

VEMANA INSTITUTE OF TECHNOLOGY

Department of Computer Science &
Engineering

DBMS LABORATORY WITH MINI PROJECT Manual V-Sem CSE 18CSL58

Prepared By:
DEPT OF CSE



DBMS LABORATORY WITH MINI PROJECT [As per Choice Based Credit System (CBCS) scheme] (Effective from the academic year 2018 - 2019) SEMESTER – V			
Subject Code	18CSL58	CIE Marks	40
Number of Contact Hours/Week	0:2:2	Exam Marks	60
Total Number of Lecture Hours	36	Exam Hours	03
CREDITS – 02			
Course objectives: This course will enable students to			
<ul style="list-style-type: none"> 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. 			
Description (If any):			
PART-A: SQL Programming (Max. Exam Mks. 50) <ul style="list-style-type: none"> Design, develop, and implement the specified queries for the following problems using Oracle, MySQL, MS SQL Server, or any other DBMS under LINUX/Windows environment. Create Schema and insert at least 5 records for each table. Add appropriate database constraints. PART-B: Mini Project (Max. Exam Mks. 30) <ul style="list-style-type: none"> Use Java, C#, PHP, Python, or any other similar front-end tool. All applications must be demonstrated on desktop/laptop as a stand-alone or web based application (Mobile apps on Android/IOS are not permitted.) Installation procedure of the required software must be demonstrated, carried out in groups and documented in the journal.			
Lab Experiments:			
1. Consider the following schema for a Library Database: 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 <ol style="list-style-type: none"> Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each branch, etc. Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017. Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulation operation. Partition the BOOK table based on year of publication. Demonstrate its working with a simple query. Create a view of all books and its number of copies that are currently available in the Library. 			



2. Consider the following schema for Order Database:
 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's average.
2. Find the name and numbers of all salesman who had more than one customer.
3. List all the salesman and indicate those who have and don't have customers in their cities (Use UNION operation.)
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 be deleted.
3. Consider the schema for Movie Database:
 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 more movies.
3. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).
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 movie title.
5. Update rating of all movies directed by 'Steven Spielberg' to 5.
4. Consider the schema for College Database:
 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, FinallA)

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 all subjects.
4. Calculate the FinallA (average of best two test marks) and update the corresponding table for all students.
5. Categorize students based on the following criterion:
 If FinallA = 17 to 20 then CAT = 'Outstanding'
 If FinallA = 12 to 16 then CAT = 'Average'
 If FinallA < 12 then CAT = 'Weak'
 Give these details only for 8th semester A, B, and C section students.





5. Consider the schema for Company Database:
 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 the project
2. Show the resulting salaries if every employee working on the 'IoT' project is given a 10 percent raise.
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).
6. 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..

Part B: Mini project

- For any problem selected, write the ER Diagram, apply ER-mapping rules, normalize the relations, and follow the application development process.
- Make sure that the application should have five or more tables, at least one trigger and one stored procedure, using suitable frontend tool.
- Indicative areas include; health care, education, industry, transport, supply chain, etc.

Course outcomes: The students should be able to:

- Create, Update and query on the database.
- Demonstrate the working of different concepts of DBMS
- Implement, analyze and evaluate the project developed for an application.

Conduction of Practical Examination:

- **Experiment distribution**
 - For laboratories having only one part: Students are allowed to pick one experiment from the lot with equal opportunity.
 - For laboratories having PART A and PART B: Students are allowed to pick one experiment from PART A and one experiment from PART B, with equal opportunity.
- Change of experiment is allowed only once and marks allotted for procedure to be made zero of the changed part only.
- Marks Distribution (Coursed to change in accoradance with university regulations)
 - For laboratories having only one part – Procedure + Execution + Viva-Voce: 15+70+15=100 Marks
 - For laboratories having PART A and PART B
 - i. Part A – Procedure + Execution + Viva = 6 + 28 + 6 = 40 Marks
 - ii. Part B – Procedure + Execution + Viva = 9 + 42 + 9 = 60 Marks

1. Consider the following schema for a Library Database:

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)

```
CREATE TABLE PUBLISHER
(NAME VARCHAR (20),
ADDRESS VARCHAR (20),
PHONE NUMBER(10),
PRIMARY KEY(NAME));
```

```
CREATE TABLE BOOK
(BOOK_ID NUMBER(4),
TITLE VARCHAR (20),
PUBLISHER_NAME VARCHAR(20),
PUB_YEAR NUMBER(5),
PRIMARY KEY(BOOK_ID),
FOREIGN KEY(PUBLISHER_NAME) REFERENCES PUBLISHER(NAME) ON DELETE CASCADE;
```

```
CREATE TABLE BOOK_AUTHORS
(BOOK_ID NUMBER(4),
AUTHOR_NAME VARCHAR (20),
PRIMARY KEY (BOOK_ID,AUTHOR_NAME),
FOREIGN KEY(BOOK_ID) REFERENCES BOOK(BOOK_ID) ON DELETE CASCADE);
```

```
CREATE TABLE LIBRARY_BRANCH
(BRANCH_ID NUMBER(5),
BRANCH_NAME VARCHAR (20),
ADDRESS VARCHAR (20),
PRIMARY KEY(BRANCH_ID));
```

```
CREATE TABLE BOOK_COPIES
(BOOK_ID NUMBER(4),
BRANCH_ID NUMBER(5),
NO_OF_COPIES NUMBER(5),
PRIMARY KEY(BOOK_ID, BRANCH_ID),
FOREIGN KEY(BOOK_ID) REFERENCES BOOK(BOOK_ID) ON DELETE CASCADE,
FOREIGN KEY(BRANCH_ID) REFERENCES LIBRARY_BRANCH(BRANCH_ID) ON DELETE CASCADE );
```

```
CREATE TABLE BOOK_LENDING
(BOOK_ID NUMBER(4),
BRANCH_ID NUMBER(5),
CARD_NO NUMBER(10),
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);

INSERT INTO PUBLISHER VALUES ('&NAME', '&ADDRESS', '&PHONE');

Apply forward slash (/), to execute the latest SQL statement.

INSERT INTO PUBLISHER VALUES ('MCGRAW-HILL', 9989076587, 'BANGALORE');
 INSERT INTO PUBLISHER VALUES ('PEARSON', 9889076565, 'NEWDELHI');
 INSERT INTO PUBLISHER VALUES ('RANDOM HOUSE', 7455679345, 'HYDRABAD');
 INSERT INTO PUBLISHER VALUES ('HACHETTE LIVRE', 8970862340, 'CHENAI');
 INSERT INTO PUBLISHER VALUES ('GRUPO PLANETA', 7756120238, 'BANGALORE');

SQL> SELECT * FROM PUBLISHER;

NAME	PHONE	ADDRESS
MCGRAW-HILL	9989076587	BANGALORE
PEARSON	9889076565	NEWDELHI
RANDOM HOUSE	7455679345	HYDRABAD
HACHETTE LIVRE	8970862340	CHENAI
GRUPO PLANETA	7756120238	BANGALORE

INSERT INTO BOOK VALUES ('1', 'DBMS', 'MCGRAW-HILL', 2017);
 INSERT INTO BOOK VALUES ('2', 'ADBMS', 'MCGRAW-HILL', 2016);
 INSERT INTO BOOK VALUES ('3', 'CN', 'PEARSON', 2016);
 INSERT INTO BOOK VALUES ('4', 'CG', 'GRUPO PLANETA', 2015);
 INSERT INTO BOOK VALUES ('5', 'OS', 'PEARSON', 2016);

SQL> SELECT * FROM BOOK;

BOOK_ID	TITLE	PUBLISHER_NAME	PUB_YEAR
1	DBMS	MCGRAW-HILL	2017
2	ADBMS	MCGRAW-HILL	2016
3	CN	PEARSON	2016
4	CG	GRUPO PLANETA	2015
5	OS	PEARSON	2016

INSERT INTO BOOK_AUTHORS VALUES ('1', 'NAVATHE'); INSERT
 INTO BOOK_AUTHORS VALUES ('2', 'NAVATHE'); INSERT INTO
 BOOK_AUTHORS VALUES ('3', 'TANENBAUM'); INSERT INTO
 BOOK_AUTHORS VALUES ('4', 'EDWARD ANGEL'); INSERT INTO
 BOOK_AUTHORS VALUES ('5', 'GALVIN');

SQL> SELECT * FROM BOOK_AUTHORS ;

BOOK_ID	AUTHOR_NAME
1	NAVATHE
2	NAVATHE
3	TANENBAUM
4	EDWARD ANGEL
5	GALVIN


```
INSERT INTO LIBRARY_BRANCH VALUES ('10','VV PURAM','BANGALORE');
INSERT INTO LIBRARY_BRANCH VALUES ('11','BIT','BANGALORE');
INSERT INTO LIBRARY_BRANCH VALUES ('12','RAJAJI NAGAR','BANGALORE');
INSERT INTO LIBRARY_BRANCH VALUES ('13','JP NAGAR','BANGALORE'); INSERT
INTO LIBRARY_BRANCH VALUES ('14','JAYANAGAR','BANGALORE');
```

```
SQL> SELECT * FROM LIBRARY_BRANCH;
```

BRANCH	BRANCH_NAME	ADDRESS
10	VV PURAM	BANGALORE
11	BIT	BANGALORE
12	RAJAJI NAGAR	BANGALORE
13	JP NAGAR	BANGALORE
14	JAYANAGAR	BANGALORE

```
INSERT INTO BOOK_COPIES VALUES ('1','10', 10);
INSERT INTO BOOK_COPIES VALUES ('1','11', 5);
INSERT INTO BOOK_COPIES VALUES ('2','12', 2);
INSERT INTO BOOK_COPIES VALUES ('2','13', 5);
INSERT INTO BOOK_COPIES VALUES ('3','14', 7);
INSERT INTO BOOK_COPIES VALUES ('5','10', 1);
INSERT INTO BOOK_COPIES VALUES ('4','11', 3);
```

```
SQL> SELECT * FROM BOOK_COPIES;
```

BOOK_ID	BRANCH	NO_OF_COPIES
1	10	10
1	11	5
2	12	2
2	13	5
3	14	7
5	10	1
4	11	3

```
INSERT INTO BOOK_LENDING VALUES ('1', '10', 101,'01-JAN-17','01-JUN-17');
INSERT INTO BOOK_LENDING VALUES ('3', '14', 101,'11-JAN-17','11-MAR-17');
INSERT INTO BOOK_LENDING VALUES ('2', '13', 101,'21-FEB-17','21-APR-17');
INSERT INTO BOOK_LENDING VALUES ('4', '11', 101,'15-MAR-17','15-JUL-17');
INSERT INTO BOOK_LENDING VALUES ('1', '11', 104,'12-APR-17','12-MAY-17')
```

```
SQL> SELECT * FROM BOOK_LENDING;
```

BOOK_ID	BRANCH	CARD_NO	DATE_OUT	DUE_DATE
1	10	101	01-JAN-17	01-JUN-17
3	14	101	11-JAN-17	11-MAR-17
2	13	101	21-FEB-17	21-APR-17
4	11	101	15-MAR-17	15-JUL-17
1	11	104	12-APR-17	12-MAY-17

Queries:

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

```
SELECT      B.BOOK_ID, B.TITLE, B.PUBLISHER_NAME, A.AUTHOR_NAME,
            C.NO_OF_COPIES, L.BRANCH_ID
FROM        BOOK B, BOOK_AUTHORS A, BOOK_COPIES C, LIBRARY_BRANCH L
WHERE       A.BOOK_ID=B.BOOK_ID AND B.BOOK_ID=C.BOOK_ID
            AND C.BRANCH_ID = L.BRANCH_ID;
```

OUTPUT:

BOOK_ID	TITLE	PUBLISHER_NAME	AUTHOR_NAME	NO_OF_COPIES	BRANCH
1	DBMS	MCGRRAW-HILL	NAVATHE	10	10
1	DBMS	MCGRRAW-HILL	NAVATHE	5	11
2	ADBMS	MCGRRAW-HILL	NAVATHE	2	12
2	ADBMS	MCGRRAW-HILL	NAVATHE	5	13
3	CN	PEARSON	TANENBAUM	7	14
5	OS	PEARSON	GALVIN	1	10
4	CG	GRUPO PLANETA	EDWARD ANGEL	3	11

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 '01-JUL-2017'
GROUP BY   CARD_NO
HAVING      COUNT (*)>3;
```

OUTPUT:

CARD_NO
101

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

Before Deleting:

SQL> select * from book;

BOOK_ID	TITLE	PUBLISHER_NAME	PUB_YEAR
1	DBMS	MCGRRAW-HILL	01-JAN-17
2	ADBMS	MCGRRAW-HILL	10-JUN-16
3	CN	PEARSON	16-SEP-16
4	CG	GRUPO PLANETA	11-SEP-15
5	OS	PEARSON	23-MAY-16

SQL> select * from book_copies;

Department of Computer Science & Engg.VIT



BOOK_ID	BRANCH	NO_OF_COPIES
1	10	10
1	11	5
2	12	2
2	13	5
3	14	7
5	10	1
4	11	3

DELETE FROM BOOK
WHERE BOOK_ID='3';

SQL> select * from book;

BOOK_ID	TITLE	PUBLISHER_NAME	PUB_YEAR
1	DBMS	MCGRAW-HILL	01-JAN-17
2	ADBMS	MCGRAW-HILL	10-JUN-16
4	CG	GRUPO PLANETA	11-SEP-15
5	OS	PEARSON	23-MAY-16

SQL> select * from book_copies;

BOOK_ID	BRANCH	NO_OF_COPIES
1	10	10
1	11	5
2	12	2
2	13	5
5	10	1
4	11	3

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

CREATE TABLE BOOKPART
PARTITION BY RANGE (PUB_YEAR)
(PARTITION P1 VALUES LESS THAN(2016),
PARTITION P2 VALUES LESS THAN (MAXVALUE))
AS SELECT * FROM BOOK;

CREATE VIEW PUBYEAR AS SELECT PUB_YEAR
FROM BOOK;

O/P: SELECT * FROM PUB_YEAR;

OUTPUT:

SQL> select table_name,partition_name from
user_tab_partitions; TABLE_NAME
PARTITION_NAME

BOOKPART	P2
BOOKPART	P1

SQL> SELECT * FROM BOOKPART PARTITION (P1);

BOOK_ID	TITLE	PUBLISHER_NAME	PUB_YEAR
4	CG	GRUPO PLANETA	2015



```
SQL> SELECT * FROM BOOKPART PARTITION (P2);
```

BOOK_ID	TITLE	PUBLISHER_NAME	PUB_YEAR
1	DBMS	MCGRRAW-HILL	2017
2	ADBMS	MCGRRAW-HILL	2016
5	OS	PEARSON	2016

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

```
CREATE VIEW BOOKCOP AS
  SELECT B.BOOK_ID, C.TITLE, B.BRANCH_ID,
         (B.NO_OF_COPIES - (SELECT COUNT(*)
                           FROM BOOK_LENDING
                           WHERE B.BOOK_ID=BOOK_ID AND
                                B.BRANCH_ID=BRANCH_ID)) AS NO_COPY
  FROM   BOOK_COPIES B, BOOK C
 WHERE  B.BOOK_ID = C.BOOK_ID;
```

OUTPUT:

```
SQL> SELECT * FROM BOOKCOP;
```

BOOK_ID	TITLE	BRANCH	NO_COPY
1	DBMS	10	9
1	DBMS	11	5
2	ADBMS	12	2
2	ADBMS	13	4
5	OS	10	1
4	CG	11	2

2. Consider the following schema for Order Database:

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)

```
CREATE TABLE SALESMAN(SALESMAN_ID NUMBER(5),
    NAME    VARCHAR(20),
    CITY    VARCHAR(20),
    COMMISSION VARCHAR(10),
    PRIMARY KEY (SALESMAN_ID));
```

```
CREATE TABLE CUSTOMER(CUSTOMER_ID VARCHAR(4),
    CUST_NAME VARCHAR (20),
    CITY VARCHAR (20),  GRADE
    NUMBER (3), SALESMAN_ID
    NUMBER(5),
    PRIMARY KEY(CUSTOMER_ID),
    FOREIGN KEY(SALESMAN_ID) REFERENCES SALESMAN(SALESMAN_ID) ON
    DELETE SET NULL);
```

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

```
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 CUSTOMER VALUES ('C1', 'PREETHI','BANGALORE', 100, '1000');
INSERT INTO CUSTOMER VALUES ('C2', 'VIVEK','MANGALORE', 300, '1000'); INSERT
INTO CUSTOMER VALUES ('C3', 'BHASKAR','CHENNAI', 400, '2000'); INSERT INTO
CUSTOMER VALUES ('C4', 'CHETHAN','BANGALORE', 200, '2000'); INSERT INTO
CUSTOMER VALUES ('C5', 'MAMATHA','BANGALORE', 400, '3000');
```

```
INSERT INTO ORDERS VALUES ('O1', 5000, '04-MAY-17', 'C1', '1000');
INSERT INTO ORDERS VALUES ('O2', 6000, '04-MAY-17', 'C1', '1000');
INSERT INTO ORDERS VALUES ('O3', 7000, '04-MAY-17', 'C2', '1000');
```

```

INSERT INTO ORDERS VALUES ('04', 450, '20-JAN-17', 'C1', '2000');
INSERT INTO ORDERS VALUES ('05', 1000, '24-FEB-17', 'C2', '2000');
INSERT INTO ORDERS VALUES ('06', 3500, '13-APR-17', 'C3', '3000');
INSERT INTO ORDERS VALUES ('07', 550, '09-MAR-17', 'C4', '2000');
INSERT INTO ORDERS VALUES ('08', 6500, '04-MAY-17', 'C5', '1000');
INSERT INTO ORDERS VALUES ('09', 7500, '09-MAR-17', 'C2', '2000');

```

```
SELECT * FROM SALESMAN;
```

SALESMAN	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 CUSTOMER;
```

CUSTOMER	CUST_NAME	CITY	GRADE	SALESMAN
C1	PREETHI	BANGALORE	100	1000
C2	VIVEK	MANGALORE	300	1000
C3	BHASKAR	CHENNAI	400	2000
C4	CHETHAN	BANGALORE	200	2000
C5	MAMATHA	BANGALORE	400	3000

```
SELECT * FROM ORDERS;
```

ORD_NO	PURCHASE_AMT	ORD_DATE	CUSTOMER	SALESMAN
01	5000	04-MAY-17	C1	1000
02	6000	04-MAY-17	C1	1000
03	7000	04-MAY-17	C2	1000
04	450	20-JAN-17	C1	2000
05	1000	24-FEB-17	C2	2000
06	3500	13-APR-17	C3	3000
07	550	09-MAR-17	C4	2000
08	6500	04-MAY-17	C5	1000
09	7500	09-MAR-17	C2	2000

Queries:

1. Count the customers with grades above Bangalore's average.

```

SELECT      GRADE, COUNT (DISTINCT CUSTOMER_ID) AS NO_OF_CUSTOMERS
FROM        CUSTOMER
GROUP BY    GRADE
HAVING      GRADE > (  SELECT AVG(GRADE)
                       FROM CUSTOMER
                       WHERE CITY='BANGALORE');

```

OUTPUT:

GRADE	NO_OF_CUSTOMER
400	2
300	1

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

```

SELECT      SALESMAN_ID, NAME
FROM        SALESMAN S
WHERE (      (SELECT COUNT (*)
              FROM CUSTOMER
              WHERE SALESMAN_ID=S.SALESMAN_ID)
        > 1);

```

OUTPUT:

SALESMAN	NAME
1000	JOHN
2000	RAVI

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

```

SELECT      S.SALESMAN_ID, S.CITY
FROM        SALESMAN S
WHERE       EXISTS (SELECT CITY FROM CUSTOMER WHERE S.CITY=CITY AND
                    S.SALESMAN_ID=SALESMAN_ID)

UNION
SELECT      SALESMAN_ID,'NO MATCH OF CITIES'
FROM        SALESMAN S
WHERE       NOT EXISTS (SELECT CITY FROM CUSTOMER WHERE S.CITY=CITY AND
                        S.SALESMAN_ID=SALESMAN_ID);

```

OUTPUT:

SALESMAN	CITY
1000	BANGALORE
2000	BANGALORE
3000	NO MATCH OF CITIES
4000	NO MATCH OF CITIES
5000	NO MATCH OF CITIES

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

```
CREATE VIEW LISTSALSMAN AS
SELECT O.ORD_DATE, S.SALESMAN_ID, S.NAME
FROM SALESMAN S, ORDERS O
WHERE S.SALESMAN_ID=O. S.SALESMAN_ID AND
O.PURCHASE_AMT=(SELECT MAX (PURCHASE_AMT)
FROM ORDERS C WHERE C. ORD_DATE=O. ORD_DATE);
```

OUTPUT:

SALESMAN	ORD_DATE
1000	04-MAY-17
3000	13-APR-17
2000	20-JAN-17
2000	24-FEB-17
2000	09-MAR-17

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

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> SELECT * FROM SALESMAN;
```

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

```
SQL> SELECT * FROM CUSTOMER;
```

CUSTOMER	CUST_NAME	CITY	GRADE	SALESMAN
C1	PREETHI	BANGALORE	100	
C2	VIVEK	MANGALORE	300	
C3	BHASKAR	CHENNAI	400	2000
C4	CHETHAN	BANGALORE	200	2000
C5	MAMATHA	BANGALORE	400	3000

```
SQL> SELECT * FROM ORDERS;
ORD_NO PURCHASE_AMT ORD_DATE CUSTOMER SALESMAN
```

02	450	20-JAN-17	C1	200
03	1000	24-FEB-17	C2	200
04	3500	13-APR-17	C3	300



3. Consider the schema for Movie Database:

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)

```
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 (20),
MOV_YEAR NUMBER (4),
MOV_LANG VARCHAR (12),
DIR_ID NUMBER (3),
PRIMARY KEY(MOV_ID),
FOREIGN KEY(DIR_ID) REFERENCES DIRECTOR(DIR_ID)ON DELETE CASCADE);
```

```
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) ON DELETE CASCADE,
FOREIGN KEY(MOV_ID) REFERENCES MOVIES(MOV_ID) ON DELETE CASCADE);
```

```
CREATE TABLE RATING
( MOV_ID NUMBER (4),
REV_STARS NUMBER(2),
FOREIGN KEY(MOV_ID) REFERENCES MOVIES(MOV_ID) ON DELETE CASCADE);
```

```
INSERT INTO ACTOR VALUES (301,'ANUSHKA','F');
INSERT INTO ACTOR VALUES (302,'PRABHAS','M');
INSERT INTO ACTOR VALUES (303,'ARAVIND','M');
INSERT INTO ACTOR VALUES (304,'JERMY','M');
INSERT INTO ACTOR VALUES (305,'KIM NEWMEN','M');
```

```
SQL> SELECT * FROM ACTOR;
```

ACT_ID	ACT_NAME	A
301	ANUSHKA	F
302	PRABHAS	M
303	ARAVIND	M
304	JERMY	M
305	KIM NEWMEN	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 DIRECTOR VALUES (64,'MAHESH', 8989776539);

```

```
SQL> SELECT * FROM DIRECTOR;
```

DIR_ID	DIR_NAME	DIR_PHONE
60	RAJAMOULI	875161100
61	HITCHCOCK	776613891
62	FARAN	998677653
63	STEVEN SPIELBERG	898977653
64	MAHESH	898977653

```

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,'PSYCHO', 2008, 'ENGLISH', 61);
INSERT INTO MOVIES VALUES (1004,'WAR HORSE', 2011, 'ENGLISH', 63);
INSERT INTO MOVIES VALUES (1005,'LAST BUS', 2016, 'KANNADA', 64);
INSERT INTO MOVIES VALUES (1006,'THE BIRDS', 2011, 'ENGLISH', 61);
INSERT INTO MOVIES VALUES (1007,'TITANIC', 2012, 'ENGLISH', 63);

```

```
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	PSYCHO	2008	ENGLISH	61
1004	WAR HORSE	2011	ENGLISH	63
1005	LAST BUS	2016	KANNADA	64
1006	THE BIRDS	2011	ENGLISH	61
1007	TITANIC	2012	ENGLISH	63

```

INSERT INTO MOVIE_CAST VALUES (301, 1002, 'HEROINE');
INSERT INTO MOVIE_CAST VALUES (301, 1001, 'HEROINE');
INSERT INTO MOVIE_CAST VALUES (303, 1005, 'HERO');
INSERT INTO MOVIE_CAST VALUES (302, 1002, 'HERO');
INSERT INTO MOVIE_CAST VALUES (302, 1001, 'HERO');
INSERT INTO MOVIE_CAST VALUES (304, 1004, 'HERO');
INSERT INTO MOVIE_CAST VALUES (305, 1005, 'HERO');
INSERT INTO MOVIE_CAST VALUES (305, 1007, 'HERO');

```

```
SQL> SELECT * FROM MOVIE_CAST;
```

ACT_ID	MOV_ID	ROLE
301	1002	HEROINE

Department of Computer Science & Engg.VIT

```

301      1001    HEROINE
303      1005    HERO
302      1002    HERO
302      1001    HERO
304      1004    HERO
305      1005    HERO
305      1007    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);
INSERT INTO RATING VALUES (1005, 3);
INSERT INTO RATING VALUES (1006, 8);
INSERT INTO RATING VALUES (1007, 0);
INSERT INTO RATING VALUES (1001, 2);
INSERT INTO RATING VALUES (1002, 5);

```

```
SQL> SELECT * FROM RATING;
```

MOV_ID	REV_STARS
1001	4
1001	2
1002	2
1002	5
1003	5
1004	4
1005	3
1006	8
1007	0

Queries:

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

```

SELECT      M.MOV_TITLE
FROM        MOVIES M,DIRECTOR D
WHERE       M.DIR_ID=D.DIR_ID AND D.DIR_NAME = 'HITCHCOCK';

```

OUTPUT:

MOV_TITLE
PSYCHO
THE BIRDS

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

```

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;

```

OUTPUT:

MOV_TITLE

BAHUBALI-1
BAHUBALI-2
LAST BUS

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

```
SELECT    A.ACT_NAME
FROM      ACTOR A
JOIN      MOVIE_CAST C 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;
```

OUTPUT:

ACT_NAME

ANUSHKA
PRABHAS
ARAVIND
KIM NEWMEN

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 movie title.

```
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;
```

OUTPUT:

MOV_TITLE	MAX(REV_STARS)
-----	-----
BAHUBALI-1	5
BAHUBALI-2	4
LAST BUS	3
PSYCHO	5
THE BIRDS	8
WAR HORSE	4

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

```

UPDATE    RATING
SET       REV_STARS=5
WHERE     MOV_ID IN ( SELECT M.MOV_ID
                       FROM   MOVIES M, DIRECTOR D
                       WHERE   M.DIR_ID=D.DIR_ID AND D.DIR_NAME
                              = 'STEVEN SPIELBERG'
                       );

```

BEFORE UPDATING

SQL> SELECT * FROM RATING;

MOV_ID	REV_STARS
1001	4
1002	2
1003	5
1004	4 <-----
1005	3
1006	8
1007	0 <-----
1001	2
1002	5

AFTER UPDATING

SQL> SELECT * FROM RATING;

MOV_ID	REV_STARS
1001	4
1002	2
1003	5
1004	5 <-----
1005	3
1006	8
1007	5 <-----
1001	2
1002	5

4. Consider the schema for College

Database: 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)

```
CREATE TABLE STUDENT (USN VARCHAR (10), SNAME
                        VARCHAR (20), ADDRESS
                        VARCHAR (20), PHONE
                        NUMBER (10), GENDER
                        CHAR (1),
                        PRIMARY KEY(USN));
```

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

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

```
CREATE TABLE SUBJECT (SUBCODE VARCHAR (8),
                       TITLE VARCHAR (20),
                       SEM NUMBER (2),
                       CREDITS NUMBER (5),
                       PRIMARY KEY (SUBCODE));
```

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

```
INSERT INTO STUDENT VALUES ('1VI16CS001','ABHILASH','BELAGAVI',8877881122,'M');
INSERT INTO STUDENT VALUES ('1VI16CS011','AMOGH','BENGALURU',7722829912,'M');
INSERT INTO STUDENT VALUES ('1VI16CS113','ANANYA','BENGALURU',7712312312,'F');
INSERT INTO STUDENT VALUES ('1VI16CS049','HARSHA','MANGALURU',8877881122,'M');
INSERT INTO STUDENT VALUES ('1VI16CS065','KRUTHI','BENGALURU',9900211201,'F');
```

```

INSERT INTO STUDENT VALUES ('1VI16CS071','MEGHA','BENGALURU',9923211099,'F');
INSERT INTO STUDENT VALUES ('1VI16CS091','MANJU','BENGALURU',7894737377,'M');
INSERT INTO STUDENT VALUES ('1VI16CS009','KIRAN','BENGALURU',7894737377,'M');
INSERT INTO STUDENT VALUES ('1VI16CS021','NAYANA','BENGALURU',7894737377,'F');
INSERT INTO STUDENT VALUES ('1VI16CS093','KUMAR','BENGALURU',7894737377,'M');
INSERT INTO STUDENT VALUES ('1VI16CS100','SWETHA','BENGALURU',7894737377,'F');

```

```

INSERT INTO STUDENT VALUES ('1VI15CS027','ANVITHA','TUMKUR',9845091341,'F');
INSERT INTO STUDENT VALUES ('1VI15CS012','AJAY','DAVANGERE',7696772121,'M');
INSERT INTO STUDENT VALUES ('1VI15CS015','ANVITHA','BELLARY',9944850121,'F');
INSERT INTO STUDENT VALUES ('1VI15CS101','NEMISA
SINHA','MANGALURU',8812332201,'M');
INSERT INTO STUDENT VALUES ('1VI15CS200','PAVAN','KALBURGI',9900232201,'M');
INSERT INTO STUDENT VALUES ('1VI15CS191','SIRI','SHIMOGA',9905542212,'F');

```

```

INSERT INTO STUDENT VALUES ('1VI14CS007','ADITYA','SHIMOGA',9905542212,'M');
INSERT INTO STUDENT VALUES ('1VI14CS018','AMOGH','MYSORE',9905541112,'M');
INSERT INTO STUDENT VALUES ('1VI14CS020','AMULYA','SHIMOGA',8812332201,'F');
INSERT INTO STUDENT VALUES ('1VI14CS051','KEERTHI','SHIMOGA',9905542212,'M');
INSERT INTO STUDENT VALUES ('1VI14CS078','MANJULA','SHIMOGA',9905541234,'F');
INSERT INTO STUDENT VALUES ('1VI14CS112','POOJA','SHIMOGA',9985541112,'F');
INSERT INTO STUDENT VALUES ('1VI14CS114','PRADEEP','SHIMOGA',9901232212,'M');

```

```

INSERT INTO STUDENT VALUES ('1VI14CS066','PRAKASH','SHIMOGA',9901232212,'M');
INSERT INTO STUDENT VALUES ('1VI14CS132','PRIYA','MYSORE',9901232212,'F');
INSERT INTO STUDENT VALUES ('1VI14CS161','SIRI','TUMKUR',9901232212,'F');

```

```
SQL> SELECT * FROM STUDENT;
```

USN	SNAME	ADDRESS	PHONE	G
1VI16CS001	ABHILASH	BELAGAVI	8877881122	M
1VI16CS011	AMOGH	BENGALURU	7722829912	M
1VI16CS113	ANANYA	BENGALURU	7712312312	F
1VI16CS049	HARSHA	MANGALURU	8877881122	M
1VI16CS065	KRUTHI	BENGALURU	9900211201	F
1VI16CS071	MEGHA	BENGALURU	9923211099	F
1VI16CS091	MANJU	BENGALURU	7894737377	M
1VI16CS009	KIRAN	BENGALURU	7894737377	M
1VI16CS021	NAYANA	BENGALURU	7894737377	F
1VI16CS093	KUMAR	BENGALURU	7894737377	M
1VI16CS100	SWETHA	BENGALURU	7894737377	F
1VI15CS027	ANVITHA	TUMKUR	9845091341	F
1VI15CS012	AJAY	DAVANGERE	7696772121	M
1VI15CS015	ANVITHA	BELLARY	9944850121	F
1VI15CS101	NEMISA SINHA	MANGALURU	8812332201	M
1VI15CS200	PAVAN	KALBURGI	9900232201	M
1VI15CS191	SIRI	SHIMOGA	9905542212	F
1VI14CS007	ADITYA	SHIMOGA	9905542212	M
1VI14CS018	AMOGH	MYSORE	9905541112	M

1VI14CS020	AMULYA	SHIMOGA	8812332201	F
1VI14CS051	KEERTHI	SHIMOGA	9905542212	M
1VI14CS078	MANJULA	SHIMOGA	9905541234	F
1VI14CS112	POOJA	SHIMOGA	9985541112	F
1VI14CS114	PRADEEP	SHIMOGA	9901232212	M
1VI14CS066	PRAKASH	SHIMOGA	9901232212	M
1VI14CS132	PRIYA	MYSORE	9901232212	F
1VI14CS161	SIRI	TUMKUR	9901232212	F

```
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 ('CSE6A', 6,'A');
INSERT INTO SEMSEC VALUES ('CSE6B', 6,'B');
```

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

```
SQL> SELECT * FROM SEMSEC;
```

SSID	SEM	S
CSE4A	4	A
CSE4B	4	B
CSE4C	4	C
CSE6A	6	A
CSE6B	6	B
CSE8A	8	A
CSE8B	8	B
CSE8C	8	C

```
INSERT INTO CLASS VALUES ('1VI16CS001','CSE4A');
INSERT INTO CLASS VALUES ('1VI16CS011','CSE4A');
INSERT INTO CLASS VALUES ('1VI16CS113','CSE4A');
INSERT INTO CLASS VALUES ('1VI16CS049','CSE4B');
INSERT INTO CLASS VALUES ('1VI16CS065','CSE4B');
INSERT INTO CLASS VALUES ('1VI16CS071','CSE4B');
INSERT INTO CLASS VALUES ('1VI16CS091','CSE4B');
INSERT INTO CLASS VALUES ('1VI16CS009','CSE4C');
INSERT INTO CLASS VALUES ('1VI16CS021','CSE4C');
INSERT INTO CLASS VALUES ('1VI16CS093','CSE4C');
INSERT INTO CLASS VALUES ('1VI16CS100','CSE4C');
INSERT INTO CLASS VALUES ('1VI15CS027','CSE6A');
INSERT INTO CLASS VALUES ('1VI15CS012','CSE6A');
INSERT INTO CLASS VALUES ('1VI15CS015','CSE6A');
INSERT INTO CLASS VALUES ('1VI15CS101','CSE6B');
INSERT INTO CLASS VALUES ('1VI15CS200','CSE6B');
INSERT INTO CLASS VALUES ('1VI15CS191','CSE6B');
INSERT INTO CLASS VALUES ('1VI14CS007','CSE8A');
INSERT INTO CLASS VALUES ('1VI14CS018','CSE8A');
INSERT INTO CLASS VALUES ('1VI14CS020','CSE8A');
```

```

INSERT INTO CLASS VALUES ('1VI14CS051','CSE8A');
INSERT INTO CLASS VALUES ('1VI14CS078','CSE8B');
INSERT INTO CLASS VALUES ('1VI14CS112','CSE8B');
INSERT INTO CLASS VALUES ('1VI14CS114','CSE8B');
INSERT INTO CLASS VALUES ('1VI14CS066','CSE8C');
INSERT INTO CLASS VALUES ('1VI14CS132','CSE8C');
INSERT INTO CLASS VALUES ('1VI14CS161','CSE8C');

```

```
SQL> SELECT * FROM CLASS;
```

USN	SSID
1VI14CS007	CSE8A
1VI14CS018	CSE8A
1VI14CS020	CSE8A
1VI14CS051	CSE8A
1VI14CS066	CSE8C
1VI14CS078	CSE8B
1VI14CS112	CSE8B
1VI14CS114	CSE8B
1VI14CS132	CSE8C
1VI14CS161	CSE8C
1VI15CS012	CSE6A
1VI15CS015	CSE6A
1VI15CS027	CSE6A
1VI15CS101	CSE6B
1VI15CS191	CSE6B
1VI15CS200	CSE6B
1VI16CS001	CSE4A
1VI16CS009	CSE4C
1VI16CS011	CSE4A
1VI16CS021	CSE4C
1VI16CS049	CSE4B
1VI16CS065	CSE4B
1VI16CS071	CSE4B
1VI16CS091	CSE4B
1VI16CS093	CSE4C
1VI16CS100	CSE4C
1VI16CS113	CSE4A

```

INSERT INTO SUBJECT VALUES ('10CS81','SA', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS82','SMAD', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS83','WNMC', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS84','WEB', 8, 4);

```

```

INSERT INTO SUBJECT VALUES ('10CS61','ME', 6, 4);
INSERT INTO SUBJECT VALUES ('10CS62','USP', 6, 4);
INSERT INTO SUBJECT VALUES ('10CS63','SD', 6, 4);
INSERT INTO SUBJECT VALUES ('10CS64','CNII', 6, 4);
INSERT INTO SUBJECT VALUES ('10CS65','CG', 6, 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);
```

```
SQL> SELECT * FROM SUBJECT;
```

SUBCODE	TITLE	SEM	CREDITS
10CS81	SA	8	4
10CS82	SMAD	8	4
10CS83	WNMC	8	4
10CS84	WEB	8	4
10CS61	ME	6	4
10CS62	USP	6	4
10CS63	SD	6	4
10CS64	CNII	6	4
10CS65	CG	6	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

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI15CS101','10CS61','CSE6B', 20, 23, 20);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI15CS101','10CS62','CSE6B', 18, 19, 19);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI15CS101','10CS63','CSE6B', 19, 20, 20);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI15CS101','10CS64','CSE6B', 20, 20, 19);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI15CS101','10CS65','CSE6B', 18, 20, 19);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI14CS007','10CS81','CSE8A', 15, 10, 12);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI14CS007','10CS82','CSE8A', 15, 20, 12);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI14CS007','10CS83','CSE8A', 5, 10, 5);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI14CS007','10CS84','CSE8A', 15, 20, 12);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI14CS078','10CS81','CSE8B', 15, 20, 12);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI14CS078','10CS82','CSE8B', 15, 20, 12);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI14CS078','10CS83','CSE8B', 10, 8, 10);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI14CS078','10CS84','CSE8B', 15, 20, 12);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI14CS066','10CS81','CSE8C', 15, 20, 12);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1VI14CS066','10CS82','CSE8C', 12, 13, 14);
```



```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1VI14CS066','10CS83','CSE8C', 15, 20, 12);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1VI14CS066','10CS84','CSE8C', 15, 20, 12);
```

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

USN	SUBCODE	SSID	TEST1	TEST2	TEST3	FINALIA
1VI15CS10	10CS61	CSE6B	20	23	20	
1VI15CS10	10CS62	CSE6B	18	19	19	
1VI15CS10	10CS63	CSE6B	19	20	20	
1VI15CS10	10CS64	CSE6B	20	20	19	
1VI15CS10	10CS65	CSE6B	18	20	19	
1VI14CS00	10CS81	CSE8A	15	10	12	
1VI14CS00	10CS82	CSE8A	15	20	12	
1VI14CS00	10CS83	CSE8A	5	10	5	
1VI14CS00	10CS84	CSE8A	15	20	12	
1VI14CS07	10CS81	CSE8B	15	20	12	
1VI14CS07	10CS82	CSE8B	15	20	12	
1VI14CS07	10CS83	CSE8B	10	8	10	
1VI14CS07	10CS84	CSE8B	15	20	12	
1VI14CS06	10CS81	CSE8C	15	20	12	
1VI14CS06	10CS82	CSE8C	12	13	14	
1VI14CS06	10CS83	CSE8C	15	20	12	
1VI14CS06	10CS84	CSE8C	15	20	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';
```

OUTPUT:

USN	SNAME	ADDRESS	PHONE	G	SEM	S
1VI16CS009	KIRAN	BENGALURU	789473737	M	4	C
1VI16CS021	NAYANA	BENGALURU	789473737	F	4	C
1VI16CS093	KUMAR	BENGALURU	789473737	M	4	C
1VI16CS100	SWETHA	BENGALURU	789473737	F	4	C

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

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

OUTPUT:

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

3.Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.

```
CREATE VIEW STU_TEST1_MARKS_VIEW
AS
SELECT    TEST1, SUBCODE
FROM      IAMARKS
WHERE     USN = '1BI15CS101';
```

OUTPUT:

```
SQL> SELECT * FROM STU_TEST1_MARKS_VIEW;
```

TEST1	SUBCODE
-----	-----
20	10CS61
12	10CS62
19	10CS63
20	10CS64
15	10CS65

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

```
UPDATE IAMARKS SET FINALIA=((TEST1+TEST2+TEST3)-
LEAST(TEST1,TEST2,TEST3))/2;
```

OUTPUT:

SQL> SELECT * FROM IAMARKS;

USN	SUBCODE	SSID	TEST1	TEST2	TEST3	FINALIA
1VI15CS101	10CS61	CSE6B	20	23	20	21.5
1VI15CS101	10CS62	CSE6B	18	19	19	19
1VI15CS101	10CS63	CSE6B	19	20	20	20
1VI15CS101	10CS64	CSE6B	20	20	19	20
1VI15CS101	10CS65	CSE6B	18	20	19	19.5
1VI14CS007	10CS81	CSE8A	15	10	12	13.5
1VI14CS007	10CS82	CSE8A	15	20	12	17.5
1VI14CS007	10CS83	CSE8A	5	10	5	7.5
1VI14CS007	10CS84	CSE8A	15	20	12	17.5
1VI14CS078	10CS81	CSE8B	15	20	12	17.5
1VI14CS078	10CS82	CSE8B	15	20	12	17.5
1VI14CS078	10CS83	CSE8B	10	8	10	10
1VI14CS078	10CS84	CSE8B	15	20	12	17.5
1VI14CS066	10CS81	CSE8C	15	20	12	17.5
1VI14CS066	10CS82	CSE8C	12	13	14	13.5
1VI14CS066	10CS83	CSE8C	15	20	12	17.5
1VI14CS066	10CS84	CSE8C	15	20	12	17.5

5. Categorize students based on the following criterion:

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,IA.SUBCODE,
(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;

```

OUTPUT:

USN	SNAME	ADDRESS	PHONE	G	SUBCODE	CAT
1VI14CS007	ADITYA	SHIMOGA	990554221	M	10CS84	OUTSTANDING
1VI14CS007	ADITYA	SHIMOGA	990554221	M	10CS83	WEAK
1VI14CS007	ADITYA	SHIMOGA	990554221	M	10CS82	OUTSTANDING
1VI14CS007	ADITYA	SHIMOGA	990554221	M	10CS81	AVERAGE
1VI14CS078	MANJULA	SHIMOGA	990554123	F	10CS84	OUTSTANDING
1VI14CS078	MANJULA	SHIMOGA	990554123	F	10CS83	WEAK
1VI14CS078	MANJULA	SHIMOGA	990554123	F	10CS82	OUTSTANDING
1VI14CS078	MANJULA	SHIMOGA	990554123	F	10CS81	OUTSTANDING
1VI14CS066	PRAKASH	SHIMOGA	990123221	M	10CS84	OUTSTANDING
1VI14CS066	PRAKASH	SHIMOGA	990123221	M	10CS83	OUTSTANDING
1VI14CS066	PRAKASH	SHIMOGA	990123221	M	10CS82	AVERAGE
1VI14CS066	PRAKASH	SHIMOGA	990123221	M	10CS81	OUTSTANDING

5. Consider the schema for Company Database:

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)

```
CREATE TABLE EMPLOYEE(SSN VARCHAR(8), Name
                        VARCHAR(10), Address
                        VARCHAR(30), Sex
                        CHAR(2),
Salary NUMBER(10),SuperSSN VARCHAR(8),DNo VARCHAR(6),
PRIMARY KEY(SSN));
```

```
CREATE TABLE DEPARTMENT(DNo VARCHAR(6),
                        DName VARCHAR(10),
                        MgrSSN VARCHAR(8),
                        MgrStartDate DATE,
PRIMARY KEY(DNo),
FOREIGN KEY(MgrSSN) REFERENCES EMPLOYEE(SSN) ON DELETE CASCADE);
```

```
CREATE TABLE DLOCATION(DNo VARCHAR(6),
                        DLoc VARCHAR(15),
PRIMARY KEY(DNo,DLoc),
FOREIGN KEY(DNo) REFERENCES DEPARTMENT(DNo) ON DELETE CASCADE);
```

```
CREATE TABLE PROJECT(PNo VARCHAR(5),
                        PName VARCHAR(10),
                        PLocation VARCHAR(14),
                        DNo VARCHAR(6),
PRIMARY KEY(PNo),
FOREIGN KEY(DNo) REFERENCES DEPARTMENT(DNo) ON DELETE CASCADE);
```

```
CREATE TABLE WORKS_ON(SSN VARCHAR(8),
                        PNo VARCHAR(5),
                        Hours NUMBER(5),
PRIMARY KEY(PNo,SSN),
FOREIGN KEY(SSN) REFERENCES EMPLOYEE(SSN) ON DELETE CASCADE,
FOREIGN KEY(PNo) REFERENCES PROJECT(PNo) ON DELETE CASCADE);
```

```
ALTER TABLE EMPLOYEE ADD FOREIGN KEY(SuperSSN) REFERENCES
EMPLOYEE(SSN) ON DELETE CASCADE;
```

```
ALTER TABLE EMPLOYEE ADD FOREIGN KEY(DNo) REFERENCES
DEPARTMENT(DNo) ON DELETE CASCADE;
```

```
INSERT INTO EMPLOYEE(SSN, Name, Address, Sex,
Salary)VALUES('100','John','VV Puram,Bangalore','M',660000);
INSERT INTO EMPLOYEE(SSN, Name, Address, Sex,
Salary)VALUES('200','Scott','MG Road,Bangalore','M',700500);
INSERT INTO EMPLOYEE(SSN, Name, Address, Sex,
Salary)VALUES('300','Smith','Jayanagar,Bangalore','M',60000
0);
```




```

INSERT INTO EMPLOYEE(SSN, Name, Address, Sex,
Salary)VALUES('400','Vani','Vijayanagar,Bangalore','F',80000
0);
INSERT INTO EMPLOYEE(SSN, Name, Address, Sex,
Salary)VALUES('500','Gopal','PB Nagar,Bangalore','M',500000);
INSERT INTO EMPLOYEE(SSN, Name, Address, Sex, Salary) VALUES(600,'Ravi','Kormangala
Bangalore','M',700000);
INSERT INTO EMPLOYEE(SSN, Name, Address, Sex, Salary) VALUES(700,'Raghu','RR Nagar
Bangalore','M',680000);
INSERT INTO EMPLOYEE(SSN, Name, Address, Sex, Salary) VALUES(800,'Vinod','RT Nagar
Bangalore','M',800000);
INSERT INTO EMPLOYEE(SSN, Name, Address, Sex, Salary) VALUES(900,'Shankar','CH pete
Bangalore','M',606000);
INSERT INTO EMPLOYEE(SSN, Name, Address, Sex, Salary) VALUES(1000,'Sagar','VV Puram
Bangalore','M',800000);

```

```

INSERT INTO DEPARTMENT VALUES('D1','Accounts','200','11-Feb-2015');
INSERT INTO DEPARTMENT VALUES('D2','Research','200','11-Mar-2016');
INSERT INTO DEPARTMENT VALUES('D3','Finance','400','16-Jun-2015');
INSERT INTO DEPARTMENT VALUES('D4','Admin','100','30-Apr-2017');
INSERT INTO DEPARTMENT VALUES('D5','Testing','400','21-Mar-2016');

```

```

INSERT INTO DLOCATION VALUES('D1','Bangalore');
INSERT INTO DLOCATION VALUES('D2','Mysore');
INSERT INTO DLOCATION VALUES('D1','Mysore');
INSERT INTO DLOCATION VALUES('D3','Bangalore');
INSERT INTO DLOCATION VALUES('D4','Mangalore');

```

```

INSERT INTO PROJECT VALUES('P1','Billing','Bangalore','D1');
INSERT INTO PROJECT VALUES('P8','IoT','Mysore','D2');
INSERT INTO PROJECT VALUES('P3','Network','Davangere','D2');
INSERT INTO PROJECT VALUES('P4','Tax','Kolar','D1');
INSERT INTO PROJECT VALUES('P5','Salary','Bangalore','D3');
INSERT INTO PROJECT VALUES('P6','Placement','Mysore','D4');
INSERT INTO PROJECT VALUES('P7','Software','Bangalore','D5');

```

```

INSERT INTO WORKS_ON VALUES('100','P1',8);
INSERT INTO WORKS_ON VALUES('200','P3',10);
INSERT INTO WORKS_ON VALUES('300','P8',10);
INSERT INTO WORKS_ON VALUES('100','P8',10);
INSERT INTO WORKS_ON VALUES('400','P4',10);
INSERT INTO WORKS_ON VALUES('400','P6',12);
INSERT INTO WORKS_ON VALUES('500','P7',10);
INSERT INTO WORKS_ON VALUES('600','P4',10);
INSERT INTO WORKS_ON VALUES('700','P5',10);
INSERT INTO WORKS_ON VALUES('800','P1',10);
INSERT INTO WORKS_ON VALUES('900','P4',10);
INSERT INTO WORKS_ON VALUES('1000','P5',10);

```

```

UPDATE EMPLOYEE SET SuperSSN='200' where
SSN='100'; UPDATE EMPLOYEE SET SuperSSN='200'
where SSN='300'; UPDATE EMPLOYEE SET
SuperSSN='200' where SSN='400'; UPDATE EMPLOYEE
SET SuperSSN='300' where SSN='200'; UPDATE
EMPLOYEE SET SuperSSN='300' where SSN='500';
UPDATE EMPLOYEE SET SuperSSN='200' where
SSN='600'; UPDATE EMPLOYEE SET SuperSSN='200'

```

Department of Computer Science & Engg.VIT




```
UPDATE EMPLOYEE SET SuperSSN='200' where SSN='800';
UPDATE EMPLOYEE SET SuperSSN='200' where SSN='900';
UPDATE EMPLOYEE SET SuperSSN='200' where
SSN='1000';
```

```
UPDATE EMPLOYEE SET DNo='D1' where SSN='100';
UPDATE EMPLOYEE SET DNo='D2' where SSN='200';
UPDATE EMPLOYEE SET DNo='D3' where SSN='300';
UPDATE EMPLOYEE SET DNo='D4' where SSN='400';
UPDATE EMPLOYEE SET DNo='D2' where SSN='500';
UPDATE EMPLOYEE SET DNo='D1' where SSN='600';
UPDATE EMPLOYEE SET DNo='D1' where SSN='700';
UPDATE EMPLOYEE SET DNo='D1' where SSN='800';
UPDATE EMPLOYEE SET DNo='D1' where SSN='900';
UPDATE EMPLOYEE SET DNo='D1' where SSN='1000';
```

```
SELECT * FROM EMPLOYEE;
```

SSN	NAME	ADDRESS	SE	SALARY	SUPERSSN	DNO
100	John	VV Puram,Bangalore	M	660000	200	D1
200	Scott	MG Road,Bangalore	M	700500	300	D2
300	Smith	Jayanagar,Bangalore	M	600000	200	D3
400	Vani	Vijayanagar,Bangalore	F	800000	200	D4
500	Gopal	PB Nagar,Bangalore	M	500000	300	D2
600	Ravi	Kormangala	M	700000	200	D1
700	Raghu	RR Nagar Bangalore	M	680000	200	D1
800	Vinod	RT Nagar Bangalore	M	800000	200	D1
900	Shankar	CH pete Bangalore	M	606000	200	D1
1000	Sagar	VV Puram Bangalore	M	800000	200	D1

```
SELECT * FROM DEPARTMENT;
```

DNO	DNAME	MGRSSN	MGRSTARTD
D1	Accounts	200	11-FEB-15
D2	Research	200	11-MAR-16
D3	Finance	400	16-JUN-15
D4	Admin	100	30-APR-17
D5	Testing	400	21-MAR-16

```
SELECT * FROM DLOCATION;
```

DNO	DLOC
D1	Bangalore
D1	Mysore
D2	Mysore
D3	Bangalore
D4	Mangalor

```
SELECT * FROM PROJECT;
```

PNO	PNAME	PLOCATION	DNO
P1	Billing	Bangalore	D1
P8	IoT	Mysore	D2
P3	Network	Davangere	D2
P4	Tax	Kolar	D1
P5	Salary	Bangalore	D3
P6	Placement	Mysore	D4
P7	Software	Bangalore	D5



```
SELECT * FROM WORKS_ON;
SSN    PNO    HOURS
```

```
-----
100    P1      8
300    P3     10
300    P8     10
100    P8     10
400    P4     10
400    P6     12
500    P7     10
400    P8     10
600    P4     10
700    P5     10
800    P1     10
900    P4     10
1000   P5     10
```

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 controls the project.

```
SELECT DISTINCT PNO
FROM PROJECT
WHERE PNO IN(
    (SELECT P.PNO
     FROM PROJECT P, DEPARTMENT D, EMPLOYEE E
     WHERE P.DNO=D.DNO AND D.MGRSSN=E.SSN AND E.Name='Scott')
    UNION
    (SELECT W.PNO
     FROM WORKS_ON W, EMPLOYEE E
     WHERE W.SSN=E.SSN AND E.Name='Scott'));
```

OUTPUT:

```
PNO
----
P1
P3
P4
P8
```

2. Show the resulting salaries if every employee working on the 'IoT' project is given a 10 percent raise.

```
SELECT E.Name, 1.1 * E.Salary AS Increased_salary
FROM EMPLOYEE E, WORKS_ON W, PROJECT P
WHERE E.SSN=W.SSN AND W.PNO=P.PNO AND P.PName='IoT';
```

OUTPUT:

```
NAME    INCREASED_SALARY
-----
John      726000
Smith     660000
Vani      880000
```

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.

```
SELECT SUM (E. Salary) AS TOTAL_SALARY,
MAX(E. Salary) AS MAX_SALARY,
MIN(E. Salary) AS MIN_SALARY, AVG(E. Salary) AS AVG_SALARY
FROM EMPLOYEE E, DEPARTMENT D
WHERE E. DNo= D. DNo AND D.DName='Accounts';
```

OUTPUT:

TOTAL_SALARY	MAX_SALARY	MIN_SALARY	AVG_SALARY
4246000	800000	606000	707666.667

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

```
SELECT E.Name
FROM EMPLOYEE E
WHERE NOT EXISTS((SELECT PNo FROM PROJECT WHERE DNo='D5')
MINUS (SELECT W.PNo FROM WORKS_ON W WHERE E.SSN=W.SSN));
```

OUTPUT:

NAME
Gopal

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(*) AS NO_OF_EMP
FROM EMPLOYEE E, DEPARTMENT D
WHERE E.DNo= D.DNo AND E.Salary>600000
GROUP BY D.DNo
HAVING COUNT(*)>=5;
```

OUTPUT:

DNO	NO_OF_EMP
D1	6