DAYANANDA SAGAR ACADEMY OF TECHNOLOGY AND MANAGEMENT

Udayapura, Kanakapura Road, Opp. Art of Living, Bangalore – 560082

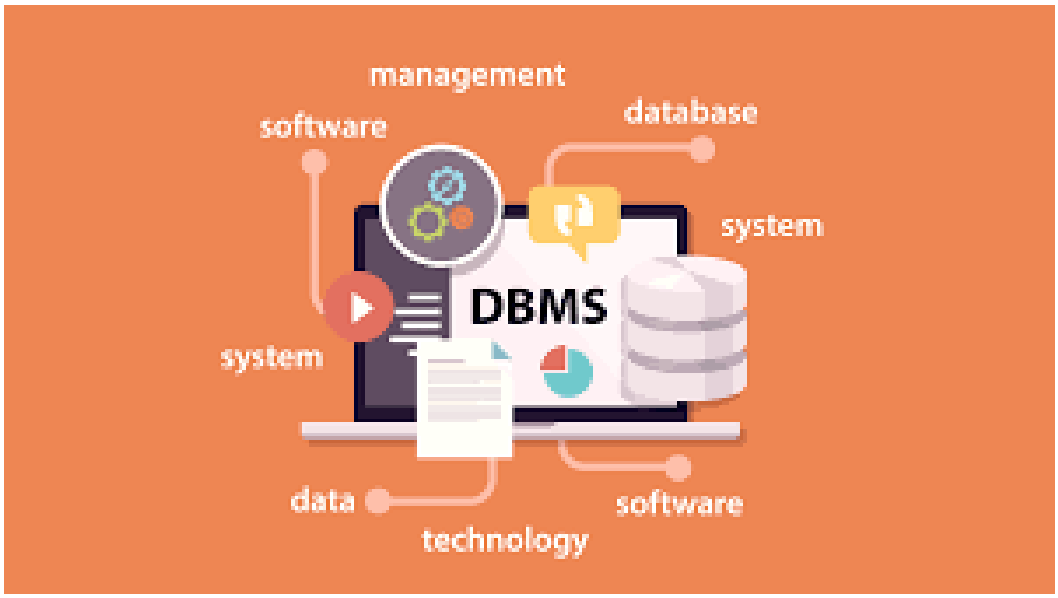
(Affliated to VTU, Belagavi, Approved by AICTE, New Delhi)

Accredited by NBA and NAAC (A+)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

2023-2024

DBMS LAB MANNUAL
(21CSL55)



Compiled by:

Prof. Lakshmi M R

Dr. Kavitha C
HOD, CSE, DSATM

Dr. M Ravishankar
Principal, DSATM

21CSL55: DBMS LABORATORY WITH MINI PROJECT

Course objectives: This course will enable students to

- Foundation knowledge in database concepts, technology and practice to groom students into well-informed database application developers.
- Strong practice in SQL programming through a variety of database problems.
- Develop database applications using front-end tools and back-end DBMS.

Database: A Database is a collection of interrelated data and a Database Management System is a a software system that enables users to define, create and maintain the database and which provides controlled access to the database

SQL: It is structured query language, basically used to pass the query to retrieve and manipulate the information from database. Depending upon the nature of query, SQL is divided into different components:

- **DDL**(Data Definition Language)
- **DML**(Data Manipulation Language)
- **DCL**(Data Control Language)

DDL: The Data Definition Language (DDL) is used to create the database (i.e. tables, keys,relationships etc), maintain the structure of the database and destroy databases and database objects.

Eg. Create, Drop, Alter, Describe, Truncate

1. **CREATE** statements: It is used to create the table.

Syntax:
CREATE TABLE table_name(columnName1 datatype(size), columnName2 datatype(size), ...);

2. **DROP statements:** To destroy an existing database, table, index, or view. If a table isdropped all records held within it are lost and cannot be recovered.

Syntax:
DROP TABLE table_name;

3. **ALTER statements:** To modify an existing database object.

- **Adding new columns:**

Syntax:

Alter table table_name Add(New_columnName1 datatype(size), New_columnName2 datatype(size),.....)

• **Dropping a columns from a table : Syntax:**

Alter table table_name DROP column columnName:

• **Modifying Existing columns:**

Syntax:

Alter table table_name Modify (columnName1 Newdatatype(Newsize));

4. **Describe statements:** To describe the structure (column and data types) of an existingdatabase, table, index, or view.

Syntax:

DESC table_name;

5. **Truncate statements:** To destroy the data in an existing database, table, index, or view. If a table is truncated all records held within it are lost and cannot be recovered but the table structure is maintained.

Syntax :

TRUNCATE TABLE table_name;

Data Manipulation Language (DML):

• A Data Manipulation Language enables programmers and users of the database to retrieve insert, delete and update data in a database. e.g. INSERT, UPDATE, DELETE, SELECT.

INSERT: INSERT statement adds one or more records to any single table in a relational database.

Syntax:

INSERT INTO tablename VALUES (expr1,expr2. .....);

UPDATE: UPDATE statement that changes the data of one or more records in a table. Eitherall the rows can be updated, or a subset may be chosen using a condition.

Syntax:

UPDATE table_name SET column_name = value [, column_name = value....] [WHERE condition]

DELETE: DELETE statement removes one or more records from a table. A subset may bedefined for deletion using a condition, otherwise all records are removed.

Syntax:

DELETE FROM tablename WHERE condition:

SELECT: SELECT statement returns a result set of records from one or more tables.

The select statement has optional clauses:

• WHERE specifies which rows to retrieve

• GROUP BY groups rows sharing a property so that an aggregate function can be applied to each group having group.

• HAVING selects among the groups defined by the GROUP BY clause.

• ORDER BY specifies an order in which to return the rows.

Syntax:

SELECT<attribute list> FROM<table list>
WHERE<condition> Where

• Attribute list is a list of attribute name whose values to be retrieved by the query.

• Table list is a list of table name required to process query.

• Condition is a Boolean expression that identifies the tuples to be retrieved by query.

Data Constraints are the business Rules which are enforced on the data being stored in a tableare called Constraints.

Types of Data Constraints

1. I/O Constraint This type of constraint determines the speed at which data can be inserted or extracted from an Oracle table. I/O Constraints is divided into two different types
   - The Primary Key Constraint
   - The Foreign Key Constraint

2. Business rule Constraint This type of constraint is applied to data prior the data being Inserted into table columns.

- Column level
- Table level

The PRIMARY KEY defined at column level

Syntax:

```
CREATETABLEtablename
(Columnname1DATATYPE
CONSTRAINT <constraintname1>
PRIMARY KEY, Columnname2
DATATYPE, columnname3
DATATYPE, ....);
```

The PRIMARY KEY defined at table level

Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE, columnname2 DATATYPE, columnname3 DATATYPE, **PRIMARY KEY (columnname1, columnname2));**

The FOREIGN KEY defined at column level

Syntax

CREATE TABLE tablename (Columnname1 tablename[(columnname)] [ON DELETE CASCADE], columnname3 DATATYPE , ...);

DATATYPE columnname2 REFERENCES DATATYPE ,

The table in which FOREIGN KEY is defined is called FOREIGN TABLE or DETAIL TABLE. The table in which PRIMARY KEY is defined and referenced by FOREIGN KEY is called PRIMARY TABLE or MASTER TABLE.

ON DELETE CASCADE is set then DELETE operation in master table will trigger the DELETE operation for corresponding records in the detail table.

The FOREIGN KEY defined at table level

Syntax:

CREATE TABLE table name (Columnname1 DATATYPE, columnname2 DATATYPE, columnname3 DATATYPE, PRIMARY KEY (columnname1, columnname2), FOREIGN KEY (columnname2) REFERENCES tablename2;

A CONSTRAINT can be given User Defined Name, the syntax is:
CONSTRAINT < constraint name><constraint definition>

The CHECK Constraint defined at column level
Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE CHECK (logical expression), columnname2 DATATYPE, columnname3 DATATYPE, ..);

The CHECK Constraint defined at table level
Syntax:

CREATE TABLE table name (Columnname1 DATATYPE, columnname2 DATATYPE, columnname3 DATATYPE, CHECK (logical expression1), CHECK (logical expression2));

The UNIQUE Constraint defined at the column level
Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE UNIQUE, columnname2 DATATYPE UNIQUE, columnname3 DATATYPE ...);

The UNIQUE Constraint defined at the the table level
Syntax:
CREATE TABLE tablename (Columnname1 DATATYPE, columnname2 DATATYPE, columnname3 DATATYPE, UNIQUE(columnname1));

NOT NULL constraint defined at column level :
Syntax:
CREATE TABLE tablename (Columnname1 DATATYPE NOT NULL, columnname2 DATATYPE NOT NULL, columnname3 DATATYPE,...);

Note:
The NOT NULL constraint can only be applied at column level.

ER- Diagram: It is an Entity –Relationship diagram which is used to represent the relationshipbetween different entities. An entity is an object in the real world which is distinguishable from other objects. The overall logical structure of a database can be expressed graphically by an ER diagram, which is built up from following components.

- Rectangles: represent entity sets.
- Ellipses: represent attributes.
- Diamonds: represent relationships among entity sets.
- Lines: link attribute to entity sets and entity sets to relationships.

Mapping Cardinalities: It expresses the number of entities to which another entity can beassociated via a relationship set. For a binary relationship set R between entity sets A and B. The Mapping Cardinalities must be one of the following.

- One to one
- One to many
- Many to one
- Many to many

LAB EXPERIMENTS

PART A: SQL PROGRAMMING

A. Consider the following schema for a LibraryDatabase:

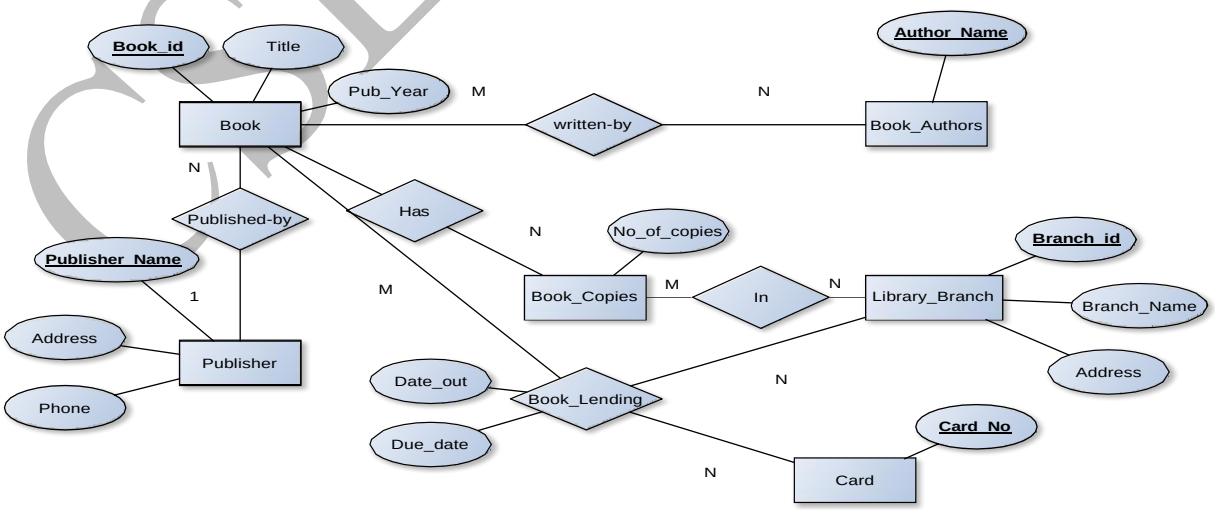
BOOK (*Book_id, Title, Publisher_Name, Pub_Year*)
BOOK_AUTHORS (Book_id, Author_*Name*)
PUBLISHER (*Name, Address, Phone*)
BOOK_COPIES (*Book_id, Branch_id, No-of_Copies*)
BOOK_LENDING (*Book_id, Branch_id, Card_No, Date_Out, Due_Date*)
LIBRARY_BRANCH (*Branch_id, Branch_Name, Address*)

Write SQL queries to

1. **Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each branch,etc.**
2. **Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun2017**
3. **Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulationoperation.**
4. **Partition the BOOK table based on year of publication. Demonstrate its working with a simplequery.**
5. **Create a view of all books and its number of copies that are currently available in the Library.**

Solution:

Entity-Relationship Diagram



Schema Diagram

Table Creation:

PUBLISHER

```
SQL>CREATE TABLE PUBLISHER(
     NAME VARCHAR(18) PRIMARY KEY,
     ADDRESS VARCHAR(10),
     PHONE VARCHAR(10));

Table created.
```

BOOK

```
SQL>CREATE TABLE BOOK(
     BOOK_ID INTEGER PRIMARY KEY,
     TITLE VARCHAR(20),
     PUBLISHER_NAME VARCHAR(20)
     PUB_YEAR NUMBER(4),
     FOREIGN KEY(PUBLISHER_NAME) REFERENCES PUBLISHER(NAME)ON
                                                          DELETECAS
                                                          ADE

     );

Table created.
```

BOOK_AUTHORS

```
SQL>CREATE TABLE BOOK_AUTHORS(
     BOOK_ID INTEGER,
     AUTHOR_NAME VARCHAR(20),
     PRIMARY KEY(BOOK_ID),
     FOREIGN KEY(BOOK_ID) REFERENCES BOOK(BOOK_ID) ON DELETE CASCADE);

Table created.
```

LIBRARY_BRANCH

```
SQL>CREATE TABLE LIBRARY_BRANCH(
     BRANCH_ID INTEGER PRIMARY KEY,
     BRANCH_NAME VARCHAR(18),
     ADDRESS VARCHAR(15));

Table created.
```

BOOK_COPIES

```
SQL>CREATE TABLE BOOK_COPIES(
     BOOK_ID INTEGER,
     BRANCH_ID INTEGER,
     NO_OF_COPIES INTEGER,
     FOREIGN KEY(BOOK_ID) REFERENCES BOOK(BOOK_ID) ON DELETE CASCADE,
     FOREIGN KEY(BRANCH_ID) REFERENCES LIBRARY_BRANCH(BRANCH_ID) ON
     DELETE CASCADE,
     PRIMARY KEY(BOOK_ID,BRANCH_ID));
```

```
Table created.
```

BOOK_LENDING

```
SQL>CREATE TABLE BOOK_LENDING(
     BOOK_ID INTEGER,
     BRANCH_ID INTEGER,
     CARD_NO INTEGER,
     DATE_OUT DATE,
     DUE_DATE DATE,
     PRIMARY KEY(BOOK_ID,BRANCH_ID,CARD_NO),
     FOREIGN KEY(BOOK_ID) REFERENCES BOOK(BOOK_ID) ON DELETE CASCADE,
     FOREIGN KEY(BRANCH_ID) REFERENCES LIBRARY_BRANCH(BRANCH_ID) ON
     DELETE CASCADE,
     );Table created.
```

Values for tables:

PUBLISHER

```
SQL>INSERT INTO PUBLISHER VALUES('PEARSON','BANGALORE','9875462530');

SQL> INSERT INTO PUBLISHER VALUES('MCGRAW','NEWDELHI','7845691234');

SQL> INSERT INTO PUBLISHER VALUES('SAPNA','BANGALORE','7845963210');
```

BOOK

```
SQL> INSERT INTO BOOK VALUES(1111,'SE','PEARSON',2005);

SQL> INSERT INTO BOOK VALUES(2222,'DBMS','MCGRAW',2004);

SQL> INSERT INTO BOOK VALUES(3333,'ANOTOMY','PEARSON',2010); SQL>

INSERT INTO BOOK VALUES(4444,'ENCYCLOPEDIA','SAPNA',2010);
```

BOOK_AUTHORS

```
SQL> INSERT INTO BOOK_AUTHORS VALUES(1111,'SOMMERVILLE');

SQL> INSERT INTO BOOK_AUTHORS VALUES(2222,'NAVATHE'); SQL>
```

```
INSERT INTO BOOK_AUTHORS VALUES(3333,'HENRY GRAY');

SQL>INSERT INTO BOOK_AUTHORS VALUES(4444,'THOMAS');
```

LIBRARY_BRANCH

```
SQL> INSERT INTO LIBRARY_BRANCH VALUES(11,'CENTRAL TECHNICAL','MG
ROAD');

SQL> INSERT INTO LIBRARY_BRANCH VALUES(22,'MEDICAL','BH ROAD');

SQL> INSERT INTO LIBRARY_BRANCH VALUES(33,'CHILDREN','SS PURAM');

SQL> INSERT INTO LIBRARY_BRANCH VALUES(44,'SECRETARIAT','SIRAGATE');

SQL> INSERT INTO LIBRARY_BRANCH VALUES(55,'GENERAL','JAYANAGAR');
```

BOOK_COPIES

```
SQL> INSERT INTO BOOK_COPIES VALUES(1111,11,5);

SQL> INSERT INTO BOOK_COPIES VALUES(3333,22,6);

SQL> INSERT INTO BOOK_COPIES VALUES(4444,33,10);

SQL> INSERT INTO BOOK_COPIES VALUES(2222,11,12);

SQL> INSERT INTO BOOK_COPIES VALUES(4444,55,3);
```

BOOK_LENDING

```
SQL> INSERT INTO BOOK_LENDING VALUES(2222,11,1,'10-JAN-2017','20-AUG-2017');

SQL> INSERT INTO BOOK_LENDING VALUES(3333,22,2,'09-JUL-2017','12-AUG-2017');

SQL> INSERT INTO BOOK_LENDING VALUES(4444,55,1,'11-APR-2017','09-AUG-2017');

SQL> INSERT INTO BOOK_LENDING VALUES(2222,11,5,'09-AUG-2017','19-AUG-2017');

SQL> INSERT INTO BOOK_LENDING VALUES(4444,33,1,'10-JUN-2017','15-AUG-2017');

SQL> INSERT INTO BOOK_LENDING VALUES(1111,11,1,'12-MAY-2017','10-JUN-2017');

SQL> INSERT INTO BOOK_LENDING VALUES(3333,22,1,'10-JUL-2017','15-JUL-2017');
```

```
SQL> SELECT * FROM BOOK;

   BOOK_ID    TITLE                PUBLISHER_NAME    PUB_YEAR
---------- ---------------------- ------------------- ----------
      1111 SE                     PEARSON                2005
      2222 DBMS                   MCGRAW                 2004
      3333 ANOTOMY                PEARSON                2010
      4444 ENCYCLOPEDIA           SAPNA                  2010
```

```
  4 rows selected.
SQL> SELECT * FROM BOOK_AUTHORS;

   BOOK_ID AUTHOR_NAME
   ---------- ------------------------------
      1111 SOMMERVILLE
      2222 NAVATHE
      3333 HENRY GRAY
      4444 THOMAS

4 rows selected.

SQL> SELECT * FROM PUBLISHER;

NAME            ADDRESS          PHONE
--------------------------------------------------- ----------
PEARSON         BANGALORE        9875462530
MCGRAW          NEWDELHI         7845691234
SAPNA           BANGALORE        7845963210

3 rows selected.

SQL> SELECT * FROM BOOK_COPIES;

BOOK_ID BRANCH_ID NO_OF_COPIES
---------- ---------- -------------
    1111         11             5
    3333         22             6
    4444         33            10
    2222         11            12
    4444         55             3

5 rows selected.

SQL> SELECT * FROM BOOK_LENDING;

   BOOK_ID BRANCH_ID CARD_NO  DATE_OUT  DUE_DATE
   ---------- ---------- ---------- --------- ---------
      2222         11        1 10-JAN-17 20-AUG-17
      3333         22        2 09-JUL-17 12-AUG-17
      4444         55        1 11-APR-17 09-AUG-17
      2222         11        5 09-AUG-17 19-AUG-17
      4444         33        1 10-JUL-17 15-AUG-17
      1111         11        1 12-MAY-17 10-JUN-17
      3333         22        1 10-JUL-17 15-JUL-17

7 rows selected.
SQL> SELECT * FROM LIBRARY_BRANCH;

      BRANCH_ID BRANCH_NAME                    ADDRESS
      ---------- ------------------------------------------------- ------------------------
              11 CENTRAL TECHNICAL     MG ROAD
```

```
                22 MEDICAL              BH ROAD
                33 CHILDREN             SS PURAM
                44 SECRETARIAT          SIRAGATE
                55 GENERAL              JAYANAGAR

5 rows selected.
```

Queries:

1) Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each branch, etc.

```
SELECT LB.BRANCH_NAME, B.BOOK_ID,TITLE,
      PUBLISHER_NAME,AUTHOR_NAME, NO_OF_COPIES
FROM BOOK B, BOOK_AUTHORS BA, BOOK_COPIES BC,
LIBRARY_BRANCH LB WHERE B.BOOK_ID = BA.BOOK_ID AND
     BA.BOOK_ID = BC.BOOK_ID AND
     BC.BRANCH_ID = LB.BRANCH_ID
```

BRANCH_NAME	BOOK_ID	TITLE	PUBLISHER_NAME	AUTHOR_NAME	NO_OF_COPIES
GENERAL	4444	ENCYCLOPEDIA	SAPNA	THOMAS	3
MEDICAL	3333	ANOTOMY	PEARSON	HENRY GRAY	6
CHILDREN	4444	ENCYCLOPEDIA	SAPNA	THOMAS	10
CENTRAL TECHNICAL	1111	SE	PEARSON	SOMMERVILLE	5
CENTRAL TECHNICAL	2222	DBMS	MCGRAW	NAVATHE	12

2) Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017.

```
SELECT CARD_NO
FROM BOOK_LENDING
WHERE DATE_OUT BETWEEN '01-JAN-2017' AND '30-JUN-2017'
GROUP BY CARD_NO
HAVING COUNT(*) > 3;
```

```
   CARD_NO
----------
         1
```

3) Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulation operation.

```
DELETE FROM BOOK
WHERE BOOK_ID = '3333';

1 row deleted.
```

```
SQL> SELECT * FROM BOOK;

  BOOK_ID TITLE                PUBLISHER_NAME      PUB_YEAR
---------- -------------------- -------------------- ------------
        - --                                          --
     1111 SE                   PEARSON                   2005
     2222 DBMS                 MCGRAW                    2004
     4444 ENCYCLOPEDIA         SAPNA                     2010

           SQL> SELECT * FROM
                BOOK_COPIES;
             BOOK_ID BRANCH_ID
                  NO_OF_COPIES
---------- ---------- ----------
                              -
     1111         11          5
     4444         33         10
     2222         11         12
     4444         55          3

            SQL> SELECT * FROM
                 BOOK_LENDING;

BOOK_ID BRANCH_ID  CARD_NO DATE_OUT DUE_DATE
---------- ---------- ---------- --------- ---------
                                      --
                                   20-AUG-
     2222         11          1 10-JAN-17 17
                                   09-AUG-
     4444         55          1 11-APR-17 17
                                   19-AUG-
     2222         11          5 09-AUG-17 17
                                   15-AUG-
     4444         33          1 10-JUN-17 17
                                   10-JUN-
     1111         11          1 12-MAY-17 17
```

4) Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.

```
CREATE VIEW V_PUBLICATION AS
SELECT PUB_YEAR
FROM BOOK;

SELECT * FROM V_PUBLICATIONS;


  PUB_YEAR
      2004
      2005
      2010
      2010
```

5) Create a view of all books and its number of copies that are currently available in the Library.

```
CREATE VIEW BOOKS_AVAILABLE AS
SELECT B.BOOK_ID, B.TITLE, C.NO_OF_COPIES
FROM LIBRARY_BRANCH L, BOOK B, BOOK_COPIES C
WHERE B.BOOK_ID = C.BOOK_ID AND
L.BRANCH_ID=C.BRANCH_ID;
```

View created.

```
SQL> SELECT * FROM BOOKS_AVAILABLE;
```

BOOK_ID	TITLE	NO_OF_COPIES
1111	SE	5
3333	ANOTOMY	6
4444	ENCYCLOPEDIA	10
2222	DBMS	12
4444	ENCYCLOPEDIA	3

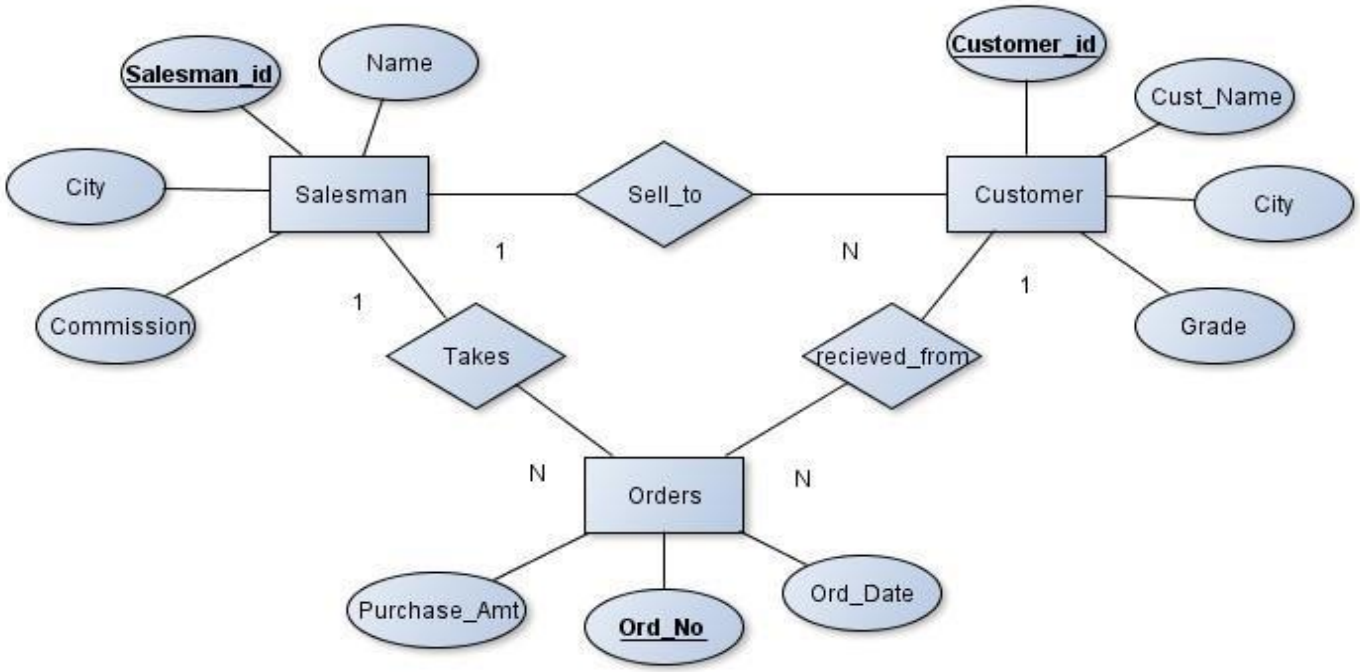
B. Consider the following schema for OrderDatabase:

SALESMAN (*Salesman_id, Name, City, Commission*)
CUSTOMER (*Customer_id, Cust_Name, City, Grade,Salesman_id*)
ORDERS (*Ord_No, Purchase_Amt, Ord_Date, Customer_id, Salesman_id*)
Write SQL queries to

1. **Count the customers with grades above Bangalore'saverage.**
2. **Find the name and numbers of all salesmen who had more than onecustomer.**
3. **List all salesmen and indicate those who have and don't have customers in their cities (Use UNIONoperation.)**
4. **Create a view that finds the salesman who has the customer with the highest order of a day.**
5. **Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also bedeleted.**

Solution:

Entity-Relationship Diagram



Schema Diagram

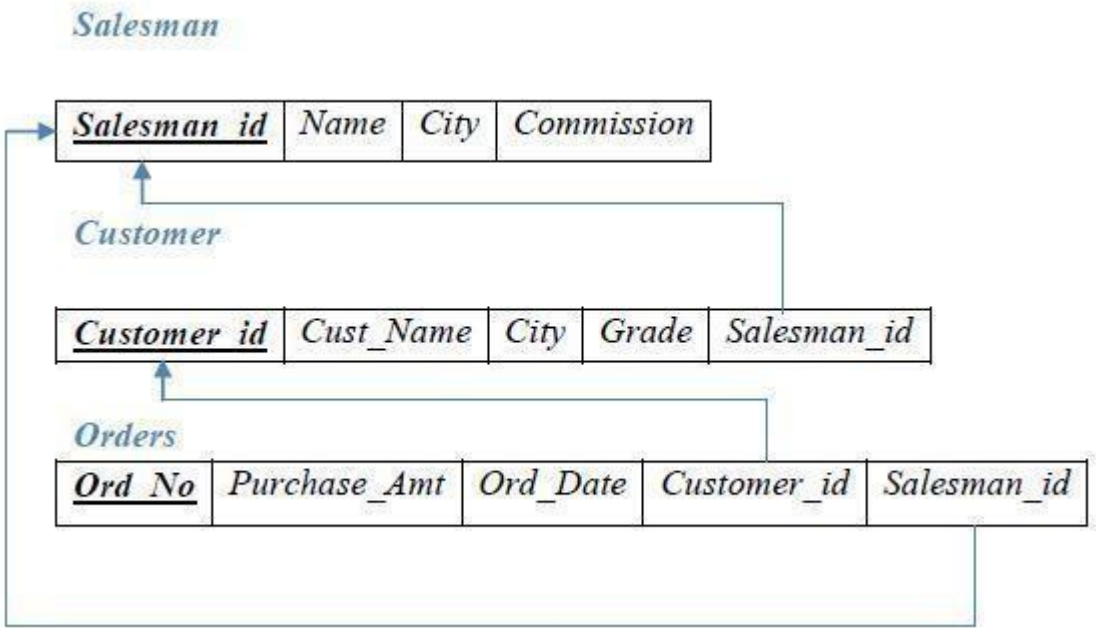


Table Creation

```
CREATE TABLE SALESMAN (SALESMAN_ID NUMBER (4),
NAME VARCHAR2 (20),
CITY VARCHAR2 (20),
COMMISSION VARCHAR2 (20), PRIMARYKEY(SALESMAN_ID));

CREATE TABLE CUSTOMER1 (CUSTOMER_ID NUMBER (4),
CUST_NAME VARCHAR2 (20),
CITY VARCHAR2 (20),
GRADE NUMBER (3),
SALESMAN_ID NUMBER (4),
PRIMARY KEY (CUSTOMER_ID),
FOREIGN KEY(SALESMAN_ID) REFERENCES SALESMAN (SALESMAN_ID) ON
DELETE SET NULL);

CREATE TABLE ORDERS (ORD_NO NUMBER (5),
PURCHASE_AMT NUMBER (10, 2),
ORD_DATE DATE,
CUSTOMER_ID NUMBER (4),
SALESMAN_ID NUMBER (4),
PRIMARY KEY (ORD_NO),
CUSTOMER_ID REFERENCES CUSTOMER1 (CUSTOMER_ID) ON DELETE
CASCADE, SALESMAN_ID REFERENCES SALESMAN (SALESMAN_ID) ON
DELETE CASCADE);
```

Table Descriptions

DESC SALESMAN;

```
SQL> DESC SALESMAN;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SALESMAN_ID                               NOT NULL NUMBER(4)
 NAME                                               VARCHAR2(15)
 CITY                                               VARCHAR2(15)
 COMMISSION                                         NUMBER(3,2)
```

DESC CUSTOMER1;

```
SQL> DESC CUSTOMER1;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CUSTOMER_ID                               NOT NULL NUMBER(4)
 CUST_NAME                                          VARCHAR2(15)
 CITY                                               VARCHAR2(15)
 GRADE                                              NUMBER(3)
 SALESMAN_ID                                        NUMBER(4)
```

DESC ORDERS;

```
SQL> DESC ORDERS;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ORD_NO                                    NOT NULL NUMBER(5)
 PURCHASE_AMT                                       NUMBER(10,2)
 ORD_DATE                                           DATE
 CUSTOMER_ID                                        NUMBER(4)
 SALESMAN_ID                                        NUMBER(4)
```

Insertion of Values to Tables

INSERT INTO SALESMAN VALUES (1000, ‘JOHN’,’BANGALORE’,’25 %’);
INSERT INTO SALESMAN VALUES (2000, ‘RAVI’,’BANGALORE’,’20 %’);
INSERT INTO SALESMAN VALUES (3000, ‘KUMAR’,’MYSORE’,’15 %’);
INSERT INTO SALESMAN VALUES (4000, ‘SMITH’,’DELHI’,’30 %’);
INSERT INTO SALESMAN VALUES (5000, ‘HARSHA’,’HYDRABAD’,’15%’);

INSERT INTO CUSTOMER1 VALUES (10, ‘PREETHI’,’BANGALORE’, 100, 1000);
INSERT INTO CUSTOMER1 VALUES (11, ‘VIVEK’,’MANGALORE’, 300, 1000);
INSERT INTO CUSTOMER1 VALUES (12, ‘BHASKAR’,’CHENNAI’, 400, 2000);
INSERT INTO CUSTOMER1 VALUES (13, ‘CHETHAN’,’BANGALORE’, 200, 2000);
INSERT INTO CUSTOMER1 VALUES (14, ‘MAMATHA’,’BANGALORE’, 400, 3000);

INSERT INTO ORDERS VALUES (50, 5000, ‘04-MAY-17’, 10, 1000);
INSERT INTO ORDERS VALUES (51, 450, ‘20-JAN-17’, 10, 2000);

INSERT INTO ORDERS VALUES (52, 1000, '24-FEB-17', 13, 2000);
INSERT INTO ORDERS VALUES (53, 3500, '13-APR-17', 14, 3000);
INSERT INTO ORDERS VALUES (54, 550, '09-MAR-17', 12, 2000);

SELECT * FROM SALESMAN;

SALESMAN_ID	NAME	CITY	COMMISSION
1000	JOHN	BANGALORE	25 %
2000	RAVI	BANGALORE	20 %
3000	KUMAR	MYSORE	15 %
4000	SMITH	DELHI	30 %
5000	HARSHA	HYDRABAD	15 %

SELECT * FROM CUSTOMER1;

CUSTOMER_ID	CUST_NAME	CITY	GRADE	SALESMAN_ID
10	PREETHI	BANGALORE	100	1000
11	VIVEK	MANGALORE	300	1000
12	BHASKAR	CHENNAI	400	2000
13	CHETHAN	BANGALORE	200	2000
14	MAMATHA	BANGALORE	400	3000

SELECT * FROM ORDERS;

ORD_NO	PURCHASE_AMT	ORD_DATE	CUSTOMER_ID	SALESMAN_ID
50	5000	04-MAY-17	10	1000
51	450	20-JAN-17	10	2000
52	1000	24-FEB-17	13	2000
53	3500	13-APR-17	14	3000
54	550	09-MAR-17	12	2000

Queries:

1. **Count the customers with grades above Bangalore'saverage.**

```
SELECT GRADE, COUNT (DISTINCT CUSTOMER_ID)
FROMCUSTOMER1
GROUP BY GRADE
HAVING GRADE > (SELECT AVG(GRADE)
FROM CUSTOMER1
WHERE CITY='BANGALORE');
```

GRADE	COUNT(DISTINCTCUSTOMER_ID)
300	1
400	2

2. Find the name and numbers of all salesmen who had more than onecustomer.

```
SELECT SALESMAN_ID, NAME
 FROM SALESMAN A
 WHERE 1 < (SELECT COUNT (*)
        FROM CUSTOMER1
 WHERE SALESMAN_ID=A.SALESMAN_ID);
```

```
SALESMAN_ID NAME
----------- --------------------
       1000 JOHN
       2000 RAVI
```

3. List all salesmen and indicate those who have and don't have customers in their cities (Use UNIONoperation.)

```
SELECT SALESMAN.SALESMAN_ID, NAME, CUST_NAME, COMMISSION
FROM SALESMAN, CUSTOMER1
WHERE SALESMAN.CITY = CUSTOMER1.CITY
UNION
SELECT SALESMAN_ID, NAME, 'NO MATCH', COMMISSION
FROM SALESMAN
WHERE NOT CITY = ANY
 (SELECT CITY
  FROM CUSTOMER1)
ORDER BY 2 DESC;
```

```
SALESMAN_ID NAME                 CUST_NAME            COMMISSION
----------- -------------------- -------------------- --------------------
       4000 SMITH                NO MATCH             30 %
       2000 RAVI                 CHETHAN              20 %
       2000 RAVI                 MAMATHA              20 %
       2000 RAVI                 PREETHI              20 %
       3000 KUMAR                NO MATCH             15 %
       1000 JOHN                 CHETHAN              25 %
       1000 JOHN                 MAMATHA              25 %
       1000 JOHN                 PREETHI              25 %
       5000 HARSHA               NO MATCH             15 %
```

4. Create a view that finds the salesman who has the customer with the highest order of a day.

```
CREATE VIEW ELITSALESMAN AS
SELECT B.ORD_DATE, A.SALESMAN_ID, A.NAME
FROM SALESMAN A, ORDERS B
```

```
WHERE A.SALESMAN_ID = B.SALESMAN_ID
AND B.PURCHASE_AMT=(SELECT MAX (PURCHASE_AMT)
                                FROM ORDERS C
                                    WHERE C.ORD_DATE = B.ORD_DATE);
```

```
ORD_DATE  SALESMAN_ID NAME
--------- ----------- --------------------
04-MAY-17        1000 JOHN
20-JAN-17        2000 RAVI
24-FEB-17        2000 RAVI
13-APR-17        3000 KUMAR
09-MAR-17        2000 RAVI
```

5. Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also bedeleted.

Use ON DELETE CASCADE at the end of foreign key definitions while creating child table orders and then execute the following:

Use ON DELETE SET NULL at the end of foreign key definitions while creating child table customers and then executes the following:

```
DELETE FROM SALESMAN
WHERE SALESMAN_ID=1000;
```

```
SQL> DELETE FROM SALESMAN
  2  WHERE SALESMAN_ID=1000;

1 row deleted.

SQL> SELECT * FROM SALESMAN;

SALESMAN_ID NAME                 CITY                 COMMISSION
----------- -------------------- -------------------- ----------
       2000 RAVI                 BANGALORE            20 %
       3000 KUMAR                MYSORE               15 %
       4000 SMITH                DELHI                30 %
       5000 HARSHA               HYDRABAD             15 %
```

C. Consider the schema for MovieDatabase:

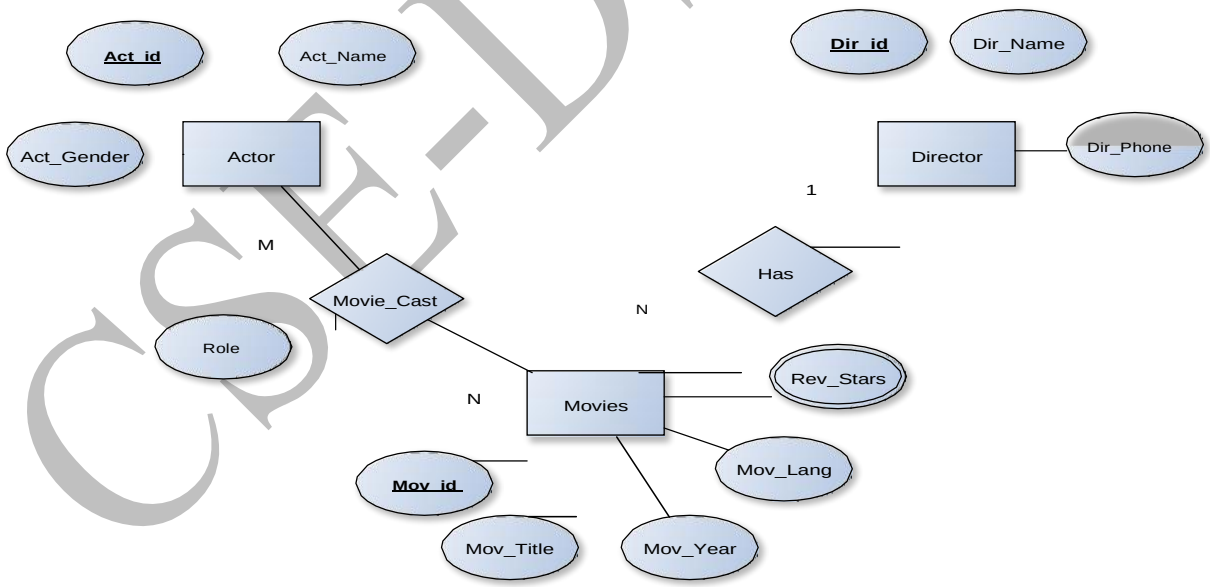
ACTOR (*Act_id*, *Act_Name, Act_Gender*)
DIRECTOR (*Dir_id*, *Dir_Name, Dir_Phone*)
MOVIES (*Mov_id*, *Mov_Title, Mov_Year, Mov_Lang, Dir_id*)
MOVIE_CAST (*Act_id*, *Mov_id*, *Role*)
RATING (*Mov_id*, *Rev_Stars*)

Write SQL queries to

1. **List the titles of all movies directed by'Hitchcock'.**
2. **Find the movie names where one or more actors acted in two or moremovies.**
3. **List all actors who acted in a movie before 2000 and also in a movieafter 2015 (use JOINoperation).**
4. **Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movietitle.**
5. **Update rating of all movies directed by 'Steven Spielberg' to5.**

Solution:

Entity-Relationship Diagram



Schema Diagram

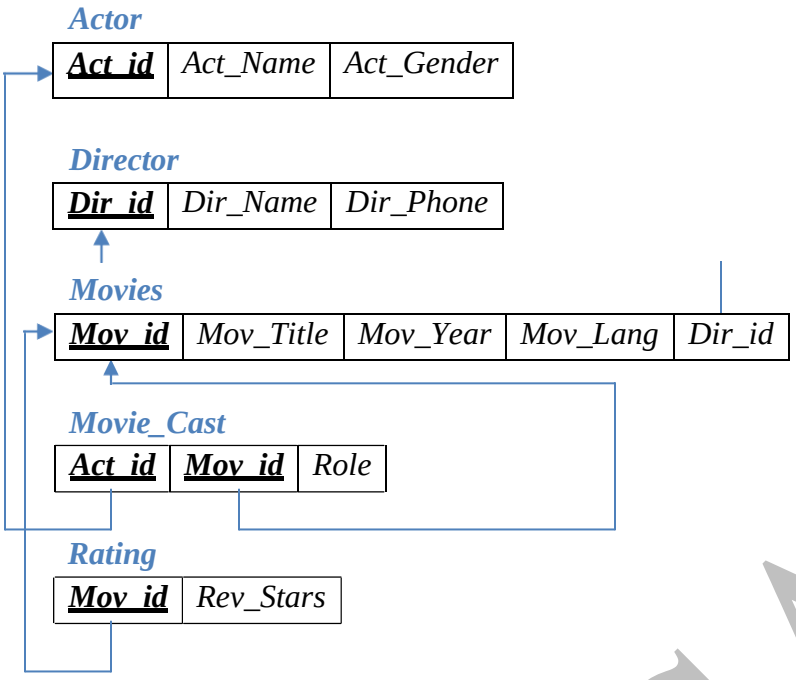


Table Creation

```
CREATE TABLE ACTOR (
ACT_ID NUMBER (3),
ACT_NAME VARCHAR (20),
ACT_GENDER CHAR (1),
PRIMARY KEY (ACT_ID));

CREATE TABLE DIRECTOR (
DIR_ID NUMBER (3),
DIR_NAME VARCHAR (20),
DIR_PHONE NUMBER (10),
PRIMARY KEY (DIR_ID));

CREATE TABLE MOVIES (
MOV_ID NUMBER (4),
MOV_TITLE  VARCHAR (25),
MOV_YEAR  NUMBER (4),
MOV_LANG  VARCHAR (12),
DIR_ID NUMBER (3),
PRIMARY KEY (MOV_ID),
FOREIGN KEY (DIR_ID) REFERENCES DIRECTOR (DIR_ID));
```

```
CREATE TABLE MOVIE_CAST (
ACT_ID NUMBER (3),
MOV_ID NUMBER (4),
ROLE  VARCHAR (10),
PRIMARY KEY (ACT_ID, MOV_ID),
FOREIGN KEY (ACT_ID) REFERENCES ACTOR (ACT_ID),
FOREIGN KEY (MOV_ID) REFERENCES MOVIES (MOV_ID));

CREATE TABLE RATING (
MOV_ID NUMBER (4),
REV_STARS VARCHAR (25),
PRIMARY KEY (MOV_ID),
FOREIGN KEY (MOV_ID) REFERENCES MOVIES (MOV_ID));
```

Table Descriptions

DESC ACTOR;

```
SQL> DESC ACTOR;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ACT_ID                                    NOT NULL NUMBER(3)
 ACT_NAME                                           VARCHAR2(20)
 ACT_GENDER                                         CHAR(1)
```

DESC DIRECTOR;

```
SQL> DESC DIRECTOR;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 DIR_ID                                    NOT NULL NUMBER(3)
 DIR_NAME                                           VARCHAR2(20)
 DIR_PHONE                                          NUMBER(10)
```

DESC MOVIES;

```
SQL> DESC MOVIES;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 MOV_ID                                    NOT NULL NUMBER(4)
 MOV_TITLE                                          VARCHAR2(25)
 MOV_YEAR                                           NUMBER(4)
 MOV_LANG                                           VARCHAR2(12)
 DIR_ID                                             NUMBER(3)
```

DESC MOVIE_CAST;

```
SQL> DESC MOVIE_CAST;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ACT_ID                                    NOT NULL NUMBER(3)
 MOV_ID                                    NOT NULL NUMBER(4)
 ROLE                                               VARCHAR2(10)
```

DESC RATING;

```
SQL> DESC RATING;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 MOV_ID                                    NOT NULL NUMBER(4)
 REV_STARS                                          VARCHAR2(25)
```

Insertion of Values to Tables

INSERT INTO ACTOR VALUES (301,'ANUSHKA','F');
INSERT INTO ACTOR VALUES (302,'PRABHAS','M');
INSERT INTO ACTOR VALUES (303,'PUNITH','M');
INSERT INTO ACTOR VALUES (304,'JERMY','M');

INSERT INTO DIRECTOR VALUES (60,'RAJAMOULI', 8751611001);
INSERT INTO DIRECTOR VALUES (61,'HITCHCOCK', 7766138911);
INSERT INTO DIRECTOR VALUES (62,'FARAN', 9986776531);
INSERT INTO DIRECTOR VALUES (63,'STEVEN SPIELBERG', 8989776530);

INSERT INTO MOVIES VALUES (1001,'BAHUBALI-2', 2017, 'TELAGU', 60);
INSERT INTO MOVIES VALUES (1002,'BAHUBALI-1', 2015, 'TELAGU', 60);
INSERT INTO MOVIES VALUES (1003,'AKASH', 2008, 'KANNADA', 61);
INSERT INTO MOVIES VALUES (1004,'WAR HORSE', 2011, 'ENGLISH', 63);

INSERT INTO MOVIE_CAST VALUES (301, 1002, 'HEROINE');
INSERT INTO MOVIE_CAST VALUES (301, 1001, 'HEROINE');
INSERT INTO MOVIE_CAST VALUES (303, 1003, 'HERO');
INSERT INTO MOVIE_CAST VALUES (303, 1002, 'GUEST');
INSERT INTO MOVIE_CAST VALUES (304, 1004, 'HERO');

INSERT INTO RATING VALUES (1001, 4);
INSERT INTO RATING VALUES (1002, 2);

INSERT INTO RATING VALUES (1003, 5);
INSERT INTO RATING VALUES (1004, 4);

SELECT * FROM ACTOR;

```
SQL> SELECT * FROM ACTOR;

    ACT_ID ACT_NAME             A
---------- -------------------- -
       301 ANUSHKA              F
       302 PRABHAS              M
       303 PUNITH               M
       304 JERMY                M
```

SELECT * FROM DIRECTOR;

```
SQL> SELECT * FROM DIRECTOR;

    DIR_ID DIR_NAME              DIR_PHONE
---------- -------------------- ----------
        60 RAJAMOULI            8751611001
        61 HITCHCOCK            7766138911
        62 FARAN                9986776531
        63 STEVEN SPIELBERG     8989776530
```

SELECT * FROM MOVIES;

```
SQL> SELECT * FROM MOVIES;

    MOV_ID MOV_TITLE                MOV_YEAR MOV_LANG        DIR_ID
---------- ------------------------ ---------- ------------ ----------
      1001 BAHUBALI-2                     2017 TELAGU               60
      1002 BAHUBALI-1                     2015 TELAGU               60
      1003 AKASH                          2008 KANNADA              61
      1004 WAR HORSE                      2011 ENGLISH              63
```

SELECT * FROM MOVIE_CAST;

```
SQL> SELECT * FROM MOVIE_CAST;

    ACT_ID     MOV_ID ROLE
---------- ---------- ----------
       301       1002 HEROINE
       301       1001 HEROINE
       303       1003 HERO
       303       1002 GUEST
       304       1004 HERO
```

SELECT * FROM RATING;

```
SQL> SELECT * FROM RATING;

    MOV_ID REV_STARS
---------- -----------
      1001 4
      1002 2
      1003 5
      1004 4
```

Queries:

1. **List the titles of all movies directed by'Hitchcock'.**

```
SELECT MOV_TITLE
FROM MOVIES
WHERE DIR_ID IN (SELECT DIR_ID
                 FROM DIRECTOR
                 WHERE DIR_NAME = 'HITCHCOCK');
```

```
MOV_TITLE
-------------------------
AKASH
```

2. **Find the movie names where one or more actors acted in two or moremovies.**

```
SELECT MOV_TITLE
FROM MOVIES M, MOVIE_CAST MV
WHERE M.MOV_ID=MV.MOV_ID AND ACT_ID IN (SELECT ACT_ID
                                        FROM MOVIE_CAST GROUP BY ACT_ID
                                        HAVING COUNT (ACT_ID)>1)
GROUP BY MOV_TITLE
HAVING COUNT (*)>1;
```

```
MOV_TITLE
------------------------
BAHUBALI-1
```

3. **List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOINoperation).**

```
SELECT ACT_NAME, MOV_TITLE, MOV_YEAR
```

```
FROM ACTOR A
JOIN MOVIE_CASTC
        ON A.ACT_ID=C.ACT_ID
JOIN MOVIES M
        ON C.MOV_ID=M.MOV_ID
WHERE M.MOV_YEAR NOT BETWEEN 2000 AND 2015;
```

OR

```
SELECT A.ACT_NAME, A.ACT_NAME, C.MOV_TITLE, C.MOV_YEAR
FROM ACTOR A, MOVIE_CAST B, MOVIES C
WHERE A.ACT_ID=B.ACT_ID
AND B.MOV_ID=C.MOV_ID
AND C.MOV_YEAR NOT BETWEEN 2000 AND 2015;
```

ACT_NAME	MOV_TITLE	MOV_YEAR
ANUSHKA	BAHUBALI-2	2017

4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movietitle.

```
SELECT MOV_TITLE, MAX (REV_STARS)
FROM MOVIES
INNER JOIN RATING USING (MOV_ID)
GROUP BY MOV_TITLE
HAVING MAX (REV_STARS)>0
ORDER BY MOV_TITLE;
```

MOV_TITLE	MAX(REV_STARS)
AKASH	5
BAHUBALI-1	2
BAHUBALI-2	4
WAR HORSE	4

5. Update rating of all movies directed by 'Steven Spielberg' to5
KL

```
UPDATE RATING
SET REV_STARS=5
WHERE MOV_ID IN (SELECT MOV_ID FROM MOVIES
                 WHERE DIR_ID IN (SELECT DIR_ID
                                  FROM DIRECTOR
                                  WHERE DIR_NAME = 'STEVEN
                                  SPIELBERG'));
```

```
SQL> SELECT * FROM RATING;

    MOV_ID REV_STARS
---------- -----------
      1001 4
      1002 2
      1003 5
      1004 5
```

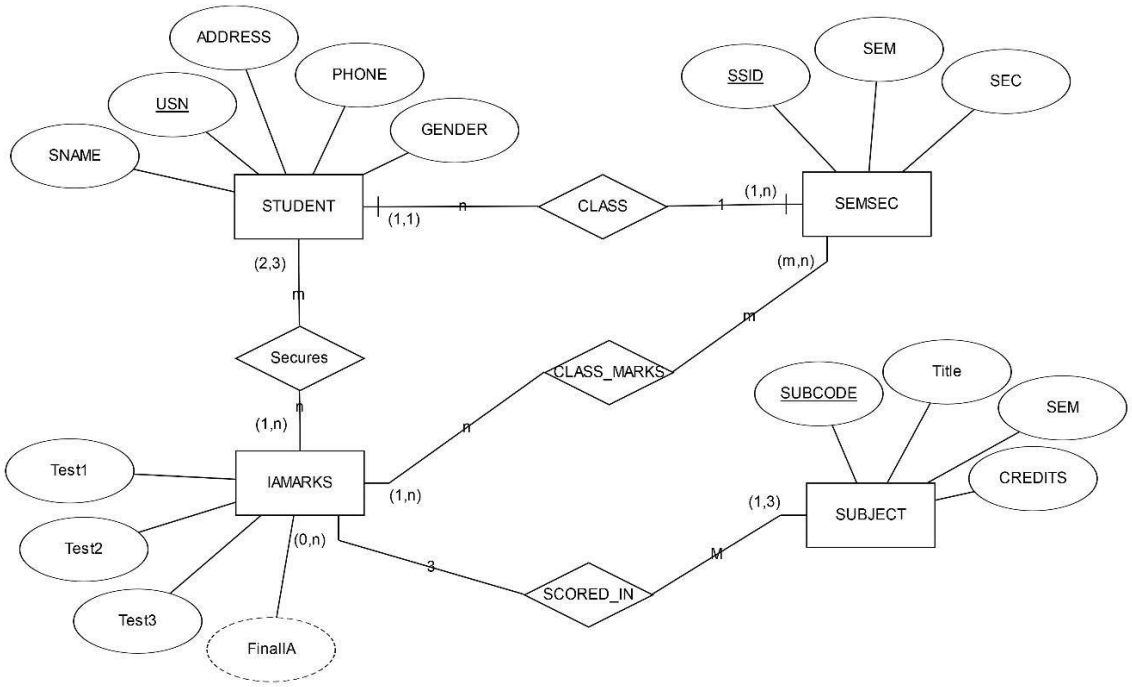
D. Consider the schema for CollegeDatabase:

STUDENT (*USN, SName, Address, Phone, Gender*)
SEMSEC (*SSID, Sem, Sec*)
CLASS (*USN, SSID*)
SUBJECT (*Subcode, Title, Sem, Credits*)
IAMARKS (*USN, Subcode, SSID, Test1, Test2, Test3, FinalIA*)
Write SQL queries to

1. **List all the student details studying in fourth semester 'C'section.**
2. **Compute the total number of male and female students in each semester and in each section.**
3. **Create a view of Test1 marks of student USN '1BI15CS101' in allsubjects.**
4. **Calculate the FinalIA (average of best two test marks) and update thecorresponding table for allstudents.**
5. **Categorize students based on the followingcriterion:**
   If FinalIA = 17 to 20 then CAT ='Outstanding'
   If FinalIA = 12 to 16 then CAT = 'Average'
   If FinalIA< 12 then CAT = 'Weak'
   Give these details only for 8th semester A, B, and C section students.

Solution:

Entity - Relationship Diagram



Schema Diagram

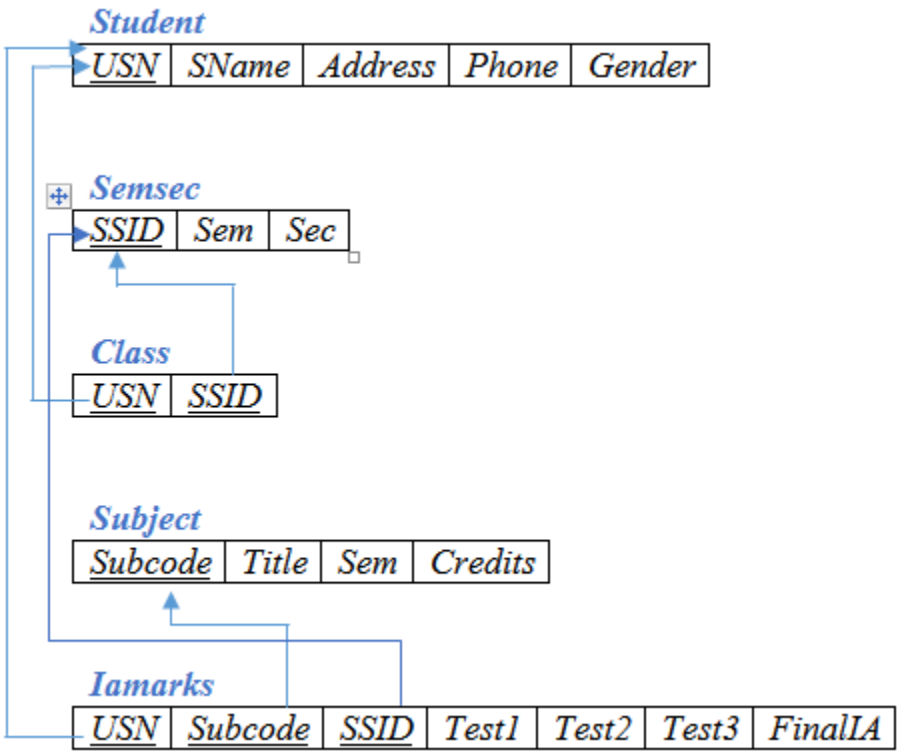


Table Creation

```
CREATE TABLE STUDENT (
 USN VARCHAR (10) PRIMARY KEY,
 SNAME VARCHAR (25),
 ADDRESS VARCHAR (25),
 PHONE NUMBER (10),
 GENDER CHAR (1));

CREATE TABLE SEMSEC (
SSID VARCHAR (5) PRIMARY KEY,
SEM NUMBER (2),
SEC CHAR (1));

CREATE TABLE CLASS (
USN VARCHAR (10),
SSID VARCHAR (5),
PRIMARY KEY (USN, SSID),
FOREIGN KEY (USN) REFERENCES STUDENT (USN),
FOREIGN KEY (SSID) REFERENCES SEMSEC (SSID));
```

```
CREATE TABLE SUBJECT (
SUBCODE VARCHAR (8),
TITLE VARCHAR (20),
SEM NUMBER (2),
CREDITS NUMBER (2),
PRIMARY KEY (SUBCODE));

 CREATE TABLE IAMARKS (
 USN VARCHAR (10),
 SUBCODE VARCHAR (8),
 SSID VARCHAR (5),
 TEST1 NUMBER (2),
 TEST2 NUMBER (2),
 TEST3 NUMBER (2),
 FINALIA NUMBER (2),
 PRIMARY KEY (USN, SUBCODE, SSID),
FOREIGN KEY (USN) REFERENCES STUDENT (USN),
 FOREIGN KEY (SUBCODE) REFERENCES SUBJECT (SUBCODE),
 FOREIGN KEY (SSID) REFERENCES SEMSEC (SSID));
```

Table Descriptions

DESC STUDENT;

```
Name
--------------------------------------------------------------------------------------------
USN
SNAME
ADDRESS
PHONE
GENDER
```

DESC SEMSEC;

```
SQL> DESC SEMSEC;
 Name
 ------------------------------------------------------------------------------------------
 SSID
 SEM
 SEC
```

DESC CLASS;

```
SQL> DESC CLASS;
 Name
 -----------------------------------------
 USN
 SSID
```

DESC SUBJECT;

```
SQL> DESC SUBJECT1;
 Name
 ----------------------------------
 SUBCODE
 TITLE
 SEM
 CREDITS
```

DESC IAMARKS;

```
SQL> DESC IAMARKS;
 Name
 -----------------------------------------------
 USN
 SUBCODE
 SSID
 TEST1
 TEST2
 TEST3
 FINALIA
```

Insertion of values to tables

INSERT INTO STUDENT VALUES ('1RN13CS020','AKSHAY','BELAGAVI', 8877881122,'M');
INSERT INTO STUDENT VALUES ('1RN13CS062','SANDHYA','BENGALURU', 7722829912,'F');
INSERT INTO STUDENT VALUES ('1RN13CS091','TEESHA','BENGALURU', 7712312312,'F');
INSERT INTO STUDENT VALUES ('1RN13CS066','SUPRIYA','MANGALURU', 8877881122,'F');
INSERT INTO STUDENTVALUES ('1RN14CS010','ABHAY','BENGALURU', 9900211201,'M');
INSERT INTO STUDENT VALUES ('1RN14CS032','BHASKAR','BENGALURU', 9923211099,'M');
INSERT INTO STUDENTVALUES ('1RN14CS025','ASMI','BENGALURU', 7894737377,'F');
INSERT INTO STUDENT VALUES ('1RN15CS011','AJAY','TUMKUR', 9845091341,'M');

```
INSERT INTO STUDENT VALUES ('1RN15CS029','CHITRA','DAVANGERE',
7696772121,'F');
INSERT INTO STUDENT VALUES ('1RN15CS045','JEEVA','BELLARY', 9944850121,'M');
INSERT INTO STUDENT VALUES ('1RN15CS091','SANTOSH','MANGALURU',
8812332201,'M');
INSERT INTO STUDENT VALUES ('1RN16CS045','ISMAIL','KALBURGI',
9900232201,'M');
INSERT INTO STUDENT VALUES ('1RN16CS088','SAMEERA','SHIMOGA',
9905542212,'F');
INSERT INTO STUDENT VALUES ('1RN16CS122','VINAYAKA','CHIKAMAGALUR',
8800880011,'M');

INSERT INTO SEMSEC VALUES ('CSE8A', 8,'A');
INSERT INTO SEMSEC VALUES (‘CSE8B', 8,'B');
INSERT INTO SEMSEC VALUES (‘CSE8C’,8,’C’);

INSERT INTO SEMSEC VALUES ('CSE7A', 7,’A’);
INSERT INTO SEMSEC VALUES (‘CSE7B’,7,'B’);
INSERT INTO SEMSEC VALUES ('CSE7C',7,'C');

INSERT INTO SEMSEC VALUES (‘CSE6A', 6,'A');
INSERT INTO SEMSEC VALUES (‘CSE6B’, 6,’B’);
INSERT INTO SEMSEC VALUES ('CSE6C’, 6,’C’);

INSERT INTO SEMSEC VALUES (‘CSE5A’, 5,'A’);
INSERT INTO SEMSEC VALUES ('CSE5B', 5,'B');
INSERT INTO SEMSEC VALUES (‘CSE5C', 5,'C');

INSERT INTO SEMSEC VALUES (‘CSE4A’, 4,’A’);
INSERT INTO SEMSEC VALUES ('CSE4B', 4,’B’);
INSERT INTO SEMSEC VALUES (‘CSE4C’, 4,'C’);

INSERT INTO SEMSEC VALUES ('CSE3A', 3,'A');
INSERT INTO SEMSEC VALUES (‘CSE3B', 3,'B');
INSERT INTO SEMSEC VALUES (‘CSE3C’, 3,’C’);

INSERT INTO SEMSEC VALUES ('CSE2A', 2,’A’);
INSERT INTO SEMSEC VALUES (‘CSE2B’, 2,'B’);
INSERT INTO SEMSEC VALUES ('CSE2C', 2,'C');
INSERT INTO SEMSEC VALUES (‘CSE1A', 1,'A');
```

```
INSERT INTO SEMSEC VALUES (‘CSE1B’, 1,’B’);
INSERT INTO SEMSEC VALUES ('CSE1C', 1,’C’);

INSERT INTO CLASS VALUES (‘1RN13CS020’,’CSE8A’);
INSERT INTO CLASS VALUES (‘1RN13CS062’,’CSE8A’);
INSERT INTO CLASS VALUES (‘1RN13CS066’,’CSE8B’);
INSERT INTO CLASS VALUES (‘1RN13CS091’,’CSE8C’);

INSERT INTO CLASS VALUES (‘1RN14CS010’,’CSE7A’);
INSERT INTO CLASS VALUES (‘1RN14CS025’,’CSE7A’);
INSERT INTO CLASS VALUES (‘1RN14CS032’,’CSE7A’);

INSERT INTO CLASS VALUES (‘1RN15CS011’,’CSE4A’);
INSERT INTO CLASS VALUES (‘1RN15CS029’,’CSE4A’);
INSERT INTO CLASS VALUES (‘1RN15CS045’,’CSE4B’);
INSERT INTO CLASS VALUES (‘1RN15CS091’,’CSE4C’);

INSERT INTO CLASS VALUES (‘1RN16CS045’,’CSE3A’);
INSERT INTO CLASS VALUES (‘1RN16CS088’,’CSE3B’);
INSERT INTO CLASS VALUES (‘1RN16CS122’,’CSE3C’);

INSERT INTO SUBJECT VALUES ('10CS81','ACA', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS82','SSM', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS83','NM', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS84','CC', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS85','PW', 8, 4);

INSERT INTO SUBJECT VALUES ('10CS71','OOAD', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS72','ECS', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS73','PTW', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS74','DWDM', 7, 4);
INSERT INTO SUBJECT VALUES (‘10CS75','JAVA', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS76','SAN', 7, 4);

INSERT INTO SUBJECT VALUES ('15CS51', 'ME', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS52','CN', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS53','DBMS', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS54','ATC', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS55','JAVA', 5, 3);
INSERT INTO SUBJECT VALUES ('15CS56','AI', 5, 3);
```

```
INSERT INTO SUBJECT VALUES ('15CS41','M4', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS42','SE', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS43','DAA', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS44','MPMC', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS45','OOC', 4, 3);
INSERT INTO SUBJECT VALUES ('15CS46','DC', 4, 3);

INSERT INTO SUBJECT VALUES ('15CS31','M3', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS32','ADE', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS33','DSA', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS34','CO', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS35','USP', 3, 3);
INSERT INTO SUBJECT VALUES ('15CS36','DMS', 3, 3);

INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS81','CSE8C', 15, 16, 18);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS82','CSE8C', 12, 19, 14);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS83','CSE8C', 19, 15, 20);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS84','CSE8C', 20, 16, 19);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS85','CSE8C', 15, 15, 12);
```

SELECT * FROM STUDENT;

```
SQL> SELECT * FROM STUDENT1;

USN         SNAME                     ADDRESS                       PHONE G
----------  ------------------------- ------------------------- ---------- -
1RN13CS020 AKSHAY                    BELAGAVI                  8877881122 M
1RN13CS062 SANDHYA                   BENGALURU                 7722829912 F
1RN13CS091 TEESHA                    BENGALURU                 7712312312 F
1RN13CS066 SUPRIYA                   MANGALURU                 8877881122 F
1RN14CS010 ABHAY                     BENGALURU                 9900211201 M
1RN14CS032 BHASKAR                   BENGALURU                 9923211099 M
1RN15CS011 AJAY                      TUMKUR                    9845091341 M
1RN15CS029 CHITRA                    DAVANGERE                 7696772121 F
1RN15CS045 JEEVA                     BELLARY                   9944850121 M
1RN15CS091 SANTOSH                   MANGALURU                 8812332201 M
1RN16CS045 ISMAIL                    KALBURGI                  9900232201 M
1RN16CS088 SAMEERA                   SHIMOGA                   9905542212 F
1RN16CS122 VINAYAKA                  CHIKAMAGALUR              8800880011 M
1RN14CS025 ASMI                      BENGALURU                 7894737377 F
```

SELECT * FROM SEMSEC;

```
SQL> SELECT * FROM SEMSEC;

SSID         SEM S
----- ---------- -
CSE8A          8 A
CSE8B          8 B
CSE8C          8 C
CSE7A          7 A
CSE7B          7 B
CSE7C          7 C
CSE6A          6 A
CSE6B          6 B
CSE6C          6 C
CSE5A          5 A
CSE5B          5 B
CSE5C          5 C
CSE4A          4 A
CSE4B          4 B
CSE4C          4 C
CSE3A          3 A
CSE3B          3 B
CSE3C          3 C
CSE2A          2 A
CSE2C          2 C
CSE2B          2 B
CSE1A          1 A
CSE1B          1 B
CSE1C          1 C
```

SELECT * FROM CLASS;

```
SQL> SELECT * FROM CLASS;

USN        SSID
---------- -----
1RN13CS020 CSE8A
1RN13CS062 CSE8A
1RN13CS066 CSE8B
1RN13CS091 CSE8C
1RN14CS010 CSE7A
1RN14CS025 CSE7A
1RN14CS032 CSE7A
1RN15CS011 CSE4A
1RN15CS029 CSE4A
1RN15CS045 CSE4B
1RN15CS091 CSE4C
1RN16CS045 CSE3A
1RN16CS088 CSE3B
1RN16CS122 CSE3C

14 rows selected.
```

SELECT * FROM SUBJECT;

SUBCODE	TITLE	SEM	CREDITS
10CS81	ACA	8	4
10CS82	SSM	8	4
10CS83	NM	8	4
10CS84	CC	8	4
10CS85	PW	8	4
10CS71	OOAD	7	4
10CS72	ECS	7	4
10CS73	PTW	7	4
10CS74	DWDM	7	4
10CS75	JAVA	7	4
10CS76	SAN	7	4
15CS51	ME	5	4
15CS52	CN	5	4
15CS53	DBMS	5	4
15CS54	ATC	5	4
15CS55	JAVA	5	3
15CS56	AI	5	3
15CS41	M4	4	4
15CS42	SE	4	4
15CS43	DAA	4	4
15CS44	MPMC	4	4
15CS45	OOC	4	3
15CS46	DC	4	3
15CS31	M3	3	4
15CS32	ADE	3	4
15CS33	DSA	3	4
15CS34	CO	3	4
15CS35	USP	3	3
15CS36	DMS	3	3

SELECT * FROM IAMARKS;

```
SQL> SELECT * FROM IAMARKS;
```

USN	SUBCODE	SSID	TEST1	TEST2	TEST3	FINALIA
1RN13CS091	10CS81	CSE8C	15	16	18	
1RN13CS091	10CS82	CSE8C	12	19	14	
1RN13CS091	10CS83	CSE8C	19	15	20	
1RN13CS091	10CS84	CSE8C	20	16	19	
1RN13CS091	10CS85	CSE8C	15	15	12	

Queries:

1. **List all the student details studying in fourth semester 'C'section.**

```
SELECT S.*, SS.SEM, SS.SEC
 FROM STUDENT S, SEMSEC SS, CLASS C
 WHERE S.USN = C.USN AND
 SS.SSID = C.SSID AND
 SS.SEM = 4 AND
```

SS.SEc=’C’;

USN	SNAME	ADDRESS	PHONE	G	SEM	S
1RN15CS091	SANTOSH	MANGALURU	8812332201	M	4	C

2. **Compute the total number of male and female students in each semester and ineach section.**

```
SELECT SS.SEM, SS.SEC, S.GENDER, COUNT (S.GENDER) AS COUNT
FROM STUDENT S, SEMSEC SS, CLASS C
WHERES.USN = C.USN AND
SS.SSID = C.SSID
GROUP BY SS.SEM, SS.SEC, S.GENDER
ORDER BY SEM;
```

SEM	S	G	COUNT
3	A	M	1
3	B	F	1
3	C	M	1
4	A	F	1
4	A	M	1
4	B	M	1
4	C	M	1
7	A	F	1
7	A	M	2
8	A	F	1
8	A	M	1
8	B	F	1
8	C	F	1

3. **Create a view of Test1 marks of student USN ‘1BI15CS101’ in allsubjects.**

```
CREATE VIEW STU_TEST1_MARKS_VIEW
AS
SELECT TEST1, SUBCODE
FROM IAMARKS
WHERE USN = '1RN13CS091';
```

TEST1	SUBCODE
15	10CS81
12	10CS82
19	10CS83
20	10CS84
15	10CS85

4. Calculate the FinalIA (average of best two test marks) and update the corresponding table for allstudents.

```
CREATE OR REPLACE PROCEDURE AVGMARKS IS
    CURSOR C_IAMARKS IS
SELECT GREATEST(TEST1,TEST2) AS A, GREATEST(TEST1,TEST3) AS B,
GREATEST(TEST3,TEST2) ASC
FROM IAMARKS
WHERE FINALIA IS NULL
FOR UPDATE;

  C_A NUMBER;
  C_B NUMBER;
  C_C NUMBER;
  C_SM NUMBER;
  C_AV NUMBER;

BEGIN
  OPEN C_IAMARKS;
  LOOP
  FETCH C_IAMARKS INTO C_A, C_B, C_C;
    EXIT WHEN C_IAMARKS%NOTFOUND;
    --DBMS_OUTPUT.PUT_LINE(C_A || ' ' || C_B || ' ' || C_C);
    IF (C_A != C_B) THEN
 C_SM:=C_A+C_B;
    ELSE
 C_SM:=C_A+C_C;
    END IF;

    C_AV:=C_SM/2;
    --DBMS_OUTPUT.PUT_LINE('SUM = '||C_SM);
    --DBMS_OUTPUT.PUT_LINE('AVERAGE = '||C_AV);
    UPDATE IAMARKS SET FINALIA=C_AV  WHERE CURRENT OF C_IAMARKS;

  END LOOP;
  CLOSE C_IAMARKS;
END;
/
```

Note: Before execution of PL/SQL procedure, IAMARKS table contents are:

SELECT * FROM IAMARKS;

```
SQL> SELECT * FROM IAMARKS;

USN        SUBCODE  SSID       TEST1      TEST2      TEST3    FINALIA
---------- -------- ----- ---------- ---------- ---------- ----------
1RN13CS091 10CS81   CSE8C          15         16         18
1RN13CS091 10CS82   CSE8C          12         19         14
1RN13CS091 10CS83   CSE8C          19         15         20
1RN13CS091 10CS84   CSE8C          20         16         19
1RN13CS091 10CS85   CSE8C          15         15         12
```

Below SQL code is to invoke the PL/SQL stored procedure from the command line:

```
BEGIN
AVGMARKS;
END;
```

```
SQL> select * from IAMARks;

USN        SUBCODE  SSID       TEST1      TEST2      TEST3    FINALIA
---------- -------- ----- ---------- ---------- ---------- ----------
1RN13CS091 10CS81   CSE8C          15         16         18         17
1RN13CS091 10CS82   CSE8C          12         19         14         17
1RN13CS091 10CS83   CSE8C          19         15         20         20
1RN13CS091 10CS84   CSE8C          20         16         19         20
1RN13CS091 10CS85   CSE8C          15         15         12         15
```

5. Categorize students based on the followingcriterion:
If FinalIA = 17 to 20 then CAT =‘Outstanding’
If FinalIA = 12 to 16 then CAT = ‘Average’
If FinalIA< 12 then CAT = ‘Weak’
Give these details only for 8th semester A, B, and C section students.

```
SELECT S.USN,S.SNAME,S.ADDRESS,S.PHONE,S.GENDER,
    (CASE
      WHEN IA.FINALIA BETWEEN 17 AND 20 THEN'OUTSTANDING'
      WHEN IA.FINALIA BETWEEN 12 AND 16 THEN 'AVERAGE'
      ELSE'WEAK'
    END) AS CAT
FROM STUDENT S, SEMSEC SS, IAMARKS IA, SUBJECT SUB
WHERE S.USN = IA.USN AND
SS.SSID = IA.SSID AND
SUB.SUBCODE = IA.SUBCODE AND
SUB.SEM = 8;
```

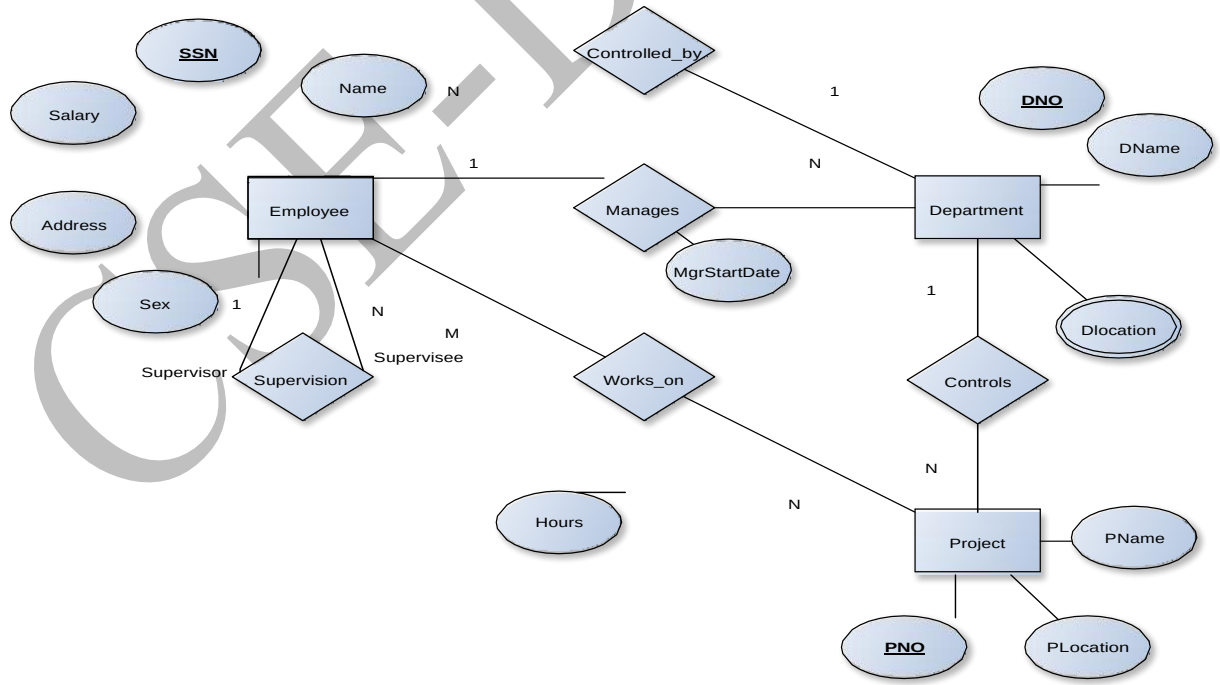
USN	SNAME	ADDRESS	PHONE	G	CAT
1RN13CS091	TEESHA	BENGALURU	7712312312	F	OutStanding
1RN13CS091	TEESHA	BENGALURU	7712312312	F	OutStanding
1RN13CS091	TEESHA	BENGALURU	7712312312	F	OutStanding
1RN13CS091	TEESHA	BENGALURU	7712312312	F	OutStanding
1RN13CS091	TEESHA	BENGALURU	7712312312	F	Average

E. Consider the schema for CompanyDatabase:

EMPLOYEE (*SSN, Name, Address, Sex, Salary, SuperSSN, DNo*)
DEPARTMENT (*DNo, DName, MgrSSN, MgrStartDate*)
DLOCATION (*DNo,DLoc*)
PROJECT (*PNo, PName, PLocation, DNo*)
WORKS_ON (*SSN, PNo, Hours*)
Write SQL queries to

1. **Make a list of all project numbers for projects that involve an employee whose last name is 'Scott', either as a worker or as a manager of the department that controls theproject.**
2. **Show the resulting salaries if every employee working on the 'IoT' project is given a 10 percentraise.**
3. **Find the sum of the salaries of all employees of the 'Accounts' department, as well as the maximum salary, the minimum salary, and the average salary in this department**
4. **Retrieve the name of each employee who works on all the projects controlled by department number 5 (use NOT EXISTS operator). For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than Rs.6,00,000.**

Entity-Relationship Diagram



Schema Diagram

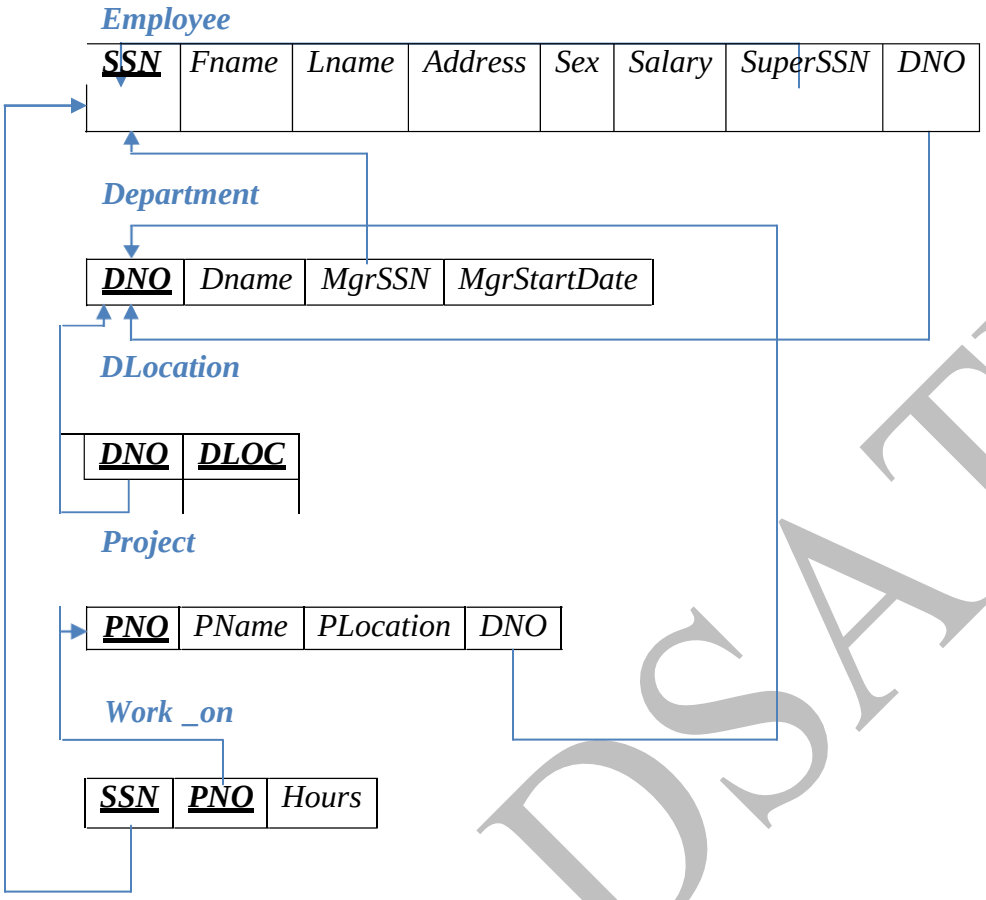


Table Creation

```
CREATE TABLE DEPARTMENT
(DNO VARCHAR2 (20) PRIMARY KEY,
DNAME VARCHAR2 (20),
MGRSTARTDATE DATE);

CREATE TABLE EMPLOYEE
(SSN VARCHAR2 (20) PRIMARY KEY,
FNAME VARCHAR2 (20),
LNAME VARCHAR2 (20),
ADDRESS VARCHAR2 (20),
SEX CHAR (1),
SALARY INTEGER,
SUPERSSN REFERENCES EMPLOYEE (SSN),
DNO REFERENCES DEPARTMENT (DNO));
```

NOTE: Once DEPARTMENT and EMPLOYEE tables are created we must alter department table to add foreign constraint MGRSSN using sql command

ALTER TABLE DEPARTMENT
ADD MGRSSN REFERENCES EMPLOYEE (SSN);

CREATE TABLE DLOCATION
(DLOC VARCHAR2 (20),
DNO REFERENCES DEPARTMENT (DNO),
PRIMARY KEY (DNO, DLOC));

CREATE TABLE PROJECT
(PNO INTEGER PRIMARY KEY,
PNAME VARCHAR2(20),
PLOCATION VARCHAR2 (20),
DNO REFERENCES DEPARTMENT (DNO));

CREATE TABLE WORKS_ON
(HOURS NUMBER (2),
SSN REFERENCES EMPLOYEE (SSN),
PNO REFERENCES PROJECT(PNO),
PRIMARY KEY (SSN, PNO));

Table Descriptions

DESC EMPLOYEE;

```
SQL> DESC EMPLOYEE;
 Name
 -----------------------------------------------------------------
 SSN
 FNAME
 LNAME
 ADDRESS
 SEX
 SALARY
 SUPERSSN
 DNO
```

DESC DEPARTMENT;

```
SQL> DESC DEPARTMENT;
 Name
 -----------------------------------------------------
 DNO
 DNAME
 MGRSTARTDATE
 MGRSSN
```

DESC DLOCATION;

```
SQL> DESC DLOCATION;
 Name
 -------------------------
 DLOC
 DNO
```

DESC PROJECT;

```
SQL> DESC PROJECT;
 Name
 ------------------------
 PNO
 PNAME
 PLOCATION
 DNO
```

DESC WORKS_ON;

```
SQL> DESC WORKS_ON;
 Name
 ----------------------------------------------
 HOURS
 SSN
 PNO
```

Insertion of values to tables

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES (‘RNSECE01’,’JOHN’,’SCOTT’,’BANGALORE’,’M’, 450000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES (‘RNSCSE01’,’JAMES’,’SMITH’,’BANGALORE’,’M’, 500000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES (‘RNSCSE02’,’HEARN’,’BAKER’,’BANGALORE’,’M’, 700000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES (‘RNSCSE03’,’EDWARD’,’SCOTT’,’MYSORE’,’M’, 500000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES (‘RNSCSE04’,’PAVAN’,’HEGDE’,’MANGALORE’,’M’, 650000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES (‘RNSCSE05’,’GIRISH’,’MALYA’,’MYSORE’,’M’, 450000);

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES (‘RNSCSE06’,’NEHA’,’SN’,’BANGALORE’,’F’, 800000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES (‘RNSACC01’,’AHANA’,’K’,’MANGALORE’,’F’, 350000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES (‘RNSACC02’,’SANTHOSH’,’KUMAR’,’MANGALORE’,’M’, 300000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES (‘RNSISE01’,’VEENA’,’M’,’MYSORE’,’M’, 600000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES (‘RNSIT01’,’NAGESH’,’HR’,’BANGALORE’,’M’, 500000);

INSERT INTO DEPARTMENT VALUES (‘1’,’ACCOUNTS’,’01-JAN-01’,’RNSACC02’);
INSERT INTO DEPARTMENT VALUES (‘2’,’IT’,’01-AUG-16’,’RNSIT01’);
INSERT INTO DEPARTMENT VALUES (‘3’,’ECE’,’01-JUN-08’,’RNSECE01’);
INSERT INTO DEPARTMENT VALUES (‘4’,’ISE’,’01-AUG-15’,’RNSISE01’);
INSERT INTO DEPARTMENT VALUES (‘5’,’CSE’,’01-JUN-02’,’RNSCSE05’);

Note: update entries of employee table to fill missing fields SUPERSSN and DNO

UPDATE EMPLOYEE SET
SUPERSSN=NULL, DNO=’3’
WHERE SSN=’RNSECE01’;

UPDATE EMPLOYEE SET
SUPERSSN=’RNSCSE02’, DNO=’5’
WHERE SSN=’RNSCSE01’;

UPDATE EMPLOYEE SET
SUPERSSN=’RNSCSE03’, DNO=’5’
WHERE SSN=’RNSCSE02’;

UPDATE EMPLOYEE SET
SUPERSSN=’RNSCSE04’, DNO=’5’
WHERE SSN=’RNSCSE03’;

UPDATE EMPLOYEE SET
DNO=’5’, SUPERSSN=’RNSCSE05’
WHERE SSN=’RNSCSE04’;

```
UPDATE EMPLOYEE SET
DNO=’5’, SUPERSSN=’RNSCSE06’
WHERE SSN=’RNSCSE05’;

UPDATE EMPLOYEE SET
DNO=’5’, SUPERSSN=NULL
WHERE SSN=’RNSCSE06’;

UPDATE EMPLOYEE SET
DNO=’1’, SUPERSSN=’RNSACC02’
WHERESSN=’RNSACC01’;

UPDATE EMPLOYEE SET
DNO=’1’, SUPERSSN=NULL
WHERE SSN=’RNSACC02’;

UPDATE EMPLOYEE SET
DNO=’4’, SUPERSSN=NULL
WHERE SSN=’RNSISE01’;

UPDATE EMPLOYEE SET
DNO=’2’, SUPERSSN=NULL
WHERE SSN=’RNSIT01’;

INSERT INTO DLOCATION VALUES (’BANGALORE’, ‘1’);
INSERT INTO DLOCATION VALUES (’BANGALORE’, ‘2’);
INSERT INTO DLOCATION VALUES (’BANGALORE’, ‘3’);
INSERT INTO DLOCATION VALUES (’MANGALORE’, ‘4’);
INSERT INTO DLOCATION VALUES (’MANGALORE’, ‘5’);

INSERT INTO PROJECT VALUES (100,’IOT’,’BANGALORE’,’5’);
INSERT INTO PROJECT VALUES (101,’CLOUD’,’BANGALORE’,’5’);
INSERT INTO PROJECT VALUES (102,’BIGDATA’,’BANGALORE’,’5’);
INSERT INTO PROJECT VALUES (103,’SENSORS’,’BANGALORE’,’3’);
INSERT INTO PROJECT VALUES (104,’BANK MANAGEMENT’,’BANGALORE’,’1’);
INSERT INTO PROJECT VALUES (105,’SALARY MANAGEMENT’,’BANGALORE’,’1’);
INSERT INTO PROJECT VALUES (106,’OPENSTACK’,’BANGALORE’,’4’);
INSERT INTO PROJECT VALUES (107,’SMART CITY’,’BANGALORE’,’2’);
```

INSERT INTO WORKS_ON VALUES (4, 'RNSCSE01', 100);
INSERT INTO WORKS_ON VALUES (6, 'RNSCSE01', 101);
INSERT INTO WORKS_ON VALUES (8, 'RNSCSE01', 102);
INSERT INTO WORKS_ON VALUES (10, 'RNSCSE02', 100);
INSERT INTO WORKS_ON VALUES (3, 'RNSCSE04', 100);
INSERT INTO WORKS_ON VALUES (4, 'RNSCSE05', 101);
INSERT INTO WORKS_ON VALUES (5, 'RNSCSE06', 102);
INSERT INTO WORKS_ON VALUES (6, 'RNSCSE03', 102);
INSERT INTO WORKS_ON VALUES (7, 'RNSECE01', 103);
INSERT INTO WORKS_ON VALUES (5, 'RNSACC01', 104);
INSERT INTO WORKS_ON VALUES (6, 'RNSACC02', 105);
INSERT INTO WORKS_ON VALUES (4, 'RNSISE01', 106);
INSERT INTO WORKS_ON VALUES (10, 'RNSIT01', 107);

SELECT * FROM EMPLOYEE;

SSN	FNAME	LNAME	ADDRESS	S	SALARY	SUPERSSN	DNO
RNSECE01	JOHN	SCOTT	BANGALORE	M	450000		3
RNSCSE01	JAMES	SMITH	BANGALORE	M	500000	RNSCSE02	5
RNSCSE02	HEARN	BAKER	BANGALORE	M	700000	RNSCSE03	5
RNSCSE03	EDWARD	SCOTT	MYSORE	M	500000	RNSCSE04	5
RNSCSE04	PAVAN	HEGDE	MANGALORE	M	650000	RNSCSE05	5
RNSCSE05	GIRISH	MALYA	MYSORE	M	450000	RNSCSE06	5
RNSCSE06	NEHA	SN	BANGALORE	F	800000		5
RNSACC01	AHANA	K	MANGALORE	F	350000	RNSACC02	1
RNSACC02	SANTHOSH	KUMAR	MANGALORE	M	300000		1
RNSISE01	VEENA	M	MYSORE	M	600000		4
RNSIT01	NAGESH	HR	BANGALORE	M	500000		2

SELECT * FROM DEPARTMENT;

```
SQL> SELECT * FROM DEPARTMENT;
```

DNO	DNAME	MGRSTARTD	MGRSSN
1	ACCOUNTS	01-JAN-01	RNSACC02
2	IT	01-AUG-16	RNSIT01
3	ECE	01-JUN-08	RNSECE01
4	ISE	01-AUG-15	RNSISE01
5	CSE	01-JUN-02	RNSCSE05

SELECT * FROM DLOCATION;

DLOC	DNO
BANGALORE	1
BANGALORE	2
BANGALORE	3
MANGALORE	4
MANGALORE	5

SELECT * FROM PROJECT;

PNO	PNAME	PLOCATION	DNO
100	IOT	BANGALORE	5
101	CLOUD	BANGALORE	5
102	BIGDATA	BANGALORE	5
103	SENSORS	BANGALORE	3
104	BANK MANAGEMENT	BANGALORE	1
105	SALARY MANAGEMENT	BANGALORE	1
106	OPENSTACK	BANGALORE	4
107	SMART CITY	BANGALORE	2

SELECT * FROM WORKS_ON;

HOURS	SSN	PNO
4	RNSCSE01	100
6	RNSCSE01	101
8	RNSCSE01	102
10	RNSCSE02	100
3	RNSCSE04	100
4	RNSCSE05	101
5	RNSCSE06	102
6	RNSCSE03	102
7	RNSECE01	103
5	RNSACC01	104
6	RNSACC02	105
4	RNSISE01	106
10	RNSIT01	107

Queries:

1. **Make a list of all project numbers for projects that involve an employee whose last name is 'Scott', either as a worker or as a manager of the department that controlsthe project.**

```
(SELECT DISTINCT P.PNO
FROM PROJECT P, DEPARTMENT D, EMPLOYEE E
WHERE E.DNO=D.DNO
AND D.MGRSSN=E.SSN
AND E.LNAME='SCOTT')
UNION
(SELECT DISTINCT P1.PNO
FROM PROJECT P1, WORKS_ON W, EMPLOYEE E1
WHERE P1.PNO=W.PNO
AND E1.SSN=W.SSN
AND E1.LNAME='SCOTT');
```

```
     PNO
--------
      100
      101
      102
      103
      104
      105
      106
      107
```

2. Show the resulting salaries if every employee working on the 'IoT' project is given a10 percentraise.

```
SELECT E.FNAME, E.LNAME, 1.1*E.SALARY AS INCR_SAL
FROM EMPLOYEE E, WORKS_ON W, PROJECT P
WHERE E.SSN=W.SSN
AND W.PNO=P.PNO
AND P.PNAME='IOT';
```

```
FNAME                LNAME                INCR_SAL
-------------------- -------------------- ----------
JAMES                SMITH                   550000
HEARN                BAKER                   770000
PAVAN                HEGDE                   715000
```

3. Find the sum of the salaries of all employees of the 'Accounts' department, as well as the maximum salary, the minimum salary, and the average salary in thisdepartment

```
SELECT SUM (E.SALARY), MAX (E.SALARY), MIN (E.SALARY), AVG
(E.SALARY)
FROM EMPLOYEE E, DEPARTMENT D
WHERE E.DNO=D.DNO
AND D.DNAME='ACCOUNTS';
```

```
SUM(E.SALARY) MAX(E.SALARY) MIN(E.SALARY) AVG(E.SALARY)
------------- ------------- ------------- -------------
       650000        350000        300000        325000
```

4. Retrieve the name of each employee who works on all the projects Controlled by department number 5 (use NOT EXISTSoperator).

```
SELECT E.FNAME, E.LNAME
FROM EMPLOYEE E
WHERE NOT EXISTS((SELECT PNO
                  FROM PROJECT
```

```
WHERE DNO=’5’)
MINUS (SELECT PNO
FROM WORKS_ON
WHERE E.SSN=SSN));
```

FNAME	LNAME
JAMES	SMITH

5. For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than Rs. 6,00,000.

```
SELECT D.DNO, COUNT (*)
FROM DEPARTMENT D, EMPLOYEE E
WHERE D.DNO=E.DNO
AND E.SALARY>600000
AND D.DNO IN (SELECT E1.DNO
        FROM EMPLOYEE E1
        GROUP BY E1.DNO
        HAVING COUNT (*)>5)
GROUP BY D.DNO;
```

DNO	COUNT(*)
5	3

Viva Questions

1. What is SQL?

Structured Query Language

2. What isdatabase?

A database is a logically coherent collection of data with some inherent meaning, representing some aspect of real world and which is designed, built and populated with data for a specificpurpose.

3. What isDBMS?

It is a collection of programs that enables user to create and maintain a database. In other words it is general-purpose software that provides the users with the processes of defining, constructing and manipulating the database for various applications.

4. What is a Databasesystem?

The database and DBMS software together is called as Database system.

5. Advantages ofDBMS?

- Redundancy iscontrolled.
- Unauthorized access isrestricted.
- Providing multiple userinterfaces.
- Enforcing integrityconstraints.
- Providing backup andrecovery.

6. Disadvantage in File ProcessingSystem?

- Data redundancy &inconsistency.
- Difficult in accessingdata.
- Dataisolation.
- Dataintegrity.
- Concurrent access is notpossible.
- SecurityProblems.

7. Describe the three levels of dataabstraction?

There are three levels of abstraction:

- Physical level: The lowest level of abstraction describes how data arestored.
- Logical level*:* The next higher level of abstraction, describes what data are stored in database and what relationship among thosedata.

- View level:The highest level of abstraction describes only part of entiredatabase.

8. Define the "integrityrules"

There are two Integrity rules.

- Entity Integrity:States that “Primary key cannot have NULLvalue”
- Referential Integrity:States that “Foreign Key can be either a NULL value or should be Primary Key value of otherrelation.

9. What is extension andintension?

Extension - It is the number of tuples present in a table at any instance. This is time dependent.

Intension -It is a constant value that gives the name, structure of table and the constraints laid on it.

10. What is DataIndependence?

Data independence means that “the application is independent of the storage structure and access strategy of data”. In other words, The ability to modify the schema definition in one level should not affect the schema definition in the next higher level.

Two types of Data Independence:

- Physical Data Independence: Modification in physical level should not affect the logicallevel.
- Logical Data Independence: Modification in logical level should affect the view level.

NOTE: Logical Data Independence is more difficult to achieve

11. What is a view? How it is related to dataindependence?

A view may be thought of as a virtual table, that is, a table that does not really exist in its own right but is instead derived from one or more underlying base table. In other words, there is no stored file that direct represents the view instead a definition of view is stored in data dictionary.

Growth and restructuring of base tables is not reflected in views. Thus the view can insulate users from the effects of restructuring and growth in the database. Hence accounts for logical data independence.

12. What is DataModel?

A collection of conceptual tools for describing data, data relationships data semantics and constraints.

13. What is E-Rmodel?

This data model is based on real world that consists of basic objects called entities and of relationship among these objects. Entities are described in a database by a set of attributes.

14. What is Object Orientedmodel?

This model is based on collection of objects. An object contains values stored in instance variables within the object. An object also contains bodies of code that operate on the object. These bodies of code are called methods. Objects that contain same types of values and the same methods are grouped together into classes.

15. What is anEntity?

It is an 'object' in the real world with an independent existence.

16. What is an Entitytype?

It is a collection (set) of entities that have same attributes.

17. What is an Entityset?

It is a collection of all entities of particular entity type in the database.

18. What is an Extension of entitytype?

The collections of entities of a particular entity type are grouped together into an entity set.

19. What is anattribute?

It is a particular property, which describes the entity.

20. What is a Relation Schema and aRelation?

A relation Schema denoted by R(A1, A2, …, An) is made up oftherelation name R and the list of attributes A_i that it contains. A relation isdefinedas a set of tuples.Letr be the relation which contains set tuples (t1,t2,t3, ...,tn). Each tuple is an ordered list of n-values t=(v1,v2, ...,vn).

21. What is degree of aRelation?

It is the number of attribute of its relation schema.

22. What isRelationship?

It is an association among two or more entities.

23. What is Relationshipset?

The collection (or set) of similar relationships.

24. What is Relationshiptype?

Relationship type defines a set of associations or a relationship set among a given set of entity types.

25. What is degree of Relationshiptype?

It is the number of entity type participating.

26. What is DDL (Data DefinitionLanguage)?

A data base schema is specified by a set of definitions expressed by a special language called DDL.

27. What is VDL (View DefinitionLanguage)?

It specifies user views and their mappings to the conceptual schema.

28. What is SDL (Storage DefinitionLanguage)?

This language is to specify the internal schema. This language may specify the mapping between two schemas.

29. What is Data Storage - DefinitionLanguage?

The storage structures and access methods used by databasesystemare specifiedbya set of definition in a special type of DDL calleddatastorage- definitionlanguage.

30. What is DML (Data ManipulationLanguage)?

This language that enable user to access or manipulate dataasorganized byappropriate datamodel.

- Procedural DML or Low level: DML requires a user to specify what data are needed and how to get thosedata.
- Non-Procedural DML or High level: DML requires a user to specify what data are needed without specifying how to get thosedata.

31. What is DMLCompiler?

It translates DML statements in a query language intolow-levelinstruction thatthe query evaluation engine canunderstand.

32. What is RelationalAlgebra?

It is a procedural query language. It consists of a set ofoperationsthat take one or two relations as input and produce a newrelation.

33. What is RelationalCalculus?

It is an applied predicate calculus specifically tailored for relational databases proposed by E.F. Codd. E.g. of languages based on it are DSL, ALPHA,QUEL.

34. What isnormalization?

It is a process of analyzing the given relation schemas basedontheir Functional Dependencies (FDs) and primary key to achieve theproperties

- Minimizingredundancy
- Minimizing insertion, deletion and updateanomalies.

35. What is FunctionalDependency?

A Functional dependency is denoted byX Y between ~~two~~sets of attributes X andYthat are subsets of R specifies a constraint on the possible tuple that can form a relation state r of

R. The constraint is for any two tuples t1 and t2 in r if t1[X] = t2[X] then they have t1[Y] = t2[Y]. This means the value of X component of a tuple uniquely determines the value of componentY.

36. When is a functional dependency F said to beminimal?

- Every dependency in F has a single attribute for its right handside.
- We cannot replace any dependencyX A in F ~~with~~ a dependencyY A where Yisa → proper subset of X and still have a set of dependency that is equivalent toF.
- We cannot remove any dependency from F and still have set of dependency that is equivalent toF.

37. What is Multivalueddependency?

Multivalued dependency denoted by X →Y specified on relation schema R, where X and Y are both subsets of R, specifies the following constraint on any relation r of R: if two tuples t1 and t2 exist in r such that t1[X] = t2[X] then t3 and t4 should also exist in r with the followingproperties

- t3[x] = t4[X] = t1[X] = t2[X]
- t3[Y] = t1[Y] and t4[Y] = t2[Y]
- t3[Z] = t2[Z] and t4[Z] = t1[Z]

where [Z = (R-(X U Y))]

38. What is Lossless joinproperty?

It guarantees that the spurious tuple generation does not occur with respect to relation schemas after decomposition.

39. What is 1 NF (NormalForm)?

The domain of attribute must include only atomic (simple, indivisible) values.

40. What is Fully Functionaldependency?

It is based on concept of full functional dependency. A functional dependency X→Y is fully functional dependency if removal of any attribute A from X means that the dependency does not hold anymore.

41. What is2NF?

A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on primary key.

42. What is3NF?

A relation schema R is in 3NF if it is in 2NF and for every FD X→A either of the following istrue

- X is a Super-key ofR.
- A is a prime attribute ofR.

In other words, if every non prime attribute is non-transitively dependent on primary key.

43. What is BCNF (Boyce-Codd NormalForm)?

A relation schema R is in BCNF if it is in 3NF and satisfies additional constraints that for every FD X⟶A, X must be a candidate key.

44. What is4NF?

A relation schema R is said to be in 4NF if for every Multivalued dependency X⟶ Y that holds over R, one of following istrue

- X is subset or equal to (or) XY =R.
- X is a superkey.

45. What is5NF?

A Relation schema R is said to be 5NF if for every join dependency {R1, R2, ...,Rn} that holds R, one the following is true

- Ri = R for somei.
- The join dependency is implied by the set of FD, over R in which the left side is key ofR.

46. What is Domain-Key NormalForm?

A relation is said to be in DKNF if all constraints and dependencies that should hold on the constraint can be enforced by simply enforcing the domain constraint and key constraint on the relation.