 12-10-2022 | 12-10-2022 |
| P2 | Peter | Boston | 08-11-2022 | 09-11-2022 |
| P3 | Michael | Washington | 21-12-2022 | 25-10-2022 |

**Sample Output:**

| Patient_id | p_first_name | city |
|---|---|---|
| P3 | Michael | Washington |

| BILL | | |
|---|---|---|
| PK | Bill_number | Varchar(6) |
| | Bill_date | Date |
| | Bill_status | Varchar(8) |
| | Bill_amount | Decimal(7,2) |
| FK | App_number | Varchar(6) |

| PAYMENT | | |
|---|---|---|
| PK | Payment_id | Varchar(6) |
| | Pay_date | Date |
| | Pay_mode | Varchar(10) |
| | Pay_amount | Decimal(7,2) |
| FK | Bill_number | Varchar(6) |

| APPOINTMENT | | |
|---|---|---|
| PK | App_number | Varchar(6) |
| | App_date | Date |
| | App_time | Varchar(8) |
| | App_reason | Varchar(30) |
| FK | Doctor_id | Varchar(6) |
| FK | Patient_id | Varchar(6) |

| PATIENT | | |
|---|---|---|
| PK | Patient_id | Varchar(6) |
| | P_first_name | Varchar(15) |
| | P_middle_name | Varchar(15) |
| | P_last_name | Varchar(15) |
| | Address | Varchar(30) |
| | City | Varchar(15) |
| | Contact_number | Varchar(10) |
| | P_age | Int |

| DOCTOR | | |
|---|---|---|
| PK | Doctor_id | Varchar(6) |
| | Dr_first_name | Varchar(15) |
| | Dr_midddle_name | Varchar(15) |
| | Dr_last_name | Varchar(15) |

3.Write a query to display the policy details such as policy id, name, minimum premium amount and the new premium amount of all the policies.

The new premium amount is calculated based on the following conditions:

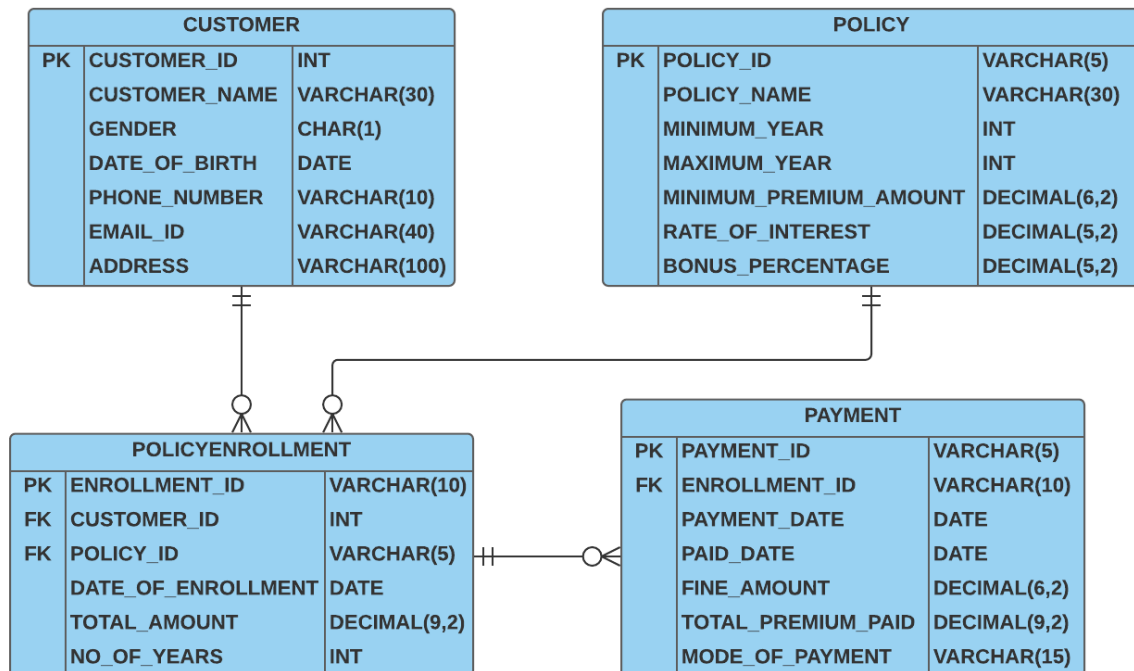If the bonus is less than 50 then increase 2000 to the minimum premium amount.

If the bonus is greater than or equal to 50 and less than 75 then increase 1000 to the minimum premium amount.

If the bonus is greater than or equal to 75 percent then increase 500 to the minimum premium amount.

Give an alias name for calculated amount as 'NEW_PREMIUM_AMOUNT'.

Sort the result set based on policy name in descending order.

| CUSTOMER | | |
|---|---|---|
| PK | CUSTOMER_ID | INT |
| | CUSTOMER_NAME | VARCHAR(30) |
| | GENDER | CHAR(1) |
| | DATE_OF_BIRTH | DATE |
| | PHONE_NUMBER | VARCHAR(10) |
| | EMAIL_ID | VARCHAR(40) |
| | ADDRESS | VARCHAR(100) |

| POLICY | | |
|---|---|---|
| PK | POLICY_ID | VARCHAR(5) |
| | POLICY_NAME | VARCHAR(30) |
| | MINIMUM_YEAR | INT |
| | MAXIMUM_YEAR | INT |
| | MINIMUM_PREMIUM_AMOUNT | DECIMAL(6,2) |
| | RATE_OF_INTEREST | DECIMAL(5,2) |
| | BONUS_PERCENTAGE | DECIMAL(5,2) |

| POLICYENROLLMENT | | |
|---|---|---|
| PK | ENROLLMENT_ID | VARCHAR(10) |
| FK | CUSTOMER_ID | INT |
| FK | POLICY_ID | VARCHAR(5) |
| | DATE_OF_ENROLLMENT | DATE |
| | TOTAL_AMOUNT | DECIMAL(9,2) |
| | NO_OF_YEARS | INT |

| PAYMENT | | |
|---|---|---|
| PK | PAYMENT_ID | VARCHAR(5) |
| FK | ENROLLMENT_ID | VARCHAR(10) |
| | PAYMENT_DATE | DATE |
| | PAID_DATE | DATE |
| | FINE_AMOUNT | DECIMAL(6,2) |
| | TOTAL_PREMIUM_PAID | DECIMAL(9,2) |
| | MODE_OF_PAYMENT | VARCHAR(15) |

4. Write a query to display the subscription information of all the customers who haven't subscribed. Give an alias name as "SUBSCRIPTION_INFO".

Sort the records based on the SUBSCRIPTION_INFO in ascending order.

**Sample Data:**

**SUBSCRIPTION_INFO**

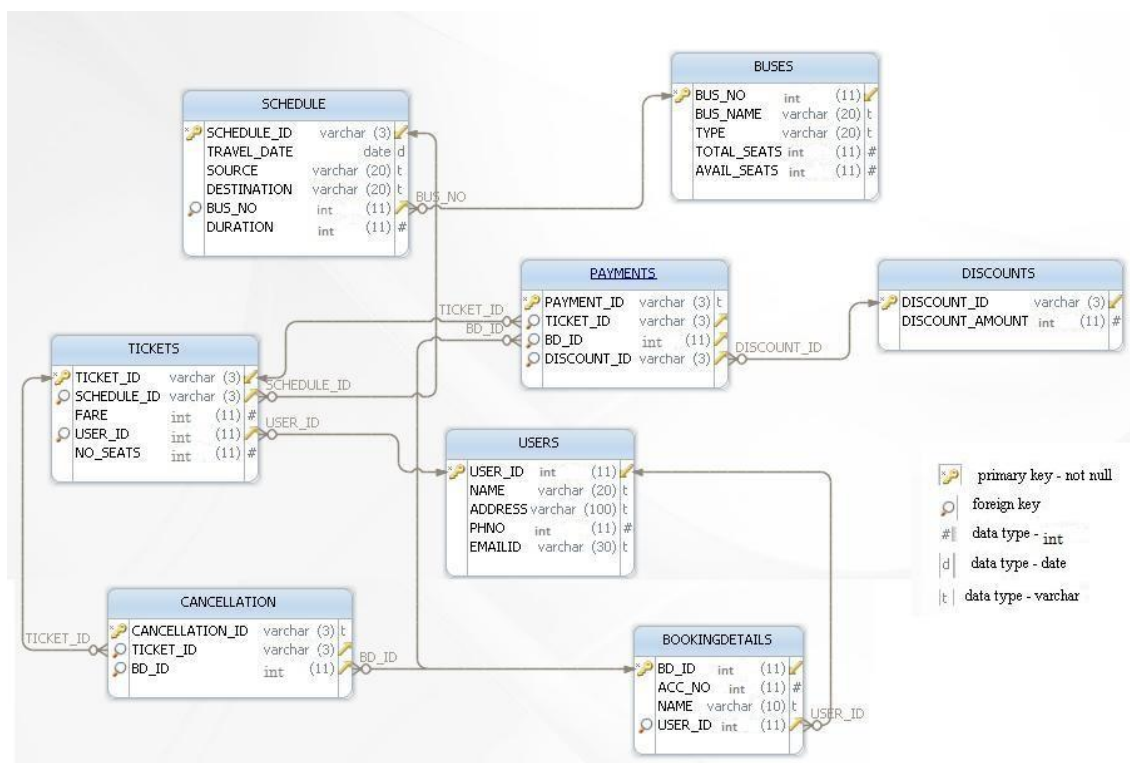**CUSTOMER_NAME** hasn't subscribed yet

**Sample Output:**

**SUBSCRIPTION_INFO**

James Jackson hasn't subscribed yet

5. Write a query to display the name and number of tickets booked by the users whose phone number is unavailable. Give an alias name as 'No_Of_Tickets' for the number of tickets booked by the user.
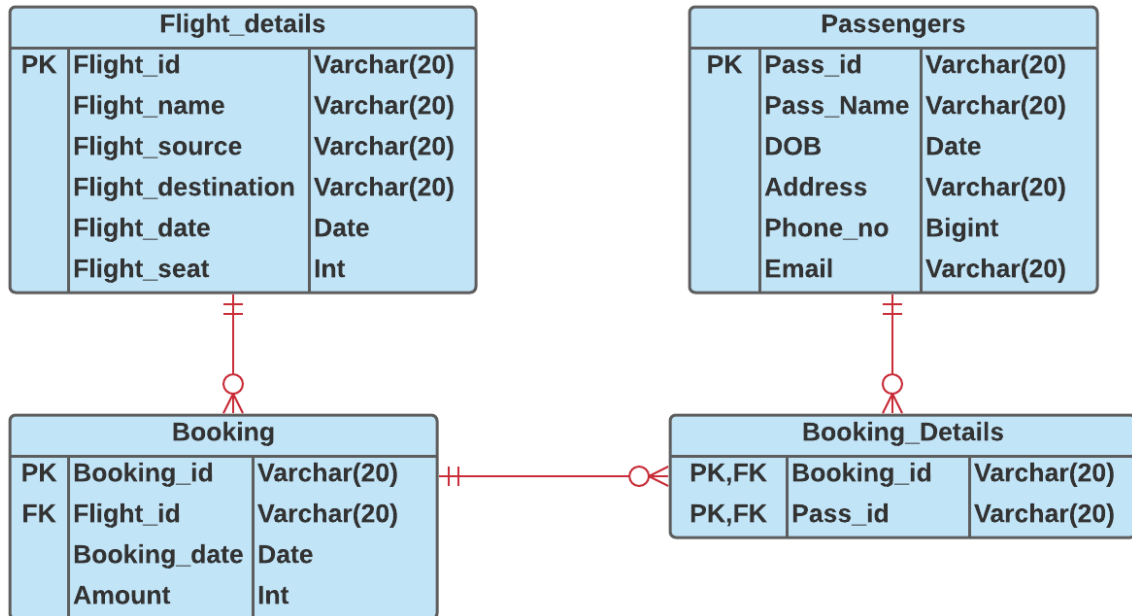
Sort the results based on the user name in descending order.

6. Write a query that displays the id, name, source, and destination of all flights that are not booked by any passengers.

Sort the results in descending order based on the flight id.

## FLIGHT MANAGEMENT SYSTEM

**Flight_details**

| PK | Flight_id | Varchar(20) |
|----|-----------|-------------|
| | Flight_name | Varchar(20) |
| | Flight_source | Varchar(20) |
| | Flight_destination | Varchar(20) |
| | Flight_date | Date |
| | Flight_seat | Int |

**Passengers**

| PK | Pass_id | Varchar(20) |
|----|---------|-------------|
| | Pass_Name | Varchar(20) |
| | DOB | Date |
| | Address | Varchar(20) |
| | Phone_no | Bigint |
| | Email | Varchar(20) |

**Booking**

| PK | Booking_id | Varchar(20) |
|----|------------|-------------|
| FK | Flight_id | Varchar(20) |
| | Booking_date | Date |
| | Amount | Int |

**Booking_Details**

| PK,FK | Booking_id | Varchar(20) |
|-------|------------|-------------|
| PK,FK | Pass_id | Varchar(20) |

7. Write a query to display the Customer's name and street of all the customers who have email. Give an alias name for the street as STREET.

Sort the records based on the STREET in descending order.

**Sample Data:**

| PHONE_NUMBER | CUSTOMER_NAME | EMAIL_ID | ADDRESS |
|--------------|---------------|----------|---------|
| 9952882017 | Will Smith | will002@gmail.com | Lombard Street,San Diego |

**Sample Output:**

| CUSTOMER_NAME | STREET |
|---------------|--------|
| Will Smith | Lombard Street |

# Mobile Prepaid System

**CUSTOMER**

| | | |
|---|---|---|
| P * | PHONE_NUMBER | VARCHAR (10) |
| | CUSTOMER_NAME | VARCHAR (20) |
| | EMAIL_ID | VARCHAR (35) |
| | ADDRESS | VARCHAR (50) |

**PLAN**

| | | |
|---|---|---|
| P * | PLAN_AMOUNT | INT(6) |
| | VALIDITY_INDAYS | INT(4) |
| | LOCAL_VOICE_INMINS_PERDAY_SN | INT(3) |
| | LOCAL_VOICE_INMINS_PERDAY_ON | INT(3) |
| | STD_VOICE_INMINS_PERDAY_SN | INT(3) |
| | STD_VOICE_INMINS_PERDAY_ON | INT(3) |
| | DATA_INGB_PERDAY | DOUBLE (3.1) |
| | SMS_PERDAY | INT (3) |
| | PLAN_TYPE | VARCHAR (10) |

**SUBSCRIPTION**

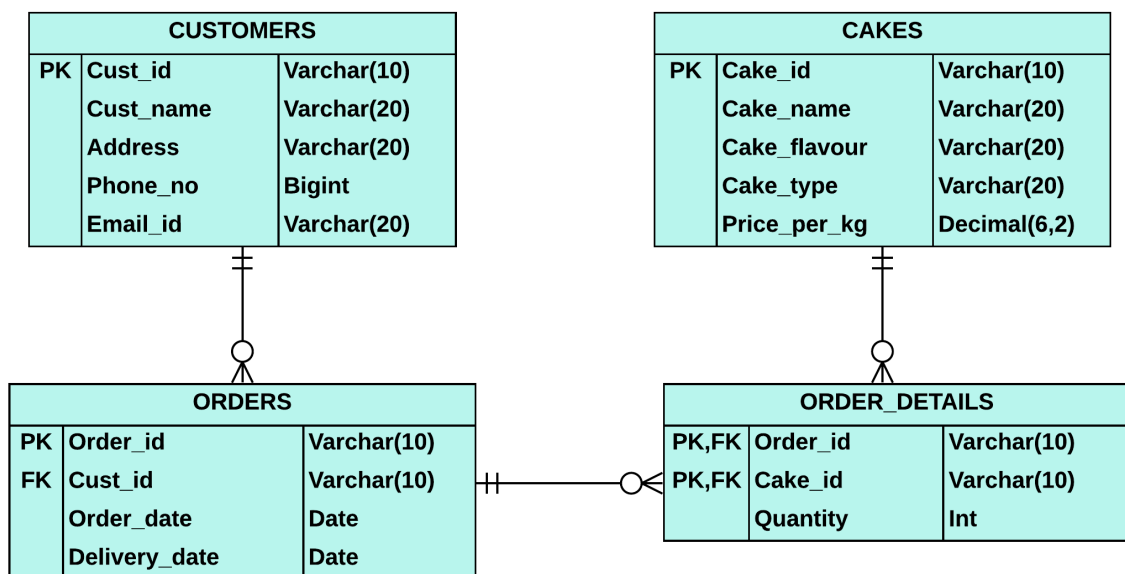| | | |
|---|---|---|
| F | PHONE_NUMBER | VARCHAR(10) |
| F | PLAN_AMOUNT | INT (6) |
| | RECHARGE_DATE | DATE |
| | EXPIRY_DATE | DATE |

P- Primary key
F- Foreign key
* - Not Null

8. Write a query to display the cake id, cake name, cake type and price per kg of the cakes that are not ordered.

Sort the results based on the cake name in ascending order.

## "BAKE  OFF"- CAKE ORDERING SYSTEM

**CUSTOMERS**

| | | |
|---|---|---|
| PK | Cust_id | Varchar(10) |
| | Cust_name | Varchar(20) |
| | Address | Varchar(20) |
| | Phone_no | Bigint |
| | Email_id | Varchar(20) |

**CAKES**

| | | |
|---|---|---|
| PK | Cake_id | Varchar(10) |
| | Cake_name | Varchar(20) |
| | Cake_flavour | Varchar(20) |
| | Cake_type | Varchar(20) |
| | Price_per_kg | Decimal(6,2) |

**ORDERS**

| | | |
|---|---|---|
| PK | Order_id | Varchar(10) |
| FK | Cust_id | Varchar(10) |
| | Order_date | Date |
| | Delivery_date | Date |

**ORDER_DETAILS**

| | | |
|---|---|---|
| PK,FK | Order_id | Varchar(10) |
| PK,FK | Cake_id | Varchar(10) |
| | Quantity | Int |

9. Write a query to display the unique name and experience of all the sailors that rode the boats. Display the experience of the sailor with no decimal value.

Give an alias name for the experience of the sailor as EXPERIENCE.

Sort the records based on the sailor's name in descending order.

(Hint: Use round().)

**"SPLASH RIDE" BOAT MANAGEMENT SYSTEM**

| Sailor_Details | | |
|---|---|---|
| PK | Sailor_id | varchar2(20) |
| | Sailor_name | varchar2(20) |
| | Phone_no | number |
| | Address | varchar2(20) |
| | DOJ | Date |

| Boat_details | | |
|---|---|---|
| PK | Boat_id | varchar2(20) |
| | Boat_name | varchar2(20) |
| | Boat_type | varchar2(20) |
| | Seat_capacity | number |

| Ride_Details | | |
|---|---|---|
| PK | Ride_id | varchar2(20) |
| FK | Sailor_id | varchar2(20) |
| FK | Boat_id | varchar2(20) |
| | DOT | DATE |
| | Shift | varchar2(20) |
| | No_passengers | number |