FACULTY OF COMPUTING
& INFORMATION TECHNOLOGY
KING ABDULAZIZ UNIVERSITY

FCIT
KAU

كلية الحاسبات
وتقنية المعلومات
جامعة الملك عبدالعزيز

Hospital Management System

Prepared For:

Dr. Zaki Alsubhi

Prepared By:

Name	ID	Section
Naif Askul	2035052	ZK
Nawaf AlGhamdi	2037159	ZK
Tareq Fuad Bajaber	2035796	ZK
Faisal Balamash	2036827	ZK

April 19, 2022

Table of Contents

Part 1

1) About the Domain

A hospital is a place where injured and sick people are treated and taken care of. It ensures the delivery of maximum health services, care, cure, and preventive services for each patient. Daily hundreds of patients visit hospitals to get treated by doctors and other employees for assistance. With that amount of data circulating, we need to accumulate them in one single unit, our project provides the accumulation, organizing, and maintenance of the data for the departments of hospitals.

The domain of the database coordinates **Departments, Employees**, **Doctors**, **Patients**, and **Rooms**. We aim to improve data access, making entities receive quick answers to the data they want from the database. Data sharing because the management system creates an environment in which users have better access to more and better-managed data and users can respond quickly to changes in their environment. Data security management system reduces the data access for entities to prevent data breaches which can cause the domain time, effort, and money if the data is used wrongly. And finally, minimize data inconsistency which is when different versions of the same data appear in different places and the probability of this happening is greatly reduced in a properly designed database management system.

The hospital that uses the database management system should have a department's unique number, The Department must have an employee managing it, and the chosen employee can manage only one department.

The employee when assigned to the department must provide their ID, first name, last name, date of birth, salary, address, and sex. An Employee can have only one supervisor and multiple supervisees. Each employee must work in a department and a department can have multiple employees that work in it.

The doctor must have ID, first name, last name, date of birth, salary, address, and sex. Doctors must be part of a department, and the department can have multiple doctors that are part of it. Every room must have a unique numeric identifier and a location inside the hospital. Each doctor is assigned to a room and a room can be assigned to multiple doctors.

Finally, when a patient is checking in, they need to provide their ID, first name, last name, date of birth, sex, and address. Patients may attend multiple doctors, and each doctor may be attended by multiple patients during his working day.

2) Identify the entities and relation

a) List of all the entities and their relations literally:

- Each **Department** must have an ID and a name.
- Each **Employee** should have an ID, first & last name, date of birth, salary, address, and sex.
- Each **Doctor** should have an ID, first & last name, date of birth, salary, sex and address.
- Each **Patient** should have an ID, first & last name, date of birth, sex, and address.
- Each **Room** must have an ID and a location.
- Each **Employee** must work in a **Department**.
- Each **Department** can have multiple **Employees** that work in it.
- One **Employee** must manage a **Department**.
- Each **Department** must have an **Employee** managing it.
- Each **Doctor** must be a part of a **Department**.
- Each **Department** can have multiple **Doctors** that are part of it.
- Each **Doctor** is assigned to a **Room**.
- Each **Room** can be assigned to multiple **Doctors.**
- Each **Patient** may attend multiple **Doctors**.
- Each **Doctor** may be attended by multiple **Patients**.
- Each **Employee** can have one **Employee** supervising him.
- Each **Employee** can supervise multiple **Employees**.

b) List of all the entities separately

- Department
- Employee
- Doctor
- Patient
- Room

c) **Map of relations between entities.**

- Employee MANAGES Department.
- Employee WORK IN Department.
- Employee(as Supervisor) SUPERVISOR Employee(as Supervisee)
- Doctor PART OF Department.

- Patient ATTEND Doctor.
- Doctor ASSIGNED to Room.

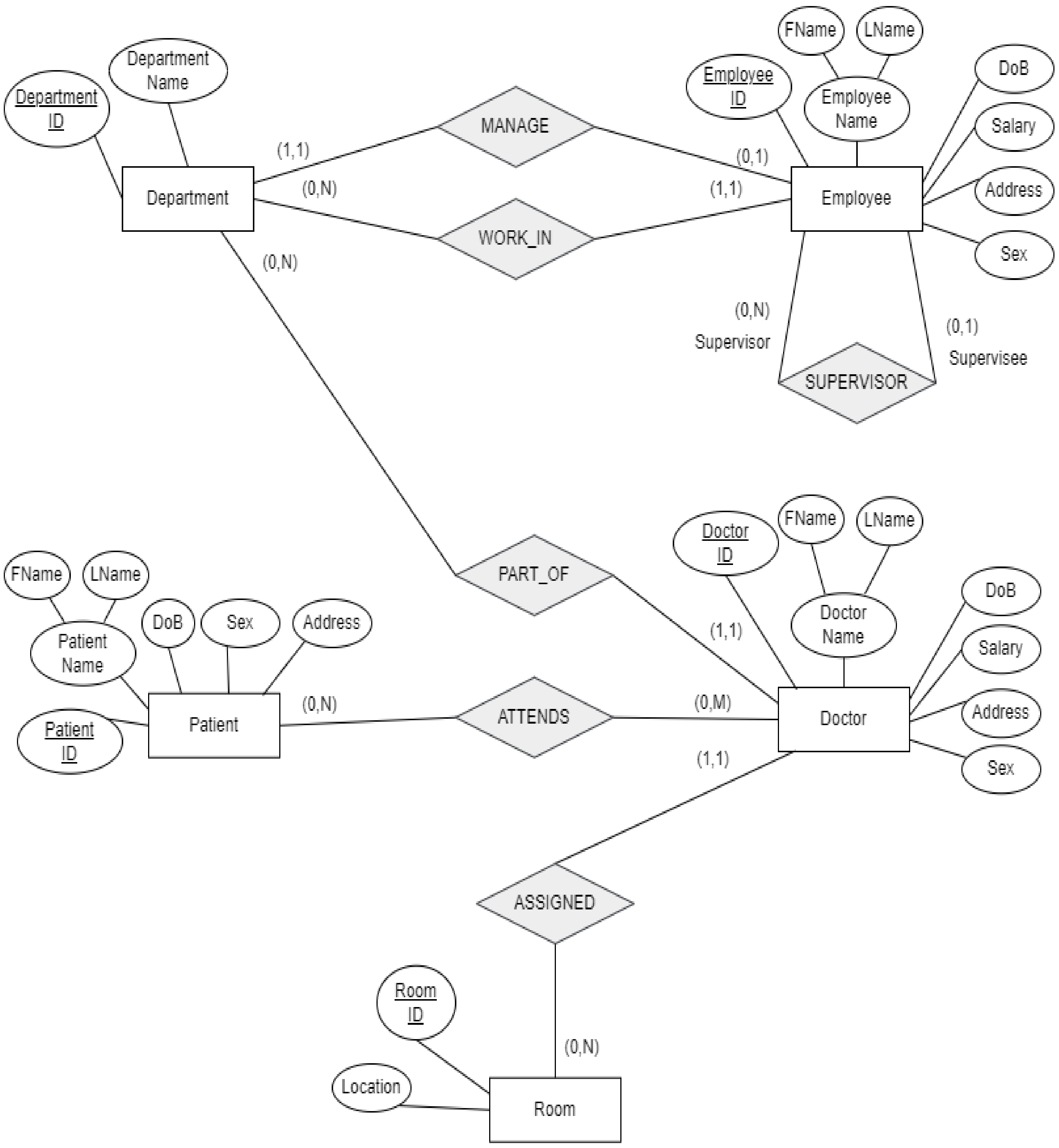
d) List of the attributes associated with each entity

- **Department:**
  - <u>Department ID</u>
  - Department Name
- **Employee:**
  - <u>Employee ID</u>
  - Employee Name (First & Last Name)
  - Date of Birth
  - Salary
  - Address
  - Sex
- **Doctor:**
  - <u>Doctor ID</u>
  - Doctor Name (First & Last Name)
  - Date of Birth
  - Salary
  - Address
  - Sex
- **Patient:**
  - <u>Patient ID</u>
  - Patient Name (First & Last Name)
  - Date of Birth
  - Address
  - Sex
- **Room:**
  - <u>Room ID</u>
  - Location

e) Identifying the candidate key, primary key, composite key, super key and etc…

- Employee ID is the primary key of Employee.
- Department ID is a foreign key in Employee.
- Supervisor ID is a foreign key in Employee.
- Department ID is the primary key of Department.
- Manager ID is a foreign Key in Department.
- Doctor ID is the primary key of Doctor.
- Room ID is a foreign key in Doctor.
- Department ID is a foreign key in Doctor.
- Patient ID is primary key in Patient.
- Room ID is primary key of Room.
- Patient ID & Doctor ID are composite key in Attends.

3) ER Diagram



Department Name
Department ID
Department
MANAGE
WORK_IN
(1,1)
(0,N)
(0,1)
(1,1)
FName
LName
Employee ID
Employee Name
DoB
Salary
Address
Sex
Employee
(0,N)
Supervisor
(0,1)
Supervisee
SUPERVISOR
(0,N)
PART_OF
Doctor ID
FName
LName
Doctor Name
DoB
Salary
Address
Sex
Doctor
(1,1)
(0,M)
(1,1)
FName
LName
Patient Name
DoB
Sex
Address
Patient ID
Patient
(0,N)
ATTENDS
ASSIGNED
(0,N)
Room ID
Location
Room

4) Ten possible queries according to the ER diagram

1. **Retrieve the Doctor ID, First name, and Last Name of all doctors.**
2. **Show the records of male employees.**
3. **Show the records of employee who are managing Departments.**
4. **Show the Department ID of the Medical Department.**
5. **Create a view which retrieves the Doctor ID, first name, and last name of all doctors who are a part of Department ID 2.**
6. **Shows the Salaries of doctors that are higher than the average.**
7. **Count how many patients attend each doctor.**
8. **Show the Patients id, Doctors id, and first name who their first name matches.**
9. **Show the Room ID's of the rooms that are not assigned to a doctor.**
10. **Show Employee ID, first name, and last name of employees who are not supervising anybody.**

Part 2

1) Translate ER into Schema

Employee

Employee ID	FName	LName	Dob	Salary	Address	Sex
Supervisor ID	Department ID					

It has Employee ID as primary key and the rest of the Employee attributes.
It has Department ID as foreign key due to the relation between Employee and Department is (1:N). It has Supervisor ID as foreign key due to recursive relationship and the relation is (1:N).

Department

Department ID	Department Name	Manager ID

It has Department ID as primary key and rest of the Department attributes.
Manager ID as foreign key due to the relation between the Department and Employee is (1:1).

Doctor

Doctor ID	FName	LName	Dob	Salary	Address	Sex
Room ID	Department ID					

It has Doctor ID as primary key and rest of the Doctor attributes.
It has Room ID and Department ID as foreign keys due to the relation between both of them are (1:M).

Patient

Patient ID	FName	LName	Dob	Address	Sex

It has Patient ID as primary key and the rest of the Patient attributes.

Room

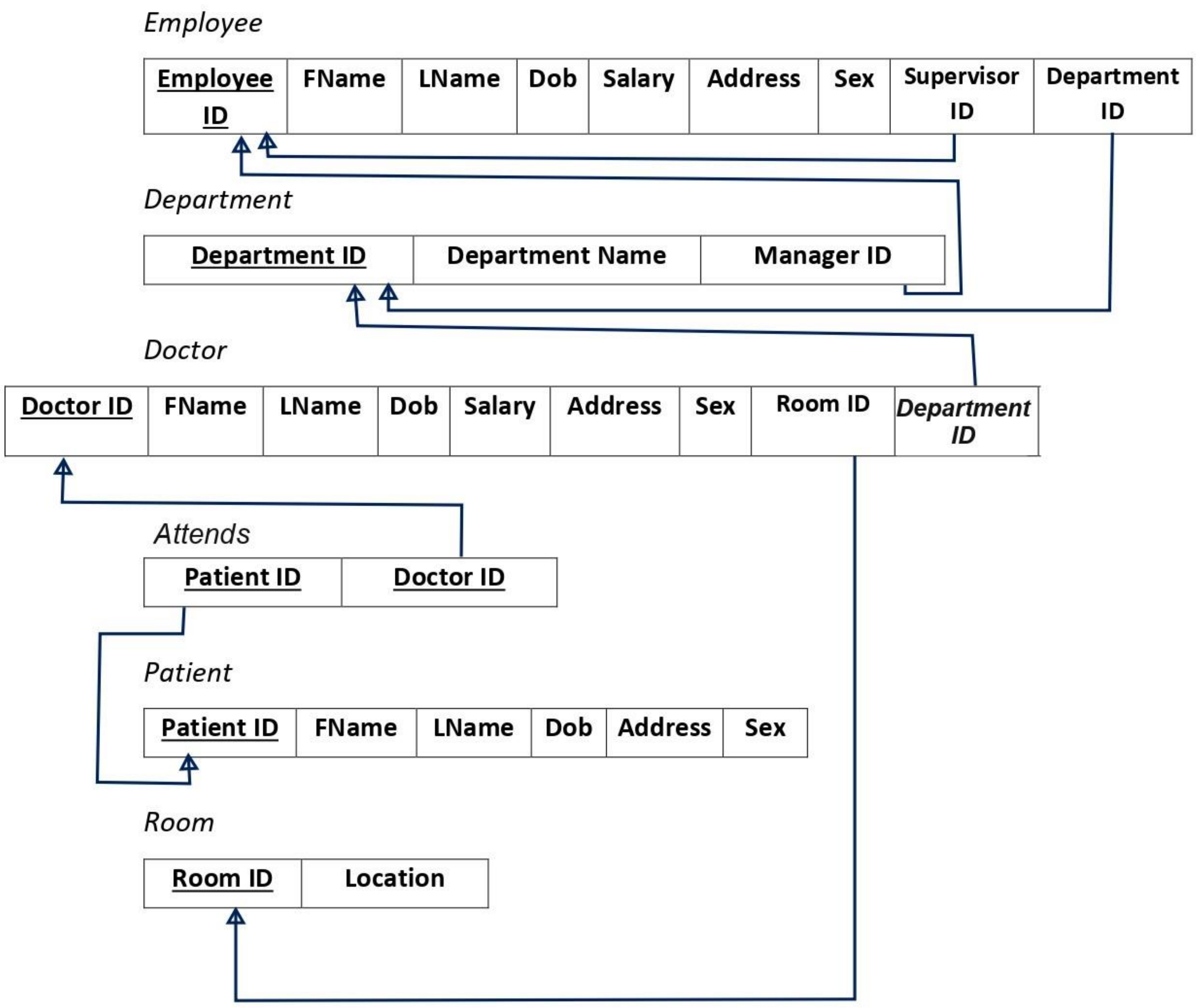
Room ID	Location

It has Room ID as primary key and Location as an attribute.

Attends

<u>**Patient ID**</u>	<u>**Doctor ID**</u>

It has the Patient ID and Doctor ID where they are composite key due to the relation between them (M:N).



2) Implementation of the Schema

```
CREATE table Department(Department_ID integer constraint D_pk
primary key, Department_Name varchar2(25) constraint deptname
unique, Manager_ID integer);
```

- ❖ Department_ID is the primary key, and the Department_Name must be unique.

```
CREATE table Employee(Employee_ID integer constraint e_pk primary
key, Fname varchar2(15), Lname varchar2(15), Dob date, salary
number(8,2), Address varchar2(25), Sex varchar2(7), Supervisor_ID
constraint svf references Employee (Employee_ID) on delete set
null, Department_ID constraint dif references
Department(Department_ID) on delete set null);
```

- ❖ Employee_ID is the primary key, Department_ID is a foreign key from Department,. and Supervisor_ID is a foreign key from the sametable(Employee).

```
ALTER table Department add constraint dc FOREIGN KEY (Manager_ID)
references Employee(Employee_ID) on delete set null;
```

- ❖ Making the Manager_ID foreign key from the Employee, couldn't be done before because the table did not exist.

```
CREATE table Room(Room_ID integer constraint R_pk primary key,
Location varchar2(25));
```

- ❖ Room_ID is the primary key.

```
CREATE table Doctor(Doctor_ID integer constraint Do_pk primary key,
Fname varchar2(15), Lname varchar2(15), Dob date, salary
number(8,2), Address varchar2(25), Sex varchar2(7), Room_ID
constraint rif references Room(Room_ID) on delete set null,
Department_ID constraint deptif references
Department(Department_ID) on delete set null);
```

- ❖ Doctor_ID is the primary key, Room_ID is a foreign key from Room, Department_ID is a foreign key from Department.

```
CREATE table Patient(Patient_ID integer constraint Patient_pk
primary key, Fname varchar2(15), Lname varchar2(15), Dob date,
Address varchar2(25), Sex varchar2(7));
```

- ❖ Patient_ID is the primary key.

```
CREATE table Attends(Patient_ID constraint pifk references
Patient(Patient_ID) on delete set null, Doctor_ID
constraint difk references Doctor(Doctor_ID) on delete set
null, constraint Attends_pk primary key
(Patient_ID,Doctor_ID));
```

- ❖ Patient_ID is a foreign key from Patient, Doctor_ID is a foreign key from Doctor, both of them are composite primary key of Attends.

3) Filling data into schema

```
INSERT into Department values(1, 'Medical Department', null);

INSERT into Department values(2, 'Medical Record Department', null);

INSERT into Department values(3, 'Paramedical Department', null);

INSERT into Department values(4, 'Pharmacy Department', null);


INSERT into Employee values(1, 'Naif', 'Askul', '2-APR-1990', 30000,
'Jeddah,Al Sanabel', 'Male', null, 1);

INSERT into Employee values(2,'Faisal', 'Balamash', '1-MAR-1995',
25000, 'Jeddah,Al Sharafeyah', 'Male', null, 1);

INSERT into Employee values(3,'Tareq', 'Bajaber', '15-JUN-1993',
26000, 'Jeddah,Al Safa', 'Male', null, 2);

INSERT into Employee values(4,'Nawaf', 'Alghamdi', '30-SEP-1993',
24000, 'Jeddah,Al Safa', 'Male', null, 2);

INSERT into Employee values(5,'Ahmed', 'Alghamdi', '2-APR-1990',
15000, 'Jeddah,Al Sanabel', 'Male', null, 2);

INSERT into Employee values(6,'Ahmed', 'Alzahrani', '2-APR-1991',
13000, 'Jeddah,Al naseem', 'Male', null, 3);

INSERT into Employee values(7,'Sarah', 'Alharbi', '2-APR-1991',
3000, 'Jeddah,Al naseem', 'Female', null, 4);

INSERT into Employee values(8,'Rahaf', 'Alzahrani', '17-MAY-1989',
9000, 'Jeddah,Al Sanabel', 'Female', null, 4);


UPDATE Department SET Manager_ID = 3 WHERE department_id = 1;

UPDATE Department SET Manager_ID = 2 WHERE department_id = 2;

UPDATE Department SET Manager_ID = 4 WHERE department_id = 3;

UPDATE Department SET Manager_ID = 1 WHERE department_id = 4;


UPDATE Employee SET Supervisor_ID = 1 WHERE Employee_ID = 2;

UPDATE Employee SET Supervisor_ID = 1 WHERE Employee_ID = 3;

UPDATE Employee SET Supervisor_ID = 1 WHERE Employee_ID = 4;

UPDATE Employee SET Supervisor_ID = 4 WHERE Employee_ID = 5;

UPDATE Employee SET Supervisor_ID = 4 WHERE Employee_ID = 6;
UPDATE Employee SET Supervisor_ID = 2 WHERE Employee_ID = 7;

UPDATE Employee SET Supervisor_ID = 3 WHERE Employee_ID = 8;


INSERT into Room values(1,'Floor 2,J1');

INSERT into Room values(2,'Floor 1,E5');

INSERT into Room values(3,'Floor 1,G5');

INSERT into Room values(4,'Floor 2,GG');


INSERT into Doctor values(1, 'Zaki', 'Alsubhi', '2-APR-1990', 23000,
'Jeddah,Al Sanabel', 'Male', 1, 1);

INSERT into Doctor values(2 , 'Marwan', 'Khalil', '1-MAR-
1995', 25000, 'Jeddah,Al Sharafeyah', 'Male', 2, 2);

INSERT into Doctor values(3 , 'Radahn', 'Almalki', '15-JUN-
1993', 15000, 'Jeddah,Al Safa', 'Male', 3, 3);

INSERT into Doctor values(4 ,'Nawaf','ALzahrani ','30-
SEP-1993',22000,'Jeddah,Al Safa', 'Male', 4, 4);


 INSERT into patient values(1, 'Ahmed', 'AlGhamdi', '2-APR-1990',
'Jeddah,Al Sanabel', 'Male');

INSERT into patient values(2, 'Ahmed', 'bader', '9-APR-1990',
'Jeddah,Al Sanabel', 'Male');

INSERT into patient values(3, 'Bader', 'saleh', '9-SEP-1987',
'Jeddah,Al Safa', 'Male');

INSERT into patient values(4, 'Sara', 'saleh', '10-SEP-1987',
'Jeddah,Al Safa', 'Female');


 INSERT into Attends values (1,1);

INSERT into Attends values (2,2);
INSERT into Attends values (3,3);
INSERT into Attends values (4,4);
```

Part 3

1) Queries expressed in algebra form

1. **Select Doctor_ID, Fname, Lname from Doctor;**

Π (Doctor_ID, Fname, Lname)(Doctor)

2. **Select * From Employee where sex = 'Male';**

σ (sex = 'Male') (Employee)

3. **Select * from Employee where Employee_ID in (select Manager_ID from department);**

σ (Employee_ID IN (Π (Manager_ID)Department))

4. **Select Department_ID from department WHERE department_name = 'Medical Department';**

Π (Department_ID) (σ (Department_name = 'Medical Department')(Department))

5. **Create VIEW MED_INS as Select Doctor_ID, fname, lname from Doctor where department_ID = 2;**

Create view MED_INS as Π (Doctor_ID, Fname, Lname) (σ (Department_ID = 2) (Doctor))

6. **Select Doctor_ID, Salary from Doctor where Salary > (Select avg(Salary) from Doctor);**

σ (Salary > (Π($\mathcal{F}$AVG Salary)(Doctor)) (Π (Doctor_ID, Salary) (Doctor))

7. **Select Doctor_ID, COUNT(Patient_ID) from Attends Group by Doctor_ID;**

$_{\text{Doctor\_ID}}\ \mathcal{F}_{\text{COUNT Patient\_ID}}$ (Attends)

8. **Select D.Doctor_ID, D.Fname, P.Patient_ID, , P.Fname from Doctor D, Patient P where D.Fname = P.Fname;**

Π(D.Doctor_ID, D.FName, P.Patient_ID, P.FName)(Doctor $\bowtie_{\text{D.Fname = P.FName}}$ Patient)

9. **Select Room_ID from Room minus Select Room_ID from Doctor;**

Π (Room_ID)(Room) $-$ Π (Room_ID)(Doctor)

10. Select Employee_ID, Address from Employee where supervisor_ID is null;

Π(Employee_ID, Address)(σ(Supervisor_ID = null)(Employee))

2) Write those queries by using the SQL language

```
1. Select Doctor_ID, Fname, Lname from Doctor;

2. Select * From Employee where sex = ‘Male’;

3. Select * from Employee where Employee_ID in (select Manager_ID from
   department);

4. Select Department_ID from department WHERE department_name =
   'Medical Department';

5. Create VIEW MED_INS as Select Doctor_ID, fname, lname from Doctor
   where  department_ID = 2;

6. Select Doctor_ID, Salary from Doctor where Salary > (Select
   avg(Salary) from Doctor);

7. Select Doctor_ID, COUNT(Patient_ID) from Attends Group by Doctor_ID;

8. Select D.Doctor_ID, D.Fname, P.Patient_ID, P.Fname from Doctor D,
   Patient P where D.Fname = P.Fname;

9. Select Room_ID from Room minus Select Room_ID from Doctor;

10. Select Employee_ID, Address from Employee where supervisor_ID is
   null;
```

3) <u>Queries that contain more complex which include the following:</u>

a) Selection from different tables at one time.

```
SELECT E.Fname, D.Department_name FROM Employee E, Department D WHERE
E.Department_id = D.Department_id;
```

b) Aggregation by using GROUP BY and HAVING statement.

```
SELECT Doctor_ID, MAX(salary), AVG(salary) from doctor GROUP BY
Doctor_ID HAVING MIN(salary) < 20000;
```

c) Nesting with aggregation

```
SELECT AVG(salary) FROM (SELECT salary FROM employee WHERE salary <=
(SELECT AVG(salary) FROM employee));
```

d) Nesting involving NOT IN or NOT EXISTS

```
SELECT Doctor_ID, Fname, Lname, Department_ID FROM Doctor WHERE
Doctor_ID NOT IN (Select department_ID from department WHERE
department_name = 'Medical Department');
```

e) Outer Join

```
SELECT E.Employee_ID, E.Employee_Name D.Department_name,
D.Department_ID, D.Manager_ID from Employee E, Department D where
E.Employee_ID = D.Manager_ID(+);
```