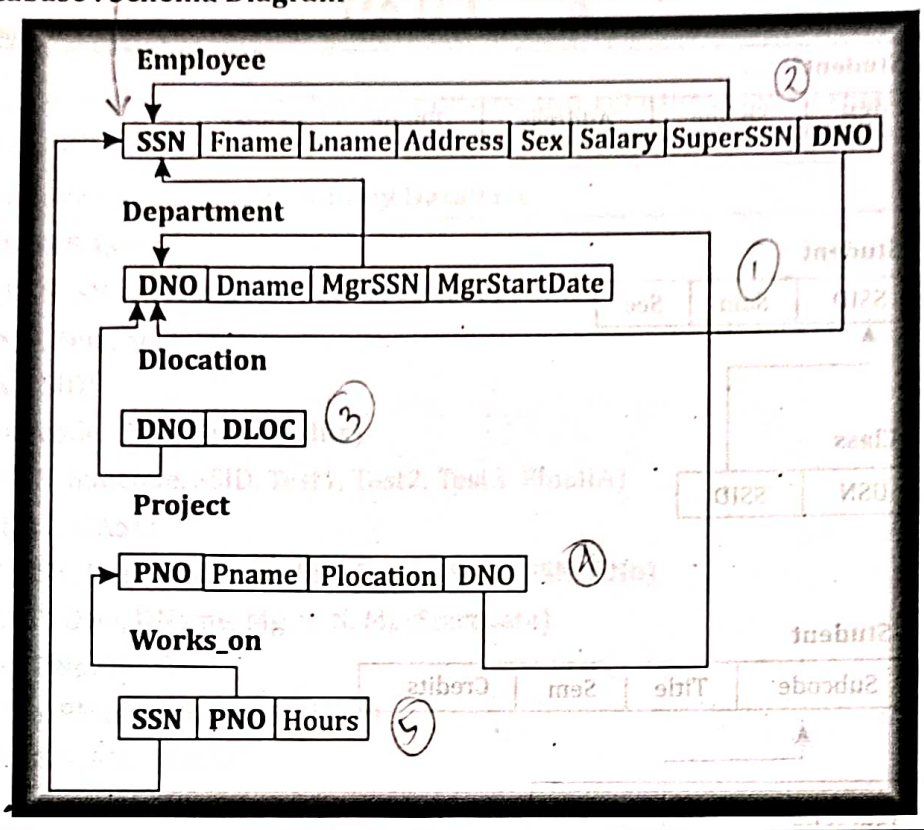
Company Database : Schema Diagram



Program 2	**Viewing all databases, Creating a Database, Viewing all Tables in a Database, Creating Tables (With and Without Constraints), Inserting/Updating/Deleting Records in a Table, Saving (Commit) and Undoing (rollback)**
Solution	Consider the Inventory Database with the following tables

Product(PID:Number;Name:Text;Price:Number)

Purchase(PO:Number;PID:Number:Qty:Number)

Steps :

1. Create database ~~Inventory~~ Company
2. Create tables Product and Purchase with and without constraint.
3. View all tables in ~~Inventory~~ Company Database
4. Insert five tuples into each relation
5. Display all the tuples in the Product and Purchase Table
6. Update the product name for the PID=40 as CAMERA Update SSN = 13 Salary = 70000
7. Delete information about the product whose PID=50
8. Perform Saving and Undoing

Step 1 : **Create database Inventory (In Oracle, creating a database means creating a user or schema)**

SQL> conn

Enter user-name: system

Enter password:tiger

Connected.

Connected to:

Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production

Version 19.3.0.0.0

SQL> alter session set "_ORACLE_SCRIPT"=true;

Session altered.

SQL> create user INVENTORY identified by tiger;

User created.

SQL> grant system to INVENTORY;

Grant succeeded.

SQL> GRANT ALL PRIVILEGES TO INVENTORY;

Grant succeeded.

SQL> conn

Enter user-name: Inventory

Enter password:

Connected.

Clear Screen;

Clear Screen

Step 2: **Create tables Product and Purchase with and without constraint.**

SQL> create table Product (PID Number(10) Primary Key, Name varchar2(20) Not Null,

2 Price Number(8,2)) ;

Table created.

SQL> create table Purchase (PO Number(10) Primary Key, PRODUCT_ID Number(10) references

2 Product(PID), Qty Number(5));

Table created.

SQL> desc Product;

Name	Null?	Type
PID	NOT NULL	NUMBER(10)
NAME	NOT NULL	VARCHAR2(20)
PRICE		NUMBER(8,2)

SQL> desc Purchase;

Name	Null?	Type
PO	NOT NULL	NUMBER(10)
PRODUCT_ID		NUMBER(10)
QTY		NUMBER(5)

Step 3 : **View all tables in Inventory Database**

SQL> select table_name, status from User_tables;

TABLE_NAME	STATUS
PRODUCT	VALID
PURCHASE	VALID

Step 4 : **Insert five tuples into each relation**

SQL> Insert into Product(PID,NAME,PRICE) values(10,'PRINTER',20000);
1 row created.
SQL> Insert into Product(PID,NAME,PRICE) values(20,'KEYBOARD',20000);
1 row created.
SQL> Insert into Product(PID,NAME,PRICE) values(30,'MONITOR',15000);
1 row created.
SQL> Insert into Product(PID,NAME,PRICE) values(40,'TABLE',25000);
1 row created.
SQL> Insert into Product(PID,NAME,PRICE) values(50,'SCANNER',14000);
1 row created.
SQL> Insert into Purchase(PO,PRODUCT_ID,Qty) values(101,10,25);
1 row created.
SQL> Insert into Purchase(PO,PRODUCT_ID,Qty) values(102,40,20);
1 row created.
SQL> Insert into Purchase(PO,PRODUCT_ID,Qty) values(107,30,40);
1 row created.
SQL> Insert into Purchase(PO,PRODUCT_ID,Qty) values(104,40,50);
1 row created.
SQL> Insert into Purchase(PO,PRODUCT_ID,Qty) values(105,40,10);
1 row created.

Step 5 : **Display all the tuples in the Product and Purchase Table**

SQL> select * from Product;

PID	NAME	PRICE
10	PRINTER	20000
20	KEYBOARD	20000
30	MONITOR	15000
40	TABLE	25000
50	SCANNER	14000

SQL> select * from Purchase;

PO	PRODUCT_ID	QTY
101	10	25
102	40	20
107	30	40
104	40	50
105	40	10

Step 6. **Update the product name for the PID=40 as CAMERA**

SQL> Update Product set Name='CAMERA' where PID=40;
1 row updated.

SQL> select * from Product;

PID	NAME	PRICE
10	PRINTER	20000
20	KEYBOARD	20000
30	MONITOR	15000
40	CAMERA	25000
50	SCANNER	14000

Step 7 : **Delete information about the product whose PID=50**

SQL> DELETE from Product where PID=50;

1 row deleted.

SQL> select * from Product;

PID	NAME	PRICE
10	PRINTER	20000
20	KEYBOARD	20000
30	MONITOR	15000
40	CAMERA	25000

Step 8: **Perform Saving and Undoing**

SQL> Insert into Product(PID,Name,Price) values(50,'MOBILE',35000);

1 row created.

SQL> Insert into Product(PID,Name,Price) values(60,'LAPTOP',70000);

1 row created.

SQL> commit;

Commit complete.

SQL> select * from Product;

PID	NAME	PRICE
10	PRINTER	20000
20	KEYBOARD	20000
30	MONITOR	15000
40	CAMERA	25000
50	MOBILE	35000
60	LAPTOP	70000

6 rows selected.

SQL> Savepoint S1;

Savepoint created.

SQL> Insert into Product(PID,Name,Price) values(70,'TABLE',50000);

1 row created.

SQL> Insert into Product(PID,Name,Price) values(80,'CHAIR',25000);

1 row created.

SQL> Rollback to S1;

Rollback complete.

SQL> select * from Product;

PID	NAME	PRICE
10	PRINTER	20000
20	KEYBOARD	20000
30	MONITOR	15000
40	CAMERA	25000
50	MOBILE	35000
60	LAPTOP	70000

6 rows selected.

Program 3	**Altering a Table, Dropping/Truncating/Renaming a table, Backing up/Restoring a Database**
Solution	Consider the Library Database with the following data and execute the quesries

Lib (BID: Number; Title:Text;Autor:Text;Publication:Text;Year_Of_Publication:Text)

Steps:

1. Create Lib table by properly specifying the constraint.
2. Rename the LIB as LIBRARY
3. Add a new column Price with Not Null constraint to the existing table Library.
4. All the constraints and views that reference the column are dropped automatically along with the column.
5. Rename the BID to BOOKID in the Library table.
6. Change the data type of the column Year_Of_Publication as Text with size 15.
7. Insert data into Library table
8. Truncate table to delete the records
9. Drop table

Step 1 : **Create Lib table by properly specifying the constraint.**

```
SQL> CREATE TABLE LIB
  2 (BID VARCHAR2(8) PRIMARY KEY,
  3  TITLE VARCHAR2(20) NOT NULL,
  4  AUTHOR VARCHAR2(20),
  5  PUBLICATION VARCHAR2(20),
  6  YEAR_OF_PUBLICATION NUMBER(4)
  7 );
```

Table created.

Name	Null?	Type
BID	NOT NULL	VARCHAR2(8)
TITLE	NOT NULL	VARCHAR2(20)
AUTHOR		VARCHAR2(20)
PUBLICATION		VARCHAR2(20)
YEAR_OF_PUBLICATION		NUMBER(4)

```
SQL> DESC LIB;
```

Step 2 : **Rename the LIB as LIBRARY**

```
SQL> ALTER TABLE LIB RENAME TO LIBRARY ;
```

Table altered.

SQL> DESC LIBRARY;

Name	Null?	Type
BID	NOT NULL	VARCHAR2(8)
TITLE	NOT NULL	VARCHAR2(20)
AUTHOR		VARCHAR2(20)
PUBLICATION		VARCHAR2(20)
YEAR_OF_PUBLICATION		NUMBER(4)

Step 3: **Add a new column Price with Not Null constraint to the existing table Library**

SQL> ALTER TABLE LIBRARY ADD PRICE NUMBER(8,2) NOT NULL;

Table altered.

SQL> DESC LIBRARY;

Name	Null	Type
BID	NOT NULL	VARCHAR2(8)
TITLE	NOT NULL	VARCHAR2(20)
AUTHOR		VARCHAR2(20)
PUBLICATION		VARCHAR2(20)
YEAR_OF_PUBLICATION		NUMBER(4)
PRICE	NOT NULL	NUMBER(8,2)

Step 4: **All the constraints and views that reference the column are dropped automatically along with the column.**

SQL> ALTER TABLE LIBRARY DROP COLUMN AUTHOR CASCADE CONSTRAINTS ;

Table altered.

SQL> DESC LIBRARY;

Name	Null	Type
BID	NOT NULL	VARCHAR2(8)
TITLE	NOT NULL	VARCHAR2(20)
PUBLICATION		VARCHAR2(20)
YEAR_OF_PUBLICATION		NUMBER(4)

Step 5: **Rename the BID to BOOKID in the Library table**

SQL> ALTER TABLE LIBRARY RENAME COLUMN BID TO BOOKID ;

Table altered.

SQL> DESC LIBRARY;

Name	Null	Type
BOOKID	NOT NULL	VARCHAR2(8)
TITLE	NOT NULL	VARCHAR2(20)
PUBLICATION		VARCHAR2(20)
YEAR_OF_PUBLICATION		NUMBER(4)
PRICE	NOT NULL	NUMBER(8,2)

Step 6 : **Change the data type of the column Year_Of_Publication as Text with size 15**

SQL> ALTER TABLE LIBRARY MODIFY YEAR_OF_PUBLICATION VARCHAR2(15);

Table altered.

SQL> DESC LIBRARY;

Name	Null	Type
BOOKID	NOT NULL	VARCHAR2(8)
TITLE	NOT NULL	VARCHAR2(20)
PUBLICATION		VARCHAR2(20)
YEAR_OF_PUBLICATION		VARCHAR2(15)
PRICE	NOT NULL	NUMBER(8,2)

Step 7 : **Insert data into Library table**

SQL> insert into LIBRARY values ('SP001', 'DBMS','SKYWARD PUBLISHERS', '2022', 300);

1 row inserted.

SQL> DESC LIBRARY;

Name	Null	Type
BOOKID	NOT NULL	VARCHAR2(8)
TITLE	NOT NULL	VARCHAR2(20)
PUBLICATION		VARCHAR2(20)
YEAR_OF_PUBLICATION		VARCHAR2(15)
PRICE	NOT NULL	NUMBER(8,2)

Step 8: **Truncate table to delete the records**

SQL> TRUNCATE TABLE LIBRARY;

Table LIBRARY truncated.

FOR VERIFICATION

SQL> DESC LIBRARY;

Name	Null	Type
BOOKID	NOT NULL	VARCHAR2(8)
TITLE	NOT NULL	VARCHAR2(20)
PUBLICATION		VARCHAR2(20)
YEAR_OF_PUBLICATION		VARCHAR2(15)
PRICE	NOT NULL	NUMBER(8,2)

Step 9: **Drop table**

SQL> DROP TABLE LIBRARY;

Table dropped.

FOR VERIFICATION

SQL> DESC LIBRARY;

ERROR:

ORA-04043: object LIBRARY does not exist

Program 4 For a given set of relation schemes, create tables and perform the following Simple Queries, Simple Queries with Aggregate functions, Queries with Aggregate functions (group by and having clause)

Solution

Consider the Salary database and execute the following simple queries

SALARYDB(EID: String;NAME:Text;DEPT:String,DOJ:Date;SALARY:Number)

Steps:

1. Create table Salary.
2. Enter five tuples into the table.
3. Display Employee Number and their Salary.
4. Find the sum of salaries of all the employees.
5. Find the sum and average salaries of employees of a particular department.
6. Find the number of employees working for each department.
7. Display Employee information in ascending and descending order of their date of joining.
8. Find the highest salary that an Employee draws.
9. Find the least salary that an Employee draws.
10. Display the details of employee whose name is Rushank and salary is greater than 50,000.

Step 1: **Create table Salary**

```
SQL>  CREATE TABLE SALARYDB  (ENO VARCHAR2(8) PRIMARY KEY, NAME VARCHAR2(15)
NOT NULL, DEPT VARCHAR2(10), DOJ DATE, SALARY NUMBER(10,2) );
Table created.

SQL> DESC SALARYDB
```

Name	Null	Type
ENO	NOT NULL	VARCHAR2(8)
NAME	NOT NULL	VARCHAR2(15)
DEPT		VARCHAR2(10)
DOJ		DATE
SALARY		NUMBER(10,2)

Step 2: **Enter five tuples into the table**

```
SQL>  INSERT INTO SALARYDB VALUES(&ENO,&NAME,&DEPT,&DOJ,&SALARY);
Enter value for eno: 'SC1010'
Enter value for name: 'AHANA'
Enter value for dept: 'HR'
Enter value for doj: '15-FEB-2010'
Enter value for salary: 60000
old 1:  INSERT INTO SALARYDB VALUES(&ENO,&NAME,&DEPT,&DOJ,&SALARY)
new 1:  INSERT INTO SALARYDB VALUES('SC1010','AHANA','HR','15-FEB-2010',60000)
1 row created.
```

```
SQL> /
Enter value for eno: 'SC1011'
Enter value for name: 'RAMESH'
Enter value for dept: 'FINANCE'
Enter value for doj: '10-MAR-2012'
Enter value for salary: 45000
old 1:  INSERT INTO SALARYDB VALUES(&ENO,&NAME,&DEPT,&DOJ,&SALARY)
new 1:  INSERT INTO SALARYDB VALUES('SC1011','RAMESH','FINANCE','10-MAR-2012',45000)
1 row created.

SQL> /
Enter value for eno: 'SC1013'
Enter value for name: 'NAVEEN'
Enter value for dept: 'MARKETING'
Enter value for doj: '8-JAN-2009'
Enter value for salary: 55000
old 1:  INSERT INTO SALARYDB VALUES(&ENO,&NAME,&DEPT,&DOJ,&SALARY)
new 1:  INSERT INTO SALARYDB VALUES('SC1013','NAVEEN','MARKETING','8-JAN-2009',55000)
1 row created.

SQL> /
Enter value for eno: 'SC1014'
Enter value for name: 'ANAGHA'
Enter value for dept: 'HR'
Enter value for doj: '14-APR-2012'
Enter value for salary: 35000
old 1:  INSERT INTO SALARYDB VALUES(&ENO,&NAME,&DEPT,&DOJ,&SALARY)
new 1:  INSERT INTO SALARYDB VALUES('SC1014','ANAGHA','HR','14-APR-2012',35000)
1 row created.

SQL> /
Enter value for eno: 'SC1015'
Enter value for name: 'RUSHANK'
Enter value for dept: 'ADMIN'
Enter value for doj: '16-MAY-2011'
Enter value for salary: 55000
old 1:  INSERT INTO SALARYDB VALUES(&ENO,&NAME,&DEPT,&DOJ,&SALARY)
new 1:  INSERT INTO SALARYDB VALUES('SC1015','RUSHANK','ADMIN','16-MAY-2011',55000)
1 row created.
```

```
SQL> /
Enter value for eno: 'SC1016'
Enter value for name: 'RUSHANK'
Enter value for dept: 'FINANCE'
Enter value for doj: '4-JUN-2008'
Enter value for salary: 25000
old 1:  INSERT INTO SALARYDB VALUES(&ENO,&NAME,&DEPT,&DOJ,&SALARY)
new 1:  INSERT INTO SALARYDB VALUES('SC1016','RUSHANK','FINANCE','4-JUN-2008',25000)
SQL> SELECT  * FROM SALARYDB;
```

ENO	NAME	DEPT	DOJ	SALARY
SC1010	AHANA	HR	15-FEB-10	60000
SC1011	RAMESH	FINANCE	10-MAR-12	45000
SC1013	NAVEEN	MARKETING	08-JAN-09	55000
SC1014	ANAGHA	HR	14-APR-12	35000
SC1015	RUSHANK	ADMIN	16-MAY-11	55000
SC1016	RUSHANK	FINANCE	04-JUN-08	25000

6 rows selected

Step 3: **Display Employee Number and their Salary**

```
SQL> SELECT ENO,SALARY FROM SALARYDB;
```

ENO	SALARY
SC1010	60000
SC1011	45000
SC1013	55000
SC1014	35000
SC1015	55000
SC1016	25000

6 rows selected.

Step 4: **Find the sum of salaries of all the employees.**

```
SQL> SELECT SUM(SALARY) AS "TOTAL_SALARY" FROM SALARYDB;
```

TOTAL_SALARY

275000

Step 5: **Find the sum and average salaries of employees of a particular department**

```
SQL>SELECT DEPT,SUM(SALARY) AS "TOTAL_SALARY",AVG(SALARY) AS "AVERAGE_SALARY" FROM
  2  SALARYDB GROUP BY DEPT ;
```

DEPT	TOTAL SALARY	AVERAGE_SALARY
ADMIN	55000	55000
HR	95000	47500
FINANCE	70000	35000
MARKETING	55000	55000

Step 6: **Find the number of employees working for each department**

SQL> SELECT DEPT,COUNT(*) AS "NUMBER_OF_EMPLOYEES" FROM SALARYDB GROUP BY DEPT;

DEPT	NUMBER_OF_EMPLOYEES
ADMIN	1
HR	2
FINANCE	2
MARKETING	1

Step 7: **Display Employee information in ascending and descending order of their date of joining.**

SQL> SELECT * FROM SALARYDB ORDER BY DOJ ASC;

ENO	NAME	DEPT	DOJ	SALARY
SC1016	RUSHANK	FINANCE	04-JUN-08	25000
SC1013	NAVEEN	MARKETING	08-JAN-09	55000
SC1010	AHANA	HR	15-FEB-10	60000
SC1015	RUSHANK	ADMIN	16-MAY-11	55000
SC1011	RAMESH	FINANCE	10-MAR-12	45000
SC1014	ANAGHA	HR	14-APR-12	35000

6 rows selected.

SQL> SELECT * FROM SALARYDB ORDER BY DOJ DESC;

ENO	NAME	DEPT	DOJ	SALARY
SC1014	ANAGHA	HR	14-APR-12	35000
SC1011	RAMESH	FINANCE	10-MAR-12	45000
SC1015	RUSHANK	ADMIN	16-MAY-11	55000
SC1010	AHANA	HR	15-FEB-10	60000
SC1013	NAVEEN	MARKETING	08-JAN-09	55000
SC1016	RUSHANK	FINANCE	04-JUN-08	25000

6 rows selected.

Step 8: **Find the highest salary that an Employee draws.**

SQL> SELECT MAX(SALARY) AS "HIGHEST_SALARY" FROM SALARYDB

HIGHEST_SALARY

60000

Step 9: **Find the least salary that an Employee draws**

SQL> SELECT MIN(SALARY) AS "LEAST_SALARY" FROM SALARYDB;

LEAST_SALARY

25000

Step 10: **Display the details of employee whose name is Rushank and salary is greater than 50000**

```
SQL>  SELECT * FROM SALARYDB
 2 WHERE NAME='RUSHANK' AND SALARY>50000 ;
```

ENO	NAME	DEPT	DOJ	SALARY
SC1015	RUSHANK	ADMIN	16-MAY-11	55000

NOTE: **TO EXECUTE PROGRAM 5, 6, 7, 8, 9 & 10 , CREATE THE FOLLOWING TABLES**

Consider the Company database with following Schema

EMP(ENO:Number;ENAME;String;EBDATE:Date;ADDRESS:Text;GENDER:Text;SALARY:Number;DEPTNO;Number)

DEPT(DNO:Number;DNAME:String;DLOCATION;String)

PROJECT(PNO:Number; PNAME:String; DNUM:Number)

WORKS_ON(ENO:Number; PNUM;Number;HOURS:Number)

Dependent (essn : number; Dependent_name String; Sex String; Bdate date; Relationship String)

CREATING TABLES BY PROPERLY SPECIFYING PRIMARY AND FOREIGN KEYS

```
SQL>  CREATE TABLE DEPT (DNO NUMBER(4) PRIMARY KEY,  DNAME VARCHAR2(20) NOT NULL,
 2  DLOCATION VARCHAR2(20) ) ;
Table created.

SQL> CREATE TABLE EMP (ENO NUMBER(6) PRIMARY KEY, ENAME VARCHAR2(20) NOT NULL,
 2  EBDATE DATE, ADDRESS VARCHAR2(20),  GENDER CHAR, SALARY NUMBER(10) NOT NULL,
 3  DEPTNO NUMBER(4) REFERENCES DEPT );
Table created.

SQL> DESC DEPT;
```

Name	Null ?	Type
DNO	NOT NULL	NUMBER(4)
DNAME	NOT NULL	VARCHAR2(20)
DLOCATION		VARCHAR2(20)

SQL> DESC EMP

Name	Null ?	Type
ENO	NOT NULL	NUMBER(6)
ENAME	NOT NULL	VARCHAR2(20)
EBDATE		DATE
ADDRESS		VARCHAR2(20)
GENDER		CHAR(1)
SALARY	NOT NULL	NUMBER(10)
DEPTNO		NUMBER(4)

```
SQL> CREATE TABLE PROJECT (PNO NUMBER(10) PRIMARY KEY, PNAME VARCHAR2(20) NOT
2 NULL, DNUM NUMBER(4) REFERENCES DEPT );
```

```
SQL> DESC PROJECT;
```

Name	Null ?	Type
PNO	NOT NULL	NUMBER(10)
PNAME	NOT NULL	VARCHAR2(20)
DNUM		NUMBER(4)

```
SQL> CREATE TABLE WORKS_ON (ENO NUMBER(6) REFERENCES EMP NOT NULL,
 2 PNUM NUMBER(10) REFERENCES PROJECT NOT NULL, HOURS NUMBER(3,1) NOT NULL,
 3 PRIMARY KEY(ENO,PNUM) ) ;
Table created.
```

```
SQL> DESC WORKS_ON;
```

Name	Null ?	Type
ENO	NOT NULL	NUMBER(6)
PNUM	NOT NULL	NUMBER(10)
HOURS	NOT NULL	NUMBER(3,1)

```
SQL> INSERT INTO DEPT(DNO,DNAME,DLOCATION) VALUES (2,'ACCOUNTS','JAYANAGAR');
1 row created.
SQL> INSERT INTO DEPT(DNO,DNAME,DLOCATION) VALUES (4,'RESEARCH','KENGERI');
1 row created.
SQL> INSERT INTO DEPT(DNO,DNAME,DLOCATION) VALUES (5,'ADMIN','SOUTHEND');
1 row created.
SQL> INSERT INTO EMP(ENO,ENAME,EBDATE,ADDRESS,GENDER,SALARY,DEPTNO)
VALUES(1001,'ANIRUDH','14-JAN-1990','BANGALORE','M',45000,4);
1 row created.
SQL> INSERT INTO EMP(ENO,ENAME,EBDATE,ADDRESS,GENDER,SALARY,DEPTNO)
VALUES(1004,'LAKSHMI','4-MAR-1998','MYSORE','F',55000,4);
1 row created.
SQL> INSERT INTO EMP(ENO,ENAME,EBDATE,ADDRESS,GENDER,SALARY,DEPTNO)
VALUES(1002,'SINHCANA','22-DEC-1990','MANGALORE','F',50000,2);
1 row created.
SQL> INSERT INTO EMP(ENO,ENAME,EBDATE,ADDRESS,GENDER,SALARY,DEPTNO)
VALUES(1007,'PRASHANT','26-JAN-1989','DHARWAD','M',20000,4);
1 row created.
SQL> INSERT INTO EMP(ENO,ENAME,EBDATE,ADDRESS,GENDER,SALARY,DEPTNO)
VALUES(1003,'VINAY','26-NOV-1990','HUBLI','M',30000,2);
1 row created.
```

SQL> INSERT INTO EMP(ENO,ENAME,EBDATE,ADDRESS,GENDER,SALARY,DEPTNO) VALUES(1005,'VIDYA','26-NOV-1978','HUBLI','F',35000,4);

1 row created.

SQL> INSERT INTO EMP(ENO,ENAME,EBDATE,ADDRESS,GENDER,SALARY,DEPTNO) VALUES(1006,'PRAJWAL','2-FEB-1974','BANGALORE','M',65000,5);

1 row created.

SQL> INSERT INTO EMP(ENO,ENAME,EBDATE,ADDRESS,GENDER,SALARY,DEPTNO) VALUES(1008,'RAJESH','2-FEB-2010','BANGALORE','M',25000,2);

1 row created.

SQL> SELECT * FROM EMP;

ENO	ENAME	EBDATE	ADDRESS	G	SALARY	DEPTNO
1001	ANIRUDH	14-JAN-90	BANGALORE	M	45000	4
1004	LAKSHMI	04-MAR-98	MYSORE	F	5500	4
1002	SINHCANA	22-DEC-90	MANGALORE	F	5000	2
1007	PRASHANT	26-JAN-89	DHARWAD	M	20000	4
1003	VINAY	26-NOV-90	HUBLI	M	3000	2
1005	VIDYA	26-NOV-78	HUBLI	F	35000	4
1006	PRAJWAL	02-FEB-74	BANGALORE	M	65000	5
1008	RAJESH	02-FEB-10	BANGALORE	M	25000	2

8 rows selected.

SQL> SELECT * FROM DEPT;

DNO	DNAME	DLOCATION
2	ACCOUNTS	JAYANAGAR
4	RESEARCH	KENGERI
5	ADMIN	SOUTHEND

SQL> INSERT INTO PROJECT (PNO,PNAME,DNUM) VALUES(10,'ERP',5);

1 row created.

SQL> INSERT INTO PROJECT (PNO,PNAME,DNUM) VALUES(20,'BANKING',2);

1 row created.

SQL> INSERT INTO PROJECT (PNO,PNAME,DNUM) VALUES(30,'CONNECT_TECH',4);

1 row created.

SQL> INSERT INTO PROJECT (PNO,PNAME,DNUM) VALUES(40,'SMART_SEEK',5);

1 row created.

SQL> INSERT INTO PROJECT (PNO,PNAME,DNUM) VALUES(50,'FINANCE',2);

1 row created.

SQL> INSERT INTO PROJECT (PNO,PNAME,DNUM) VALUES(60,'ANALYTICA',4);

1 row created.

SQL> INSERT INTO PROJECT (PNO,PNAME,DNUM) VALUES(70,'MARKET_RESEARCH',4);

1 row created.

SQL> INSERT INTO PROJECT (PNO,PNAME,DNUM) VALUES(80,'SMART_SEARCH',4);
1 row created.

SQL> SELECT * FROM PROJECT;

PNO	PNAME	DNUM
10	ERP	5
20	BANKING	2
30	CONNECT_TECH	4
40	SMART_SEEK	5
50	FINANCE	2
60	ANALYTICA	4
70	MARKET_RESEARCH	4
80	SMART_SEARCH	4

8 rows selected.

SQL> INSERT INTO WORKS_ON(ENO,PNUM,HOURS) VALUES (1001,10,4.5);
1 row created.
SQL> INSERT INTO WORKS_ON(ENO,PNUM,HOURS) VALUES (1002,10,6);
1 row created.
SQL> INSERT INTO WORKS_ON(ENO,PNUM,HOURS) VALUES (1008,10,4);
1 row created.
SQL> INSERT INTO WORKS_ON(ENO,PNUM,HOURS) VALUES (1006,20,4);
1 row created.
SQL> INSERT INTO WORKS_ON(ENO,PNUM,HOURS) VALUES (1004,20,8);
1 row created.
SQL> INSERT INTO WORKS_ON(ENO,PNUM,HOURS) VALUES (1005,40,8);
1 row created.
SQL> INSERT INTO WORKS_ON(ENO,PNUM,HOURS) VALUES (1003,50,8);
1 row created.
SQL> INSERT INTO WORKS_ON(ENO,PNUM,HOURS) VALUES (1007,60,5);
1 row created.

SQL> SELECT * FROM WORKS_ON;

ENO	PNUM	HOURS
1001	10	4.5
1002	10	6
1008	10	4
1006	20	4
1004	20	8
1005	40	8
1003	50	8
1007	60	5

8 rows selected.

Program 5 **Execute the Following Queries**

(a) How the resulting salaries if every employee working on the 'Research' Departments is given a 10% raise.

(b) 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

Solution

(a) How the resulting salaries if every employee working on the 'Research' Departments is given a 10 percent raise.

```
SQL> SELECT E.ENO,E.ENAME,D.DNAME ,1.1*E.SALARY AS "INC_SALARY" FROM EMP E, DEPT D
  2  WHERE E.DEPTNO=D.DNO AND D.DNAME='RESEARCH';
```

ENO	ENAME	DNAME	INC_SALARY
1001	ANIRUDH	RESEARCH	49500
1004	LAKSHM	RESEARCH	60500
1007	PRASHANT	RESEARCH	22000
1005	VIDYA	RESEARCH	38500

$\frac{1}{100} \times 10$

(b) 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.

```
SQL> SELECT MAX(E.SALARY),MIN(E.SALARY),SUM(E.SALARY),AVG(E.SALARY) FROM EMP E, DEPT D
  2  WHERE E.DEPTNO=D.DNO AND D.DNAME='ACCOUNTS';
```

MAX(E.SALARY)	MIN(E.SALARY)	SUM(E.SALARY)	AVG(E.SALARY)
50000	25000	105000	35000

Program 6 **Execute the Following Queries**

(a) Retrieve the name of each employee Controlled by Department number 5 (use EXISTS operator).

(b) Retrieve the name of each dept and number of employees working in each Department which has at least 2 employees.

Solution

(a) Retrieve the name of each employee Controlled by Department number 5 (use EXISTS operator).

```
SQL> SELECT E.ENAME FROM EMP E
  2  WHERE EXISTS(SELECT D.DNO FROM DEPT D WHERE E.DEPTNO=D.DNO AND E.DEPTNO=5);
```

ENAME
PRAJWAL

(b) Retrieve the name of each dept and number of employees working in each department which has at least 2 employees.

```
SQL> SELECT D.DNAME,COUNT(*) FROM EMP E,DEPT D
  2  WHERE E.DEPTNO=D.DNO  GROUP BY D.DNAME  HAVING COUNT(*)>=2;
```

DNAME	COUNT(*)
ACCOUNTS	3
RESEARCH	4

at least - min 2
max ∞

Program 7 **Execute the Following Queries**

(a) For each project, retrieve the project number, the project name, and the number of employee who work on that project.(use GROUP BY)

(b) Retrieve the name of employees who born in the year 1990's

Solution

(a) For each project, retrieve the project number, the project name, and the number of employee who work on that project.(use GROUP BY)

```
SQL> SELECT P.PNO,P.PNAME,COUNT(*) AS "NO_OF_EMP" FROM PROJECT P,WORKS_ON W WHERE
2      P.PNO=W.PNUM GROUP BY P.PNO,P.PNAME ;
```

PNO	PNAME	NO_OF_EMP
10	ERP	3
60	ANALYTICA	1
20	BANKING	2
40	SMART_SEEK	1
50	FINANCE	1

(b) Retrieve the name of employees who born in the year 1990.

```
SQL>  SELECT ENAME,EBDATE FROM EMP  WHERE EBDATE LIKE '%-%-90' ;
```

Pattern matching

ENAME	EBDATE
ANIRUDH	14-JAN-90
SINHCANA	22-DEC-90
VINAY	26-NOV-90

Program 8 **Execute the Following Queries**

For each Department that has more than five employees, retrieve the department number and number of employees who are making salary more than 40000.

Solution

1. For each Department that has more than Five employees, retrieve the department number and number of employees who are making salary more than 40000.

```
SQL> SELECT D.DNAME, D.DNO,COUNT(*) AS "NO_OF_EMP" FROM EMP E,DEPT D
  2  WHERE E.DEPTNO=D.DNO AND E.SALARY>40000 AND
  3  D.DNO IN (SELECT DEPTNO FROM EMP GROUP BY DEPTNO HAVING COUNT(*)>=5)
  4  GROUP BY D.DNO,D.DNAME;
no rows selected
```

2. For each Department that has more than Two employees, retrieve the department number and number of employees who are making salary more than 40,000.

```
SQL>  SELECT D.DNAME, D.DNO,COUNT(*) FROM EMP  E,DEPT  D
2  WHERE E.DEPTNO=D.DNO AND E.SALARY>40000 AND
3  D.DNO IN (SELECT DEPTNO FROM EMP14 GROUP BY DEPTNO HAVING COUNT(*)>2)
4 GROUP BY D.DNO,D.DNAME;
```

DNAME	DNO	COUNT(*)
ACCOUNTS	2	1
RESEARCH	4	2

Program 9 **For each project on which more than two employees work, retrieve the project number, project name and the number of employees who work on that project.**

Solution

```
SQL> SELECT P.PNO,P.PNAME,COUNT(*)  AS "NO_OF_EMP_WORKING" FROM PROJECT P,WORKS_ON W
  2    WHERE P.PNO=W.PNUM  GROUP BY P.PNO,P.PNAME  HAVING COUNT(*)>2;
```

PNO	PNAME	NO_OF_EMP_WORKING
10	ERP	3

Program 10 **For a given set of relation tables perform the following: Creating Views (with and without check option), Dropping views, Selecting from a view**

Solution

What is a View?

In Oracle, view is a virtual table that does not physically exist. It is stored in Oracle data dictionary and do not store any data. It can be executed when called. A view is created by a query joining one or more tables. Views are used for security purposes because they provide encapsulation of the name of the table. Data is in the virtual table, not stored permanently. Views display only selected data. We can also use joins in the Select statement in deriving the data for the view.

NOTE: Let us consider the tables Emp And Dept table Created For Program Number 5, 6, 7, 8, and 9. We will create the view on both Emp and Dept table.

1. **Without Check Option:**

Step 1: **Create View**

```
SQL> CREATE VIEW  EMP_DEPT AS  (SELECT E.ENO,E.ENAME,E.SALARY, E.DEPTNO, D.DNAME
  2        FROM EMP E,DEPT D  WHERE E.DEPTNO=D.DNO);
View created.
```

Step 2: **Display all the rows of a View.**

```
SQL> SELECT * FROM EMP_DEPT;
```

ENO	ENAME	SALARY	DEPTNO	DNAME
1001	ANIRUDH	45000	4	RESEARCH
1004	LAKSHMI	55000	4	RESEARCH
1002	SINHCANA	50000	2	ACCOUNTS
1007	PRASHANT	20000	4	RESEARCH
1003	VINAY	30000	2	ACCOUNTS
1005	VIDYA	35000	4	RESEARCH
1006	PRAJWAL	65000	5	ADMIN
1008	RAJESH	25000	2	ACCOUNTS

8 rows selected.

Step 3: **Insert records into a View**

SQL> INSERT INTO EMP_DEPT(ENO,ENAME,SALARY,DEPTNO) VALUES(1009,'SRIKANTH',90000,5);

1 row created.

Step 4: **Display all the rows of a View.**

SQL> SELECT * FROM EMP_DEPT;

ENO	ENAME	SALARY	DEPTNO	DNAME
1001	ANIRUDH	45000	4	RESEARCH
1004	LAKSHMI	55000	4	RESEARCH
1002	SINHCANA	50000	2	ACCOUNTS
1007	PRASHANT	20000	4	RESEARCH
1003	VINAY	30000	2	ACCOUNTS
1005	VIDYA	35000	4	RESEARCH
1006	PRAJWAL	65000	5	ADMIN
1008	RAJESH	25000	2	ACCOUNTS
1009	SRIKANTH	90000	5	ADMIN

9 rows selected.

Step 5: **Drop View**

SQL> DROP VIEW EMP_DEPT;

View dropped.

2. With Check Option :

The WITH CHECK OPTION clause in SQL is a very useful clause for views. It is applicable to a updatable view. If the view is not updatable, then there is no meaning of including this clause in the CREATE VIEW statement.

The WITH CHECK OPTION clause is used to prevent the insertion of rows in the view where the condition in the WHERE clause in CREATE VIEW statement is not satisfied.

If we have used the WITH CHECK OPTION clause in the CREATE VIEW statement, and if the UPDATE or INSERT clause does not satisfy the conditions then they will return an error.

Step 1: Let us create simple view on EMP table with check option of salary less than 50000 in where condition.

SQL> CREATE VIEW EMP_VIEW AS

2 (SELECT ENO,ENAME,SALARY FROM EMP WHERE SALARY <=50000) WITH CHECK OPTION;

View EMP_VIEW created.

Step 2: **Display all the rows of a View.**

SQL> SELECT * FROM EMP_VIEW;

ENO	ENAME	SALARY
1001	ANIRUDH	45000
1002	SINHCANA	50000
1007	PRASHANT	20000
1003	VINAY	30000
1005	VIDYA	35000
1008	RAJESH	25000

6 rows selected.

Step 3: Insert a row where employee salary is less than 50000.

SQL> INSERT INTO EMP_VIEW(ENO,ENAME,SALARY) VALUES(1011,'SNIGDHA',39000);

1 row inserted.

Step 4: **Display all the rows of a View.**

SQL> SELECT * FROM EMP_VIEW;

ENO	ENAME	SALARY
1001	ANIRUDH	45000
1002	SINHCANA	50000
1007	PRASHANT	20000
1003	VINAY	30000
1005	VIDYA	35000
1008	RAJESH	25000
1011	SNIGDHA	39000

7 rows selected.

Step 5 : **Insert a row where employee salary is greater than 50000. This will give an eroor.**

SQL> INSERT INTO EMP_VIEW(ENO,ENAME,SALARY) VALUES(1012,'SMAYAN',99999);

ORA-01402: view WITH CHECK OPTION where-clause violation

Step 6 : **Drop View**

SQL> DROP VIEW EMP_VIEW;

View dropped.

PART - B

Create the following tables with properly specifying Primary keys, Foreign keys and Execute the following queries.

BRANCH (Branchid, Branchname, HOD)

STUDENT (USN, Name, Address, Branchid, sem)

BOOK (Bookid, Bookname, Authorid, Publisher, Branchid)

AUTHOR (Authorid, Authorname, Country, age)

BORROW (USN, Bookid, Borrowed_Date)

Bid Bname Hod

usn name Add Bid Sem

Bookid Bname Authorid

Step 1 : **Create College database**

```
SQL> conn
Enter user-name: system
Enter password:tiger
Connected.
Connected to:
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.3.0.0.0
SQL> alter session set "_ORACLE_SCRIPT"=true;
Session altered.
SQL> create user COLLEGEDB identified by tiger;
User created.
SQL> grant system to COLLEGEDB;
Grant succeeded.
SQL> GRANT ALL PRIVILEGES TO COLLEGEDB;
Grant succeeded.
SQL> conn
Enter user-name: COLLEGEDB
Enter password:tiger
Connected.
```

Step 2 : **Create all the tables by properly specifying key constraints.**

```
SQL> CREATE TABLE BRANCH (BRID NUMBER(5) PRIMARY KEY, BRNAME VARCHAR2(15) NOT NULL,
2 HOD VARCHAR2(10) );
Table created.

SQL> DESC BRANCH;
```

Name	Null?	Type
BRID	NOT NULL	NUMBER(5)
BRNAME	NOT NULL	VARCHAR2(15)
HOD		VARCHAR2(10)

SQL> CREATE TABLE STUDENT (USN VARCHAR2(15) PRIMARY KEY, NAME VARCHAR2(15) NOT NULL,
2 ADDRESS VARCHAR2(15), BRID NUMBER(5) REFERENCES BRANCH, SEM VARCHAR2(10));
Table created.

SQL> DESC STUDENT;

Name	Null?	Type
USN	NOT NULL	VARCHAR2(15)
NAME	NOT NULL	VARCHAR2(15)
ADDRESS		VARCHAR2(15)
BRID		NUMBER(5)
SEM		VARCHAR2(10)

SQL> CREATE TABLE AUTHOR (AID VARCHAR2(10) PRIMARY KEY, ANAME VARCHAR2(15) NOT NULL,
2 COUNTRY VARCHAR2(15), AGE NUMBER(4));
Table created.

SQL> DESC AUTHOR;

Name	Null?	Type
AID	NOT NULL	VARCHAR2(10)
ANAME	NOT NULL	VARCHAR2(15)
COUNTRY		VARCHAR2(15)
AGE		NUMBER(4)

SQL> CREATE TABLE BOOK (BKID VARCHAR2(10) PRIMARY KEY, BKNAME VARCHAR2(15) NOT NULL,
2 AID VARCHAR2(10) REFERENCES AUTHOR, PUBLISHER VARCHAR2(20) NOT NULL,
3 BRID NUMBER(5) REFERENCES BRANCH);
Table created.

SQL> DESC BOOK;

Name	Null?	Type
BKID	NOT NULL	VARCHAR2(10)
BKNAME	NOT NULL	VARCHAR2(15)
AID		VARCHAR2(10)
PUBLISHER	NOT NULL	VARCHAR2(20)
BRID		NUMBER(5)

SQL> CREATE TABLE BORROW (USN VARCHAR2(15) REFERENCES STUDENT, BKID VARCHAR2(10)
2 REFERENCES BOOK , BORROW_DATE DATE, PRIMARY KEY(USN,BKID));
Table created.

SQL> DESC BORROW;

Name	Null?	Type
USN	NOT NULL	VARCHAR2(15)
BKID	NOT NULL	VARCHAR2(10)
BORROW_DATE		DATE

Program 1 **Viewing all databases, Creating a Databases, Viewing all Tables in a Database, Creating Tables (With and Without Constraints), Inserting/Updating/Deleting Records in a Table, Saving (Commit) and Undoing(rollback)**

Solution

Step 1: **View all tables in COLLEGE Database**

SQL> SELECT TABLE_NAME, STATUS FROM USER_TABLES;

TABLE_NAME	STATUS
BRANCH	VALID
STUDENT	VALID
AUTHOR	VALID
BOOK	VALID
BORROW	VALID

Step 2 : **Inserting Records into a Table.**

SQL> INSERT INTO BRANCH(BRID,BRNAME,HOD) VALUES(10,'BCA','SANTOSH S');

1 row created.

SQL> INSERT INTO BRANCH(BRID,BRNAME,HOD) VALUES(20,'BBA','RASHMI E');

1 row created.

SQL> INSERT INTO BRANCH(BRID,BRNAME,HOD) VALUES(30,'BCOM','RAMESH A');

1 row created.

SQL> INSERT INTO BRANCH(BRID,BRNAME,HOD) VALUES(40,'BSC','ADITYA P');

1 row created.

SQL> INSERT INTO BRANCH(BRID,BRNAME,HOD) VALUES(50,'BA','ASHA P');

1 row created.

SQL> SELECT * FROM BRANCH;

BRID	BRNAME	HOD
10	BCA	SANTOSH S
20	BBA	RASHMI E
30	BCOM	RAMESH A
40	BSC	ADITYA P
50	BA	ASHA P

SQL> INSERT INTO STUDENT(USN,NAME,ADDRESS,BRID,SEM) VALUES('SCAS202201','ANURADHA','JAYANAGR',10,'II SEM');

1 row created.

SQL> INSERT INTO STUDENT(USN,NAME,ADDRESS,BRID,SEM) VALUES('SCAS202202','MANULA','BASAVANGUDI',10,'II SEM');

1 row created.

SQL> INSERT INTO STUDENT(USN,NAME,ADDRESS,BRID,SEM) VALUES('SCAC202204','ARUN','JPNAGAR',30,'II SEM');

1 row created.

SQL> INSERT INTO STUDENT(USN,NAME,ADDRESS,BRID,SEM) VALUES('SCAS202204','ABHI','GIRINAGAR',40,'II SEM');

1 row created.

SQL> INSERT INTO STUDENT(USN,NAME,ADDRESS,BRID,SEM) VALUES('SCAA202201','DEEPTI','GIRINAGAR',50,'IV SEM');

1 row created.

SQL> SELECT * FROM STUDENT;

USN	NAME	ADDRESS	BRID	SEM
SCAS202201	ANURADHA	JAYANAGR	10	II SEM
SCAS202202	MANULA	BASAVANGUDI	10	II SEM
SCAS202203	LAKSHMI	BASAVANGUDI	10	IV SEM
SCAC202203	RENUKA	HANUMANTNAGAR	20	II SEM
SCAC202204	ARUN	JPNAGAR	30	II SEM
SCAS202204	ABHI	GIRINAGAR	40	II SEM
SCAA202201	DEEPTI	GIRINAGAR	50	IV SEM

7 rows selected.

SQL> INSERT INTO AUTHOR(AID,ANAME,COUNTRY,AGE) VALUES('NEPCOMP01','ARUNA','INDIA',36);

1 row created.

SQL> INSERT INTO AUTHOR(AID,ANAME,COUNTRY,AGE) VALUES('NEPCOMP02','SUMA','INDIA',38);

1 row created.

SQL> INSERT INTO AUTHOR(AID,ANAME,COUNTRY,AGE) VALUES('NEPCOMM02','SANGEETHA','INDIA',42);

1 row created.

SQL> INSERT INTO AUTHOR(AID,ANAME,COUNTRY,AGE) VALUES('NEPCOMM01','DILIP','INDIA',39);

1 row created.

SQL> INSERT INTO AUTHOR(AID,ANAME,COUNTRY,AGE) VALUES('NEPSCI01','SHEKAR','INDIA',44);

1 row created.

SQL> SELECT * FROM AUTHOR;

AID	ANAME	COUNTRY	AGE
NEPCOMP01	ARUNA	INDIA	36
NEPCOMP02	SUMA	INDIA	38
NEPCOMM02	SANGEETHA	INDIA	42
NEPCOMM01	DILIP	INDIA	39
NEPSCI01	SHEKAR	INDIA	44

5 rows selected.

SQL> INSERT INTO BOOK(BKID,BKNAME,AID,PUBLISHER,BRID) VALUES('NEPDBMS','DBMS','NEPCOMP02','SKYWARD',10);

1 row created.

SQL> INSERT INTO BOOK(BKID,BKNAME,AID,PUBLISHER,BRID) VALUES('NEPSE','SE','NEPCOMP02','SKYWARD',10);

1 row created.

SQL> INSERT INTO BOOK(BKID,BKNAME,AID,PUBLISHER,BRID) VALUES('NEPJAVA','JAVA','NEPCOMM01','OXFORD',20);

1 row created.

SQL> INSERT INTO BOOK(BKID,BKNAME,AID,PUBLISHER,BRID) VALUES('NEPMATHS','MATHS','NEPSCI01','OXFORD',30);

1 row created.

SQL> INSERT INTO BOOK(BKID,BKNAME,AID,PUBLISHER,BRID) VALUES('NEPPHY','PHYSICS','NEPCOMM02','SHREE',40);

1 row created.

SQL> SELECT * FROM BOOK;

BKID	BKNAME	AID	PUBLISHER	BRID
NEPDBMS	DBMS	NEPCOMP02	SKYWARD	10
NEPSE	SE	NEPCOMP02	SKYWARD	10
NEPJAVA	JAVA	NEPCOMM01	OXFORD	20
NEPMATHS	MATHS	NEPSCI01	OXFORD	30
NEPPHY	PHYSICS	NEPCOMM02	SHREE	40

SQL> INSERT INTO BORROW(USN,BKID,BORROW_DATE) VALUES('SCAS202201','NEPDBMS','20-MAY-2022');

1 row created.

SQL> INSERT INTO BORROW(USN,BKID,BORROW_DATE) VALUES('SCAS202201','NEPSE','28-MAY-2022');

1 row created.

SQL> INSERT INTO BORROW(USN,BKID,BORROW_DATE) VALUES('SCAC202204','NEPMATHS','6-JUN-2022');

1 row created.

SQL> INSERT INTO BORROW(USN,BKID,BORROW_DATE) VALUES('SCAA202201','NEPPHY','12-JUN-2022');

1 row created.

SQL> INSERT INTO BORROW(USN,BKID,BORROW_DATE) VALUES('SCAS202203','NEPPHY','18-JUN-2022');

1 row created.

SQL>INSERT INTO BORROW(USN,BKID,BORROW_DATE) VALUES('SCAS202201','NEPMATHS','5-JUN-2022');

1 row created.

SQL> INSERT INTO BORROW(USN,BKID,BORROW_DATE) VALUES ('SCAS202201','NEPJAVA','5-JUN-2022');

1 row created.

SQL> SELECT * FROM BORROW;

USN	BKID	BORROW_DA
SCAS202201	NEPDBMS	20-MAY-22
SCAS202201	NEPSE	28-MAY-22
SCAC202204	NEPMATHS	06-JUN-22
SCAA202201	NEPPHY	12-JUN-22
SCAS202203	NEPPHY	18-JUN-22
SCAS202201	NEPMATHS	05-JUN-22
SCAS202201	NEPJAVA	05-JUN-22

7 rows selected.

Step 3 : **Updating Records in a Table.**

SQL> UPDATE BOOK SET PUBLISHER='SKYWARD' WHERE BKID='NEPJAVA';

1 row updated.

SQL> SELECT * FROM BOOK;

BKID	BKNAME	AID	PUBLISHER	BRID
NEPDBMS	DBMS	NEPCOMP02	SKYWARD	10
NEPSE	SE	NEPCOMP02	SKYWARD	10
NEPJAVA	JAVA	NEPCOMM01	SKYWARD	20
NEPMATHS	MATHS	NEPSCI01	OXFORD	30
NEPPHY	PHYSICS	NEPCOMM02	SHREE	40

5 rows selected.

Step 4 : **Deleting Records from a Table.**

SQL> DELETE BORROW WHERE BKID='NEPPHY';

2 rows deleted.

SQL> SELECT * FROM BORROW;

USN	BKID	BORROW_DA
SCAS202201	NEPDBMS	20-MAY-22
SCAS202201	NEPSE	28-MAY-22
SCAC202204	NEPMATHS	06-JUN-22
SCAS202201	NEPMATHS	05-JUN-22
SCAS202201	NEPJAVA	05-JUN-22

5 rows selected.

SStep 5 : **Perform Saving (COMMIT)**

SQL> INSERT INTO BRANCH(BRID,BRNAME,HOD) VALUES(60,'MCA','BALAJI');

1 row created.

SQL> INSERT INTO BRANCH(BRID,BRNAME,HOD) VALUES(70,'MBA','VENKAT');

1 row created.

SQL> COMMIT;

Commit complete.

SQL> SELECT * FROM BRANCH;

BRID	BRNAME	HOD
10	BCA	SANTOSH S
20	BBA	RASHMI E
30	BCOM	RAMESH A
40	BSC	ADITYA P
50	BA	ASHA P
60	MCA	BALAJI
70	MBA	VENKAT

7 rows selected.

Step 6 : **Perform Undoing (ROLL BACK)**

SQL> SAVEPOINT S1;

Savepoint created.

SQL> INSERT INTO BRANCH(BRID,BRNAME,HOD) VALUES(80,'MATHS','VEENA') ;

1 row created.

SQL> INSERT INTO BRANCH(BRID,BRNAME,HOD) VALUES(90,'CHEMISTRY','JALAJA');

1 row created.

SQL> ROLL BACK S1;

Rollback complete.

SQL> SELECT * FROM BRANCH;

BRID	BRNAME	HOD
---	---	---
10	BCA	SANTOSH S
20	BBA	RASHMI E
30	BCOM	RAMESH A
40	BSC	ADITYA P
50	BA	ASHA P
60	MCA	BALAJI
70	MBA	VENKAT

7 rows selected.

Program 2

(a) List the details of Students who are all studying in 2nd sem BCA.

(b) List the students who are not borrowed any books.

Solution

(a) List the details of Students who are all studying in 2nd sem BCA.

```
SQL>  SELECT * FROM STUDENT S,BRANCH BR  WHERE S.BRID=BR.BRID
 2  AND S.SEM='II SEM' AND BR.BRNAME='BCA'  ;
```

USN	NAME	ADDRESS	BRID	SEM	BRID BRNAME	BRNAME	HOD
SCAS202201	ANURADHA	JAYANAGR	10	II SEM	10	BCA	SANTOSH S
SCAS202202	MANULA	BASAVANGUDI	10	II SEM	10	BCA	SANTOSH S

(b) List the students who are not borrowed any books.

```
SQL> SELECT * FROM STUDENT S
 2  WHERE S.USN NOT IN(SELECT B.USN FROM BORROW B);
```

USN	NAME	ADDRESS	BRID SEM	SEM
SCAA202201	DEEPTI	GIRINAGAR	50	IV SEM
SCAC202203	RENUKA	HANUMANTNAGAR	20	II SEM
SCAS202202	MANULA	BASAVANGUDI	10	II SEM
SCAS202203	LAKSHMI	BASAVANGUDI	10	IV SEM
SCAS202204	ABHI	GIRINAGAR	40	II SEM

Program 3

(a) Display the USN, Student name, Branch_name, Book_name, Author_name, Books_Borrowed_Date of 2nd sem BCA Students who borrowed books.

(b) Display the number of books written by each Author.

Solution

(a) Display the USN, Student name, Branch_name, Book_name, Author_name, Books_Borrowed_Date of 2nd sem BCA Students who borrowed books.

```
SQL> SELECT S.USN,S.NAME,S.SEM,BR.BRNAME,BK.BKNAME,A.ANAME,B.BORROW_DATE
2       FROM STUDENT S,BRANCH BR,BOOK BK,AUTHOR A,BORROW B
3       WHERE S.BRID=BR.BRID AND S.BRID=BK.BRID  AND A.AID=BK.AID
4       AND B.USN=S.USN  AND BK.BKID=B.BKID AND S.SEM='II SEM' AND BR.BRNAME='BCA';
```

USN	NAME	SEM	BRNAME	BKNAME	ANAME	BORROW_DATE
SCAS202201	ANURADHA	II SEM	BCA	DBMS	SUMA	20-05-22
SCAS202201	ANURADHA	II SEM	BCA	SE	SUMA	28-05-22

(b) Display the number of books written by each Author.

```
SQL> SELECT A.ANAME,COUNT(DISTINCT BK.BKID) AS "NO_OF_BOOKS" FROM AUTHOR A,BOOK BK
 2  WHERE A.AID=BK.AID  GROUP BY A.ANAME ;
```

ANAME	NO_OF_BOOKS
SUMA	2
SHEKAR	1
SANGEETHA	1
DILIP	1

Program 4

(a) Display the student details who borrowed more than two books.

(b) Display the student details who borrowed books of more than one Author.

Solution

(a) Display the student details who borrowed more than two books.

```
SQL> SELECT S.NAME  FROM STUDENT S,BORROW B   WHERE S.USN=B.USN
 2     GROUP BY S.NAME HAVING COUNT(DISTINCT B.BKID)>2 ;
```

NAME
ANURADHA

(b) Display the student details who borrowed books of more than one Author.

```
SQL>  SELECT S.NAME,COUNT(DISTINCT BK.AID)  FROM STUDENT S,BOOK BK,BORROW B
 2  WHERE S.USN=B.USN AND B.BKID=BK.BKID  GROUP BY S.NAME
 3 HAVING COUNT(DISTINCT BK.AID)>1;
```

NAME	COUNT(DISTINCTBK.AID)
ANURADHA	3

Program 5

(a) Display the Book names in descending order of their names.

(b) List the details of students who borrowed the books which are all published by the same publisher.

Solution

(a) Display the Book names in descending order of their names.

```
SQL> SELECT * FROM BOOK  ORDER BY BKNAME DESC;
```

BKID	BKNAME	AID	PUBLISHER	BRID
NEPSE	SE	NEPCOMP02	SKYWARD	10
NEPPHY	PHYSICS	NEPCOMM02	SHREE	40
NEPMATHS	MATHS	NEPSCI01	OXFORD	30
NEPJAVA	JAVA	NEPCOMM01	SKYWARD	20
NEPDBMS	DBMS	NEPCOMP02	SKYWARD	10

(b) List the details of students who borrowed the books which are all published by the same publisher.

```
SQL> SELECT S.NAME,COUNT(BK.PUBLISHER) FROM STUDENT S,BOOK BK,BORROW B
  2 WHERE S.USN=B.USN AND B.BKID=BK.BKID GROUP BY S.NAME
  3 HAVING COUNT(BK.PUBLISHER)>1;
```

NAME	COUNT(BK.PUBLISHER)
ANURADHA	4

NOTE: PROGRAM 6 to 8 requires below schema to be created.

STUDENT (USN, name, date_of_birth, branch, mark1, mark2, mark3, total, GPA)

Program 6 **Perform the following:**

Creating Tables (With and Without Constraints), Inserting/Updating/Deleting Records in a Table, Saving (Commit) and Undoing (rollback)

Solution

Step 1: **Creating Table (With and Without Constraints)**

```
SQL> CREATE TABLE STUDENT (USN VARCHAR2(10) PRIMARY KEY,
  2 NAME VARCHAR2(20) NOT NULL, DOB DATE, BRANCH VARCHAR2(10) NOT NULL,
  3 MARK1 NUMBER(4) NOT NULL, MARK2 NUMBER(4) NOT NULL, MARK3 NUMBER(4) NOT NULL,
  4 TOTAL NUMBER(4), GPA NUMBER(4,2) ) ;
Table created.

SQL> DESC STUDENT;
```

Name	Null?	Type
USN	NOT NULL	VARCHAR2(10)
NAME	NOT NULL	VARCHAR2(20)
DOB		DATE
BRANCH	NOT NULL	VARCHAR2(10)
MARK1	NOT NULL	NUMBER(4)
MARK2	NOT NULL	NUMBER(4)
MARK3	NOT NULL	NUMBER(4)
TOTAL		NUMBER(4)
GPA		NUMBER(4,2)

Step 2: **Inserting Records into a Table.**

```
SQL> INSERT INTO STUDENT(USN,NAME,DOB,BRANCH,MARK1,MARK2,MARK3,TOTAL,GPA)
VALUES('SCA202201','SANJANA','24-AUG-2004','BCA',85,96,97,NULL,NULL) ;

1 row created.

SQL> INSERT INTO STUDENT(USN,NAME,DOB,BRANCH,MARK1,MARK2,MARK3,TOTAL,GPA) VA
LUES('SCAC202201','ANIRUDH','10-OCT-2004','BCOM',75,85,65,NULL,NULL) ;

1 row created.
```

SQL> INSERT INTO STUDENT(USN,NAME,DOB,BRANCH,MARK1,MARK2,MARK3,TOTAL,GPA) VALUES('SCAB202201','AKASH','11-NOV-2004','BBA',75,85,83,NULL,NULL);

1 row created.

SQL> INSERT INTO STUDENT(USN,NAME,DOB,BRANCH,MARK1,MARK2,MARK3,TOTAL,GPA) VALUES('SCA202202','TANDRA','1-DEC-2004','BCA',84,56,63,NULL,NULL);

1 row created.

SQL> INSERT INTO STUDENT(USN,NAME,DOB,BRANCH,MARK1,MARK2,MARK3,TOTAL,GPA) VALUES('SCA202203','ANUSHA','1-JAN-2005','BCA',68,72,78,NULL,NULL);

1 row created.

SQL> SELECT * FROM STUDENT;

USN	NAME	DOB	BRANCH	MARK1	MARK2	MARK3	TOTAL	GPA
SCA202201	SANJANA	24-AUG-04	BCA	85	96	97		
SCAC202201	ANIRUDH	10-OCT-04	BCOM	75	85	65		
SCAB202201	AKASH	11-NOV-04	BBA	75	85	83		
SCA202202	TANDRA	01-DEC-04	BCA	84	56	63		
SCA202203	ANUSHA	01-JAN-05	BCA	68	72	78		

5 rows selected.

Step 3: **Inserting Records into a Table and Saving (Commit)**

SQL> INSERT INTO STUDENT(USN,NAME,DOB,BRANCH,MARK1,MARK2,MARK3,TOTAL,GPA) VALUES('SCAC202203','SAHANA','15-JAN-2005','BCOM',45,65,68,NULL,NULL);

1 row created.

SQL> INSERT INTO STUDENT(USN,NAME,DOB,BRANCH,MARK1,MARK2,MARK3,TOTAL,GPA) VALUES('SCAC202204','YOHAN','15-FEB-2004','BCOM',55,76,78,NULL,NULL);

1 row created.

SQL> INSERT INTO STUDENT(USN,NAME,DOB,BRANCH,MARK1,MARK2,MARK3,TOTAL,GPA) VALUES('SCAC202205','SRIKANTH','05-MAY-1977','BCOM',99,89,79,NULL,NULL);

SQL> COMMIT;

Commit complete.

Step 4: **Deleting Records into a Table and Saving (Commit)**

SQL> DELETE STUDENT WHERE USN='SCAC202205';

1 row deleted.

Step 5: **Undoing (rollback)**

SQL> SAVEPOINT S1 ;

Savepoint created.

SQL> INSERT INTO STUDENT(USN,NAME,DOB,BRANCH,MARK1,MARK2,MARK3,TOTAL,GPA) VALUES('SCAB202204','SATYA','15-MAR-2004','BBA',68,77,74,NULL,NULL);

1 row created.

SQL> INSERT INTO STUDENT(USN,NAME,DOB,BRANCH,MARK1,MARK2,MARK3,TOTAL,GPA) VALUES('SCAB202205','DHANU','25-MAR-2004','BBA',68,85,90,NULL,NULL);

1 row created.

SQL> ROLL BACK;

Rollback complete.

SQL> SELECT * FROM STUDENT;

USN	NAME	DOB	BRANCH	MARK1	MARK2	MARK3	TOTAL	GPA
SCA202201	SANJANA	24-AUG-04	BCA	85	96	97		
SCAC202201	ANIRUDH	10-OCT-04	BCOM	75	85	65		
SCAB202201	AKASH	11-NOV-04	BBA	75	85	83		
SCA202202	TANDRA	01-DEC-04	BCA	84	56	63		
SCA202203	ANUSHA	01-JAN-05	BCA	68	72	78		
SCAC202203	SAHANA	15-JAN-05	BCOM	45	65	68		
SCAC202204	YOHAN	15-FEB-04	BCOM	55	76	78		

7 rows selected.

Step 6: **Updating Records in a Table.**

SQL> UPDATE STUDENT SET TOTAL=MARK1+MARK2+MARK3;

7 rows updated.

SQL> SELECT * FROM STUDENT;

USN	NAME	DOB	BRANCH	MARK1	MARK2	MARK3	TOTAL	GPA
SCA202201	SANJANA	24-AUG-04	BCA	85	96	97	278	
SCAC202201	ANIRUDH	10-OCT-04	BCOM	75	85	65	225	
SCAB202201	AKASH	11-NOV-04	BBA	75	85	83	243	
SCA202202	TANDRA	01-DEC-04	BCA	84	56	63	203	
SCA202203	ANUSHA	01-JAN-05	BCA	68	72	78	218	
SCAC202203	SAHANA	15-JAN-05	BCOM	45	65	68	178	
SCAC202204	YOHAN	15-FEB-04	BCOM	55	76	78	209	

7 rows selected.

Program 7 **Execute the following queries:**

(a) Find the GPA score of all the students.

(b) Find the students who born on a particular year of birth from the date_of_birth column.

Solution

(a) Find the GPA score of all the students.

SQL> UPDATE STUDENT SET GPA=((100*TOTAL)/300)/10 ;

7 rows updated.

SQL> SELECT * FROM STUDENT;

USN	NAME	DOB	BRANCH	MARK1	MARK2	MARK3	TOTAL	GPA
---	---	---	---	---	---	---	---	---
SCA202201	SANJANA	24-AUG-04	BCA	85	96	97	278	9.27
SCAC202201	ANIRUDH	10-OCT-04	BCOM	75	85	65	225	7.5
SCAB202201	AKASH	11-NOV-04	BBA	75	85	83	243	8.1
SCA202202	TANDRA	01-DEC-04	BCA	84	56	63	203	6.77
SCA202203	ANUSHA	01-JAN-05	BCA	68	72	78	218	7.27
SCAC202203	SAHANA	15-JAN-05	BCOM	45	65	68	178	5.93
SCAC202204	YOHAN	15-FEB-04	BCOM	55	76	78	209	6.97

7 rows selected.

(b) Find the students who born on a particular year of birth from the date_of_birth column.

SQL> SELECT USN,NAME,BRANCH,DOB FROM STUDENT WHERE DOB LIKE '%-%-04';

USN	NAME	BRANCH	DOB
SCA202201	SANJANA	BCA	24-AUG-04
SCAC202201	ANIRUDH	BCOM	10-OCT-04
SCAB202201	AKASH	BBA	11-NOV-04
SCA202202	TANDRA	BCA	01-DEC-04
SCAC202204	YOHAN	BCOM	15-FEB-04

Program 8

(a) List the students who are studying in a particular branch of study.
(b) Find the maximum GPA score of the student branch-wise.

Solution

(a) List the students who are studying in a particular branch of study.

SQL> SELECT USN,NAME,BRANCH,DOB FROM STUDENT WHERE BRANCH='BCA';

USN	NAME	BRANCH	DOB
SCA202201	SANJANA	BCA	24-AUG-04
SCA202202	TANDRA	BCA	01-DEC-04
SCA202203	ANUSHA	BCA	01-JAN-05

(b) Find the maximum GPA score of the student branch-wise.

SQL> SELECT BRANCH,MAX(GPA) FROM STUDENT GROUP BY BRANCH;

BRANCH	MAX (GPA)
BCA	9.27
BCOM	7.5
BBA	8.1

VIVA QUESTIONS

1. What is DBMS ?

Ans. The database management system is a collection of programs that enables user to store, retrieve, update and delete information from a database.

2. What is RDBMS ?

Ans. Relational Database Management system (RDBMS) is a database management system (DBMS) that is based on the relational model. Data from relational database can be accessed or reassembled in many different ways without having to reorganize the database tables. Data from relational database can be accessed using an API , Structured Query Language (SQL).

3. What is SQL ?

Ans. Structured Query Language(SQL) is a language designed specifically for communicating with databases. SQL is an ANSI (American National Standards Institute) standard.

4. What are the different type of SQL's ?

Ans.
1. **DDL** – Data Definition Language: DDL is used to define the structure that holds the data. For example, table.
2. **DML** – Data Manipulation Language: DML is used for manipulation of the data itself. Typical operations are Insert, Delete, Update and retrieving the data from the table.
3. **DCL** – Data Control Language: DCL is used to control the visibility of data like granting database access and set privileges to create tables etc.

5. What are the Advantages of SQL.

Ans.
1. SQL is not a proprietary language used by specific database vendors. Almost every major DBMS supports SQL, so learning this one language will enable programmers to interact with any database like ORACLE, SQL ,MYSQL etc.
2. SQL is easy to learn. The statements are all made up of descriptive English words, and there aren't that many of them.
3. SQL is actually a very powerful language and by using its language elements you can perform very complex and sophisticated database operations.

6. What is a field in a database ?

Ans. A field is an area within a record reserved for a specific piece of data.

Examples: Employee Name, Employee ID etc

7. What is a Record in a database ?

Ans. A record is the collection of values / fields of a specific entity: i.e. an Employee, Salary etc.

8. What is a Table in a database ?

Ans. A table is a collection of records of a specific type. For example, employee table, salary table etc.

9. What is a database transaction?

Ans. Database transaction takes database from one consistent state to another. At the end of the transaction the system must be in the prior state if the transaction fails or the status of the system should reflect the successful completion if the transaction goes through.

10. What are properties of a transaction?

Ans. Properties of the transaction can be summarized as ACID Properties.

1. **Atomicity:** A transaction consists of many steps. When all the steps in a transaction gets completed, it will get reflected in DB or if any step fails, all the transactions are rolled back.

2. **Consistency:** The database will move from one consistent state to another, if the transaction succeeds and remain in the original state, if the transaction fails.
3. **Isolation:** Every transaction should operate as if it is the only transaction in the system.
4. **Durability:** Once a transaction has completed successfully, the updated rows/records must be available for all other transactions on a permanent basis.

11. What is a Database Lock ?

Ans. Database lock tells a transaction, if the data item in questions is currently being used by other transactions.

12. What are the type of locks ?

1. Shared Lock: When a shared lock is applied on data item, other transactions can only read the item, but can't write into it.
2. Exclusive Lock: When an exclusive lock is applied on data item, other transactions can't read or write into the data item.

13. What are the different type of normalization?

Ans. In database design, we start with one single table, with all possible columns. A lot of redundant data would be present since it's a single table. The process of removing the redundant data, by splitting up the table in a well defined fashion is called normalization.

1. **First Normal Form (1NF):** A relation is said to be in first normal form if and only if all underlying domains contain atomic values only. After 1NF, we can still have redundant data.
2. **Second Normal Form (2NF):** A relation is said to be in 2NF if and only if it is in 1NF and every non key attribute is fully dependent on the primary key. After 2NF, we can still have redundant data.
3. **Third Normal Form (3NF):** A relation is said to be in 3NF, if and only if it is in 2NF and every non key attribute is non-transitively dependent on the primary key.

14. What are constraints? Explain different types of constraints.

Ans. Constraints enable the RDBMS enforce the integrity of the database automatically, without needing you to create triggers, rule or defaults.

Types of constraints: NOT NULL, CHECK, UNIQUE, PRIMARY KEY, FOREIGN KEY

For an explanation of these constraints see books online for the pages titled: "Constraints" and "CREATE TABLE", "ALTER TABLE"

15. What is a primary key?

Ans. A primary key is a column whose values uniquely identify every row in a table Primary key values can never be reused. If a row is deleted from the table, its primary key may not be assigned to any new rows in the future. To define a field as primary key, following conditions had to be met :

1. No two rows can have the same primary key value.
2. Every row must have a primary key value
3. The primary key field cannot be null
4. Values in primary key columns can never be modified or updated

16. What is a Composite Key ?

Ans. A Composite primary key is a type of candidate key, which represents a set of columns whose values uniquely identify every row in a table.For example - if "Employee_ID" and "Employee Name" in a table is combined to uniquely identify a row its called a Composite Key.

17. What is a Composite Primary Key ?

Ans. A Composite primary key is a set of columns whose values uniquely identify every row in a table. What it means is that, a table which contains composite primary key will be indexed based on the columns specified in the primary key. This key will be referred in Foreign Key tables.

For example - if the combined effect of columns, "Employee_ID" and "Employee Name" in a table is required to uniquely identify a row, its called a Composite Primary Key. In this case, both the columns will be represented as primary key.

18. What is a Foreign Key ?

Ans. When a "one" table's primary key field is added to a related "many" table in order to create the common field which relates the two tables, it is called a foreign key in the "many" table.

For example, the salary of an employee is stored in salary table. The relation is established via foreign key column "Employee_ID_Ref" which refers "Employee_ID" field in the Employee table.

19. What is a Unique Key ?

Ans. Unique key is same as primary with the difference being the existence of null. Unique key field allows one value as NULL value.

20. Define SQL Insert Statement ?

Ans. SQL INSERT statement is used to add rows to a table. For a full row insert, SQL Query should start with "insert into " statement followed by table name and values command, followed by the values that need to be inserted into the table. The insert can be used in several ways:

1. To insert a single complete row.
2. To insert a single partial row.

21. Define SQL Update Statement ?

Ans. SQL Update is used to update data in a row or set of rows specified in the filter condition.

The basic format of an SQL UPDATE statement is, Update command followed by table to be updated and SET command followed by column names and their new values followed by filter condition that determines which rows should be updated.

22. Define SQL Delete Statement ?

Ans. SQL Delete is used to delete a row or set of rows specified in the filter condition.

The basic format of an SQL DELETE statement is, DELETE FROM command followed by table name followed by filter condition that determines which rows should be updated.

23. What are wild cards used in database for Pattern Matching ?

Ans. SQL Like operator is used for pattern matching. SQL 'Like' command takes more time to process. So before using "like" operator, consider suggestions given below on when and where to use wild card search.

1. Don't overuse wild cards. If another search operator will do, use it instead.
2. When you do use wild cards, try not to use them at the beginning of the search pattern, unless absolutely necessary. Search patterns that begin with wild cards are the slowest to process.
3. Pay careful attention to the placement of the wild card symbols. If they are misplaced, you might not return the data you intended.

24. Define Joins and explain different type of joins?

Ans. In order to avoid data duplication, data is stored in related tables. Join keyword is used to fetch data from related tables. "Join" return rows when there is at least one match in both tables. Right Outer Join: Return all rows from the right table, even if there are no matches in the left table.

Left Outer Join: Return all rows from the left table, even if there are no matches in the right table.

Full Outer Join: Return rows when there is a match in one of the tables.

25. What is Self-Join?

Ans. Self-join is query used to join a table to itself. Aliases should be used for the same table comparison.

26. What is Cross Join?

Ans. Cross Join will return all records where each row from the first table is combined with each row from the second table.

27. What is a view?

Ans. The views are virtual tables. Unlike tables that contain data, views simply contain queries that dynamically retrieve data when used.

28. What is a materialized view?

Ans. Materialized views are also a view but are disk based. Materialized views get updates on specific duration, base upon the interval specified in the query definition. We can index materialized view.

29. What are the advantages and disadvantages of views in a database?

Ans. Advantages:

1. Views don't store data in a physical location.
2. The view can be used to hide some of the columns from the table.
3. Views can provide Access Restriction, since data in 70 [sertion, update and deletion is not possible with the view.

Disadvantages:

1. When a table is dropped, associated view become irrelevant.
2. Since the view is created when a query requesting data from view is triggered, its a bit slow.
3. When views are created for large tables, it occupies more memory.

30. What is a stored procedure?

Ans. Stored Procedure is a function which contains a collection of SQL Queries. The procedure can take inputs, process them and send back output.

31. What are the advantages a stored procedure?

Ans. Stored Procedures are precomplied and stored in the database. This enables the database to execute the queries much faster. Since many queries can be included in a stored procedure, round trip time to execute multiple queries from source code to database and back is avoided.

32. What is a trigger?

Ans. Database triggers are sets of commands that get executed when an event(Before Insert, After Insert, On Update, On delete of a row) occurs on a table, views

33. Explain the difference between DELETE , TRUNCATE and DROP commands?

Ans. Once delete operation is performed, Commit and Rollback can be performed to retrieve data.

Once the truncate statement is executed, Commit and Rollback statement cannot be performed. Where condition can be used along with delete statement but it can't be used with truncate statement.

Drop command is used to drop the table or keys like primary, foreign from a table.

34. What is the difference between Cluster and Non cluster Index?

Ans. A clustered index reorders the way records in the table are physically stored. There can be only one clustered index per table. It makes data retrieval faster.

A non clustered index does not alter the way it was stored but creates a completely separate object within the table. As a result insert and update command will be faster.

35. What is Union, minus and Interact commands?

Ans. MINUS operator is used to return rows from the first query but not from the second query. INTERSECT operator is used to return rows returned by both the queries