INTRODUCTION TO DATABASE [E]

TOPIC: COURSE PROJECT SUBMISSION

Luxury Resort Management System

SUBMITTED TO: **MS. JUENA AHMED NOSHIN**
FACULTY, CS, AIUB.

Course: Advance Database Management System

Section: E

SUBMITTED BY: **GROUP # 8**

Submission Date: **12th march, 2023.**

Name	ID	Contribution
Ahmed Farhan Amin	21-44804-1	Introduction, Cover Page, User Interface, Data Insertion, Query writing, Conclusion
MUHAMMAD AKIB - AL - ISLAM	20-42289-1	ER Diagram, Schema diagram
Maisha Shams	20-43359-1	Contents, Table Creation, Normalizations, sequence, index, roles
Fabiha Alam	20-44096-2	Scenario Description, Relational Algebra, Use-case Diagram, Class Diagram

Contents **Page**

Introduction:

Our project is to create a luxury resort management system, which will streamline the resort management process, provide excellent customer service, and enhance the overall guest experience. The system will be built using C# .NET framework and Oracle 10g database, which will ensure optimal performance and scalability.

Overview:

The luxury resort management system will consist of several modules that work together to manage different aspects of the resort's operations. Some of the main modules include reservation management, guest check-in and check-out, room and inventory management, billing and payment processing, and reporting.

The reservation management module will allow guests to book their stay online, and resort staff will be able to manage reservations and room assignments. The guest check-in and check-out module will streamline the check-in and check-out process, reducing wait times and improving the guest experience.

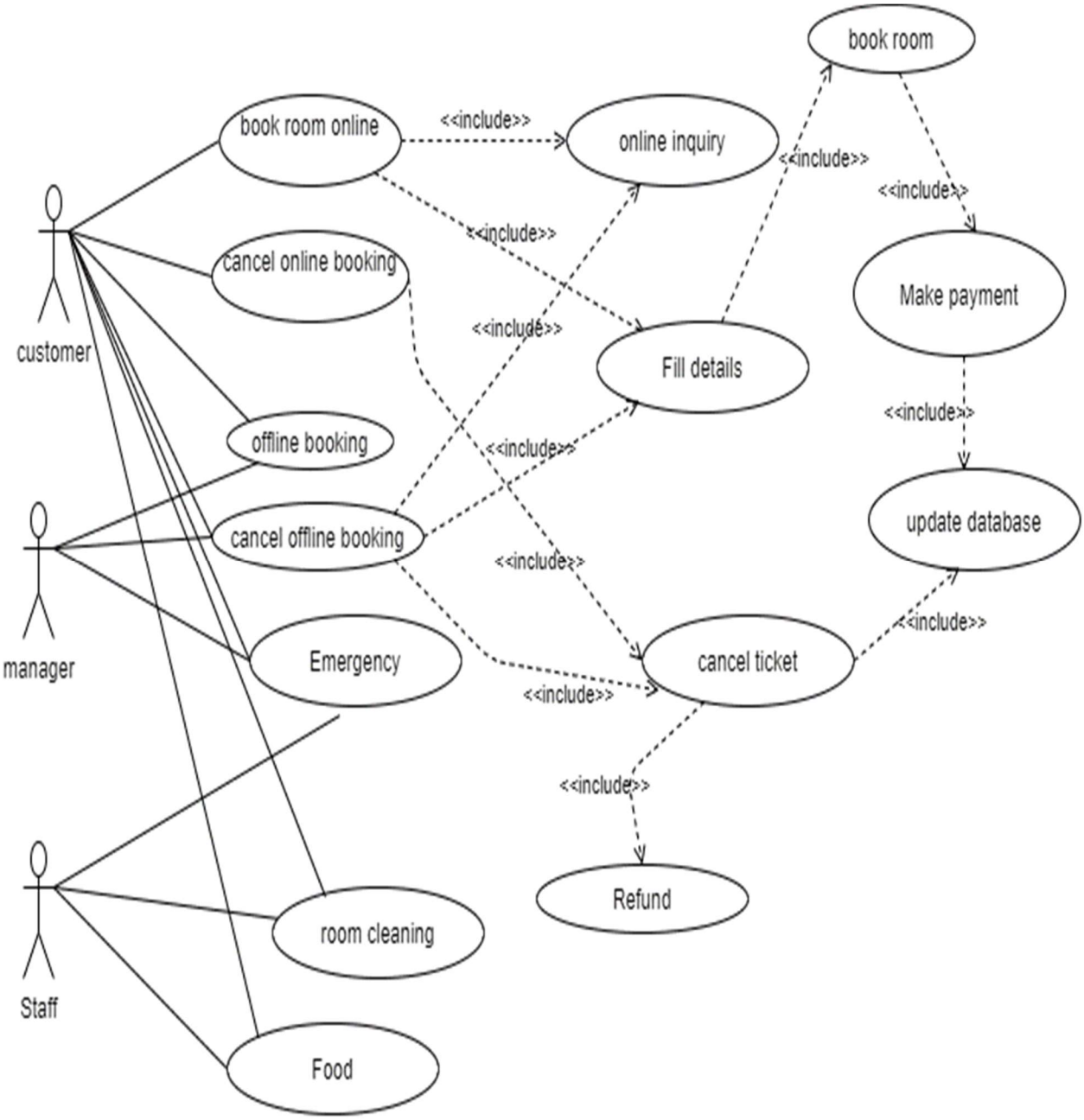
The room and inventory management module will allow staff to manage room availability, inventory, and maintenance schedules, ensuring that all rooms are clean, well-maintained, and ready for guest occupancy.

The billing and payment processing module will allow staff to manage guest billing, payment processing, and refunds, reducing errors and improving financial management.

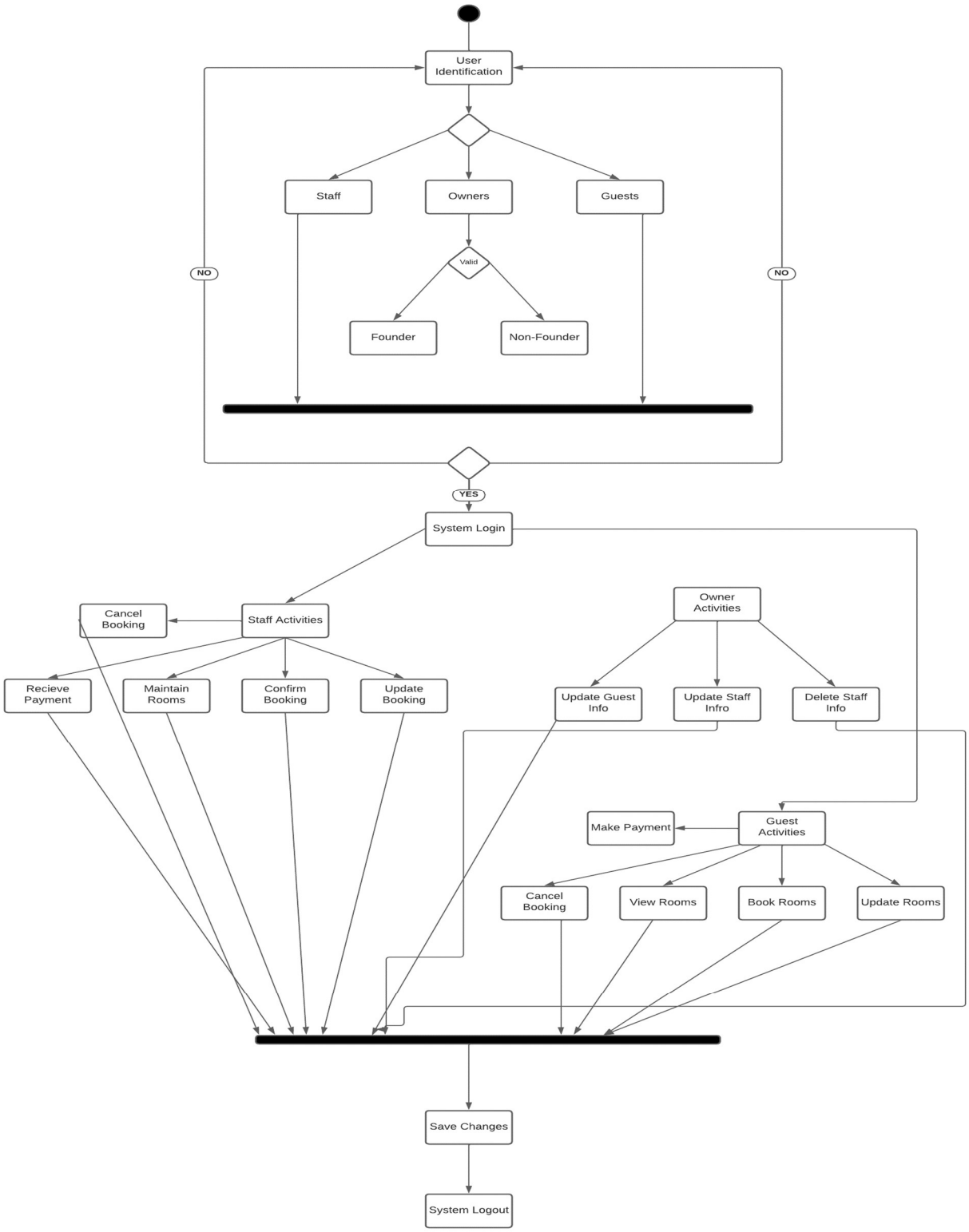
Finally, the reporting module will provide resort management with detailed reports on occupancy rates, revenue, and other key metrics, allowing them to make informed business decisions.

Overall, our luxury resort management system will improve the guest experience, streamline operations, and improve financial management. By leveraging the power of C# .NET framework and Oracle 10g database, we can create a robust and scalable system that meets the needs of any luxury resort.

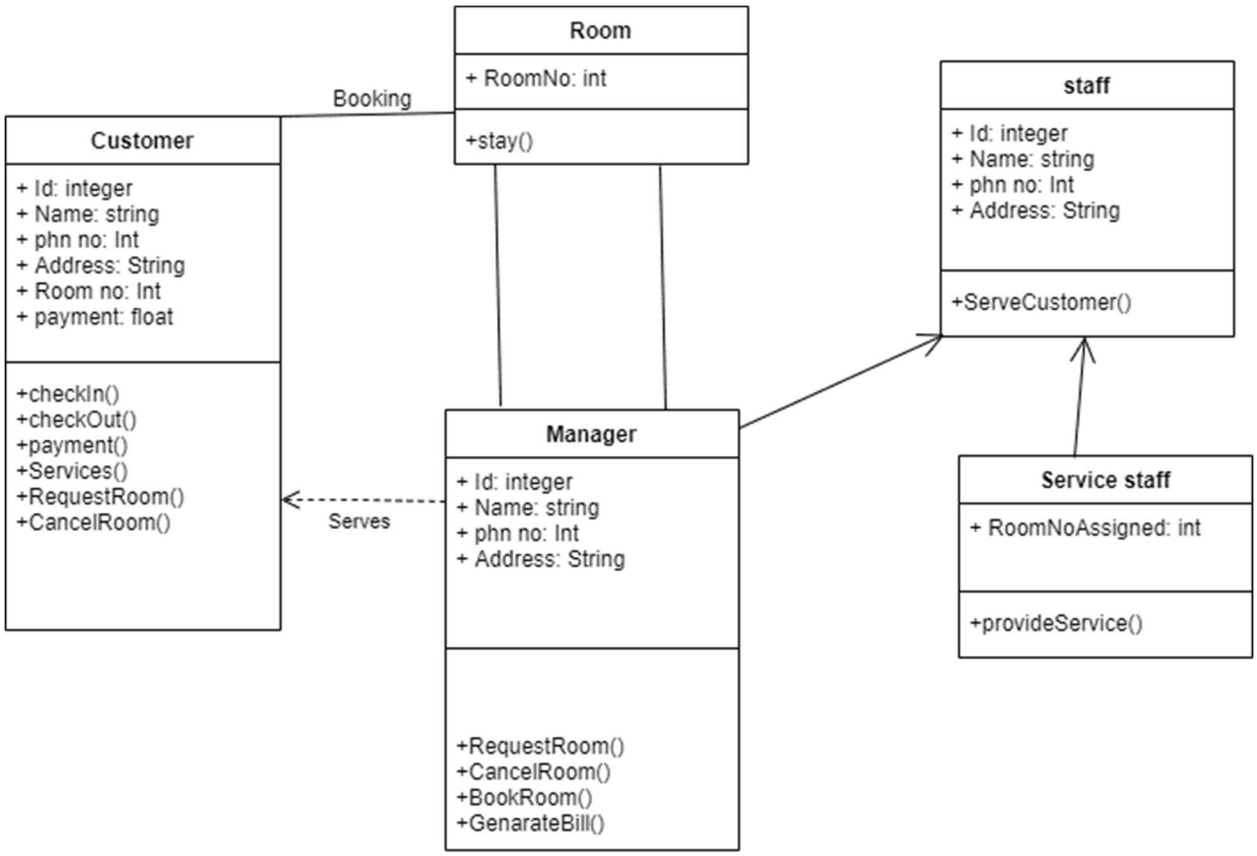
Use case diagram:

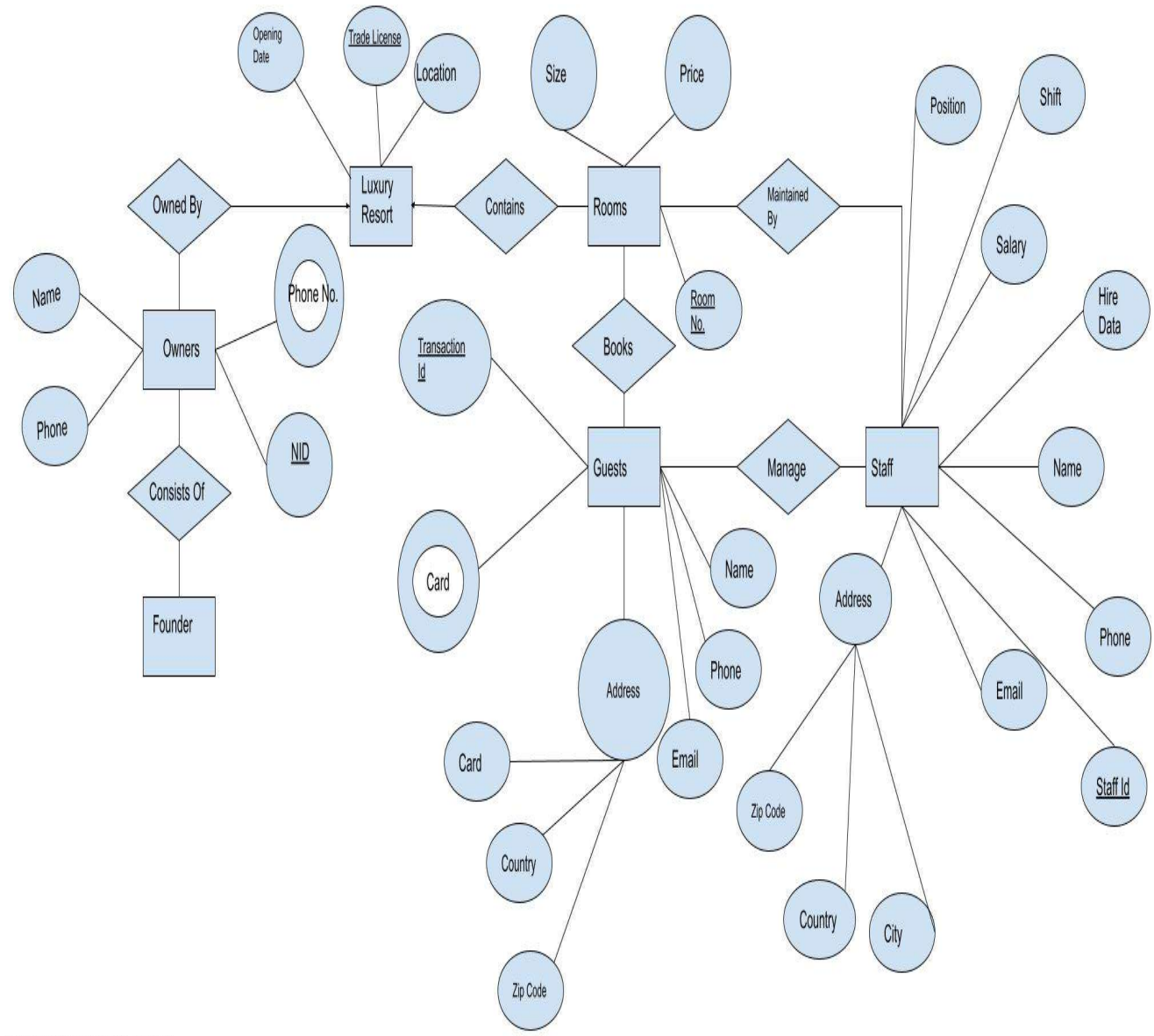


Activity diagram:



Class diagram:

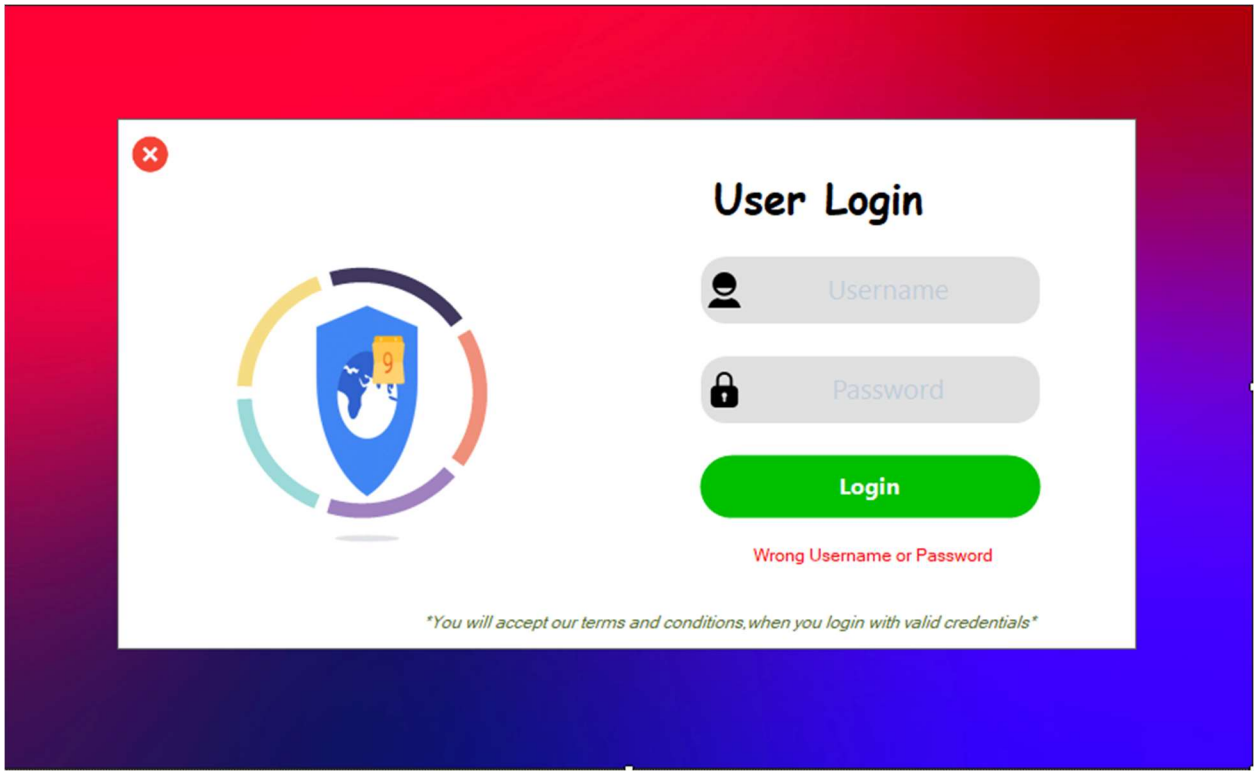


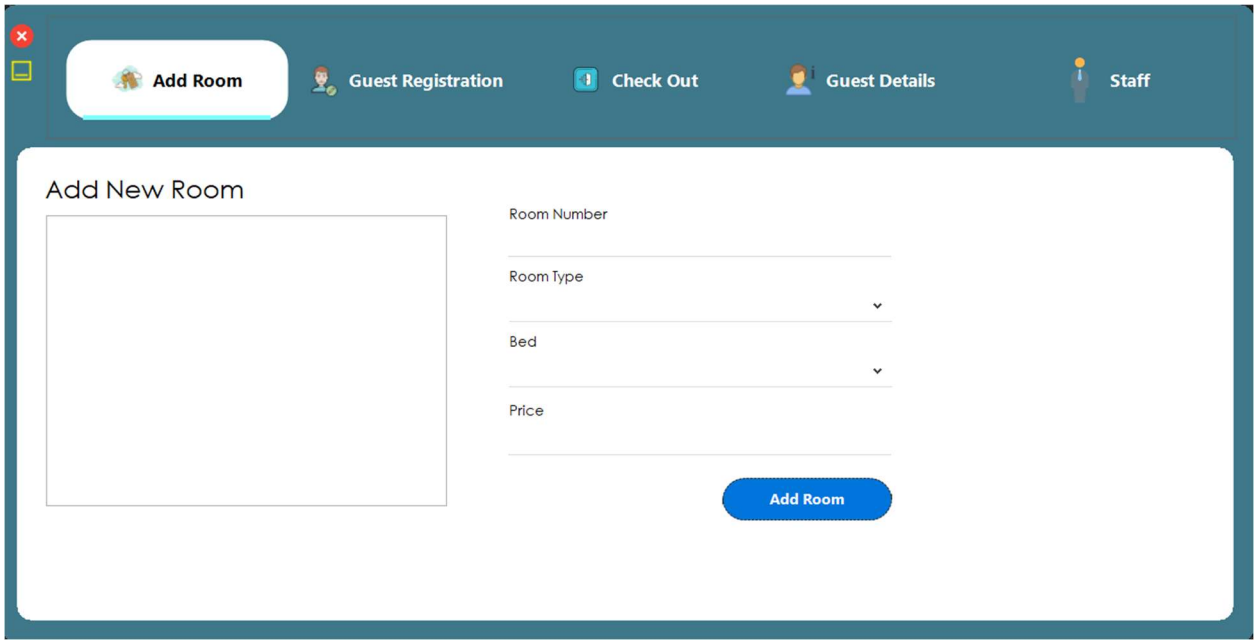


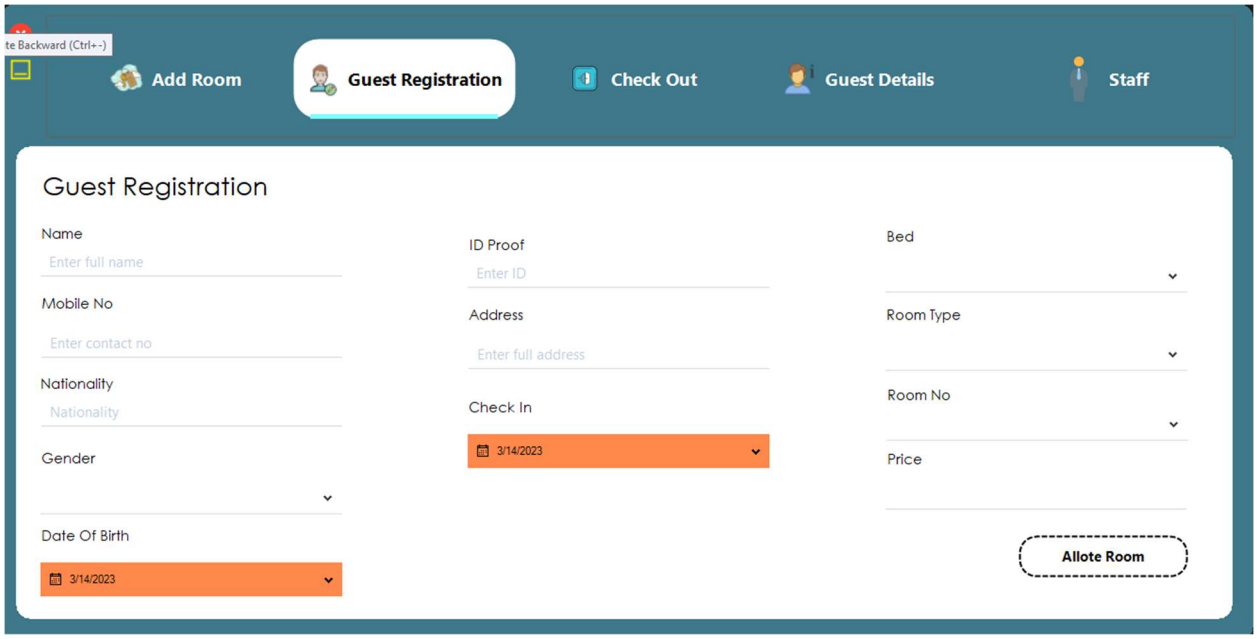
Name : Muhammad Akib-Al-Islam
ID : 20-42289-1

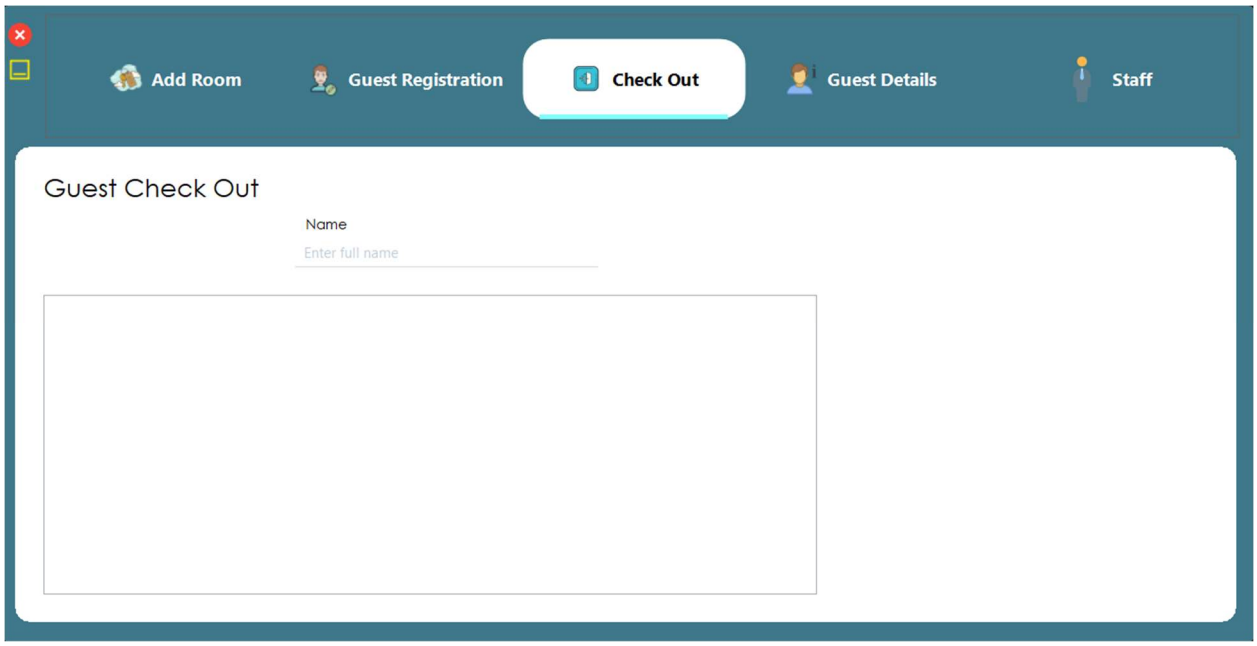
Figure: ER Diagram for Luxury Resort Management System

Interfaces created:






te Backward (Ctrl+-)
Add Room
Guest Registration
Check Out
Guest Details
Staff
Guest Registration
Name
Enter full name
Mobile No
Enter contact no
Nationality
Nationality
Gender
Date Of Birth
3/14/2023
ID Proof
Enter ID
Address
Enter full address
Check In
3/14/2023
Bed
Room Type
Room No
Price
Allote Room


Add Room
Guest Registration
Check Out
Guest Details
Staff
Guest Check Out
Name
Enter full name

Normalization

1. **CONSISTS OF**: (FNID, FNAME, FEMAIL, FPHONE, NID, NAME, PHONE, EMAIL)

   1NF- PHONE AND FPHONE IS A MULTIVALUED ATTRIBUTE

   2NF- FNID, FNAME, FEMAIL, FPHONE

   NID, NAME, PHONE, EMAIL

   3NF- NO TRANSITIVE DEPENDENCY

   FNID, FNAME, FEMAIL, FPHONE

   NID, NAME, PHONE, EMAIL

   Final table for **CONSISTS OF:**

   1. FNID, FNAME, FEMAIL
   2. FNID, FPHONE- Composite PK
   3. NID, NAME, EMAIL, FNID
   4. NID, PHONE- Composite PK

2. OWNED BY: (TRADE_LICENSE, LOCATION, OPENING_DATE, NID, NAME, EMAIL, PHONE)

   1NF- PHONE IS A MULTIVALUED ATTRIBUTE

   2NF- TRADE_LICENSE, LOCATION, OPENING_DATE

   NID, NAME, PHONE, EMAIL

   3NF- NO TRANSITIVE DEPENDENCY

   TRADE_LICENSE, LOCATION, OPENING_DATE

   NID, NAME, PHONE, EMAIL

   Final table for **OWNED BY:**

   5. TRADE_LICENSE, LOCATION, OPENING_DATE
   6. NID, NAME, EMAIL, TRADE_LICENSE
   7. NID, PHONE- Composite PK

3. **CONTAINS**: (TRADE_LICENSE, LOCATION, OPENING_DATE, ROOM_NO, R_SIZE, R_PRICE)
   1NF- NO MULTIVALUED ATTRIBUTE
   2NF- TRADE_LICENSE, LOCATION, OPENING_DATE
   ROOM_NO, R_SIZE, R_PRICE
   3NF- NO TRANSITIVE DEPENDENCY
   TRADE_LICENSE, LOCATION, OPENING_DATE

ROOM_NO, R_SIZE, R_PRICE

FINAL TABLE FOR **CONTAINS**:

8. TRADE_LICENSE, LOCATION, OPENING_DATE
9. ROOM_NO, R_SIZE, R_PRICE, TRADE_LICENSE

4. **BOOKS**: (ROOM_NO, R_SIZE, R_PRICE, TRANSACTION_ID, CARD, EMAIL, NAME, PHONE, GADDRESS)
   1NF- CARD IS A MULTIVALUED ATTRIBUTE
   2NF- ROOM_NO, R_SIZE, R_PRICE
   TRANSACTION_ID, CARD, EMAIL, NAME, PHONE, GADDRESS
   3NF- TRANSITIVE DEPENDENCY EXISTS FOR GADDRESS
   ROOM_NO, R_SIZE, R_PRICE
   TRANSACTION_ID, CARD, EMAIL, NAME, PHONE
   GADDRESS_ID, COUNTRY, CITY, ZIPCODE
   FINAL TABLE FOR **BOOKS:**
   10. ROOM_NO, R_SIZE, R_PRICE
   11. TRANSACTION_ID, EMAIL, NAME, PHONE
   12. TRANSACTION_ID, CARD - Composite PK
   13. GADDRESS_ID, COUNTRY, CITY, ZIPCODE
   14. GR_ID, ROOM_NO, TRANSACTION_ID

5. **MAINTAINED BY**: (ROOM_NO, R_SIZE, R_PRICE, STAFF_ID, PHONE, HIREDATE, POSITION, SHIFT, SALARY, NAME, EMAIL, SADDRESS)
   1NF- NO MULTIVALUED ATTRIBUTE
   2NF- ROOM_NO, R_SIZE, R_PRICE
   STAFF_ID, PHONE, HIREDATE, POSITION, SHIFT, SALARY, NAME, EMAIL, SADDRESS
   3NF- TRANSITIVE DEPENDENCY EXISTS FOR SADDRESS
   ROOM_NO, R_SIZE, R_PRICE
   STAFF_ID, PHONE, HIREDATE, POSITION, SHIFT, SALARY, NAME, EMAIL
   SADDRESS_ID, COUNTRY, CITY, ZIPCODE
   FINAL TABLE FOR **MAINTAINED BY:**
   15. ROOM_NO, R_SIZE, R_PRICE
   16. STAFF_ID, PHONE, HIREDATE, POSITION, SHIFT, SALARY, NAME, EMAIL
   17. SADDRESS_ID, COUNTRY, CITY, ZIPCODE
   18. RS_ID, ROOM_NO, STAFF_ID

6. **MANAGE**: (TRANSACTION_ID, CARD, EMAIL, NAME, PHONE, GADDRESS, STAFF_ID, PHONE, HIREDATE, POSITION, SHIFT, SALARY, NAME, EMAIL, SADDRESS)
   1NF- CARD IS A MULTIVALUED ATTRIBUTE
   2NF- TRANSACTION_ID, CARD, EMAIL, NAME, PHONE, GADDRESS
   STAFF_ID, PHONE, HIREDATE, POSITION, SHIFT, SALARY, NAME, EMAIL, SADDRESS
   3NF- TRANSITIVE DEPENDENCY EXISTS FOR SADDRESS AND GADDRESS
   TRANSACTION_ID, CARD, EMAIL, NAME, PHONE

STAFF_ID, PHONE, HIREDATE, POSITION, SHIFT, SALARY, NAME, EMAIL
SADDRESS_ID, COUNTRY, CITY, ZIPCODE
GADDRESS_ID, COUNTRY, CITY, ZIPCODE

FINAL TABLE FOR **MANAGE:**

19. TRANSACTION_ID, EMAIL, NAME, PHONE
20. TRANSACTION_ID, CARD - Composite PK
21. GADDRESS_ID, COUNTRY, CITY, ZIPCODE
22. STAFF_ID, PHONE, HIREDATE, POSITION, SHIFT, SALARY, NAME, EMAIL
23. SADDRESS_ID, COUNTRY, CITY, ZIPCODE
24. SG_ID, TRANSACTION_ID, STAFF_ID

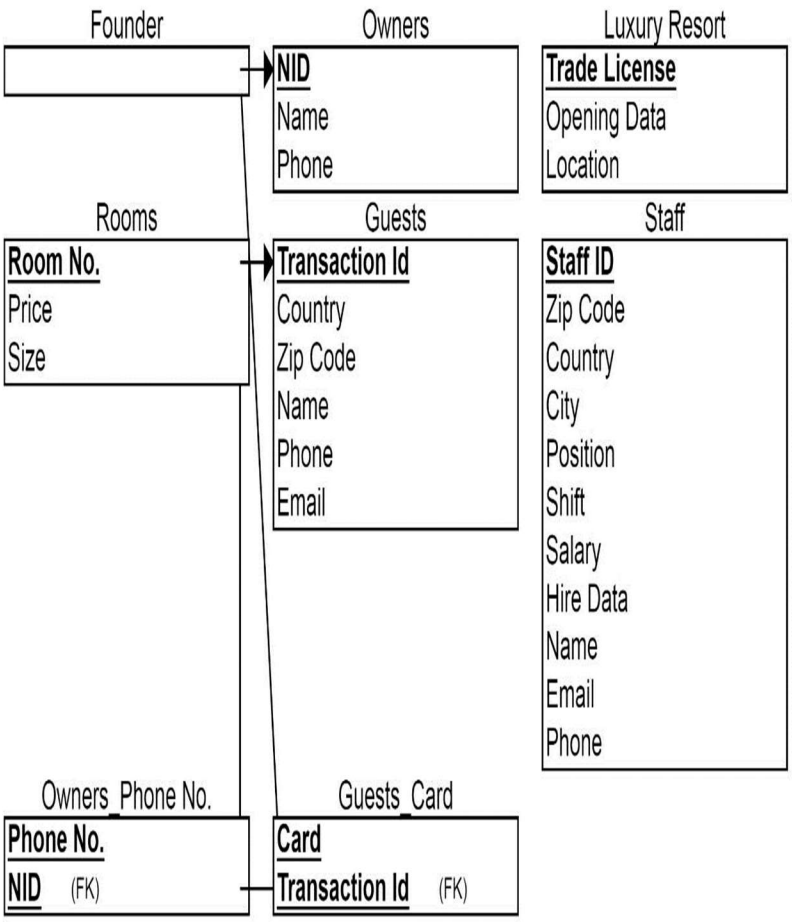
FINAL TABLES:

1. **FNID, FNAME, FEMAIL**
2. **FNID, FPHONE- Composite PK**
3. **NID, NAME, EMAIL, FNID**
4. **NID, PHONE- Composite PK**
5. **TRADE_LICENSE, LOCATION, OPENING_DATE**
6. **NID, NAME, EMAIL, TRADE_LICENSE**
7. **NID, PHONE- Composite PK**
8. **TRADE_LICENSE, LOCATION, OPENING_DATE**
9. **ROOM_NO, R_SIZE, R_PRICE, TRADE_LICENSE**
10. **ROOM_NO, R_SIZE, R_PRICE**
11. **TRANSACTION_ID, EMAIL, NAME, PHONE**
12. **TRANSACTION_ID, CARD - Composite PK**
13. **GADDRESS_ID, COUNTRY, CITY, ZIPCODE**
14. **GR_ID, ROOM_NO, TRANSACTION_ID**
15. **ROOM_NO, R_SIZE, R_PRICE**
16. **STAFF_ID, PHONE, HIREDATE, POSITION, SHIFT, SALARY, NAME, EMAIL**
17. **SADDRESS_ID, COUNTRY, CITY, ZIPCODE**
18. **RS_ID, ROOM_NO, STAFF_ID**
19. **TRANSACTION_ID, EMAIL, NAME, PHONE**
20. **TRANSACTION_ID, CARD - Composite PK**
21. **GADDRESS_ID, COUNTRY, CITY, ZIPCODE**
22. **STAFF_ID, PHONE, HIREDATE, POSITION, SHIFT, SALARY, NAME, EMAIL**
23. **SADDRESS_ID, COUNTRY, CITY, ZIPCODE**
24. **SG_ID, TRANSACTION_ID, STAFF_ID**

Evaluated Final Tables:

1. **FOUNDER- FNID, FNAME, FEMAIL**
2. **FOUNDER_CONTACT- FNID, FPHONE- Composite PK**
3. **OWNERS- NID, NAME, EMAIL, FNID, TRADE_LICENSE**
4. **OWNERS _CONTACT- NID, PHONE- Composite PK**
5. **LUXURY_RESORT- TRADE_LICENSE, LOCATION, OPENING_DATE**
6. **ROOMS**- ROOM_NO, R_SIZE, R_PRICE, TRADE_LICENSE
7. **GUESTS-** TRANSACTION_ID, EMAIL, NAME, PHONE
8. **TRANSACTION_INFO**- TRANSACTION_ID, CARD - Composite PK
9. **STAFF**- STAFF_ID, PHONE, HIREDATE, POSITION, SHIFT, SALARY, NAME, EMAIL
10. **STAFF_ADDRESS_INFO-** SADDRESS_ID, COUNTRY, CITY, ZIPCODE
11. **GUEST_ ADDRESS_INFO**- GADDRESS_ID, COUNTRY, CITY, ZIPCODE
12. **BOOKING**- GR_ID, ROOM_NO, TRANSACTION_ID
13. **ROOM_MANAGEMENT**- RS_ID, ROOM_NO, STAFF_ID
14. **GUEST_MANAGEMENT**- SG_ID, TRANSACTION_ID, STAFF_ID

Schema Diagram



Name:
Muhammad
Akib-Al-Islam

ID: 20-42289-1

```
16.create table Bill_related_to_waiter(Bill_Number number(20),primary key(Bill_Number),Description varchar2(200),B_Date DATE,B_Time TIMESTAMP,Total_Amount number(20),WEmployee_ID number(20)
FOREIGN KEY (WEmployee_ID) REFERENCES waiters(WEmployee_ID));

17.create table waiters(WEmployee_ID number(20),primary key(WEmployee_ID),Waiter_Name varchar2(50),Phone_Number number(20) );
```

Table Creation

1.FOUNDER table

```
CREATE TABLE FOUNDER
(
   FNID int NOT NULL,
   FNAME VARCHAR2(255) NOT NULL,
   FEMAIL VARCHAR2(255) NOT NULL,
   PRIMARY KEY(FNID)
);
```

Object Type TABLE Object FOUNDER

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
FOUNDER	FNID	Number	-	-	0	1	-	-	-
	FNAME	Varchar2	255	-	-	-	-	-	-
	FEMAIL	Varchar2	255	-	-	-	-	-	-
									1 - 3

2. FOUNDER_CONTACT table

```
CREATE TABLE FOUNDER_CONTACT (
   FNID int NOT NULL,
   FPHONE int NOT NULL,
   PRIMARY KEY (FNID, FPHONE)
);
```

Object Type TABLE Object FOUNDER_CONTACT

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
FOUNDER_CONTACT	FNID	Number	-	-	0	1	-	-	-
	FPHONE	Number	-	-	0	2	-	-	-
									1 - 2

3. OWNERS table

```
CREATE TABLE OWNERS (
   NID int NOT NULL,
   NAME VARCHAR2(255) NOT NULL,
```

```
  EMAIL VARCHAR2(255) NOT NULL,
  FNID int NOT NULL,
  TRADE_LICENSE VARCHAR2(255) NOT NULL,
   PRIMARY KEY(NID)
);
```

Object Type TABLE Object OWNERS

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
OWNERS	NID	Number	-	-	0	1	-	-	-
	NAME	Varchar2	255	-	-	-	-	-	-
	EMAIL	Varchar2	255	-	-	-	-	-	-
	FNID	Number	-	-	0	-	-	-	-
	TRADE_LICENSE	Varchar2	255	-	-	-	-	-	-

1 - 5

Alter table OWNERS add constraint fk1 FOREIGN KEY (FNID) REFERENCES FOUNDER (FNID);

Alter table OWNERS add constraint fk2 FOREIGN KEY (TRADE_LICENSE) REFERENCES LUXURY_RESORT (TRADE_LICENSE);

4. OWNERS_CONTACT table

```
CREATE TABLE OWNERS_CONTACT (
   NID int NOT NULL,
   PHONE int NOT NULL,
   PRIMARY KEY (NID, PHONE)
);
```

Object Type TABLE Object OWNERS_CONTACT

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
OWNERS_CONTACT	NID	Number	-	-	0	1	-	-	-
	PHONE	Number	-	-	0	2	-	-	-

1 - 2

5. LUXURY_RESORT table

```
CREATE TABLE LUXURY_RESORT (
   TRADE_LICENSE VARCHAR2(255) NOT NULL,
```

```
  LOCATION VARCHAR2(255) NOT NULL,
  OPENING_DATE DATE NOT NULL,
  PRIMARY KEY(TRADE_LICENSE)
);
```

Object Type **TABLE** Object **LUXURY_RESORT**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
LUXURY_RESORT	TRADE_LICENSE	Varchar2	255	-	-	1	-	-	-
	LOCATION	Varchar2	255	-	-	-	-	-	-
	OPENING_DATE	Date	7	-	-	-	-	-	-

1 - 3

6. ROOMS table

```
CREATE TABLE ROOMS (
  ROOM_NO int NOT NULL,
  R_SIZE int NOT NULL,
  R_PRICE int NOT NULL,
  TRADE_LICENSE VARCHAR2(255) NOT NULL,
  PRIMARY KEY(ROOM_NO)
);
```

Alter table ROOMS add constraint fk3 FOREIGN KEY (TRADE_LICENSE) REFERENCES LUXURY_RESORT (TRADE_LICENSE);

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **ROOMS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ROOMS	ROOM_NO	Number	-	-	0	1	-	-	-
	R_SIZE	Number	-	-	0	-	-	-	-
	R_PRICE	Number	-	-	0	-	-	-	-
	TRADE_LICENSE	Varchar2	255	-	-	-	-	-	-

1 - 4

7. GUESTS table

```
CREATE TABLE GUESTS (
   TRANSACTION_ID int NOT NULL,
   NAME VARCHAR2(255) NOT NULL,
   EMAIL VARCHAR2(255) NOT NULL,
   PHONE int NOT NULL,
   PRIMARY KEY(TRANSACTION_ID)
);
```

Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **GUESTS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
GUESTS	TRANSACTION_ID	Number	-	-	0	1	-	-	-
	NAME	Varchar2	255	-	-	-	-	-	-
	EMAIL	Varchar2	255	-	-	-	-	-	-
	PHONE	Number	-	-	0	-	-	-	-

1 - 4

8. TRANSACTION_INFO table

```
CREATE TABLE TRANSACTION_INFO (
   TRANSACTION_ID int NOT NULL,
   CARD int NOT NULL,
   PRIMARY KEY (TRANSACTION_ID, CARD)
);
```

Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **GUESTS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
GUESTS	TRANSACTION_ID	Number	-	-	0	1	-	-	-
	NAME	Varchar2	255	-	-	-	-	-	-
	EMAIL	Varchar2	255	-	-	-	-	-	-
	PHONE	Number	-	-	0	-	-	-	-

1 - 4

9. LUXURY_RESORT table

```
CREATE TABLE STAFF (
   STAFF_ID int NOT NULL,
   NAME VARCHAR2(255) NOT NULL,
   EMAIL VARCHAR2(255) NOT NULL,
   PHONE int NOT NULL,
   HIREDATE DATE NOT NULL,
   POSITION VARCHAR2(255) NOT NULL,
   SHIFT VARCHAR2(255) NOT NULL,
   SALARY VARCHAR2(255) NOT NULL,
   PRIMARY KEY(STAFF_ID)
);
```

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **STAFF**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
STAFF	STAFF_ID	Number	-	-	0	1	-	-	-
	NAME	Varchar2	255	-	-	-	-	-	-
	EMAIL	Varchar2	255	-	-	-	-	-	-
	PHONE	Number	-	-	0	-	-	-	-
	HIREDATE	Date	7	-	-	-	-	-	-
	POSITION	Varchar2	255	-	-	-	-	-	-
	SHIFT	Varchar2	255	-	-	-	-	-	-
	SALARY	Varchar2	255	-	-	-	-	-	-

1 - 8

10. STAFF_ADDRESS_INFO table

```
CREATE TABLE STAFF_ADDRESS_INFO (
   SADDRESS_ID INT NOT NULL,
   COUNTRY VARCHAR2(255) NOT NULL,
   CITY VARCHAR2(255) NOT NULL,
```

```
    ZIPCODE int NOT NULL
);
```

Results Explain Describe Saved SQL History

Object Type TABLE Object STAFF_ADDRESS_INFO

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
STAFF_ADDRESS_INFO	SADDRESS_ID	Number	-	-	0	1	-	-	-
	COUNTRY	Varchar2	255	-	-	-	-	-	-
	CITY	Varchar2	255	-	-	-	-	-	-
	ZIPCODE	Number	-	-	0	-	-	-	-

1 - 4

11. GUEST_ADDRESS_INFO table

```
CREATE TABLE GUEST_ADDRESS_INFO (
    GADDRESS_ID INT NOT NULL,
    COUNTRY VARCHAR2(255) NOT NULL,
    CITY VARCHAR2(255) NOT NULL,
    ZIPCODE int NOT NULL
);
```

Results Explain Describe Saved SQL History

Object Type TABLE Object GUEST_ADDRESS_INFO

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
GUEST_ADDRESS_INFO	GADDRESS_ID	Number	-	-	0	1	-	-	-
	COUNTRY	Varchar2	255	-	-	-	-	-	-
	CITY	Varchar2	255	-	-	-	-	-	-
	ZIPCODE	Number	-	-	0	-	-	-	-

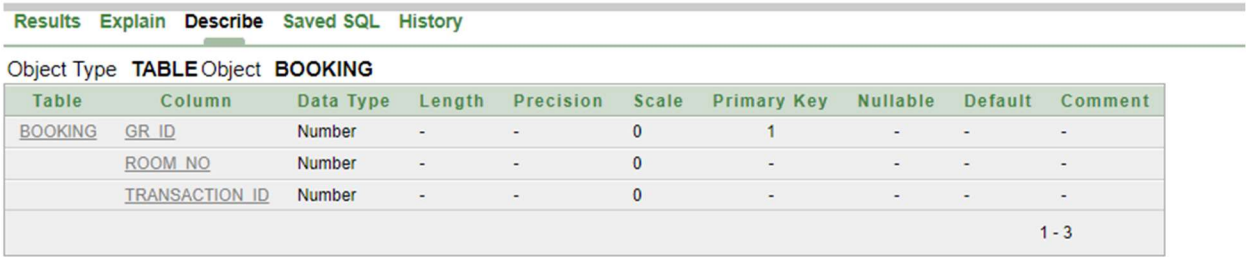
1 - 4

12. BOOKING table

```
CREATE TABLE BOOKING (
    GR_ID int NOT NULL,
    ROOM_NO int NOT NULL,
    TRANSACTION_ID int NOT NULL,
    PRIMARY KEY(GR_ID)
);
```

ALTER TABLE BOOKING ADD CONSTRAINT fk4 FOREIGN KEY (ROOM_NO) REFERENCES ROOMS(ROOM_NO);

Alter table BOOKING add constraint fk5 FOREIGN KEY (TRANSACTION_ID) REFERENCES GUESTS (TRANSACTION_ID);



Results Explain Describe Saved SQL History

Object Type TABLE Object BOOKING

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
BOOKING	GR_ID	Number	-	-	0	1	-	-	-
	ROOM_NO	Number	-	-	0	-	-	-	-
	TRANSACTION_ID	Number	-	-	0	-	-	-	-

1 - 3

13. ROOM_MANAGEMENT table

```
CREATE TABLE ROOM_MANAGEMENT (
   RS_ID int NOT NULL,
   ROOM_NO int NOT NULL,
   STAFF_ID int NOT NULL,
   PRIMARY KEY(RS_ID)
);
```

Alter table ROOM_MANAGEMENT add constraint fk6 FOREIGN KEY (ROOM_NO) REFERENCES ROOMS(ROOM_NO);

Alter table ROOM_MANAGEMENT add constraint fk7 FOREIGN KEY (STAFF_ID) REFERENCES STAFF(STAFF_ID);

Results Explain Describe Saved SQL History

Object Type TABLE Object ROOM_MANAGEMENT

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ROOM_MANAGEMENT	RS_ID	Number	-	-	0	1	-	-	-
	ROOM_NO	Number	-	-	0	-	-	-	-
	STAFF_ID	Number	-	-	0	-	-	-	-

1 - 3

14. GUEST _MANAGEMENT table

```
CREATE TABLE GUEST_MANAGEMENT (
  SG_ID int NOT NULL,
  TRANSACTION_ID int NOT NULL,
  STAFF_ID int NOT NULL,
  PRIMARY KEY(SG_ID)
);
```

Alter table GUEST_MANAGEMENT add constraint fk8 FOREIGN KEY (TRANSACTION_ID) REFERENCES GUESTS (TRANSACTION_ID);

Alter table GUEST_MANAGEMENT add constraint fk9 FOREIGN KEY (STAFF_ID) REFERENCES STAFF (STAFF_ID);

Results Explain Describe Saved SQL History

Object Type TABLE Object GUEST_MANAGEMENT

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
GUEST_MANAGEMENT	SG_ID	Number	-	-	0	1	-	-	-
	TRANSACTION_ID	Number	-	-	0	-	-	-	-
	STAFF_ID	Number	-	-	0	-	-	-	-

1 - 3

SEQUENCES

FOR FOUNDER

```
create sequence F_id
increment by 1
start with 202101
nocache
nocycle
```

FOR FOUNDER_CONTACT

```
create sequence FC_id
increment by 1
start with 202101
nocache
nocycle
```

FOR OWNERS

```
create sequence O_id
increment by 1
start with 202101
nocache
nocycle
```

FOR OWNERS_contact

```
create sequence Oc_id
increment by 1
start with 202101
nocache
nocycle
```

FOR ROOMS

```
create sequence R_id
increment by 1
start with 001
nocache
nocycle
```

FOR GUESTS

```
create sequence G_id
increment by 1
start with 1
nocache
nocycle
```

FOR TRANSACTION_INFO

```
create sequence tsc_id
increment by 1
start with 1
nocache
nocycle
```

FOR STAFF

```
create sequence S_id
increment by 1
start with 1
nocache
nocycle
```

FOR STAFF_ADDRESS_INFO

```
create sequence SA_id
increment by 1
start with 1
nocache
nocycle
```

FOR GUEST_ ADDRESS_INFO

```
create sequence GA_id
increment by 1
start with 1
nocache
nocycle
```

FOR BOOKING

```
create sequence B_id
increment by 1
start with 1
nocache
nocycle
```

FOR ROOM_MANAGEMENT
create sequence RM_id
increment by 1
start with 1
nocache
nocycle

FOR GUEST_MANAGEMENT
create sequence GM_id
increment by 1
start with 1
nocache
nocycle

INSERTION

1.FOUNDER TABLE

- INSERT INTO FOUNDER (FNID, FNAME, FEMAIL)
VALUES (F_id.NEXTVAL, 'John Smith', 'johnsmith@example.com');
- INSERT INTO FOUNDER (FNID, FNAME, FEMAIL)
VALUES (F_id.NEXTVAL, 'Jane Doe', 'janedoe@example.com');
- INSERT INTO FOUNDER (FNID, FNAME, FEMAIL)
VALUES (F_id.NEXTVAL, 'Peter Lee', 'peterlee@example.com');
- INSERT INTO FOUNDER (FNID, FNAME, FEMAIL)
VALUES (F_id.NEXTVAL, 'Sarah Kim', 'sarahkim@example.com');
- INSERT INTO FOUNDER (FNID, FNAME, FEMAIL)
VALUES (F_id.NEXTVAL, 'David Johnson', 'davidjohnson@example.com');

2. FOUNDER_CONTACT table

- INSERT INTO FOUNDER_CONTACT (FNID, FPHONE) VALUES (FC_id.NEXTVAL, 1234567890);
- INSERT INTO FOUNDER_CONTACT (FNID, FPHONE) VALUES (FC_id.NEXTVAL, 2345678901);
- INSERT INTO FOUNDER_CONTACT (FNID, FPHONE) VALUES (FC_id.NEXTVAL, 3456789012);
- INSERT INTO FOUNDER_CONTACT (FNID, FPHONE) VALUES (FC_id.NEXTVAL, 4567890123);
- INSERT INTO FOUNDER_CONTACT (FNID, FPHONE) VALUES (FC_id.NEXTVAL, 5678901234);

3. LUXURY_RESORT table

- INSERT INTO LUXURY_RESORT (TRADE_LICENSE, LOCATION, OPENING_DATE) VALUES ('ABC123', 'Miami', TO_DATE('2022-01-01', 'yyyy-mm-dd'));
- INSERT INTO LUXURY_RESORT (TRADE_LICENSE, LOCATION, OPENING_DATE) VALUES ('DEF456', 'Hawaii', TO_DATE('2021-07-15', 'yyyy-mm-dd'));
- INSERT INTO LUXURY_RESORT (TRADE_LICENSE, LOCATION, OPENING_DATE) VALUES ('GHI789', 'Bali', TO_DATE('2023-05-10', 'yyyy-mm-dd'));
- INSERT INTO LUXURY_RESORT (TRADE_LICENSE, LOCATION, OPENING_DATE) VALUES ('JKL012', 'Phuket', TO_DATE('2022-12-01', 'yyyy-mm-dd'));
- INSERT INTO LUXURY_RESORT (TRADE_LICENSE, LOCATION, OPENING_DATE) VALUES ('MNO345', 'Maldives', TO_DATE('2023-02-14', 'yyyy-mm-dd'));

4. ROOMS table

- INSERT INTO ROOMS (ROOM_NO, R_SIZE, R_PRICE, TRADE_LICENSE) VALUES (R_id.NEXTVAL, 2, 300, 'ABC123');
- INSERT INTO ROOMS (ROOM_NO, R_SIZE, R_PRICE, TRADE_LICENSE) VALUES (R_id.NEXTVAL, 3, 500, 'ABC123');
- INSERT INTO ROOMS (ROOM_NO, R_SIZE, R_PRICE, TRADE_LICENSE) VALUES (R_id.NEXTVAL, 2, 400, 'DEF456');
- INSERT INTO ROOMS (ROOM_NO, R_SIZE, R_PRICE, TRADE_LICENSE) VALUES (R_id.NEXTVAL, 4, 700, 'DEF456');
- INSERT INTO ROOMS (ROOM_NO, R_SIZE, R_PRICE, TRADE_LICENSE) VALUES (R_id.NEXTVAL, 2, 500, 'GHI789');

5. OWNERS table

- INSERT INTO OWNERS (NID, NAME, EMAIL, FNID, TRADE_LICENSE) VALUES (O_id.NEXTVAL, 'John Doe', 'johndoe@example.com', 202101, 'ABC123');
- INSERT INTO OWNERS (NID, NAME, EMAIL, FNID, TRADE_LICENSE) VALUES (O_id.NEXTVAL, 'Jane Smith', 'janesmith@example.com', 202102, 'DEF456');
- INSERT INTO OWNERS (NID, NAME, EMAIL, FNID, TRADE_LICENSE) VALUES (O_id.NEXTVAL, 'Bob Johnson', 'bobjohnson@example.com', 202103, 'GHI789');
- INSERT INTO OWNERS (NID, NAME, EMAIL, FNID, TRADE_LICENSE) VALUES (O_id.NEXTVAL, 'Sara Lee', 'saralee@example.com', 202104, 'JKL012');
- INSERT INTO OWNERS (NID, NAME, EMAIL, FNID, TRADE_LICENSE) VALUES (O_id.NEXTVAL, 'Mike Jones', 'mikejones@example.com', 202105, 'MNO345');

6. OWNERS_CONTACT table

- INSERT INTO OWNERS_CONTACT (NID, PHONE) VALUES (Oc_id.NEXTVAL, 123456789);

- INSERT INTO OWNERS_CONTACT (NID, PHONE) VALUES (Oc_id.NEXTVAL, 234567890);
- INSERT INTO OWNERS_CONTACT (NID, PHONE) VALUES (Oc_id.NEXTVAL, 345678901);
- INSERT INTO OWNERS_CONTACT (NID, PHONE) VALUES (Oc_id.NEXTVAL ,456789012);
- INSERT INTO OWNERS_CONTACT (NID, PHONE) VALUES (Oc_id.NEXTVAL, 567890123);

7. GUESTS table

- INSERT INTO GUESTS (TRANSACTION_ID, NAME, EMAIL, PHONE) VALUES (G_id.NEXTVAL, 'John Doe', 'john.doe@example.com', 123456789);
- INSERT INTO GUESTS (TRANSACTION_ID, NAME, EMAIL, PHONE) VALUES (G_id.NEXTVAL, 'Jane Smith', 'jane.smith@example.com', 234567890);
- INSERT INTO GUESTS (TRANSACTION_ID, NAME, EMAIL, PHONE) VALUES (G_id.NEXTVAL, 'Bob Johnson', 'bob.johnson@example.com', 345678901);
- INSERT INTO GUESTS (TRANSACTION_ID, NAME, EMAIL, PHONE) VALUES (G_id.NEXTVAL, 'Alice Jones', 'alice.jones@example.com', 456789012);
- INSERT INTO GUESTS (TRANSACTION_ID, NAME, EMAIL, PHONE) VALUES (G_id.NEXTVAL, 'Sam Brown', 'sam.brown@example.com', 567890123);

8. TRANSACTION_INFO table

- INSERT INTO TRANSACTION_INFO (TRANSACTION_ID, CARD) VALUES (tsc_id.NEXTVAL, 1234567890123456);
- INSERT INTO TRANSACTION_INFO (TRANSACTION_ID, CARD) VALUES (tsc_id.NEXTVAL, 2345678901234567);
- INSERT INTO TRANSACTION_INFO (TRANSACTION_ID, CARD) VALUES (tsc_id.NEXTVAL, 3456789012345678);
- INSERT INTO TRANSACTION_INFO (TRANSACTION_ID, CARD) VALUES (tsc_id.NEXTVAL, 4567890123456789);
- INSERT INTO TRANSACTION_INFO (TRANSACTION_ID, CARD) VALUES (tsc_id.NEXTVAL, 5678901234567890);

9. STAFF table

- INSERT INTO STAFF (STAFF_ID, NAME, EMAIL, PHONE, HIREDATE, POSITION, SHIFT, SALARY)
  VALUES (S_id.NEXTVAL, 'John Smith', 'john.smith@example.com', 1234567890,
  TO_DATE ('2022-01-01', 'YYYY-MM-DD'), 'Front Desk Receptionist', 'Day', '20000');
- INSERT INTO STAFF (STAFF_ID, NAME, EMAIL, PHONE, HIREDATE, POSITION, SHIFT, SALARY)

VALUES (S_id.NEXTVAL, 'Jane Doe', 'jane.doe@example.com', 2345678901, TO_DATE('2022-02-01', 'YYYY-MM-DD'), 'Housekeeping', 'Evening', '18000');

- INSERT INTO STAFF (STAFF_ID, NAME, EMAIL, PHONE, HIREDATE, POSITION, SHIFT, SALARY)
  VALUES (S_id.NEXTVAL, 'David Lee', 'david.lee@example.com', 3456789012, TO_DATE('2022-01-15', 'YYYY-MM-DD'), 'Restaurant Server', 'Morning', '25000');
- INSERT INTO STAFF (STAFF_ID, NAME, EMAIL, PHONE, HIREDATE, POSITION, SHIFT, SALARY)
  VALUES (S_id.NEXTVAL, 'Emily Chen', 'emily.chen@example.com', 4567890123, TO_DATE('2022-03-01', 'YYYY-MM-DD'), 'Spa Therapist', 'Afternoon', '23000');
- INSERT INTO STAFF (STAFF_ID, NAME, EMAIL, PHONE, HIREDATE, POSITION, SHIFT, SALARY)
  VALUES (S_id.NEXTVAL, 'Michael Kim', 'michael.kim@example.com', 5678901234, TO_DATE('2022-02-15', 'YYYY-MM-DD'), 'Pool Attendant', 'Morning', '$21000');

10. STAFF_ADDRESS_INFO table

- INSERT INTO STAFF_ADDRESS_INFO (SADDRESS_ID, COUNTRY, CITY, ZIPCODE)
  VALUES (SA_id.NEXTVAL, 'USA', 'New York', 10001);
- INSERT INTO STAFF_ADDRESS_INFO (SADDRESS_ID, COUNTRY, CITY, ZIPCODE)
  VALUES (SA_id.NEXTVAL, 'Canada', 'Toronto', '324242');
- INSERT INTO STAFF_ADDRESS_INFO (SADDRESS_ID, COUNTRY, CITY, ZIPCODE)
  VALUES (SA_id.NEXTVAL ,'UK', 'London', '536363');
- INSERT INTO STAFF_ADDRESS_INFO (SADDRESS_ID, COUNTRY, CITY, ZIPCODE)
  VALUES (SA_id.NEXTVAL, 'Australia', 'Sydney', '2000');
- INSERT INTO STAFF_ADDRESS_INFO (SADDRESS_ID, COUNTRY, CITY, ZIPCODE)
  VALUES (SA_id.NEXTVAL, 'India', 'Mumbai', 400001);

11. GUEST_ADDRESS_INFO table

- INSERT INTO GUEST_ADDRESS_INFO (GADDRESS_ID, COUNTRY, CITY, ZIPCODE)
  VALUES (GA_id.NEXTVAL, 'USA', 'San Francisco', 94105);
- INSERT INTO GUEST_ADDRESS_INFO (GADDRESS_ID, COUNTRY, CITY, ZIPCODE)
  VALUES (GA_id.NEXTVAL ,'Canada', 'Montreal', '87977');
- INSERT INTO GUEST_ADDRESS_INFO (GADDRESS_ID, COUNTRY, CITY, ZIPCODE)
  VALUES (GA_id.NEXTVAL, 'UK', 'Edinburgh', '57647858');
- INSERT INTO GUEST_ADDRESS_INFO (GADDRESS_ID, COUNTRY, CITY, ZIPCODE)
  VALUES (GA_id.NEXTVAL, 'Australia', 'Melbourne', '3000');
- INSERT INTO GUEST_ADDRESS_INFO (GADDRESS_ID, COUNTRY, CITY, ZIPCODE)

VALUES (GA_id.NEXTVAL, 'India', 'Bangalore', 560001);

12. BOOKING table

- INSERT INTO BOOKING (GR_ID, ROOM_NO, TRANSACTION_ID)
  VALUES (B_id.NEXTVAL, 1, 1);
- INSERT INTO BOOKING (GR_ID, ROOM_NO, TRANSACTION_ID)
  VALUES (B_id.NEXTVAL, 2, 2);
- INSERT INTO BOOKING (GR_ID, ROOM_NO, TRANSACTION_ID)
  VALUES (B_id.NEXTVAL, 3, 3);
- INSERT INTO BOOKING (GR_ID, ROOM_NO, TRANSACTION_ID)
  VALUES (B_id.NEXTVAL, 4, 4);
- INSERT INTO BOOKING (GR_ID, ROOM_NO, TRANSACTION_ID)
  VALUES (B_id.NEXTVAL, 5, 5);

13. BOOKING table

- INSERT INTO ROOM_MANAGEMENT (RS_ID, ROOM_NO, STAFF_ID)
  VALUES (RM_id.NEXTVAL, 1, 1);
- INSERT INTO ROOM_MANAGEMENT (RS_ID, ROOM_NO, STAFF_ID)
  VALUES (RM_id.NEXTVAL ,2, 2);
- INSERT INTO ROOM_MANAGEMENT (RS_ID, ROOM_NO, STAFF_ID)
  VALUES (RM_id.NEXTVAL, 3, 3);
- INSERT INTO ROOM_MANAGEMENT (RS_ID, ROOM_NO, STAFF_ID)
  VALUES (RM_id.NEXTVAL, 4, 4);
- INSERT INTO ROOM_MANAGEMENT (RS_ID, ROOM_NO, STAFF_ID)
  VALUES (RM_id.NEXTVAL, 5, 5);

14. BOOKING table

- INSERT INTO GUEST_MANAGEMENT (SG_ID, TRANSACTION_ID, STAFF_ID)
  VALUES (GM_id.NEXTVAL, 1,1);
- INSERT INTO GUEST_MANAGEMENT (SG_ID, TRANSACTION_ID, STAFF_ID)
  VALUES (GM_id.NEXTVAL, 2,2);
- INSERT INTO GUEST_MANAGEMENT (SG_ID, TRANSACTION_ID, STAFF_ID)
  VALUES (GM_id.NEXTVAL, 3,3);

- INSERT INTO GUEST_MANAGEMENT (SG_ID, TRANSACTION_ID, STAFF_ID) VALUES (GM_id.NEXTVAL, 4,4);
- INSERT INTO GUEST_MANAGEMENT (SG_ID, TRANSACTION_ID, STAFF_ID) VALUES (GM_id.NEXTVAL, 5,5);

SCREENSHOTS OF TABLES

1.FOUNDER table

FNID	FNAME	FEMAIL
202101	John Smith	johnsmith@example.com
202102	John Smith	johnsmith@example.com
202103	Peter Lee	peterlee@example.com
202104	Sarah Kim	sarahkim@example.com
202105	David Johnson	davidjohnson@example.com

5 rows returned in 0.00 seconds CSV Export

2. FOUNDER_CONTACT table

Results Explain Describe Saved

FNID	FPHONE
202101	2345678901
202102	3456789012
202103	4567890123
202104	5678901234

4 rows returned in 0.01 seconds

3. OWNERS table

Results Explain Describe Saved SQL History

NID	NAME	EMAIL	FNID	TRADE_LICENSE
202112	John Doe	johndoe@example.com	202101	ABC123
202113	Jane Smith	janesmith@example.com	202102	DEF456
202114	Bob Johnson	bobjohnson@example.com	202103	GHI789
202115	Sara Lee	saralee@example.com	202104	JKL012
202116	Mike Jones	mikejones@example.com	202105	MNO345

5 rows returned in 0.00 seconds CSV Export

4. OWNERS_CONTACT table

NID	PHONE
202101	123456789
202102	234567890
202103	345678901
202104	567890123

4 rows returned in 0.00 seconds CSV Ex

5. LUXURY_RESORT table

Results Explain Describe Saved SQL History

TRADE_LICENSE	LOCATION	OPENING_DATE
ABC123	Miami	01-JAN-22
DEF456	Hawaii	15-JUL-21
GHI789	Bali	10-MAY-23
JKL012	Phuket	01-DEC-22
MNO345	Maldives	14-FEB-23

5 rows returned in 0.00 seconds CSV Export

6. ROOMS table

Results Explain Describe Saved SQL History

ROOM_NO	R_SIZE	R_PRICE	TRADE_LICENSE
1	2	300	ABC123
2	3	500	ABC123
3	2	400	DEF456
4	4	700	DEF456
5	2	500	GHI789

5 rows returned in 0.00 seconds CSV Export

7. GUESTS table

Results Explain Describe Saved SQL History

TRANSACTION_ID	NAME	EMAIL	PHONE
1	John Doe	john.doe@example.com	123456789
2	Jane Smith	jane.smith@example.com	234567890
3	Bob Johnson	bob.johnson@example.com	345678901
4	Alice Jones	alice.jones@example.com	456789012
5	Sam Brown	sam.brown@example.com	567890123

5 rows returned in 0.00 seconds CSV Export

8. TRANSACTION_INFO table

TRANSACTION_ID	CARD
1	1234567890123456
2	2345678901234567
3	3456789012345678
4	4567890123456789
5	5678901234567890

5 rows returned in 0.00 seconds CSV Export

9. STAFF table

Results Explain Describe Saved SQL History

STAFF_ID	NAME	EMAIL	PHONE	HIREDATE	POSITION	SHIFT	SALARY
1	John Smith	john.smith@example.com	1234567890	01-JAN-22	Front Desk Receptionist	Day	20000
2	Jane Doe	jane.doe@example.com	2345678901	01-FEB-22	Housekeeping	Evening	18000
3	David Lee	david.lee@example.com	3456789012	15-JAN-22	Restaurant Server	Morning	25000
4	Emily Chen	emily.chen@example.com	4567890123	01-MAR-22	Spa Therapist	Afternoon	23000
5	Michael Kim	michael.kim@example.com	5678901234	15-FEB-22	Pool Attendant	Morning	$21000

5 rows returned in 0.01 seconds CSV Export

10. STAFF_ADDRESS_INFO table

Results Explain Describe Saved SQL History

SADDRESS_ID	COUNTRY	CITY	ZIPCODE
1	USA	New York	10001
2	Canada	Toronto	324242
3	UK	London	536363
4	Australia	Sydney	2000
5	India	Mumbai	400001

5 rows returned in 0.00 seconds CSV Export

11. GUEST_MANAGEMENT table

Results Explain Describe Saved SQL History

SG_ID	TRANSACTION_ID	STAFF_ID
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5

5 rows returned in 0.02 seconds CSV Export

12. GUEST_ ADDRESS_INFO table

GADDRESS_ID	COUNTRY	CITY	ZIPCODE
1	USA	San Francisco	94105
2	Canada	Montreal	87977
3	UK	Edinburgh	57647858
4	Australia	Melbourne	3000
5	India	Bangalore	560001

5 rows returned in 0.02 seconds CSV Export

13. BOOKING table

GR_ID	ROOM_NO	TRANSACTION_ID
6	1	1
7	2	2
8	3	3
9	4	4
10	5	5

5 rows returned in 0.00 seconds CSV Export

14. ROOM_MANAGEMENT table

Results Explain Describe Saved SQL History

RS_ID	ROOM_NO	STAFF_ID
5	1	1
6	2	2
7	3	3
8	4	4
9	5	5

5 rows returned in 0.00 seconds CSV Export

INDEX

FOR FOUNDER

```
CREATE INDEX f_idx
ON FOUNDER (FNAME,FEMAIL );
```

FOR FOUNDER_CONTACT

```
create INDEX FC_idx
ON FOUNDER_CONTACT (FPHONE );
```

FOR OWNERS

```
create INDEX O_idx
ON OWNERS (NAME, EMAIL );
```

FOR OWNERS_contact

```
create INDEX Oc_idX
ON OWNERS_CONTACT (PHONE );
```

FOR ROOMS

```
create INDEX R_idX
ON ROOMS(R_SIZE, R_PRICE);
```

FOR GUESTS

```
create INDEX G_id
ON GUESTS (EMAIL, PHONE);
```

FOR TRANSACTION_INFO

```
create INDEX tsc_idX
ON TRANSACTION_INFO (CARD);
```

FOR STAFF

```
create INDEX S_idx
ON STAFF (HIREDATE,POSITION);
```

FOR STAFF_ADDRESS_INFO

```
create INDEX SA_idX
ON STAFF_ADDRESS_INFO (COUNTRY, CITY);
```

FOR GUEST_ ADDRESS_INFO

```
create INDEX GA_idX
ON GUEST_ ADDRESS_INFO( CITY, ZIPCODE);
```

FOR BOOKING
create INDEX B_idX
ON BOOKING (ROOM_NO)

FOR ROOM_MANAGEMENT
create INDEX RM_idX
ON ROOM_MANAGEMENT (ROOM_NO)

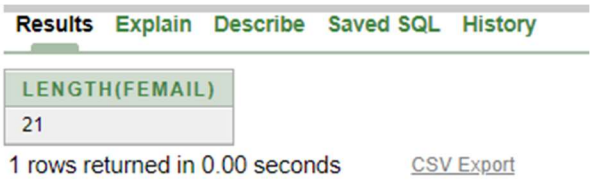
FOR GUEST_MANAGEMENT
create INDEX GM_idX
ON GUEST_MANAGEMENT (STAFF_ID)

QUERY WRITING

SINGLE ROW FUNCTION

1 . What is the length of the founder's email address for founder with FNID 3?

```
SELECT LENGTH(FEMAIL)
FROM FOUNDER
WHERE FNID = 202101;
```

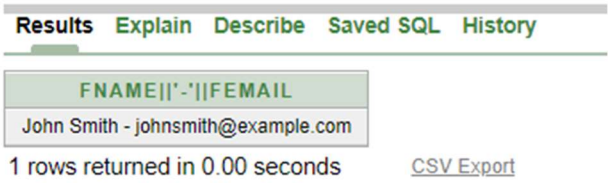


2 . What is the uppercase version of the location of the luxury resort with trade license ABC123?

```
SELECT UPPER(LOCATION)
FROM LUXURY_RESORT
WHERE TRADE_LICENSE = 'ABC123';
```

3. What is the concatenation of the founder's name and email address for the founder with FNID 1?

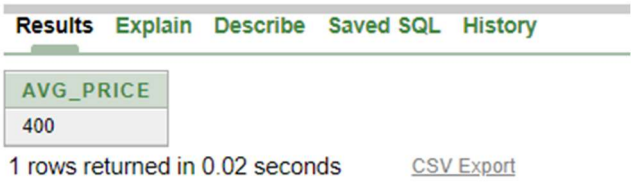
```
SELECT FNAME || ' - ' || FEMAIL
FROM FOUNDER
WHERE FNID = 202101;
```



GROUP FUNCTION

1. What is the average price of rooms in the luxury resort "ABC123"?

```
SELECT AVG(R_PRICE) AS AVG_PRICE
FROM ROOMS
WHERE TRADE_LICENSE = 'ABC123';
```



2. How many guests have made transactions at the luxury resort "DEF456"?

```
SELECT COUNT(DISTINCT TRANSACTION_ID) AS GUEST_COUNT
FROM BOOKING
WHERE ROOM_NO IN (
```

```
SELECT ROOM_NO FROM ROOMS WHERE TRADE_LICENSE = 'DEF456'
);
```

Results Explain Describe Saved SQL History

GUEST_COUNT
2

1 rows returned in 0.02 seconds CSV Export

3. How many staff members are employed on each shift?

```
SELECT SHIFT, COUNT(*) as num_employees
FROM STAFF
GROUP BY SHIFT;
```

Results Explain Describe Saved SQL History

SHIFT	NUM_EMPLOYEES
Day	1
Morning	2
Evening	1
Afternoon	1

4 rows returned in 0.02 seconds CSV Export

SUBQUERY

1. What is the average price of rooms at the luxury resort located in "Miami"?

```
SELECT AVG(R_PRICE)
FROM ROOMS
WHERE TRADE_LICENSE IN (
  SELECT TRADE_LICENSE
  FROM LUXURY_RESORT
  WHERE LOCATION = 'Miami'
);
```

Results Explain Describe Saved SQL History

AVG(R_PRICE)
400

1 rows returned in 0.00 seconds CSV Export

2. Find all guests who have booked a room managed by a staff member with a salary greater than $50,000.

```
SELECT NAME, EMAIL
FROM GUESTS
WHERE TRANSACTION_ID IN (
   SELECT TRANSACTION_ID
   FROM BOOKING
   WHERE ROOM_NO IN (
   SELECT ROOM_NO
   FROM ROOM_MANAGEMENT
   WHERE STAFF_ID IN (
   SELECT STAFF_ID
   FROM STAFF
   WHERE SALARY > '$50000'
      )
   )
);
```

Results Explain Describe Saved SQL History

NAME	EMAIL
John Doe	john.doe@example.com
Jane Smith	jane.smith@example.com
Alice Jones	alice.jones@example.com
Bob Johnson	bob.johnson@example.com

4 rows returned in 0.03 seconds CSV Export

3. Find all guests who have booked a room in a luxury resort that opened before 2022:

```
SELECT NAME, EMAIL
FROM GUESTS
WHERE TRANSACTION_ID IN (
   SELECT TRANSACTION_ID
   FROM BOOKING
```

```
  WHERE ROOM_NO IN (
    SELECT ROOM_NO
    FROM ROOMS
    WHERE TRADE_LICENSE IN (
      SELECT TRADE_LICENSE
      FROM LUXURY_RESORT
      WHERE OPENING_DATE < TO_DATE('2022-01-01', 'yyyy-mm-dd')
    )
  )
);
```

Results Explain Describe Saved SQL History

NAME	EMAIL
Alice Jones	alice.jones@example.com
Bob Johnson	bob.johnson@example.com

2 rows returned in 0.00 seconds CSV Export

JOINING

1. Return the name, email, and phone number of all the guests who have made a transaction using a particular card.

```
SELECT G.NAME, G.EMAIL, G.PHONE
FROM GUESTS G
INNER JOIN TRANSACTION_INFO TI ON G.TRANSACTION_ID = TI.TRANSACTION_ID
WHERE TI.CARD = 1234567890123456;
```

Results Explain Describe Saved SQL History

NAME	EMAIL	PHONE
John Doe	john.doe@example.com	123456789

1 rows returned in 0.00 seconds CSV Export

2. Return the name, email, and phone number of all the staff members who work the evening shift.

```
SELECT S.NAME, S.EMAIL, S.PHONE
FROM STAFF S
WHERE S.SHIFT = 'Evening';
```

Results Explain Describe Saved SQL History

NAME	EMAIL	PHONE
Jane Doe	jane.doe@example.com	2345678901

1 rows returned in 0.00 seconds CSV Export

3. Return the name and email of all the guests who have booked a room at the luxury resort in Hawaii.

```
SELECT G.NAME, G.EMAIL
FROM GUESTS G
INNER JOIN BOOKING B ON G.TRANSACTION_ID = B.TRANSACTION_ID
INNER JOIN ROOMS R ON B.ROOM_NO = R.ROOM_NO
INNER JOIN LUXURY_RESORT L ON R.TRADE_LICENSE = L.TRADE_LICENSE
WHERE L.LOCATION = 'Hawaii';
```

Results Explain Describe Saved SQL History

NAME	EMAIL
Bob Johnson	bob.johnson@example.com
Alice Jones	alice.jones@example.com

2 rows returned in 0.03 seconds CSV Export

VIEW

1. Create a view that shows the names and contact information of all the resort owners along with the location and trade license of their respective resorts.

```
CREATE VIEW OwnerResortInfo AS
SELECT OWNERS.NAME, OWNERS.EMAIL, OWNERS_CONTACT.PHONE,
LUXURY_RESORT.LOCATION, LUXURY_RESORT.TRADE_LICENSE
FROM OWNERS
JOIN OWNERS_CONTACT ON OWNERS.NID = OWNERS_CONTACT.NID
JOIN LUXURY_RESORT ON OWNERS.TRADE_LICENSE =
LUXURY_RESORT.TRADE_LICENSE;
```

2. Create a view that shows the room numbers, sizes, prices, and the location of the resorts where those rooms are located.

CREATE VIEW RoomLocationInfo AS
SELECT ROOMS.ROOM_NO, ROOMS.R_SIZE, ROOMS.R_PRICE,
LUXURY_RESORT.LOCATION
FROM ROOMS
JOIN LUXURY_RESORT ON ROOMS.TRADE_LICENSE = LUXURY_RESORT.TRADE_LICENSE;

```
CREATE VIEW RoomLocationInfo AS
SELECT ROOMS.ROOM_NO, ROOMS.R_SIZE, ROOMS.R_PRICE, LUXURY_RESORT.LOCATION
FROM ROOMS
JOIN LUXURY_RESORT ON ROOMS.TRADE_LICENSE = LUXURY_RESORT.TRADE_LICENSE;
```

Results Explain Describe Saved SQL History

View created.

Results Explain Describe Saved SQL History

ROOM_NO	R_SIZE	R_PRICE	LOCATION
1	2	300	Miami
2	3	500	Miami
3	2	400	Hawaii
4	4	700	Hawaii
5	2	500	Bali

5 rows returned in 0.00 seconds CSV Export

3. Create a view that shows the room numbers, sizes, and prices of all the available rooms (i.e., those not booked by any guest).

CREATE VIEW AvailableRooms AS
SELECT ROOMS.ROOM_NO, ROOMS.R_SIZE, ROOMS.R_PRICE
FROM ROOMS
WHERE ROOMS.ROOM_NO NOT IN (SELECT ROOM_NO FROM BOOKING);

```
CREATE VIEW AvailableRooms AS
SELECT ROOMS.ROOM_NO, ROOMS.R_SIZE, ROOMS.R_PRICE
FROM ROOMS
WHERE ROOMS.ROOM_NO NOT IN (SELECT ROOM_NO FROM BOOKING);
```

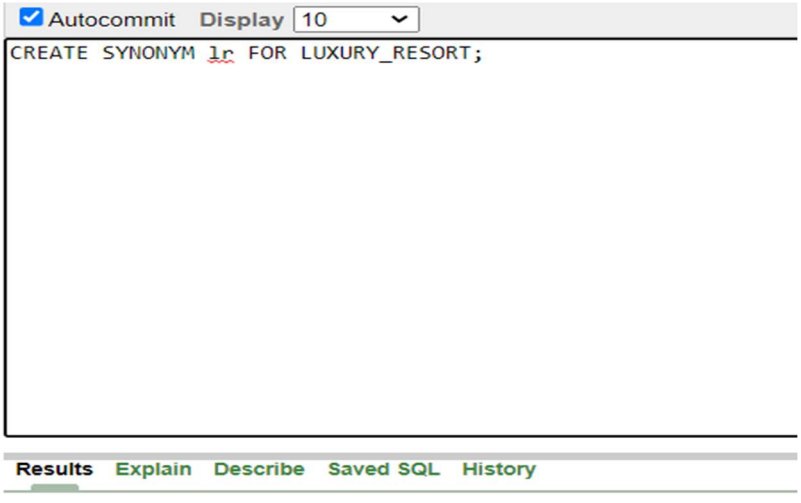
Results Explain Describe Saved SQL History

```
View created.
```

SYNONYM

1. Create a synonym for the LUXURY_RESORT table.

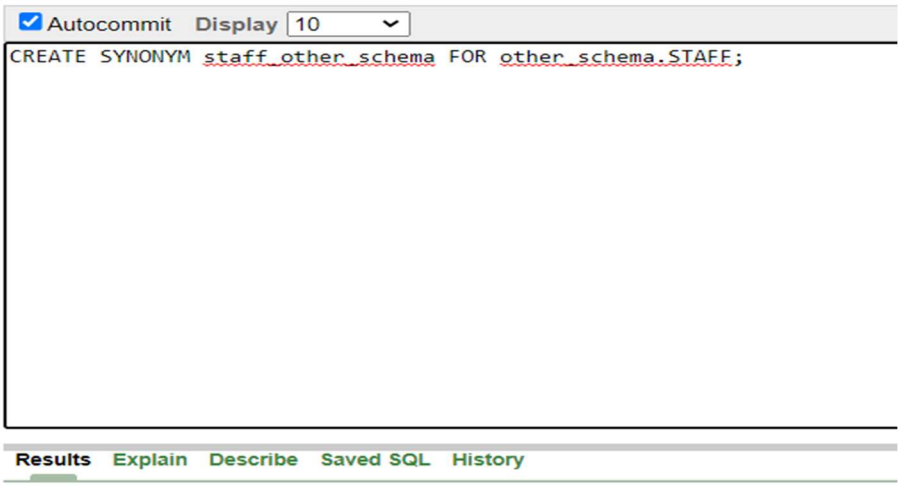
CREATE SYNONYM lr FOR LUXURY_RESORT;




```
Synonym created.
```

2. Create a synonym for the STAFF table in a different schema.

CREATE SYNONYM staff_other_schema FOR other_schema.STAFF;



3. Create a synonym for the ROOMS table in a specific schema.

CREATE SYNONYM my_schema.rooms FOR ROOMS;

USERS CREATED

- **CREATE USER NAOMI IDENTIFIED BY naomi;**
- **CREATE ROLE GUEST;**
- **CREATE ROLE STAFF;**
- **CREATE USER FABIHA IDENTIFIED BY ALAM;**
- **CREATE USER AKIB IDENTIFIED BY ISLAM;**
- **GRANT update, select ON GUEST;**
- **GRANT GUEST to MAISHA;**
- **CREATE USER FARHAN IDENTIFIED BY AMIN;**

- **CREATE USER FABIHA IDENTIFIED BY ALAM;**
- **GRANT STAFF to FARHAN;**
- **CREATE ROLE OWNER;**
- **GRANT CREATE SESSION, CREATE TABLE, CREATE SEQUENCE, CREATE VIEW, CREATE PROCEDURE TO OWNER;**
- **GRANT OWNER to NAOMI;**
- **GRANT CREATE TABLE, CREATE SEQUENCE, CREATE VIEW, CREATE PROCEDURE TO STAFF;**
- **GRANT CREATE ANY SYNONYM TO SCOTT;**

RELATIONAL ALGEBRA

1.List the names and email addresses of all the staff members who were hired before January 1, 2022, and earn a salary greater than 50,000.

π NAME, EMAIL(σ HIREDATE < '2022-01-01' ^ SALARY > 50000(STAFF))

2.Find the transaction IDs of all guests who used a credit card for payment.

π TRANSACTION_ID(TRANSACTION_INFO)

3.List the names of all luxury resorts that opened after December 31, 2019.

π LOCATION(σ OPENING_DATE > '2019-12-31'(LUXURY_RESORT))

4.Find the phone numbers of all founders and owners associated with a luxury resort.

π FPHONE(FOUNDER_CONTACT) ⋈ π FPHONE(OWNERS_CONTACT) ⋈ π TRADE_LICENSE(σ LOCATION = 'luxury resort name'(LUXURY_RESORT))

5.Find the transaction IDs and phone numbers of all guests who stayed in rooms larger than 500 sq. ft. and paid a price greater than 200 per night.

π TRANSACTION_ID, PHONE((π TRANSACTION_ID, ROOM_NO(BOOKING) ⋈ π ROOM_NO(σ R_SIZE > 500 ∧ R_PRICE > 200(ROOMS)))) ⋈ GUESTS

Conclusion:

The project aims to create a luxury resort management system using C# .NET framework and Oracle 10g database to streamline operations, improve the guest experience, and enhance financial management.

The system will consist of modules for reservation management, guest check-in and check-out, room and inventory management, billing and payment processing, and reporting. It will also prioritize security and compliance with regulations such as GDPR and CCPA.

To improve the existing project for the final term, the team could focus on further improving the user experience of the system, such as enhancing the reservation process, adding more payment options, and integrating with other tools used by the resort. Additionally, the team could explore adding features such as a loyalty program or mobile app to further enhance the guest experience.

Another area of improvement could be in the reporting module, where the team could add more metrics and analytics to provide deeper insights into the resort's operations and performance.

Overall, the team could prioritize testing and quality assurance to ensure that the system is reliable and scalable and focus on continuous improvement through user feedback and data analysis.