EXPERIMENT NO: 1

ER-DIGRAM

Design an ER diagram for the following requirement:-

A university registrar's office maintains data about the following entities:-

(a) Courses, including numbers, title, credits, syllabus, and prerequisites;

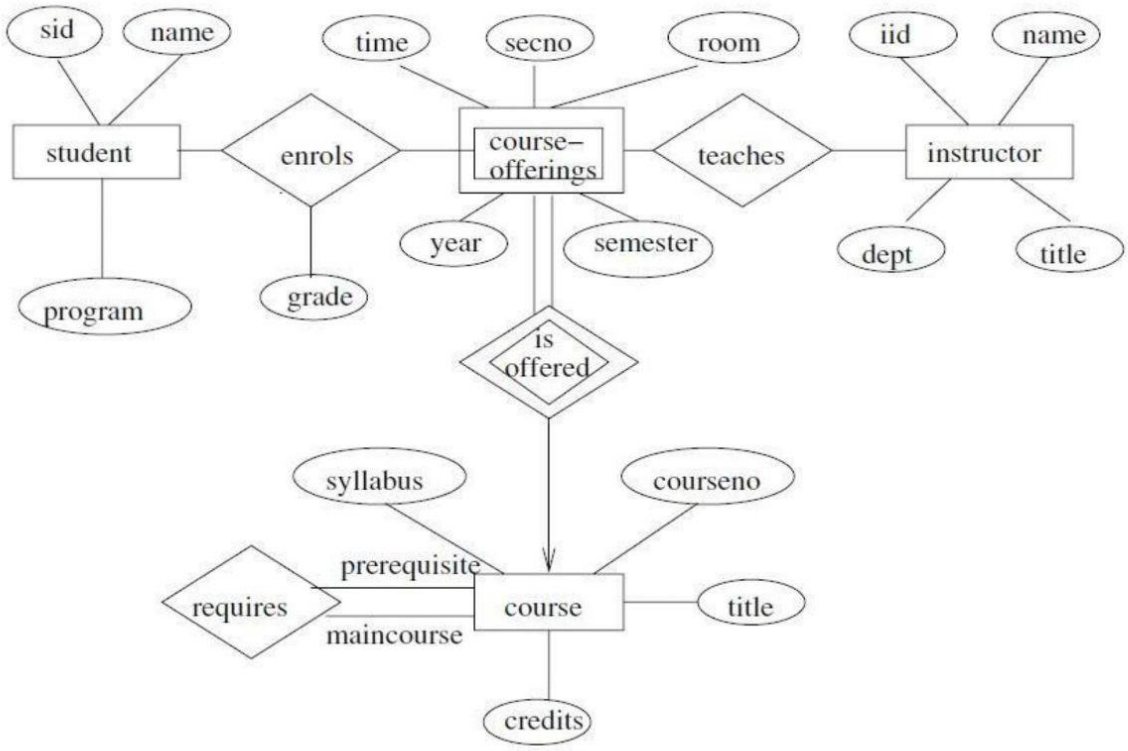
(b) Course offerings, including course number, year, semester, section number, Instructors, timings, and classroom;

(c) Students, including student-id, name, and program;

(d) Instructors, including identification number, name, department, and title.

Further, the enrollment of students in courses and grades awarded to students in each course they are enrolled for must be appropriately modeled.

Construct an E-R diagram for the registrar's office. Document all assumptions that you make about the mapping constraints.



EXPERIMENT NO: 2
BASIC SQL COMMANDS

AIM:

To study the basic sql queries such as:
SELECT
INSERT
UPDATE
DELETE

QUESTIONS

Create a table named Employee and populate the table as shown below.

EMP_ID	EMP_NAME	DEPT	SALARY
1	MICHAEL	PRODUCTION	2500
2	JOE	PRODUCTION	2500
3	SMITH	SALES	2250
4	DAVID	MARKETING	2900
5	RICHARD	SALES	1600
6	JESSY	MARKETING	1800
7	JANE	SALES	2000
8	JANET	PRODUCTION	3000
9	NEVILLE	MARKETING	2750
10	RICHARDSON	SALES	1800

```
CREATE TABLE Employee (Emp_idint,Emp_namevarchar(15),Deptvarchar(10),Salary int)

INSERT INTO Employee values(1,'Michael','Production',2500)
INSERT INTO Employee values(2,'Joe','Production',2500)
INSERT INTO Employee values(3,'Smith','Sales',2250)
INSERT INTO Employee values(4,'David','Marketing',2900)
INSERT INTO Employee values(5,'Richard','Sales',1600)
INSERT INTO Employee values(6,'Jessy','Marketing',1800)
INSERT INTO Employee values(7,'Jane','Sales',2000)
INSERT INTO Employee values(8,'Janet','Production',3000)
INSERT INTO Employee values(9,'Neville','Marketing',2750)
INSERT INTO Employee values(10,'Richardson','Sales',1800)
```

1. Display the details of all the employees.

```
SELECT * FROM Employee
```

EMP_ID	EMP_NAME	DEPT	SALARY
1	MICHAEL	PRODUCTION	2500
2	JOE	PRODUCTION	2500
3	SMITH	SALES	2250
4	DAVID	MARKETING	2900
5	RICHARD	SALES	1600
6	JESSY	MARKETING	1800
7	JANE	SALES	2000
8	JANET	PRODUCTION	3000
9	NEVILLE	MARKETING	2750
10	RICHARDSON	SALES	1800

2. Display the names and id's of all employees.

```
SELECT Emp_id,Emp_name FROM Employee
```

EMP_ID	EMP_NAME
1	MICHAEL
2	JOE
3	SMITH
4	DAVID
5	RICHARD
6	JESSY
7	JANE
8	JANET
9	NEVILLE
10	RICHARDSON

3. Delete the entry corresponding to employee id:10.

```
DELETE FROM Employee WHERE Emp_id=10
```

EMP_ID	EMP_NAME	DEPT	SALARY
1	MICHAEL	PRODUCTION	2500
2	JOE	PRODUCTION	2500
3	SMITH	SALES	2250
4	DAVID	MARKETING	2900
5	RICHARD	SALES	1600
6	JESSY	MARKETING	1800
7	JANE	SALES	2000
8	JANET	PRODUCTION	3000
9	NEVILLE	MARKETING	2750

4. Insert a new tuple to the table. The salary field of the new employee should be kept NULL.

```
INSERT INTO Employee values(10,'Richardson','Sales',NULL)
```

EMP_ID	EMP_NAME	DEPT	SALARY
1	MICHAEL	PRODUCTION	2500
2	JOE	PRODUCTION	2500
3	SMITH	SALES	2250
4	DAVID	MARKETING	2900
5	RICHARD	SALES	1600
6	JESSY	MARKETING	1800
7	JANE	SALES	2000
8	JANET	PRODUCTION	3000
9	NEVILLE	MARKETING	2750
10	RICHARDSON	SALES	NULL

5. Find the details of all employees working in the marketing department.

```
SELECT * FROM Employee WHERE Dept='Marketing'
```

EMP_ID	EMP_NAME	DEPT	SALARY
4	DAVID	MARKETING	2900
6	JESSY	MARKETING	1800
9	NEVILLE	MARKETING	2750

6. Add the salary details of the newly added employee.

```
UPDATE Employee set Salary=1900 WHERE Emp_id=10

SELECT * FROM Employee
```

EMP_ID	EMP_NAME	DEPT	SALARY
1	MICHAEL	PRODUCTION	2500
2	JOE	PRODUCTION	2500
3	SMITH	SALES	2250
4	DAVID	MARKETING	2900
5	RICHARD	SALES	1600
6	JESSY	MARKETING	1800
7	JANE	SALES	2000
8	JANET	PRODUCTION	3000
9	NEVILLE	MARKETING	2750
10	RICHARDSON	SALES	1900

7. Update the salary of Richard to 1900$.

```
UPDATE Employee set Salary=1900 WHERE Emp_name='Richardson';
SELECT * FROM Employee
```

EMP_ID	EMP_NAME	DEPT	SALARY
1	MICHAEL	PRODUCTION	2500
2	JOE	PRODUCTION	2500
3	SMITH	SALES	2250
4	DAVID	MARKETING	2900
5	RICHARD	SALES	1900
6	JESSY	MARKETING	1800
7	JANE	SALES	2000
8	JANET	PRODUCTION	3000
9	NEVILLE	MARKETING	2750
10	RICHARDSON	SALES	1900

8. Find the details of all employees who working for marketing and has a salary greater than 2000$.

```
SELECT * FROM Employee WHERE Dept='Marketing' AND Salary>2000
```

EMP_ID	EMP_NAME	DEPT	SALARY
4	DAVID	MARKETING	2900
9	NEVILLE	MARKETING	2750

9. List the names of all employees working in sales department and marketing Department.

```
SELECT emp_name FROM Employee WHERE Dept='Marketing' OR Dept='Sales'
```

EMP_NAME
SMITH
DAVID
RICHARD
JESSY
JANE
NEVILLE
RICHARDSON

10. List the names and department of all employees whose salary is between2300$ and 3000$.

```
SELECT Emp_name,Dept FROM Employee WHERE Salary BETWEEN 2300 AND 3000
```

EMP_NAME	DEPT
MICHAEL	PRODUCTION
JOE	PRODUCTION
DAVID	MARKETING
JANET	PRODUCTION
NEVILLE	MARKETING
RICHARDSON	SALES

11. Update the salary of all employees working in production department 12%.

UPDATE Employee SET Salary=Salary+salary*0.12 WHERE Dept='Production'

SELECT * FROM Employee

EMP_ID	EMP_NAME	DEPT	SALARY
1	MICHAEL	PRODUCTION	2800
2	JOE	PRODUCTION	2800
3	SMITH	SALES	2250
4	DAVID	MARKETING	2900
5	RICHARD	SALES	1900
6	JESSY	MARKETING	1800
7	JANE	SALES	2000
8	JANET	PRODUCTION	3360
9	NEVILLE	MARKETING	2750
10	RICHARDSON	SALES	1900

12. Display the names of all employees whose salary is less than 2000$ or working for the sales department.

```
SELECT Emp_name FROM Employee WHERE Salary<2000 OR Dept='Sales'
```

EMP_NAME
SMITH
RICHARD
JESSY
JANE
RICHARDSON

RESULT

The query was executed and the output was obtained

EXPERIMENT NO: 3
CREATION OF DATABASE SCHEMA USING DDL COMMANDS

AIM:

To solve queries using DDL commands.

THEORETICAL BACKGROUND:

SQL (Structured Query Language) is a computer language aimed to store, manipulate, and retrieve data stored in relational databases.In order to communicate with the database, SQL supports the following categories of commands:

Data Definition Language (DDL) - create, alter, truncate and drop commands.
Data Manipulation Language (DML) - insert, select, delete and update commands.
Transaction Control Language (TCL) - Commit savepoint and rollback commands.
Data Control Language (DCL) - grant and revoke commands.

CREATE Command

Syntax:
create table < table name > (column datatype,column datatype,….);

ALTER Command

It is used to modify the structure of the table.
Syntax:

alter table <table name> add column_name datatype;
alter table <table name> drop column column_name;
alter table <table name> rename column old_name to new_name;

TRUNCATE Command

It is used to delete records stored in a table and the structure has to be retained as it is.
Syntax:
truncate table <table name>;

DROP Command

It is used to drop a table.
Syntax:
drop table <table name>

Questions:

1. Create a table Student with the following fields - Rollno, Name, Mark1,Mark2.

   create table Student(Rollno int,Name varchar(20),Mark1 int,Mark2 int);

2. Alter the table ‘Student’ to Add a New Column ‘Mark3’ as int.

   alter table Student add Mark3 int;

3. Alter the table ‘Student’ to Delete the Column ‘Mark2’ .

   alter table Student drop column Mark2;

4. Alter Table ‘Student’ to Rename Column ‘Mark3’ to ‘Mark2’.

   alter table Student rename Mark3 to Mark2;

5. Delete the Table ‘Student’.

   Drop Table Student;

RESULT:

The query was executed successfully and output was verified.

EXPERIMENT NO: 4
Study Of Constraints

AIM:

Study About Different Constraints.

THEORETICAL BACKGROUND:

CONTRAINTS

In DBMS (Database Management Systems), constraints are **guidelines or limitations imposed on database tables to maintain the integrity, correctness, and consistency of the data**. Constraints can be used to enforce data linkages across tables, verify that data is unique, and stop the insertion of incorrect data.

Types Of Constraints

1. NOTNULL
2. UNIQUE
3. PRIMARY KEY
4. FOREIGN KEY
5. CHECK
6. DEFAULT

Questions:

1. Create a table with name customer and fields as follows:
   cust_id number, name varchar2(20), hname varchar2(20), street varchar2(20), phone integer.

   create table customer(cust_id int,name varchar(20),hname varchar(20),street varchar(20),phone int);

2. Create table items with the following fields:
   item_code number, name char(1) , Current_stock number, Unitprice number

   create table items(item_code number(5), name varchar2(20), stock_number int, unit_price int);

3) Create table order with fields as follows

Order_id number, Cust_id number, Item_code number,Order_quantity number, Delivery_date date, payment_mode char(1).

create table order(order_id int, cust_id int, item_code int(5), order_date date, expiry_date date, delivery_date date, payment_mode char(1));

4). Alter the table to add a primary key to cust_id in customer.

alter table customer add constraint p1k primary key(cust_id);

5). Alter the table to add a primary key to item_code in items.

alter table items add constraint p2k primary key(item_code);

6). Alter the table to set foreign key constraint in order to link with customer using cust_id field.

alter table order add constraint foreign key references customer(cust_id);

7). Add a not null constraint to the field order_date.

alter table order modify order_date constraint c1 not null;

8). Add a unique constraint for the order_id in order.

alter table order modify order_id constraint c2 unique;

9). Add a check constraint to the field payment_mode in order such that it takes three values'D' (for cheque), 'R' (for cash) and 'C' (for credit card).

alter table order add constraint c3 check (payment_mode in ('D', 'R', 'C'));

10). Add a new field 'remark' as char(10) to the order and set the default value as 'direct'.

alter table order add remark char(10) default 'direct';

11). Drop all the Tables customer , order and items.

Drop Table customer;
Drop Table items;
Drop Table Order;

RESULT:

The query was executed successfully and output was verified.