جامعة جدة
University of Jeddah

Database Project

Building Bridge System

Course Number: CCCS-215

Course Section: A1

Database Topic:

Our project aims to develop a comprehensive database system for managing multiple apartment complexes. The system will be designed to handle various aspects of apartment complex management, including occupant information, apartment availability, rent or buying options, payment methods, and staff management.

The goal of our project is to better the management of apartment complexes by providing a database system. This system will display and update information on apartment availability, occupant status, and financial transactions, and keeping updated with rent payment processes. Additionally, the system will help show services like security and maintenance employees on ground and will add to final bill. Our goal is to create a user-friendly and efficient tool that meets the needs of property management companies, apartment complex owners, and staff responsible for managing these complexes.

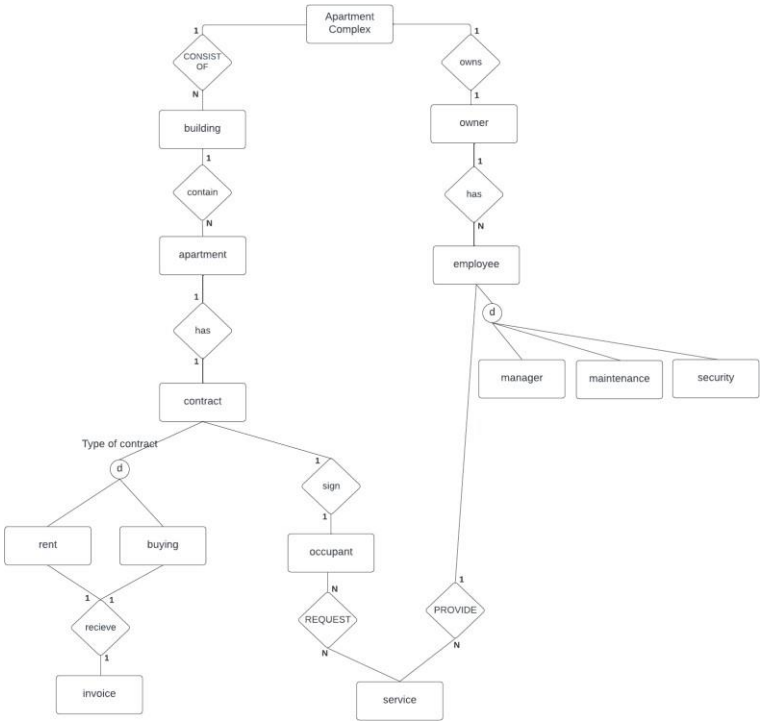
Information Needs:

The database system should display apartment details, including the building it belongs to and its location. It needs to track apartment availability, occupant information, and financial transactions such as rent payments and purchases. Supporting various payment methods and managing staff details, including schedules and tasks, are essential. Additionally, the system should monitor services like security and maintenance, integrating them into the final bill. Efficiency in information retrieval is crucial for the system's performance.

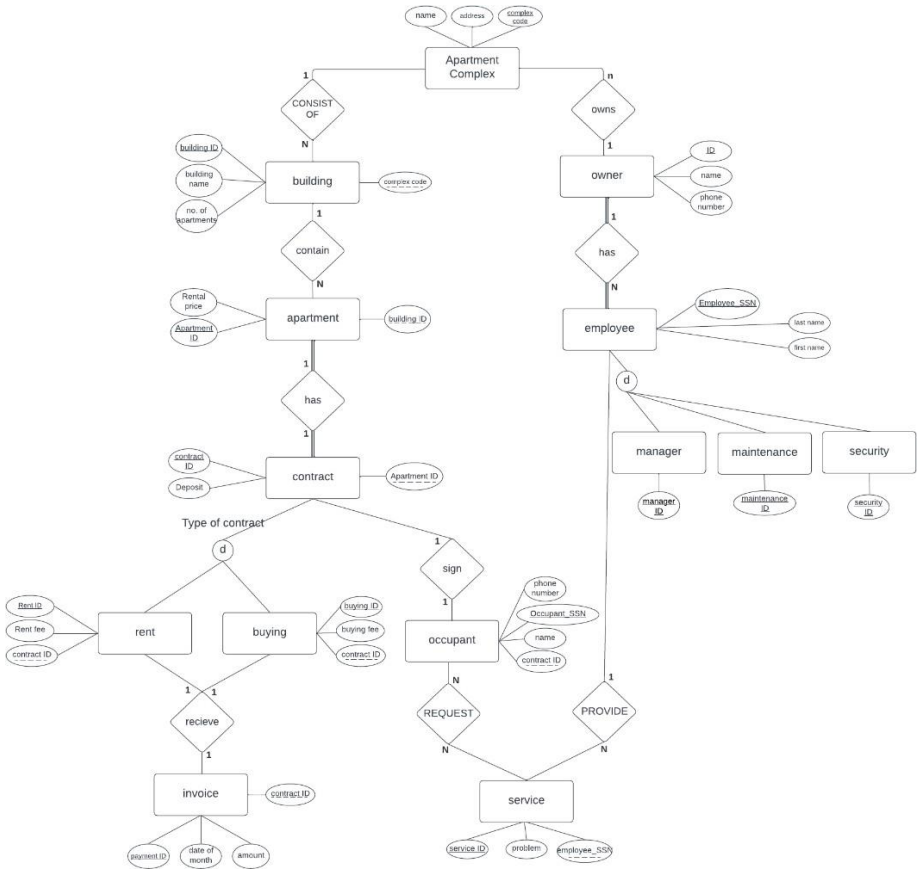
List of Entities:

- Apartment Complex
- Owner
- Building
- Apartment
- contract
- Occupant
- manager
- Maintenance
- Security
- Employee
- Rent
- invoice

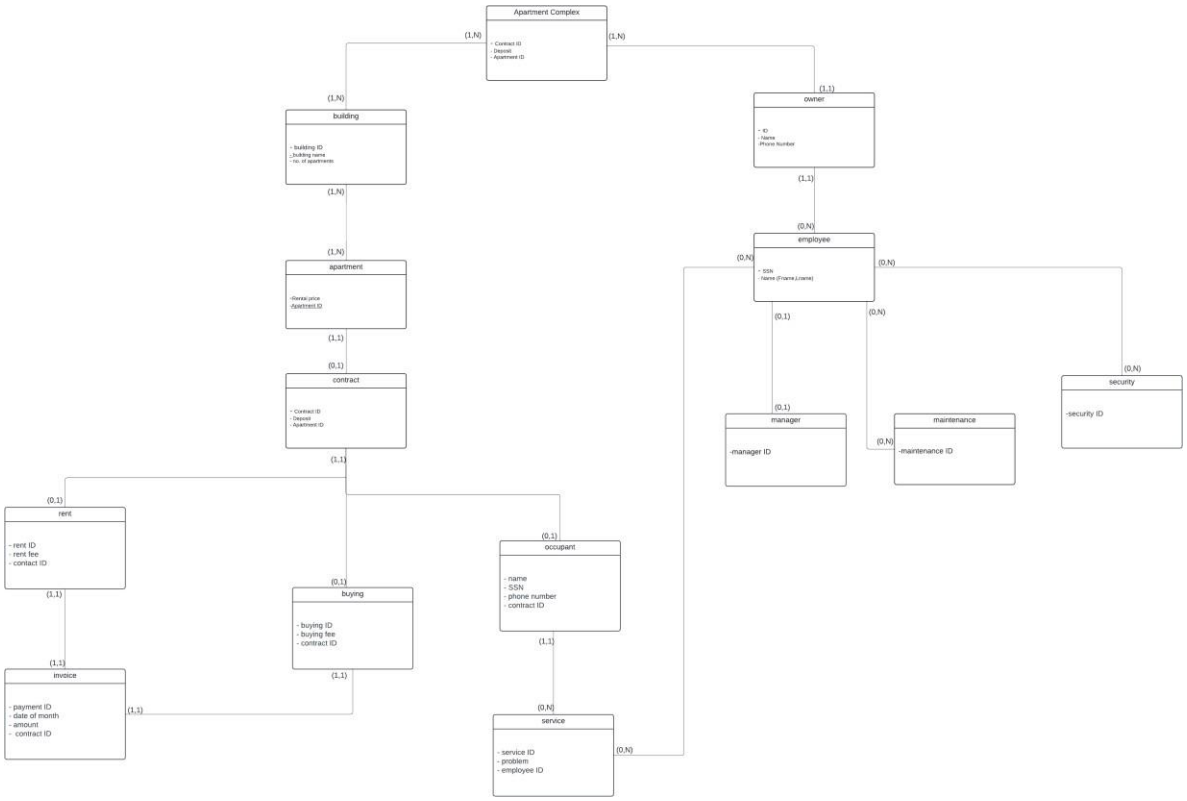
Conceptual Model:



Logical Model:

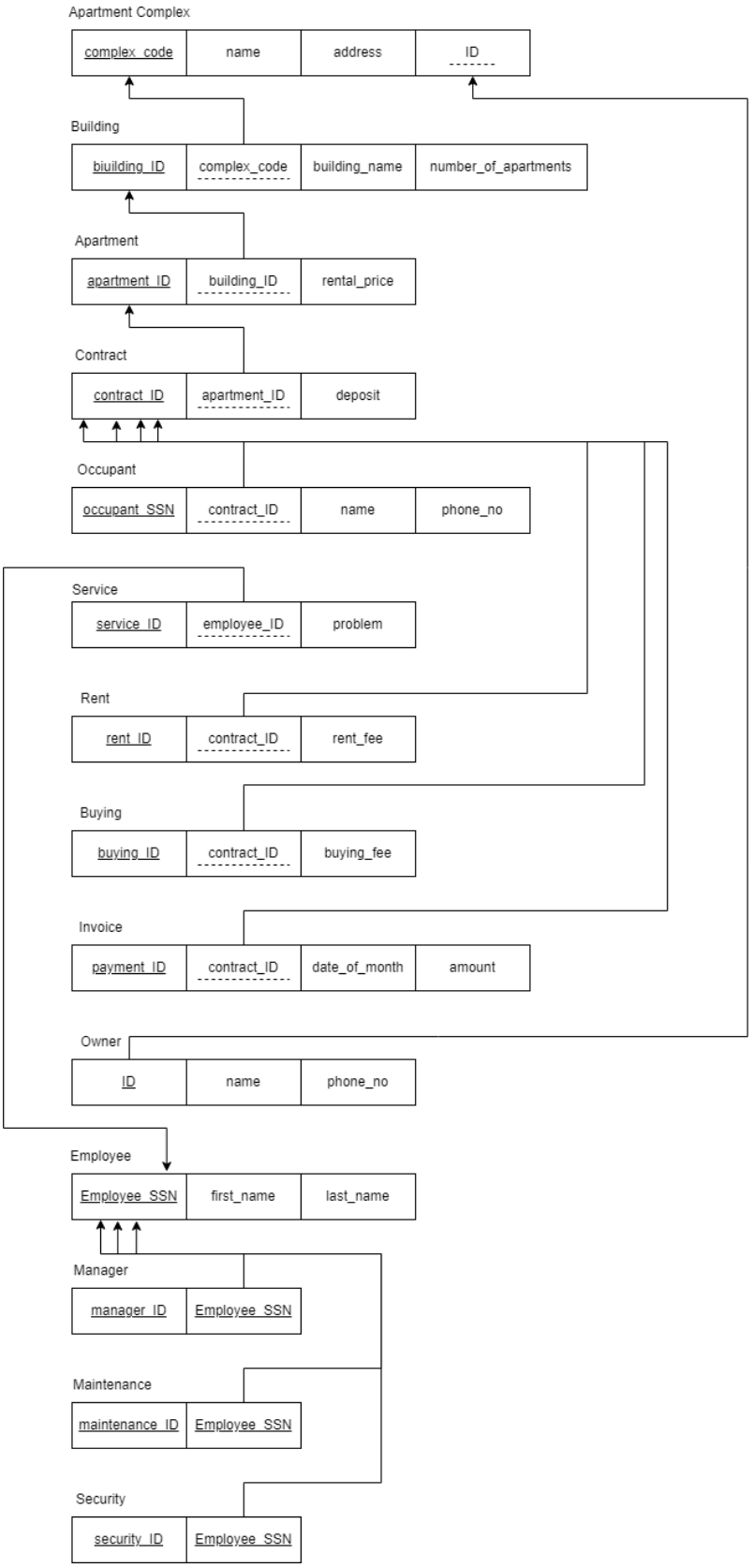


UML Class:


Apartment Complex
- Contract ID
- Deposit
- Apartment ID
(1,N)
(1,N)
building
- building ID
- building name
- no. of apartments
(1,N)
(1,N)
apartment
-Rental price
-Apartment ID
(1,1)
(0,1)
contract
- Contract ID
- Deposit
- Apartment ID
(1,1)
(0,1)
rent
- rent ID
- rent fee
- contract ID
(1,1)
(1,1)
invoice
- payment ID
- date of month
- amount
- contract ID
(1,1)
(0,1)
buying
- buying ID
- buying fee
- contract ID
(1,1)
(0,1)
occupant
- name
- SSN
- phone number
- contract ID
(1,1)
(0,N)
service
- service ID
- problem
- employee ID
(0,N)
(1,1)
owner
- ID
- Name
-Phone Number
(1,1)
(0,N)
employee
- SSN
- Name (Fname,Lname)
(0,N)
(0,N)
(0,1)
(0,N)
(0,1)
manager
-manager ID
(0,N)
maintenance
-maintenance ID
(0,N)
security
-security ID

Phase 2:

Relations Diagram:



Normalization:

1. 1NF:

- ✓ Remove repeating group.
- ✓ Each relation has a primary key.
- ✓ All attributes have functional dependency either full or partial of the key.
- ✓ All attribute atomic (no composite or multivalued).

Schema is already in 1NF. No multivalued or composite attributes with functional dependencies. Occupant and Employee and Owner each have different name for their SSN columns.

2. 2NF

- ✓ The relation is in 1 NF.
- ✓ Remove partial dependency and all non-key attributes are full dependent on the primary key.

Schema is in 2NF. Each entities' attributes are fully dependent on 1 primary key and are defined by it.

Apartment complex is only defined by the apartment code.

Each of building, apartment, and contract use a unique id while having a foreign key from the previous entity.

Rent, buying, invoice, and occupant have unique id's and have a foreign key from contract.

Service contains a unique id and a foreign key from employee.

Owner has a unique ID. Employee has SSN that goes to its different types manager, maintenance, security who each have a unique ID.

3. 3NF

- ✓ The relation is in 2NF.
- ✓ Remove transitive dependencies.

Schema satisfies all rules. There is no non-key attribute that defines any other attribute.

Creating Tables:

Create

1- Create owner table:

SQL Worksheet

```
CREATE TABLE OWNER(
    ID number PRIMARY KEY,
    name varchar2(20),
    phone_number number
);
```

```
Table created.
```

2- Create Apartment_Complex table:

SQL Worksheet

```
CREATE TABLE Apartment_Complex (
    complex_code NUMBER PRIMARY KEY,
    address VARCHAR2(30),
    name VARCHAR2(20),
    ID number,
    CONSTRAINT fk_complex_code
    FOREIGN KEY (ID)
    REFERENCES OWNER(ID)
);
```

```
Table created.
```

3- Create employee table:

SQL Worksheet

```
CREATE TABLE employee(
    employee_SSN number PRIMARY KEY,
    first_name VARCHAR2(20),
    last_name VARCHAR(20),
    ID number,
    CONSTRAINT fk_employee
    FOREIGN KEY (ID)
    REFERENCES OWNER(ID)
);
```

```
Table created.
```

4- Create manger table:

SQL Worksheet

```
CREATE TABLE manger(
    manger_id number PRIMARY KEY,
    employee_SSN number,
    CONSTRAINT fk_manger
    FOREIGN KEY (employee_SSN)
    REFERENCES employee(employee_SSN)
);
```

```
Table created.
```

5- Create maintenance table:

SQL Worksheet

```
CREATE TABLE maintenance(
    maintenance_id number PRIMARY KEY,
    employee_SSN number,
    CONSTRAINT fk_maintenance
    FOREIGN KEY (employee_SSN)
    REFERENCES employee(employee_SSN)
);
```

```
Table created.
```

6- Create security table:

SQL Worksheet

```sql
CREATE TABLE security(
      security_id number PRIMARY KEY,
      employee_SSN number,
      CONSTRAINT fk_security
      FOREIGN KEY (employee_SSN)
      REFERENCES employee(employee_SSN)
  );
```

```
Table created.
```

7- Create service table:

SQL Worksheet

```sql
CREATE TABLE service(
      survice_id number PRIMARY KEY,
      problem varchar2(30),
      employee_SSN number ,
      CONSTRAINT fk_service
      FOREIGN KEY (employee_SSN)
      REFERENCES employee(employee_SSN)
  );
```

```
Table created.
```

8- Create building table:

SQL Worksheet

```sql
CREATE TABLE building (
      building_ID NUMBER PRIMARY KEY,
      building_name VARCHAR2(20),
      no_of_apartment NUMBER,
      complex_code NUMBER,
      CONSTRAINT complex_code_fk FOREIGN KEY (complex_code)
      REFERENCES Apartment_Complex(complex_code)
  );
```

```
Table created.
```

9- Create apartment table:

SQL Worksheet

```sql
CREATE TABLE apartment(
    apartment_ID NUMBER PRIMARY KEY,
    rental_price NUMBER,
    building_ID NUMBER,
    CONSTRAINT building_ID_fk FOREIGN KEY (building_ID)
    REFERENCES building(building_ID)
);
```

```
Table created.
```

10- Create contract table:

SQL Worksheet

```sql
CREATE TABLE contract (
    contract_ID NUMBER PRIMARY KEY,
    deposit NUMBER,
    apartment_ID NUMBER,
    CONSTRAINT apartment_ID_fk FOREIGN KEY (apartment_ID)
    REFERENCES apartment(apartment_ID)
);
```

```
Table created.
```

11- Create occupant table:

SQL Worksheet

```sql
CREATE TABLE occupant(
    occupant_SSN NUMBER PRIMARY KEY,
    name VARCHAR2(20),
    phone_number NUMBER,
    contact_ID NUMBER,
    CONSTRAINT occupant_fk FOREIGN KEY (contact_ID)
    REFERENCES contract(contract_ID)
);
```

```
Table created.
```

12- Create buying table:

SQL Worksheet

```sql
CREATE TABLE buying(
    buying_ID NUMBER PRIMARY KEY,
    buying_fee NUMBER,
    contract_ID NUMBER,
    CONSTRAINT buying_fk FOREIGN KEY (contract_ID)
    REFERENCES contract(contract_ID)
);
```

```
Table created.
```

13- Create rent table:

SQL Worksheet

```sql
CREATE TABLE rent(
    rent_ID NUMBER PRIMARY KEY,
    rent_fee NUMBER,
    contract_ID NUMBER,
    CONSTRAINT rent_fk FOREIGN KEY (contract_ID)
    REFERENCES contract(contract_ID)
);
```

```
Table created.
```

14- Create invoice table:

SQL Worksheet

```sql
CREATE TABLE invoice (
    payment_ID NUMBER PRIMARY KEY,
    date_of_month NUMBER,
    amount NUMBER,
    contract_ID NUMBER,
    CONSTRAINT invoice_fk FOREIGN KEY (contract_ID)
    REFERENCES contract(contract_ID)
);
```

```
Table created.
```

15- Create occupant_service table:

SQL Worksheet Clear

```sql
CREATE TABLE occupant_service (
    occupant_SSN NUMBER,
    service_id NUMBER,
    CONSTRAINT pk_occupant_service PRIMARY KEY (occupant_SSN, service_id)
);
```

```
Table created.
```

Insert :

1- Insert into owner table:

SQL Worksheet

```sql
Insert into owner(ID,name, phone_number) Values(1, 'Lujain Omar', '050000000');
Insert into owner(ID,name, phone_number) Values(2, 'Majed mohmmed', '051000000');
Insert into owner(ID,name, phone_number) Values(3, 'Haneen Saleh', '051000009');
Insert into owner(ID,name, phone_number) Values(4, 'Rama Saead', '051200000');
Insert into owner(ID,name, phone_number) Values(5, 'Saad Samir', '051000800');
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

2- Insert into Apartment_Complex table:

SQL Worksheet

```sql
Insert into apartment_complex(complex_code,address,name, ID) Values(0561, 'AlRawdha', 'AlRawdha Complex', 1);
Insert into apartment_complex(complex_code,address,name, ID) Values(0562, 'AlSafa', 'AlSafa Complex', 3);
Insert into apartment_complex(complex_code,address,name, ID) Values(0563, 'AlMarwa', 'AlMarwa Complex', 4);
Insert into apartment_complex(complex_code,address,name, ID) Values(0564, 'Alryan', 'Alryan Complex', 2);
Insert into apartment_complex(complex_code,address,name, ID) Values(0565, 'AlSalama', 'AlSalama Complex', 5);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

3- Insert into employee table:

SQL Worksheet

```sql
Insert into employee(employee_SSN, first_name, last_name, ID) Values(1110, 'Mohammed', 'Khalid', 1);
Insert into employee(employee_SSN, first_name, last_name, ID) Values(1112, 'Salem', 'Al-ghamdi', 2);
Insert into employee(employee_SSN, first_name, last_name, ID) Values(1122, 'Jamila', 'Mohammed', 3);
Insert into employee(employee_SSN, first_name, last_name, ID) Values(1132, 'Nada', 'Khalid', 4);
Insert into employee(employee_SSN, first_name, last_name, ID) Values(1142, 'Jawaher', 'Sameer', 5);

Insert into employee(employee_SSN, first_name, last_name, ID) Values(2111, 'Abdullah', 'Zain', 1);
Insert into employee(employee_SSN, first_name, last_name, ID) Values(2211, 'Sara', 'Hady', 2);
Insert into employee(employee_SSN, first_name, last_name, ID) Values(2311, 'Ammar', 'Abullaziz', 4);
Insert into employee(employee_SSN, first_name, last_name, ID) Values(2133, 'Shourg', 'Faisal', 3);
Insert into employee(employee_SSN, first_name, last_name, ID) Values(2131, 'Yara', 'Nasser', 3);

Insert into employee(employee_SSN, first_name, last_name, ID) Values(3112, 'Sami', 'Alsulami', 1);
Insert into employee(employee_SSN, first_name, last_name, ID) Values(3312, 'Fatima', 'Musleh', 5);
Insert into employee(employee_SSN, first_name, last_name, ID) Values(3122, 'Eman', 'Khalid', 3);
```

```
1 row(s) inserted.
```

4- Insert into manger table :

SQL Worksheet

```sql
Insert into manger(manger_id, employee_SSN) Values(2220, 1110);
Insert into manger(manger_id, employee_SSN) Values(2221, 1112);
Insert into manger(manger_id, employee_SSN) Values(2222, 1122);
Insert into manger(manger_id, employee_SSN) Values(2223, 1132);
Insert into manger(manger_id, employee_SSN) Values(2224, 1142);
```

5- Insert into maintenance table:

SQL Worksheet

```sql
Insert into maintenance(maintenance_id, employee_SSN) Values(2346, 2111);
Insert into maintenance(maintenance_id, employee_SSN) Values(2344, 2131);
Insert into maintenance(maintenance_id, employee_SSN) Values(2345, 2211);
Insert into maintenance(maintenance_id, employee_SSN) Values(2347, 2311);
Insert into maintenance(maintenance_id, employee_SSN) Values(2348, 2133);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

6- Insert into security table:

SQL Worksheet

```sql
Insert into security (security_ID, employee_SSN) Values(2444, 3112);
Insert into security (security_ID, employee_SSN) Values(2424, 3312);
Insert into security (security_ID, employee_SSN) Values(2434, 3122);
Insert into security (security_ID, employee_SSN) Values(2414, 3142);
Insert into security (security_ID, employee_SSN) Values(2422, 3115);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

7- Insert into service table:

SQL Worksheet

```sql
Insert into service(survice_id , problem, employee_SSN) Values(143, 'Plumbing', 2133);
Insert into service(survice_id , problem, employee_SSN) Values(234, 'Losing car keys', 3115);
Insert into service(survice_id, problem, employee_SSN) Values(323, 'where to put bicycle', 3312);
Insert into service(survice_id, problem, employee_SSN) Values(422, 'door knob fell', 2131);
Insert into service(survice_id, problem, employee_SSN) Values(223, 'car wash', 3142);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

8- Insert into building table:

SQL Worksheet

```sql
Insert into building(building_ID, building_name, no_of_apartment, complex_code) Values(21, 'Gardenia', 14, 0562);
Insert into building(building_ID, building_name, no_of_apartment, complex_code) Values(22, 'Gardenia', 8, 0563);
Insert into building(building_ID, building_name, no_of_apartment, complex_code) Values(11, 'Panorama', 10, 0561);
Insert into building(building_ID, building_name, no_of_apartment, complex_code) Values(33, 'Panorama', 6, 0564);
Insert into building(building_ID, building_name, no_of_apartment, complex_code) Values(12, 'Panorama', 9, 0565);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

9- Insert into apartment table:

SQL Worksheet

```sql
Insert into apartment(apartment_ID, rental_price, building_ID) Values(53, 34000, 22);
Insert into apartment(apartment_ID, rental_price, building_ID) Values(54, 34000, 22);
Insert into apartment(apartment_ID, rental_price, building_ID) Values(55, 30000, 21);
Insert into apartment(apartment_ID, rental_price, building_ID) Values(56, 30000, 21);
Insert into apartment(apartment_ID, rental_price, building_ID) Values(57, 30000, 21);

Insert into apartment(apartment_ID, rental_price, building_ID) Values(41, 27000, 11);
Insert into apartment(apartment_ID, rental_price, building_ID) Values(42, 27000, 11);
Insert into apartment(apartment_ID, rental_price, building_ID) Values(43, 35000, 33);
Insert into apartment(apartment_ID, rental_price, building_ID) Values(14, 30000, 12);
Insert into apartment(apartment_ID, rental_price, building_ID) Values(45, 30000, 12);
```

```
1 row(s) inserted.

1 row(s) inserted.
```

10- Insert into contract table:

SQL Worksheet

```sql
INSERT INTO contract (contract_ID, deposit, apartment_ID)
VALUES (563, 4000, 141);
INSERT INTO contract (contract_ID, deposit, apartment_ID)
VALUES (564, 4000, 142);
INSERT INTO contract (contract_ID, deposit, apartment_ID)
VALUES (565, 4000, 143);
INSERT INTO contract (contract_ID, deposit, apartment_ID)
VALUES (566, 4000, 144);
INSERT INTO contract (contract_ID, deposit, apartment_ID)
VALUES (567, 4000, 145);
INSERT INTO contract (contract_ID, deposit, apartment_ID)
VALUES (568, 5000, 153);
INSERT INTO contract (contract_ID, deposit, apartment_ID)
```

```sql
VALUES (568, 5000, 153);
INSERT INTO contract (contract_ID, deposit, apartment_ID)
VALUES (569, 5000, 154);
INSERT INTO contract (contract_ID, deposit, apartment_ID)
VALUES (570, 5000, 155);
INSERT INTO contract (contract_ID, deposit, apartment_ID)
VALUES (571, 5000, 156);
INSERT INTO contract (contract_ID, deposit, apartment_ID)
VALUES (572, 5000, 157);
```

11- Insert into occupant table:

SQL Worksheet

```sql
INSERT INTO occupant (occupant_SSN, name, phone_number, contact_ID )
VALUES (1001, 'Raghad', '0500000', 566);
INSERT INTO occupant (occupant_SSN, name, phone_number, contact_ID )
VALUES (1002, 'Batool', '0500000', 567);
INSERT INTO occupant (occupant_SSN, name, phone_number, contact_ID )
VALUES (1003, 'Hanan', '0500000', 568);
INSERT INTO occupant (occupant_SSN, name, phone_number, contact_ID )
VALUES (1004, 'Shoroog', '0500000', 569);
INSERT INTO occupant (occupant_SSN, name, phone_number, contact_ID )
VALUES (1005, 'Elaf', '0500000', 570);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

12- Insert into rent table:

SQL Worksheet

```sql
INSERT INTO rent (rent_ID, rent_fee, contract_ID)
VALUES (11, 4000, 566);
INSERT INTO rent (rent_ID, rent_fee, contract_ID)
VALUES (12, 4000, 567);
INSERT INTO rent (rent_ID, rent_fee, contract_ID)
VALUES (13, 4000, 568);
INSERT INTO rent (rent_ID, rent_fee, contract_ID)
VALUES (14, 4000, 569);
INSERT INTO rent (rent_ID, rent_fee, contract_ID)
VALUES (15, 4000, 570);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

13- Insert into buying table:

SQL Worksheet

```sql
INSERT INTO rent (rent_ID, rent_fee, contract_ID)
VALUES (11, 4000, 566);
INSERT INTO rent (rent_ID, rent_fee, contract_ID)
VALUES (12, 4000, 567);
INSERT INTO rent (rent_ID, rent_fee, contract_ID)
VALUES (13, 4000, 568);
INSERT INTO rent (rent_ID, rent_fee, contract_ID)
VALUES (14, 4000, 569);
INSERT INTO rent (rent_ID, rent_fee, contract_ID)
VALUES (15, 4000, 570);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

14- Insert into invoice table:

SQL Worksheet Clear

```sql
INSERT INTO invoice (payment_id, date_of_month, amount, contract_id)
VALUES (180, 2, 4000, 563);
INSERT INTO invoice (payment_id, date_of_month, amount, contract_id)
VALUES (181, 5, 4000, 564);
INSERT INTO invoice (payment_id, date_of_month, amount, contract_id)
VALUES (182, 7 ,4000, 565);
INSERT INTO invoice (payment_id, date_of_month, amount, contract_id)
VALUES (183, 1, 5000, 566);
INSERT INTO invoice (payment_id, date_of_month, amount, contract_id)
VALUES (184, 9, 5000, 567);
INSERT INTO invoice (payment_id, date_of_month, amount, contract_id)
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

15- Insert into occupant_service table:

SQL Worksheet

```sql
INSERT INTO occupant_service (occupant_SSN ,service_id ) VALUES(1001,143);
INSERT INTO occupant_service (occupant_SSN ,service_id ) VALUES(1002,234);
INSERT INTO occupant_service (occupant_SSN ,service_id ) VALUES(1003,323);
INSERT INTO occupant_service (occupant_SSN ,service_id ) VALUES(1004,422);
INSERT INTO occupant_service (occupant_SSN ,service_id ) VALUES(1005,223);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

Output of tables:

Select statements:

1- Select owner table:

SQL Worksheet

```sql
SELECT *
FROM owner;
```

ID	NAME	PHONE_NUMBER
1	Lujain Omar	50000000
2	Majed mohmmed	51000000
3	Haneen Saleh	51000009
4	Rama Saead	51200000
5	Saad Samir	51000800

2- Select apartment_complex table:

SQL Worksheet

```
SELECT *
FROM apartment_complex;
```

COMPLEX_CODE	ADDRESS	NAME	ID
561	AlRawdha	AlRawdha Complex	1
562	AlSafa	AlSafa Complex	3
563	AlMarwa	AlMarwa Complex	4
564	Alryan	Alryan Complex	2
565	AlSalama	AlSalama Complex	5

3- Select employee table:

SQL Worksheet Clear Find

```
SELECT *
FROM employee;
```

EMPLOYEE_SSN	FIRST_NAME	LAST_NAME	ID
1110	Mohammed	Khalid	1
1112	Salem	Al-ghamdi	2
1122	Jamila	Mohammed	3
1132	Nada	Khalid	4
1142	Jawaher	Sameer	5
2111	Abdullah	Zain	1

1132	Nada	Khalid	4
1142	Jawaher	Sameer	5
2111	Abdullah	Zain	1
2211	Sara	Hady	2
2311	Ammar	Abullaziz	4
2133	Shourg	Faisal	3
2131	Yara	Nasser	3
3112	Sami	Alsulami	1
3312	Fatima	Musleh	5

SQL Worksheet

2133	Shourg	Faisal	3
2131	Yara	Nasser	3
3112	Sami	Alsulami	1
3312	Fatima	Musleh	5
3122	Eman	Khalid	3
3142	Alanoud	Jameel	4
3115	Raad	Yousef	2

Download CSV

15 rows selected.

4- Select manger table:

SQL Worksheet

```
SELECT *
FROM manger;
```

MANGER_ID	EMPLOYEE_SSN
2220	1110
2221	1112
2222	1122
2223	1132
2224	1142

Download CSV

5- Select maintenance table:

SQL Worksheet

```
SELECT *
FROM maintenance;
```

MAINTENANCE_ID	EMPLOYEE_SSN
2346	2111
2344	2131
2345	2211
2347	2311
2348	2133

Download CSV

6- Select security table:

SQL Worksheet

```
SELECT *
FROM security ;
```

SECURITY_ID	EMPLOYEE_SSN
2444	3112
2424	3312
2434	3122
2414	3142
2422	3115

Download CSV

7- Select Occupant

```
select * from occupant
```

OCCUPANT_SSN	NAME	PHONE_NUMBER	CONTACT_ID
1001	Raghad	500000	566
1002	Batool	500000	567
1003	Hanan	500000	568
1004	Shoroog	500000	569
1005	Elaf	500000	570

8- Select rent

```
select * from rent
```

RENT_ID	RENT_FEE	CONTRACT_ID
11	4000	566
12	4000	567
13	4000	568
14	4000	569
15	4000	570

9- select buying

```
select * from buying
```

BUYING_ID	BUYING_FEE	CONTRACT_ID
22	27000	563
23	27000	564
24	27000	565
25	27000	566
26	27000	567

10- Select invoice

```
select * from invoice
```

PAYMENT_ID	DATE_OF_MONTH	AMOUNT	CONTRACT_ID
180	2	4000	563
181	5	4000	564
182	7	4000	565
183	1	5000	566
184	9	5000	567

11- Select service

```
select * from service
```

SURVICE_ID	PROBLEM	EMPLOYEE_SSN
143	Plumbing	2133
234	Losing car keys	3115
323	where to put bicycle	3312
422	door knob fell	2131
223	car wash	3142

12- Select apartment

```
select * from apartment
```

APARTMENT_ID	RENTAL_PRICE	BUILDING_ID
153	34000	22
154	34000	22
155	30000	21
156	30000	21
157	30000	21
141	27000	11
142	27000	11
143	35000	33
144	30000	12
145	30000	12
14	30000	12
53	34000	22

54	34000	22
55	30000	21
56	30000	21
57	30000	21
41	27000	11
42	27000	11
43	35000	33
45	30000	12

13- Select Building

```
select * from building
```

BUILDING_ID	BUILDING_NAME	NO_OF_APARTMENT	COMPLEX_CODE
21	Gardenia	14	562
22	Gardenia	8	563
11	Panorama	10	561
33	Panorama	6	564
12	Panorama	9	565

14- Select Contract

```
select* from contract
```

CONTRACT_ID	DEPOSIT	APARTMENT_ID
563	4000	141
564	4000	142
565	4000	143
566	4000	144
567	4000	145
568	5000	153
569	5000	154
570	5000	155
571	5000	156
572	5000	157

Queries:

1- Retrieves data from two tables, contract and occupant, using an **INNER JOIN** operation

SQL Worksheet

```
SELECT contract.contract_ID, occupant.name
FROM contract
INNER JOIN occupant ON contract.contract_ID = occupant.contact_ID
ORDER BY occupant.name;
```

CONTRACT_ID	NAME
567	Batool
570	Elaf
568	Hanan
566	Raghad
569	Shoroog

2- Retrieves the **minimum** name and complex_code for each **group** of records with the same address from the apartment_complex

SQL Worksheet

```
SELECT MIN(name) AS name, address, MIN(complex_code) AS complex_code
FROM apartment_complex
GROUP BY address
ORDER BY complex_code;
```

NAME	ADDRESS	COMPLEX_CODE
AlRawdha Complex	AlRawdha	561
AlSafa Complex	AlSafa	562
AlMarwa Complex	AlMarwa	563
Alryan Complex	Alryan	564
AlSalama Complex	AlSalama	565

2024 Oracle · Live SQL 24.1.3, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 · Data

3- retrieves data from the occupant table using **nested** select.

SQL Worksheet

```
SELECT occupant.occupant_SSN,occupant.name, occupant.phone_number
FROM occupant
WHERE contact_ID IN (
    SELECT contract_ID
    FROM contract
    WHERE apartment_ID IN (
        SELECT apartment_ID
        FROM apartment
        WHERE rental_price > 30000
    ));
```

OCCUPANT_SSN	NAME	PHONE_NUMBER
1003	Hanan	500000
1004	Shoroog	500000

4- Display service ID and the employee who handled it

SQL Worksheet

Clear

```sql
SELECT service.survice_id, service.employee_SSN
FROM service
WHERE problem IN ('Losing car keys', 'Plumbing')
ORDER BY survice_id;
```

SURVICE_ID	EMPLOYEE_SSN
143	2133
234	3115

Procedure:

1- procedure, **GetRecords**, is designed to retrieve records from the employee table based on the provided first_name and last_name parameters.

```sql
CREATE OR REPLACE PROCEDURE GetRecords(
    temp_first_name IN employee.first_name%TYPE,
    temp_last_name IN employee.last_name%TYPE,
    emp_record OUT employee%ROWTYPE)
AS
BEGIN
    SELECT *
    INTO emp_record
    FROM employee
    WHERE first_name = temp_first_name AND last_name = temp_last_name;
END;
```

Procedure created.

Result:

```
DECLARE
    v_emp_record employee%ROWTYPE;
BEGIN
    GetRecords('Mohammed', 'Khalid', v_emp_record);
    DBMS_OUTPUT.PUT_LINE('First Name: ' || v_emp_record.first_name);
    DBMS_OUTPUT.PUT_LINE('Last Name: ' || v_emp_record.last_name);
END;
```

```
Statement processed.
First Name: Mohammed
Last Name: Khalid
```

2- The provided procedure, named **Update_Record**, is designed to update records in the invoice table.

SQL Worksheet

```
CREATE PROCEDURE Update__Record(
    date_of_month number,
    amount number)
AS
BEGIN
    UPDATE invoice
    SET amount = 3600
    WHERE date_of_month = 1;
END;
```

```
Procedure created.
```

Result:

```
exec update__record(1, 5000)
```

```
Statement processed.
```

Now when we see invoice again:

```
select * from invoice
where date_of_month = 1;
```

PAYMENT_ID	DATE_OF_MONTH	AMOUNT	CONTRACT_ID
183	1	3600	566

3- A stored procedure which checks for an attribute value, compares it with the value of an argument, and displays the results to the screen, **using cursor.**

SQL Worksheet Clear Find

```
CREATE OR REPLACE PROCEDURE rentalcheck (MinRentalPrice NUMBER) AS
    CURSOR c_rental IS
        SELECT apartment_ID, rental_price
        FROM apartment
        WHERE rental_price > MinRentalPrice;
BEGIN
    FOR v_cursrec IN c_rental LOOP
        dbms_output.put_line(v_cursrec.apartment_ID || ' ' || v_cursrec.rental_price);
    END LOOP;
END rentalcheck;
```

```
Procedure created.
```

Result:

```
exec rentalcheck(27000)
```

```
Statement processed.
153 34000
154 34000
155 30000
156 30000
157 30000
143 35000
144 30000
145 30000
14 30000
53 34000
54 34000
55 30000
56 30000
57 30000
43 35000
```