Contents

[Task 1: Analysis and Design 2](#_Toc43242188)

[Task 2: SQL Statemenṭs 6](#_Toc43242189)

[Task 3: Assessment and Evaluation 14](#_Toc43242190)

[Conclusion 15](#_Toc43242191)

## Task 1: Analysis and Design

**Q.1. Produce the following documents:**

**a) Identify and discuss your chosen organisation, with a minimum of SIX (6) entities, and outline the likely data and information that the company would require. List and explain the functions and transactions that a database application might perform that would support some of the day to day functions of the organisation. Summarise the advantages of a DBMS to the organisation.**

Ans.1.

1. Here, I have chosen ‘Employee Payroll Management’ of any MNC that employs

individual and manages all the transactions related to employees salary generation.

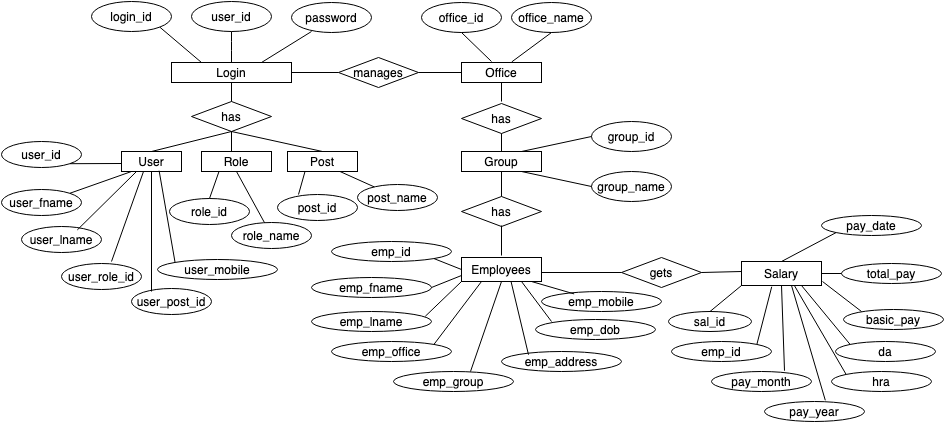
Each office has multiple users with different roles and posts. Only users of that office would be able to create, update or delete the employees from the system and takes care of the salary generation of the employees. Each office has its own users that login to the system using their password. Now, respective users can create a new group, new employees in the system or can work on the existing employees. Offices can have multiple groups and each having none or more employees tagged to that group. Every active employee tagged to any group of an office is eligible for getting a salary. This system will hold data about Offices, Users, Roles, Posts, Groups, Employees and Salary details to manage employee payroll generation.

The transaction of Employee Payroll Management system includes creating new groups in the office, creating new employees, allocating employees to groups, and generating monthly payrolls of the employees. Admin can create new users, roles and posts. Also can allocate users with different roles and posts. Here, I am using 8 entities – User, Role, Post, Login, Office, Group, Employees and Salary.

**Advantages of DBMS**

Here, I have kept all the tables in 3NF which reduces the redundant data. No duplicate data saves storage and improves access time. Data inconsistency has also been taken care of here. Data security is also maintained as only allowed users can do the desired changes leading to data privacy also. Data stored in tables are easily accessible. Also, the database keeps the backup of data. So in case of failure or crash, full recovery of data can be easily done.

**b)Draw an ERD (Entity Relationship Diagram) that shows data entities that have been identified in part a) and how they relate to one another. The ERD should support the transactions that have been identified in part a) and should be fully normalised to 3rd normal form (3NF).**

****

***Figure 1* :** ER Diagram of Employee Payroll Management System

**c)Create a data dictionary for the entity relationship diagram using the entities identified in part b). The data dictionary should identify the tables, attributes, primary and foreign keys, data types and any constraints/ business rules from your chosen organisation.**

**1.OFFICE**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **P/F** | **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** |
| P | office\_id | Integer | NNNNNN | 6 | Unique ID for all the offices. |
|  | office\_name | Text |  | 50 | Name of the office. |

**2.OFF\_GROUP**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **P/F** | **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** |
| P | group\_id | Integer | NNNNNN | 6 | Unique ID for all the groups. |
|  | group\_name | Text |  | 30 | Name of the group. |
| F | office\_id | Integer | NNNNNN | 6 | Name of the office to which group belongs. |

**3.ROLE**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **P/F** | **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** |
| P | role\_id | Integer | NNNNNN | 6 | Unique ID for all the roles. |
|  | role\_name | Text |  | 30 | Name of all the roles. |

**4.POST**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **P/F** | **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** |
| P | post\_id | Integer | NNNNNN | 6 | Unique ID for all the posts. |
|  | post\_name | Text |  | 30 | Name of all the posts. |

**5.USER**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **P/F** | **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** |
| P | user\_id | Integer | NNNNNN | 6 | Unique ID for all the users. |
|  | user\_fname | Text |  | 50 | First Name of the user. |
|  | user\_lname | Text |  | 50 | Last Name of the user. |
|  | user\_role\_id | Integer | NNNNNN | 6 | Role id of the user. |
|  | user\_post\_id | Integer | NNNNNN | 6 | Post id of the user. |
|  | user\_mobile | bigint | NNNNNNNNNN | 20 | User mobile number. |
| F | office\_id | Integer | NNNNNN | 6 | User office id. |

**6.LOGIN**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **P/F** | **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** |
| P | login\_id | Integer | NNNNNN | 6 | Unique ID for all the logins. |
| F | user\_id | Integer | NNNNNN | 6 | User id of the users. |
|  | password | Text |  | 30 | Password of the users. |

**7.EMPLOYEE**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **P/F** | **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** |
| P | emp\_id | Integer | NNNNNN | 12 | Unique ID for all the employees. |
|  | emp\_fname | Text |  | 50 | First Name of the employee. |
|  | emp\_lname | Text |  | 50 | Last Name of the employee. |
| F | emp\_office | Integer | NNNNNN | 6 | Office id of the employee. |
| F | emp\_group | Integer | NNNNNN | 6 | Group id of the employee. |
|  | emp\_mobile | bigint | NNNNNNNNNN | 20 | Employee mobile number. |
|  | emp\_address | Text |  | 50 | Employee address. |
|  | emp\_dob | Date | YYYY-MM-DD |  | Employee Date of Birth |

**8.SALARY**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **P/F** | **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** |
| P | sal\_id | Integer | NNNNNN | 6 | Unique ID for all the salary generated. |
| F | emp\_id | Integer | NNNNNN | 6 | Salary generated for employee. |
|  | pay\_month | Integer | NN | 2 | Salary generated for the month. |
|  | pay\_year | Integer | NNNN | 4 | Salary generated for the year. |
|  | basic\_pay | Integer | NNNNNNNNNNNN | 12 | Basic Pay of the employee. |
|  | da | Integer | NNNNNNNNNNNN | 12 | Dearness Allowance of the employee. |
|  | hra | Integer | NNNNNNNNNNNN | 12 | House Rent Allowance of the employee. |
|  | total\_pay | Integer | NNNNNNNNNNNN | 12 | Total Pay of the employee (basic+da+hra). |
|  | pay\_date | Date | YYYY-MM-DD |  | Salary disbursement date |

## Task 2: SQL Statemenṭs

**Q.2.**

**a) Create all of the normalised tables using SQL. Show your SQL scripts and the finished tables.**

Ans.a)

**1. OFFICE**

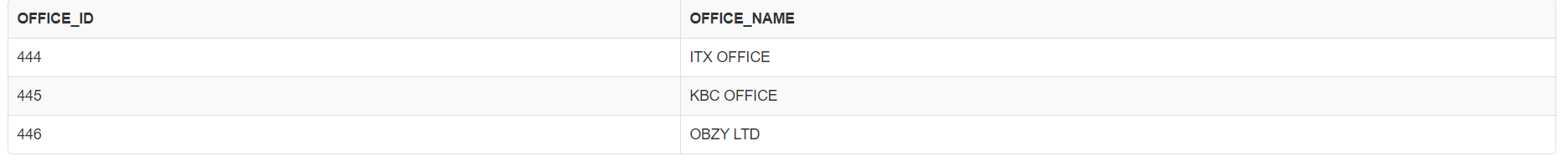
CREATE TABLE OFFICE(

OFFICE\_ID INT(6) NOT NULL, /\*Office ID \*/

OFFICE\_NAME VARCHAR(50) NOT NULL, /\*Office Name \*/

CONSTRAINT OFFICE\_PKEY PRIMARY KEY(OFFICE\_ID)

);



**2. OFF\_GROUP**

CREATE TABLE OFF\_GROUP(

GROUP\_ID INT(6) NOT NULL, /\* Group ID \*/

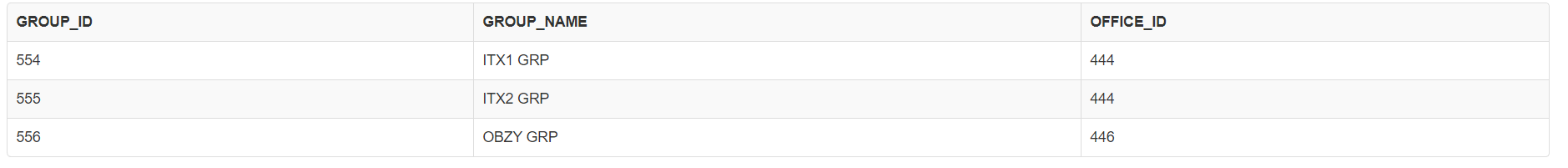
GROUP\_NAME VARCHAR(30) NOT NULL, /\*Group Name \*/

OFFICE\_ID INT(6) NOT NULL, /\* Office ID \*/

CONSTRAINT OFF\_GROUP\_PKEY PRIMARY KEY(GROUP\_ID),

CONSTRAINT OFF\_GROUP\_FKEY FOREIGN KEY(OFFICE\_ID) REFERENCES OFFICE(OFFICE\_ID)

);



**3. ROLE**

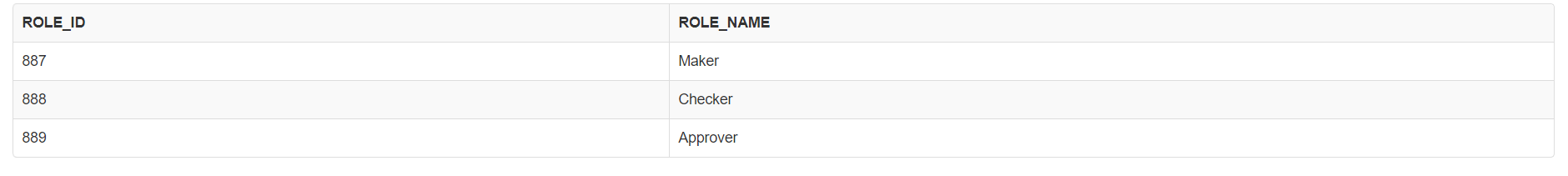
CREATE TABLE ROLE(

ROLE\_ID INT(6) NOT NULL, /\*Role ID \*/

ROLE\_NAME VARCHAR(30) NOT NULL, /\* Role Name \*/

CONSTRAINT ROLE\_PKEY PRIMARY KEY(ROLE\_ID)

);



**4. POST**

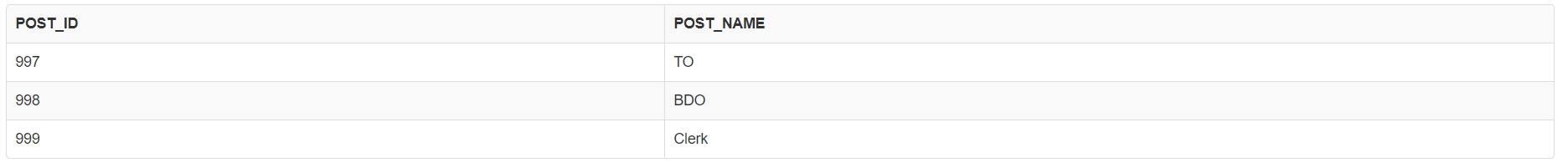
CREATE TABLE POST(

POST\_ID INT(6) NOT NULL, /\*Post ID \*/

POST\_NAME VARCHAR(30) NOT NULL, /\* Post Name \*/

CONSTRAINT ROLE\_PKEY PRIMARY KEY(POST\_ID)

);



**5. USER**

CREATE TABLE USER(

USER\_ID INT(6) NOT NULL, /\* User ID \*/

USER\_FNAME VARCHAR(50) NOT NULL, /\*User First name \*/

USER\_LNAME VARCHAR(50) NOT NULL, /\*User Last name \*/

USER\_ROLE\_ID INT(6), /\* User Role ID \*/

USER\_POST\_ID INT(6), /\*User Post ID \*/

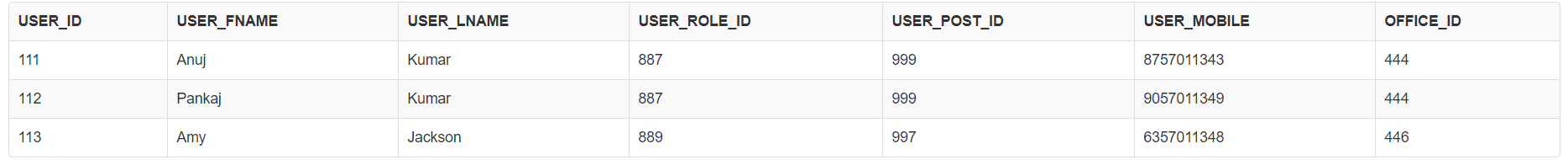
USER\_MOBILE BIGINT(20) NOT NULL, /\* User mobile no \*/

OFFICE\_ID INT(6) NOT NULL, /\* User Office\*/

CONSTRAINT USER\_PKEY PRIMARY KEY(USER\_ID),

CONSTRAINT USER\_FKEY FOREIGN KEY(OFFICE\_ID) REFERENCES OFFICE(OFFICE\_ID)

);



**6. LOGIN**

CREATE TABLE LOGIN(

LOGIN\_ID INT(6) NOT NULL, /\* Login ID \*/

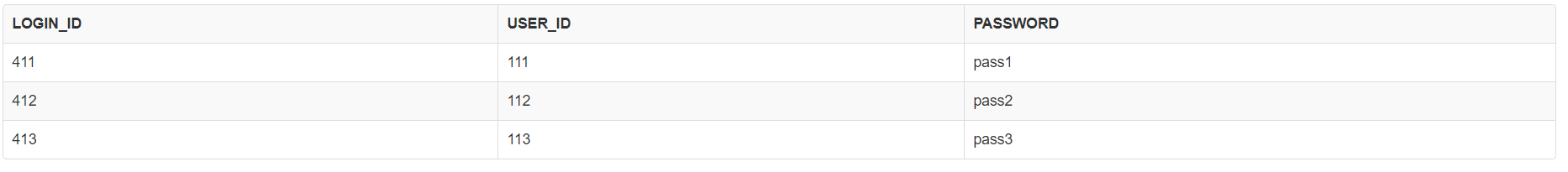
USER\_ID INT(6) NOT NULL, /\* Username\*/

PASSWORD VARCHAR(30) NOT NULL, /\* Password \*/

CONSTRAINT LOGIN\_PKEY PRIMARY KEY(LOGIN\_ID),

CONSTRAINT LOGIN\_FKEY FOREIGN KEY(USER\_ID) REFERENCES USER(USER\_ID)

);



**7. EMPLOYEE**

CREATE TABLE EMPLOYEE(

EMP\_ID INT(12) NOT NULL, /\* Employee ID \*/

EMP\_FNAME VARCHAR(50) NOT NULL, /\*Employee First Name \*/

EMP\_LNAME VARCHAR(50) NOT NULL, /\*Employee Last Name \*/

EMP\_OFFICE INT(6), /\* Employee Office ID \*/

EMP\_GROUP INT(6), /\* Employee Group ID \*/

EMP\_ADDRESS VARCHAR(50), /\*Employee Address \*/

EMP\_DOB DATE, /\* Employee Date of Birth \*/

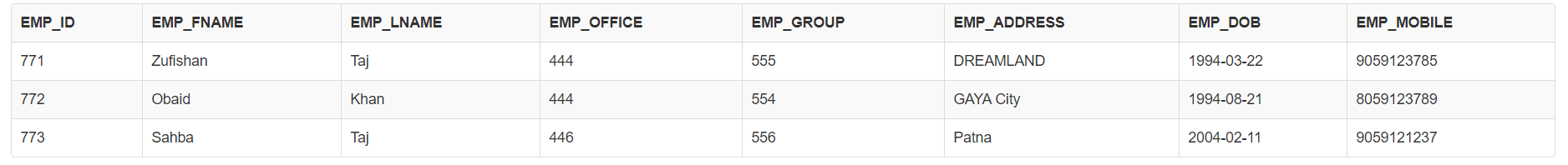
EMP\_MOBILE BIGINT(20), /\* Employee Mobile Number \*/

CONSTRAINT EMPLOYEE\_PKEY PRIMARY KEY(EMP\_ID),

CONSTRAINT EMPLOYEE1\_FKEY FOREIGN KEY(EMP\_OFFICE) REFERENCES OFFICE(OFFICE\_ID),

CONSTRAINT EMPLOYEE2\_FKEY FOREIGN KEY(EMP\_GROUP) REFERENCES OFF\_GROUP(GROUP\_ID)

);



**8. SALARY**

CREATE TABLE SALARY(

SAL\_ID INT(6) NOT NULL, /\*Salary ID \*/

EMP\_ID INT(6) NOT NULL, /\*Employee ID \*/

PAY\_MONTH INT(2) NOT NULL, /\* Pay Month \*/

PAY\_YEAR INT(4) NOT NULL, /\* Pay Year\*/

BASIC\_PAY INT(12) NOT NULL, /\*Basic Pay \*/

DA INT(12) NOT NULL, /\*Dearness Allowance \*/

HRA INT(12) NOT NULL, /\* House Rent Allowance \*/

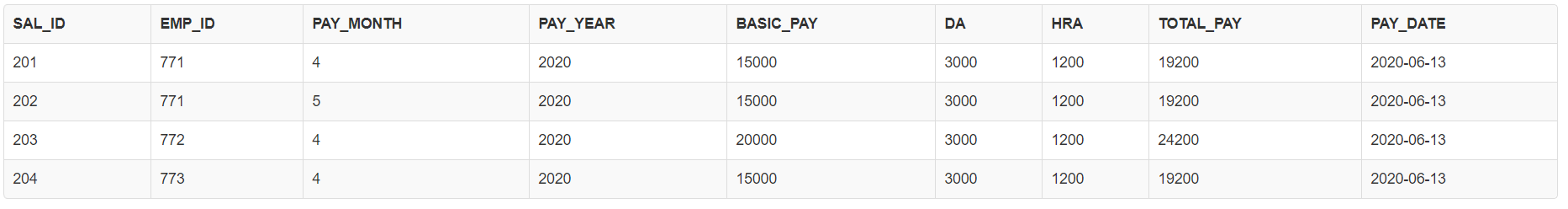
TOTAL\_PAY INT(12) NOT NULL, /\*Total Pay \*/

PAY\_DATE DATE NOT NULL,

CONSTRAINT SALARY\_PKEY PRIMARY KEY(SAL\_ID),

CONSTRAINT SALARY\_FKEY FOREIGN KEY(EMP\_ID) REFERENCES EMPLOYEE(EMP\_ID)

);



1. **Enter sample data in all tables (minimum THREE (3) rows per table).**

**1.** **OFFICE**

INSERT INTO OFFICE VALUES(444, 'ITX OFFICE');

INSERT INTO OFFICE VALUES(445, 'KBC OFFICE');

INSERT INTO OFFICE VALUES(446, 'OBZY LTD');

**2. OFF\_GROUP**

INSERT INTO OFF\_GROUP VALUES(554, 'ITX1 GRP', 444);

INSERT INTO OFF\_GROUP VALUES(555, 'ITX2 GRP', 444);

INSERT INTO OFF\_GROUP VALUES(556, 'OBZY GRP', 446);

**3.ROLE**

INSERT INTO ROLE VALUES(887, 'Maker');

INSERT INTO ROLE VALUES(888, 'Checker');

INSERT INTO ROLE VALUES(889, 'Approver');

**4.POST**

INSERT INTO POST VALUES(997, 'TO');

INSERT INTO POST VALUES(998, 'BDO');

INSERT INTO POST VALUES(999, 'Clerk');

**5.USER**

INSERT INTO USER VALUES(111, 'Anuj', 'Kumar', 887, 999, 8757011343, 444);

INSERT INTO USER VALUES(112, 'Pankaj', 'Kumar', 887, 999, 9057011349, 444);

INSERT INTO USER VALUES(113, 'Amy', 'Jackson', 889, 997, 6357011348, 446);

**6. LOGIN**

INSERT INTO LOGIN VALUES(411, 111, 'pass1');

INSERT INTO LOGIN VALUES(412, 112, 'pass2');

INSERT INTO LOGIN VALUES(413, 113, 'pass3');

**7. EMPLOYEE**

INSERT INTO EMPLOYEE VALUES(771, 'Aishwarya', 'Rai', 444, 555, 'DREAMLAND', '1994-03-22', 9059123785);

INSERT INTO EMPLOYEE VALUES(772, 'Shahrukh', 'Khan', 444, 554, 'GAYA City', '1994-08-21', 8059123789);

INSERT INTO EMPLOYEE VALUES(773, 'Alia', 'Bhatt', 446, 556, 'Patna', '2004-02-11', 9059121237);

**8.SALARY**

INSERT INTO SALARY VALUES(201, 771, 4, 2020, 15000, 3000, 1200, 19200, now());

INSERT INTO SALARY VALUES(202, 771, 5, 2020, 15000, 3000, 1200, 19200, now());

INSERT INTO SALARY VALUES(203, 772, 4, 2020, 20000, 3000, 1200, 24200, now());

INSERT INTO SALARY VALUES(204, 773, 4, 2020, 15000, 3000, 1200, 19200, now());

**c) Write at least FIVE (5) queries that show your understanding of SQL. SQL statements that should be used should include: Select, Update, Delete, From, Where, And, Count, Ascending, Order By. TWO (2) of the queries should join TWO (2) or more tables together.**

1. SQL Query to fetch login id, user name, user role name and user post name of user having user\_id=111.

select a.login\_id, b.user\_fname, b.user\_lname, c.role\_name, d.post\_name

from login a,

user b,

role c,

post d

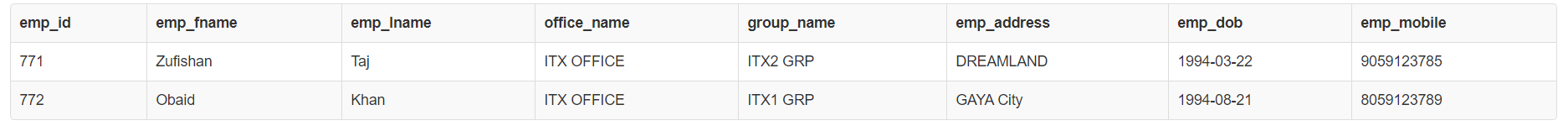
where a.user\_id=b.user\_id

and b.user\_role\_id=c.role\_id

and b.user\_post\_id=d.post\_id

and b.user\_id=111;

**output:**



1. SQL Query to update user mobile no of user having user\_id=111.

update USER

set USER\_MOBILE = 8985451234

where user\_id=111;

1. SQL Query Delete a record of a user having user\_id=113 .

delete from login where user\_id=113;

delete from user where user\_id=113;

1. SQL Query to depict count statement.

select count(a.emp\_id)

from employee a,

office b

where a.emp\_office=b.office\_id

and b.office\_id=444;

**output:**



1. SQL Query to depict order by clause.

select a.emp\_id, a.emp\_fname, a.emp\_lname, b.office\_name, c.group\_name, a.emp\_address, a.emp\_dob,a.emp\_mobile

from employee a,

office b,

off\_group c

where a.emp\_office=b.office\_id

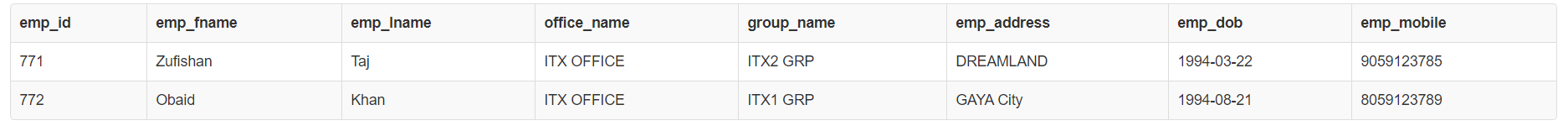
and a.emp\_group=c.group\_id

and b.office\_id=c.office\_id

and a.emp\_office=444

order by a.emp\_id asc;

**output:**



## Task 3: Assessment and Evaluation

While creating an ‘Employee Payroll Management System’, following things were taken into consideration:- How table structure would look like? What all data would it store? Constraints, data types, size of each field were also taken care of in order to improve the accessibility of data along with the storage.

Here, data is stored efficiently in the system which is the basic requirement of any DBMS. I have chosen MySQL DBMS. Users can easily fetch and manipulate data using SQL statements (queries).Any update or delete statements executed are committed into the system. Thus data integrity is maintained. Users have the privilege to either commit or rollback the modification if required thus maintaining the state of data. Also, there is a facility for data backup.

I faced challenges while creating the ER Diagram of my model. There were fields that got repeated in different tables. I analysed and then tried to create my tables in 3NF. The ordering of table creation as well as considering foreign key constraints were a bit time taking. After creating tables one by one and inserting data and how the relation can be made using one field of table to another (JOINS) lead to an efficient development of my database. Accordingly, the ER diagram was modified.

## Conclusion

In this assignment, I have created an ‘Employee Payroll Management System’ that creates and saves new employees and stores data efficiently. All data are stored in 3NF form which leads to no data duplication and data inconsistency. Using MySQL DBMS, data can be easily modified using SQL DML statements that includes (SELECT, INSERT, UPDATE, DELETE). DBMS has made life easier by storing data efficiently. Paper work has come to an end. This system is tested with dummy data and is working fine. Only users of respective office has the right to access data of employees of that office.Payroll generation has become quite easy and faster through this system. The records are easily accessible for any employee. DBMS also keeps backup, thus in case of failure data are recovered easily without any loss. Thus resulting into an efficient system to record details of employees of any organization and generating their pay bills on time efficiently.