SRI BALAJI CHOCKALINGAM ENGINEERING COLLEGE

A.C.S NAGAR(IRUMBEDU), ARNI,

T.V.MALAI DT.-632 317.

Department

Of

Information Technology

C3481-Database Management Systems Laboratory

SRI BALAJI CHOCKALINGAM ENGINEERING COLLEGE

A.C.S NAGAR(IRUMBEDU), ARNI, T.V.MALAI DT.-632 317

Department

of

Information Technology

BONAFIDE CERTIFICATE

*Certified that this is a bonafide record of work done by*_____

Of Second Year / IV Semester ***B.Tech Information Technology*** *in the Anna University Practical Examination during the year* ***2022- 2023*** *in* ***CS3481 DATABASE MANAGEMENT SYSTEMS Laboratory.***

Register No. :

Submitted for Practical Examination held on_____

Staff In-Charge

Head of the Department

Internal Examiner

External Examiner

EX NO: 1
DATE:

1.Create a database table, add constraints (primary key, unique, check, Not null), insert rows, update and delete rows using SQL DDL and DML commands.

Aim:

To Create a database table, add constraints (primary key, unique, check, Not null), insert rows, update and delete rows using SQL DDL and DML commands.

Description :

Step 1: Create the table with its essential attributes.

Step 2: Insert attributes values into the table.

Step 3: Execute different commands and extract information from the table.

Commands:

SQL> Create table Student(Stud_name varchar(20), Stud_id varchar2(10), Stud_dept varchar2(20), Stud_age varchar(5));

Table created.

SQL> desc Student;

Name	Null? Type
STUD_NAME	CHAR(20)
STUD_ID	VARCHAR(10)
STUD_DEPT	VARCHAR(20)
STUD_AGE	NUMBER(5)

SQL> Alter table Student add (Stud_addr varchar2 (20)); Table altered.

SQL> desc Student;

Name	Null? Type
STUD _ NAME	CHAR(20)

STUD_ID VARCHAR(10)
STUD_DEPT VARCHAR(20)
STUD_AGE VARCHAR(10)
STUD_ADDR VARCHAR(20)
SQL> Alter table Student modify (Stud_age number(10)); Table altered.

SQL> desc Student;

Name Null? Type

STUD _ NAME CHAR(20)
STUD_ID VARCHAR(10)
STUD_DEPT VARCHAR(20)
STUD_AGE NUMBER(10)
STUD_ADDR VARCHAR(20)
SQL> Alter table Student drop (Stud_age number(10)); Table altered.

SQL> desc Student;

Name Null? Type
STUD_NAME CHAR(20)
STUD_ID VARCHAR(10)
STUD_DEPT VARCHAR(20)
STUD_ADDR VARCHAR(20)
SQL> Truncate table Student;

Table truncated.

SQL> desc Student

Name Null? Type
STUD_NAME CHAR(20)
STUD_ID VARCHAR(10)
STUD_DEPT VARCHAR(20)
STUD_AGE NUMBER(10)
STUD_ADDR VARCHAR(20)

Rename
Syntax

Alter table table_name rename new_table_name

SQL> alter table student rename student1;

SQL> desc student1;
Name Null? Type
STUD_NAME CHAR(20)
STUD_ID VARCHAR(10)
STUD_DEPT VARCHAR(20)
STUD_AGE NUMBER(10)
STUD_ADDR VARCHAR(20)
SQL> Drop table

Student1; Table dropped.

SQL> desc Student1;

ERROR: ORA-04043: object Student1 does not exist

SQL> Create table Student(Stud_name char(20), Stud_id varchar2(10), Stud_ dept varchar2(20), Stud_age number(5));

Table created.
SQL> desc
Student;

Name Null? Type
STUD_NAME CHAR(20)
STUD_ID VARCHAR(10)
STUD_DEPT VARCHAR(20)
STUD_AGE NUMBER(5)
SQL> Insert into Student1 values('&stud_name', '&stud_id', '&stud_dept', '&stud_rollno');

SQL>Insert into Student1 values ('Ram', '101',

'MECH', '104') 1 row created.

SQL>Insert into Student1 values (‘Vicky’, ‘102’, ‘EEE’, ‘105’) 1 row created.

SQL>Insert into Student1 values (‘Saddiq’, ‘102’, ‘CSE’, ‘101’) 1 row created.

SQL>Insert into Student1 values (‘David’, ‘104’, ‘EEE’, ‘103’) 1 row created.

SQL> select * from Student1;

STUD_NAME	STUD_ID	STUD_DEPT	STUD_ROLLNO
Ram	101	MECH	104
Vicky	102	EEE	105
Saddiq	103	CSE	101
David	104	EEE	103

4 rows selected

SQL> Update Student1 set stud_id=’109’ where stud_name=’Saddiq’;

SQL> select * from Student1;

STUD_NAME	STUD_ID	STUD_DEPT	STUD_ROLLNO
Ram	101	MECH	104
Vicky	102	EEE	105
Saddiq	103	CSE	101
David	104	EEE	103

1 row updated.

SQL> select * from Student1;

STUD_NAME	STUD_ID	STUD_DEPT	STUD_ROLLNO
Ram	101	MECH	104
Vicky	102	EEE	105
David	104	EEE	103

SQL> create table con (empid decimal(10) not null, empname varchar(20),empsalary decimal(10), check(empsalary>10000));

SQL> insert into con values (‘1’,’kumar’,’20000’) 1 row created

SQL> insert into con values(‘2’,’raja’,’9000’)

*

ERROR at line 1:

ORA-02290: check constraint (SCOTT.SYS_C0010283) violated

Result:

Thus the commands were created and executed successfully.

EX NO: 2
DATE:

2. Create a set of tables, add foreign key constraints and incorporate referential integrity.

Aim :

To create a set of tables, add foreign key constraints and incorporate referential integrity.

Description :

Step 1: Create the table with its essential attributes.

Step 2: Insert attributes values into the table.

Step 3: Execute different commands and extract information from the table.

Commands:

mySQL> create database chinnu;

Query OK, 1 row affected (0.09 sec)

mySQL> use chinnu;

Database changed

mySQL> create table emp1(ename varchar(30)not null,eid varchar(20)not null);
Query OK, 0 rows affected (0.27 sec)

mySQL> insert into emp1 values('abcde',11);

Query OK, 1 row affected (0.12 sec)

mySQL> insert into emp1 values('fghij',12);

Query OK, 1 row affected (0.09 sec)

mySQL> insert into emp1 values('klmno',null);

ERROR 1048 (23000): Column 'eid' cannot be null

mySQL> select*from emp1;

+-------+-----+

| ename | eid |

+-------+-----+

| abcde | 11 |

| fghij | 12 |

+-------+-----+

2 rows in set (0.00 sec)

mySQL> create table depts(dname varchar(30)not null,did varchar(20)not null check(did insert into depts values('Sales',9876);

Query OK, 1 row affected (0.09 sec)

mySQL> insert into depts values('Marketing',5432);

Query OK, 1 row affected (0.09 sec)

mySQL> insert into depts values('Accounts',7532);

Query OK, 1 row affected (0.15 sec)

mySQL> select * from depts;

+-----------+------+

| dname | did |

+-----------+------+

| Sales | 9876 |

| Marketing | 5432 |

| Accounts | 7532 |

+-----------+------+

3 rows in set (0.00 sec)

mySQL> create table airports(aname varchar(30)not null,aid varchar(20)not null,acity varchar(30)check(acity in('Chennai','Hyderabad','Banglore')));

Query OK, 0 rows affected (0.39 sec)

mySQL> insert into airports values('abcde',100,'Chennai');

Query OK, 1 row affected (0.08 sec)

mySQL> insert into airports values('fghij',101,'Hyderabad');

Query OK, 1 row affected (0.16 sec)

mySQL> insert into airports values('klmno',102,'Banglore');

Query OK, 1 row affected (0.09 sec)

mySQL> insert into airports values('pqrst',103,'Mumbai');

ERROR 3819 (HY000): Check constraint 'airports_chk_1' is violated.

mySQL> select * from airports;

+-------+-----+-----------+

| aname | aid | acity |

+-------+-----+-----------+

| abcde | 100 | Chennai |

| fghij | 101 | Hyderabad |

| klmno | 102 | Banglore |

+-------+-----+-----------+

3 rows in set (0.00 sec)

mySQL> create table book(bname varchar(30)not null,bid varchar(20)not null unique);

Query OK, 0 rows affected (0.43 sec)

mySQL> insert into book values('Fairy Tales',1000);

Query OK, 1 row affected (0.10 sec)

mySQL> insert into book values('Comics',1001);

Query OK, 1 row affected (0.02 sec)

mySQL> insert into book values('Bedtime Stories',1002);

Query OK, 1 row affected (0.07 sec)

mySQL> select * from book;

+----------+------+

| bname | bid |

+----------+------+

| Fairy Tales | 1000 |

| Comics | 1001 |

| Bedtime Stories | 1002 |

+----------+------+

3 rows in set (0.00 sec)

mySQL> create table orders(oname varchar(30)not null,oid varchar(20)not null unique);

Query OK, 0 rows affected (0.18 sec)

mySQL> insert into orders values('Chair',2005);

Query OK, 1 row affected (0.09 sec)

mySQL> insert into orders values('Table',2006);

Query OK, 1 row affected (0.14 sec)

mySQL> insert into orders values('Chair',2007);

Query OK, 1 row affected (0.14 sec)

mySQL> select * from orders;

```
+-------+------+
 | oname | oid |
 +-------+------+
| Chair | 2005 |
 | Table | 2006 |
 | Chair | 2007 |
+-------+------+
```

3 rows in set (0.00 sec)

mySQL> create table custo(cname varchar(30)not null,cid varchar(20)not null primary key);

Query OK, 0 rows affected (0.25 sec)

mySQL> insert into custo values('Jones',506);

Query OK, 1 row affected (0.03 sec)

mySQL> insert into custo values('Hayden',507);

Query OK, 1 row affected (0.25 sec)

mySQL> insert into custo values('Rocky',508);

Query OK, 1 row affected (0.15 sec)

mySQL> select * from custo;

+--------+-----+

| cname | cid |

+--------+-----+

| Jones | 506 |

| Hayden | 507 |

| Rocky | 508 |

+--------+-----+

3 rows in set (0.00 sec)

mySQL> create table branches(bname varchar(20)not null,bid varchar(20)not null,primary key(bname,bid));

Query OK, 0 rows affected (0.13 sec)

mySQL> insert into branches values('Anna Nagar',1005);

Query OK, 1 row affected (0.15 sec)

mySQL> insert into branches values('Adyar',1006);

Query OK, 1 row affected (0.14 sec)

mySQL> insert into branches values('Anna Nagar',1007);

Query OK, 1 row affected (0.14 sec)

mySQL> select * from branches;

+------------+------+

| bname | bid |

+------------+------+

| Adyar | 1006 |

| Anna Nagar | 1005 |

| Anna Nagar | 1007 |

+------------+------+

3 rows in set (0.00 sec)

mySQL> create table dept(city varchar(20),dno varchar(5) primary key);

Query OK, 0 rows affected (0.22 sec)

mySQL> insert into dept values('Chennai',11);

Query OK, 1 row affected (0.05 sec)

mySQL> insert into dept values('Hyderabad',22);

Query OK, 1 row affected (0.06 sec)

mySQL> select * from dept;

+-----------+-----+

| city | dno |

+-----------+-----+

| Chennai | 11 |

| Hyderabad | 22 |

+-----------+-----+

2 rows in set (0.00 sec)

mySQL> create table semp(ename varchar(20),dno varchar(5)references dept(dno));

Query OK, 0 rows affected (0.45 sec)

mySQL> insert into semp values('x',11);

Query OK, 1 row affected (0.05 sec)

mySQL> insert into semp values('y',22);

Query OK, 1 row affected (0.13 sec)

mySQL> select * from semp;

```
+-------+------+
| ename | dno |
+-------+------+
| x | 11 |
| y | 22 |
+-------+------+
```

2 rows in set (0.00 sec)

mySQL> alter table semp add(eddress varchar(20));

Query OK, 0 rows affected (0.34 sec) Records: 0 Duplicates: 0 Warnings: 0

mySQL> update semp set eddress='10 Gandhi Road' where dno=11;

Query OK, 1 row affected (0.19 sec) Rows matched: 1 Changed: 1 Warnings: 0

mySQL> update semp set eddress='12 M.G.R Road'where dno=22;

Query OK, 1 row affected (0.08 sec) Rows matched: 1 Changed: 1 Warnings: 0

mySQL> select *from semp;

```
+-------+------+----------------+
| ename | dno | eddress |
+-------+------+----------------+
| x | 11 | 10 Gandhi Road |
| y | 22 | 12 M.G.R Road |
```

+-------+------+----------------+

2 rows in set (0.00 sec)

Result:

Thus the commands were created and executed successfully.

EX NO: 3
DATE:

3. Query the database tables using different 'where' clause conditions and also implement aggregate functions.

Aim:

To Query the database tables using different 'where' clause conditions and also implementing aggregate functions.

Description:

Step 1: Create the table with its essential attributes.

Step 2: Insert attributes values into the table.

Step 3: Execute different commands and extract information from the table.

Commands:

mySQL> create table emp(empno varchar(2),ename varchar(20),job varchar(10),deptno varchar(2),salary varchar(5));

Query OK, 0 rows affected (0.18 sec)

mySQL> insert into emp values(1,'Mathi','ASP',1,10000);

Query OK, 1 row affected (0.14 sec)

mySQL> insert into emp values(2,'Arjun','ASP',2,15000);

Query OK, 1 row affected (0.07 sec)

mySQL> insert into emp values(3,'Gugan','ASP',1,15000);

Query OK, 1 row affected (0.13 sec)

mySQL> insert into emp values(4,'Karthik','Prof',2,30000);

Query OK, 1 row affected (0.15 sec)

mySQL> insert into emp values(5,'Akalya','AP',1,10000);

Query OK, 1 row affected (0.13 sec)

mySQL> select * from emp;

```
+-------+---------+------+--------+--------+
| empno | ename | job | deptno | salary |
+-------+---------+------+--------+--------+
| 1 | Mathi | ASP | 1 | 10000 |
| 2 | Arjun | ASP | 2 | 15000 |
| 3 | Gugan | ASP | 1 | 15000 |
| 4 | Karthik | Prof | 2 | 30000 |
| 5 | Akalya | AP | 1 | 10000 |
+-------+---------+------+--------+--------+
5 rows in set (0.00 sec)
mySQL>  select count(*)from emp;
+----------+
| count(*) |
+----------+
| 5 |
+----------+
1 row in set (0.13 sec)
mySQL>  select * from emp where ename like 'A%';
+-------+--------+------+--------+--------+
| empno | ename | job | deptno | salary |
+-------+--------+------+--------+--------+
| 2 | Arjun | ASP | 2 | 15000 |
| 5 | Akalya | AP | 1 | 10000 |
```

```
+-------+--------+------+--------+--------+
 2 rows in set (0.00 sec)
 mySQL>  select * from emp where ename not like 'A%';
+-------+---------+------+--------+--------+
| empno | ename | job | deptno | salary |
+-------+---------+------+--------+--------+
| 1 | Mathi | ASP | 1 | 10000 |
| 3 | Gugan | ASP | 1 | 15000 |
| 4 | Karthik | Prof | 2 | 30000 |
+-------+---------+------+--------+--------+
3 rows in set (0.00 sec)
 mySQL>  select * from emp where salary between 15000 and 30000;
+-------+---------+------+--------+--------+
 | empno | ename | job | deptno | salary |
+-------+---------+------+--------+--------+
| 2 | Arjun | ASP | 2 | 15000 |
| 3 | Gugan | ASP | 1 | 15000 |
| 4 | Karthik | Prof | 2 | 30000 |
+-------+---------+------+--------+--------+
 3 rows in set (0.06 sec)
 mySQL>  select sum(salary),avg(salary)from emp;
 +-------------+-------------+
| sum(salary) | avg(salary) |
```

+-------------+-------------+

| 80000 | 16000 |

+-------------+-------------+

1 row in set (0.09 sec)

mySQL> select max(salary)as max_salary,min(sal) as min_salary from emp;

ERROR 1054 (42S22): Unknown column 'sal' in 'field list'

mySQL> select max(salary)as max_salary,min(salary) as min_salary from emp;

+------------+------------+

| max_salary | min_salary |

+------------+------------+

| 30000 | 10000 |

+------------+------------+

1 row in set (0.00 sec)

mySQL> select * from emp1 where ename like 'A%';

+-------+-----+

| ename | eid |

+-------+-----+

| abcde | 11 |

+-------+-----+

1 row in set (0.04 sec)

mySQL> select * from emp1 where ename not like 'A%';

+-------+-----+

| ename | eid |

```
+-------+-----+
| fghij | 12 |
+-------+-----+
```

Result:

Thus the commands were created and executed successfully.

EX NO: 4
DATE:

4.Query the database tables and explore sub queries and simple join operations.

Aim :

To Query the database tables and explore sub queries and simple join operations.

Description :

Step 1: Create the table with its essential attributes.

Step 2: Insert attributes values into the table.

Step 3: Execute different commands and extract information from the table.

Commands:

mySQL> create table Emp(Empno varchar(10),EmpName varchar(30),Salary varchar(30),address varchar(30),Result varchar(10));

Query OK, 0 rows affected (0.07 sec)

mySQL> create table dept (DEPTNO varchar(10),DName varchar(30),LOC varchar(30));

Query OK, 0 rows affected (0.07 sec)

mySQL> ALTER TABLE Emp Add (DEPTNO varchar(10));

Query OK, 0 rows affected (0.05 sec)

Records: 0 Duplicates: 0 Warnings: 0

mySQL> INSERT INTO Emp(Empno,EmpName,Job,DEPTNO,Salary)VALUES(1,"Mathi","AP",1,10000);

Query OK, 1 row affected (0.01 sec)

mySQL> INSERT INTO Emp(Empno,EmpName,Job,DEPTNO,Salary)VALUES(2,"Arjun","ASP",2,12000);

Query OK, 1 row affected (0.01 sec)

mySQL> INSERT INTO Emp(Empno,EmpName,Job,DEPTNO,Salary)VALUES(3,"Gugan","ASP",2,20000);

Query OK, 1 row affected (0.01 sec)

mySQL> INSERT INTO Emp(Empno,EmpName,Job,DEPTNO,Salary)VALUES(4,"Karthik","AP",1,15000);

Query OK, 1 row affected (0.01 sec)

mySQL> ALTER TABLE Emp DROP COLUMN Result;

Query OK, 0 rows affected (0.16 sec)

Records: 0 Duplicates: 0 Warnings: 0

mySQL> ALTER TABLE Emp DROP COLUMN address;

Query OK, 0 rows affected (0.14 sec)

Records: 0 Duplicates: 0 Warnings: 0

mySQL> SELECT* from Emp;

+-------+---------+--------+------+--------+

| Empno | EmpName | Salary | job | DEPTNO |

+-------+---------+--------+------+--------+

| 1 | Mathi | 10000 | AP | 1 |

| 2 | Arjun | 12000 | ASP | 2 |

| 3 | Gugan | 20000 | ASP | 2 |

| 4 | Karthik | 15000 | AP | 1 |

+-------+---------+--------+------+--------+

4 rows in set (0.00 sec)

mySQL> INSERT INTO dept (DEPTNO,DName,LOC)VALUES(1,"ACCOUNTING","NEWYORK");

Query OK, 1 row affected (0.01 sec)

mySQL> INSERT INTO dept (DEPTNO,DName,LOC)VALUES(1,"RESEARCH","DALLAS");

Query OK, 1 row affected (0.01 sec)

mySQL> INSERT INTO dept (DEPTNO,DName,LOC)VALUES(20.,"SALES","CHICAGO");

Query OK, 1 row affected (0.01 sec)

mySQL> INSERT INTO dept (DEPTNO,DName,LOC)VALUES(40.,"OPERATIONS","BOSTON");

Query OK, 1 row affected (0.01 sec)

mySQL> SELECT*from dept;

+--------+------------+---------+

| DEPTNO | DName | LOC |

+--------+------------+---------+

| 1 | ACCOUNTING | NEWYORK |

| 1 | RESEARCH | DALLAS |

| 20 | SALES | CHICAGO |

| 40 | OPERATIONS | BOSTON |

+--------+-----------+---------+

4 rows in set (0.00 sec)

mySQL> SELECT * from Emp where Salary>(SELECT max(Salary) from Emp where Empno=1);

+-------+---------+--------+------+--------+

| Empno | EmpName | Salary | job | DEPTNO |

+-------+---------+--------+------+--------+

| 2 | Arjun | 12000 | ASP | 2 |

| 3 | Gugan | 20000 | ASP | 2 |

| 4 | Karthik | 15000 | AP | 1 |

+-------+---------+--------+------+--------+

3 rows in set (0.00 sec)

mySQL> select*from Emp,dept where Emp.DEPTNO=dept.DEPTNO;

+-------+---------+--------+------+--------+--------+------------+---------+

| Empno | EmpName | Salary | job | DEPTNO | DEPTNO | DName | LOC |

+-------+---------+--------+------+--------+--------+------------+---------+

| 4 | Karthik | 15000 | AP | 1 | 1 | ACCOUNTING | NEWYORK |

| 1 | Mathi | 10000 | AP | 1 | 1 | ACCOUNTING | NEWYORK |

| 4 | Karthik | 15000 | AP | 1 | 1 | RESEARCH | DALLAS |

| 1 | Mathi | 10000 | AP | 1 | 1 | RESEARCH | DALLAS |

+-------+---------+--------+------+--------+--------+------------+---------+

4 rows in set (0.00 sec)

mySQL> select*from Emp,dept where Emp.DEPTNO!=dept.DEPTNO;

```
+-------+---------+--------+------+--------+--------+------------+---------+
| Empno | EmpName | Salary | job  | DEPTNO | DEPTNO | DName      | LOC     |
+-------+---------+--------+------+--------+--------+------------+---------+
| 3     | Gugan   | 20000  | ASP  | 2      | 1      | ACCOUNTING | NEWYORK |
| 2     | Arjun   | 12000  | ASP  | 2      | 1      | ACCOUNTING | NEWYORK |
| 3     | Gugan   | 20000  | ASP  | 2      | 1      | RESEARCH   | DALLAS  |
| 2     | Arjun   | 12000  | ASP  | 2      | 1      | RESEARCH   | DALLAS  |
| 4     | Karthik | 15000  | AP   | 1      | 20     | SALES      | CHICAGO |
| 3     | Gugan   | 20000  | ASP  | 2      | 20     | SALES      | CHICAGO |
| 2     | Arjun   | 12000  | ASP  | 2      | 20     | SALES      | CHICAGO |
| 1     | Mathi   | 10000  | AP   | 1      | 20     | SALES      | CHICAGO |
| 4     | Karthik | 15000  | AP   | 1      | 40     | OPERATIONS | BOSTON  |
| 3     | Gugan   | 20000  | ASP  | 2      | 40     | OPERATIONS | BOSTON  |
| 2     | Arjun   | 12000  | ASP  | 2      | 40     | OPERATIONS | BOSTON  |
| 1     | Mathi   | 10000  | AP   | 1      | 40     | OPERATIONS | BOSTON  |
+-------+---------+--------+------+--------+--------+------------+---------+
12 rows in set (0.00 sec)
```

Result:

Thus the commands were created and executed successfully.

EX NO: 5
DATE:

5. Query the database tables and explore natural, equi and outer joins.

Aim :

To Query the database tables and explore natural, equi and outer joins.

Description :

Step 1: Create the table with its essential attributes.

Step 2: Insert attributes values into the table.

Step 3: Execute different commands and extract information from the table.

Commands:

mySQL> create database modeld;

Query OK, 1 row affected (0.05 sec)

mySQL> use modeld;

Database changed

mySQL> create table student(Regno varchar(10),Name varchar(30),mark1 varchar(30),mark2 varchar(30),**Result** varchar(10));

Query OK, 0 rows affected (1.05 sec)

mySQL> insert into student values(101,'john',89,80,'pass');

Query OK, 1 row affected (0.27 sec)

mySQL> insert into student values(102,'raja',70,80,'pass');

Query OK, 1 row affected (0.08 sec)

mySQL> insert into student values(103,'sharin',70,90,'pass');

Query OK, 1 row affected (0.05 sec)

mySQL> insert into student values(104,'sam',90,95,'pass');

Query OK, 1 row affected (0.06 sec)

mySQL> select*from student;

```
+-------+--------+-------+-------+--------+
| Regno | Name   | mark1 | mark2 | Result |
+-------+--------+-------+-------+--------+
| 101   | john   | 89    | 80    | pass   |
| 102   | raja   | 70    | 80    | pass   |
| 103   | sharin | 70    | 90    | pass   |
| 104   | sam    | 90    | 95    | pass   |
+-------+--------+-------+-------+--------+
```

4 rows in set (0.00 sec)

mySQL> create table stud1(sname varchar(30),grade varchar(10));

Query OK, 0 rows affected (0.35 sec)

mySQL> insert into stud1 values('john','s');

Query OK, 1 row affected (0.06 sec)

mySQL> insert into stud1 values('raj','s');

Query OK, 1 row affected (0.05 sec)

mySQL> insert into stud1 values('sam','a');

Query OK, 1 row affected (0.07 sec)

mySQL> insert into stud1 values('sharin','a');

Query OK, 1 row affected (0.08 sec)

mySQL> insert into stud1 values('smith','null');

Query OK, 1 row affected (0.21 sec)

mySQL> select*from stud1;

+--------+-------+

| sname | grade |

+--------+-------+

| john | s |

| raj | s |

| sam | a |

| sharin | a |

| smith | null |

+--------+-------+

5 rows in set (0.00 sec)

mySQL> SELECT Name,Regno,**Result** FROM student RIGHT JOIN stud1 ON student.Name = stud1.sname;

+--------+-------+--------+

| Name | Regno | **Result** |

+--------+-------+--------+

| john | 101 | pass |

| NULL | NULL | NULL |

| sam | 104 | pass |

| sharin | 103 | pass |

| NULL | NULL | NULL |

+--------+-------+--------+

5 rows in set (0.06 sec)

mySQL> select Name,Regno,**Result** from student inner join stud1 on

student.Name = stud1.sname;

+--------+-------+--------+

| Name | Regno | **Result** |

+--------+-------+--------+

| john | 101 | pass |

| sam | 104 | pass |

| sharin | 103 | pass |

+--------+-------+--------+

3 rows in set (0.00 sec)

mySQL> select Name,Regno,**Result** from student left join stud1 on student.Name = stud1.sname;

+--------+-------+--------+

| Name | Regno | **Result** |

+--------+-------+--------+

| john | 101 | pass |

| raja | 102 | pass |

| sharin | 103 | pass |

| sam | 104 | pass |

+--------+-------+--------+

4 rows in set (0.00 sec)

mymySQL> select Name,Regno,**Result** from student natural join STUD1;

+--------+-------+--------+

| Name | Regno | **Result** |

```
+--------+-------+--------+
| sam    | 104   | pass   |
| sharin | 103   | pass   |
| raja   | 102   | pass   |
| john   | 101   | pass   |
| sam    | 104   | pass   |
| sharin | 103   | pass   |
| raja   | 102   | pass   |
| john   | 101   | pass   |
| sam    | 104   | pass   |
| sharin | 103   | pass   |
| raja   | 102   | pass   |
| john   | 101   | pass   |
| sam    | 104   | pass   |
| sharin | 103   | pass   |
| raja   | 102   | pass   |
| john   | 101   | pass   |
| sam    | 104   | pass   |
| sharin | 103   | pass   |
| raja   | 102   | pass   |
| john   | 101   | pass   |
+--------+-------+--------+
20 rows in set (0.25 sec)
```

mySQL> select distinct ename from employee x,dept y where x.deptno=y.deptno;

+---------+

| ename |

+---------+

| mathi |

| karthik |

| gugan |

| arjun |

+---------+

4 rows in set (0.26 sec)

mySQL> select distinct*from employee x where x.salary>=(select avg(salary)from employee);

+-------+---------+------+--------+--------+

| empno | ename | job | deptno | salary |

+-------+---------+------+--------+--------+

| 3 | gugan | asp | 2 | 20000 |

| 4 | karthik | ap | 2 | 15000 |

+-------+---------+------+--------+--------+

2 rows in set (0.27 sec)

Result:

Thus the commands were created and executed successfully.

EX NO: 6
DATE:

6.Write user-defined functions and stored procedure in SQL.

Aim :

To write user-defined functions and stored procedure in SQL

Description :

Step 1: Create the table with its essential attributes.

Step 2: Insert attributes values into the table.

Step 3: Execute different commands and extract information from the table.

Commands:

SQL>create table ititems(itemid number(3),actualprice number(5),ord number(4),prodid number(4));

Table Created.

SQL>insert into ititems values(101,2000,500,201);

1 row created.

SQL>insert into ititems values(102,3000,1600,202);

1 row created.

SQL>insert into ititems values(103,4000,600,202);

1 row created.

SQL>select*from ititems;

itemid	actualprice	ordid	prodid
101	2000	500	201
102	3000	1600	202

103 4000 600 202

SQL>create procedure itsum(identity number,total number)is price number;

2.null_price exception:

3.begin

4.select actualprice into price from ititems where itemid=identity;

5.if price is null then

6.raise null_price;

7.else

8.update ititems set actualprice=actualprice+total where itemid=identity;

9.end if;

10.exception

11.when null_price then

12.dbms_output.put_line('price is null');

13.end;

14. /

procedure created.

SQL>exec itsum(101,500);

procedure created successfully completed.

SQL>select*from ititems;

itemid	actualprice	ordid	prodid
101	2500	500	201
102	3000	1600	202

103 4000 600 202

SQL>set serveroutput on;

SQL>create procedure yyy(a IN number)is price number;

2.begin

3.select actualprice into price from ititems where itemid=a;

4.dbms_output.put_line('Actual price is'||price);

5.if price is null then

6.dbms_output.put_line('price is null');

7.end if;

8.end;

9. /

procedure created.

SQL>exec yyy(103);

Actual price is 4000

procedure created successfully completed.

SQL>set serveroutput on;

SQL>create procedure zzz(a in number,b out number)is identity number;

2.begin

3.select ordid into identity from ititems where itemid=a;

4.if identity<1000 then

5.b:=100;

6.end if;

7.end;

8. /

procedure created.

SQL>declare

2.a number;

3.b number;

4.begin

5.zzz(101,b);

6.dbms_output.put_line('The value of b is'||b);

7.end;

8. /

The value of b is 100

procedure successfully completed.

Result:

Thus the commands were created and executed successfully.

EX NO: 7
DATE:

7.Execute complex transactions and realize DCC AND TCL commands

Aim :

To Execute complex transactions and realize DCC AND TCL commands.

Description :

Step 1: Create the table with its essential attributes.

Step 2: Insert attributes values into the table.

Step 3: Execute different commands and extract information from the table.

Commands:

mysql> create table employee(empno varchar(10),ename varchar(30),job varchar(10),deptno varchar(10),salary varchar(30));

Query OK, 0 rows affected (0.63 sec)

mysql> insert into employee values(1,'mathi','ap',1,10000);

Query OK, 1 row affected (0.23 sec)

mysql> insert into employee values(2,'arjun','asp',2,15000);

Query OK, 1 row affected (0.10 sec)

mysql> insert into employee values(3,'gugan','asp',1,15000);

Query OK, 1 row affected (0.06 sec)

mysql> insert into employee values(4,'karthik','ap',2,30000);

Query OK, 1 row affected (0.10 sec)

```
mysql> select*from employee;
+-------+---------+------+--------+--------+
| empno | ename   | job  | deptno | salary |
+-------+---------+------+--------+--------+
| 1     | mathi   | ap   | 1      | 10000  |
| 2     | arjun   | asp  | 2      | 15000  |
| 3     | gugan   | asp  | 1      | 15000  |
| 4     | karthik | ap   | 2      | 30000  |
+-------+---------+------+--------+--------+
mysql> select user();
+----------------+
| user()         |
+----------------+
| root@localhost |
+----------------+
1 row in set (0.03 sec)
mysql> show grants for 'kavitha'@'localhost';
+-----------------------------------------+
| Grants for kavitha@localhost            |
+-----------------------------------------+
| GRANT USAGE ON *.* TO `kavitha`@`localhost` |
```

```
+-----------------------------------------+

1 row in set (0.03 sec)

mysql> grant select on *.* to 'kavitha'@'localhost';

Query OK, 0 rows affected (0.37 sec)

mysql> show grants for 'kavitha'@'localhost';

+-----------------------------------------------------------------------------+

| Grants for kavitha@localhost                                                |

+-----------------------------------------------------------------------------+

| GRANT SELECT ON *.* TO `kavitha`@`localhost`                                |

| GRANT INSERT, UPDATE (`empno`) ON `modeldb`.`employee` TO `kavitha`@`localhost` |

+-----------------------------------------------------------------------------+

2 rows in set (0.04 sec)

mysql> system mysql -u kavitha -p;

Enter password: *******

Welcome to the MySQL monitor.  Commands end with ; or \g.

Your MySQL connection id is 20

Server version: 8.0.33 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its

affiliates. Other names may be trademarks of their respective

owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

```
mysql> select user();
+------------------+
| user()           |
+------------------+
| kavitha@localhost |
+------------------+
1 row in set (0.00 sec)
mysql> use modeldb;
Database changed
mysql> insert into employee values(1,'mathi','ap',1,10000);
Query OK, 1 row affected (0.14 sec)
mysql> insert into employee values(2,'arjun','asp',2,15000);
Query OK, 1 row affected (0.21 sec)
mysql> insert into employee values(3,'gugan','asp',1,15000);
Query OK, 1 row affected (0.08 sec)
mysql> insert into employee values(4,'karthik','ap',2,30000);
Query OK, 1 row affected (0.05 sec)
mysql> select*from employee;
+-------+---------+------+--------+--------+
| empno | ename   | job  | deptno | salary |
+-------+---------+------+--------+--------+
| 1     | mathi   | ap   | 1      | 10000  |
| 2     | arjun   | asp  | 2      | 12000  |
```

| 3 | gugan | asp | 2 | 20000 |

| 4 | karthik | ap | 2 | 15000 |

| 1 | mathi | ap | 1 | 10000 |

| 2 | arjun | asp | 2 | 15000 |

| 3 | gugan | asp | 1 | 15000 |

| 4 | karthik | ap | 2 | 30000 |

+-------+---------+------+--------+--------+

8 rows in set (0.00 sec)

mysql> update employee set salary=25000 where ename="mathi";

Query OK, 2 rows affected (0.16 sec)

Rows matched: 2 Changed: 2 Warnings: 0

mysql> update employee set salary=30000 where ename="arjun";

Query OK, 2 rows affected (0.06 sec)

Rows matched: 2 Changed: 2 Warnings: 0

mysql> select*from employee;

+-------+---------+------+--------+--------+

| empno | ename | job | deptno | salary |

+-------+---------+------+--------+--------+

| 1 | mathi | ap | 1 | 25000 |

| 2 | arjun | asp | 2 | 30000 |

| 3 | gugan | asp | 2 | 20000 |

| 4 | karthik | ap | 2 | 15000 |

| 1 | mathi | ap | 1 | 25000 |

| 2 | arjun | asp | 2 | 30000 |

| 3 | gugan | asp | 1 | 15000 |

| 4 | karthik | ap | 2 | 30000 |

+-------+---------+------+--------+--------+

8 rows in set (0.00 sec)

mysql> revoke insert on employee from 'kavitha'@'localhost';

Query OK, 0 rows affected (0.09 sec)

mysql> insert into employee values(1,'mathi','asp',1,20000);

ERROR 1142 (42000): INSERT command denied to user 'kavitha'@'localhost' for table 'employee'

mysql> use kavi2db;

Database changed

mysql> create table employees(empno varchar(10),ename varchar(30),job varchar(10),deptno varchar(10),salary varchar(30));

Query OK, 0 rows affected (0.63 sec)

mysql> insert into employees values(1,'mathi','ap',1,10000);

Query OK, 1 row affected (0.23 sec)

mysql> insert into employees values(2,'arjun','asp',2,15000);

Query OK, 1 row affected (0.10 sec)

mysql> insert into employees values(3,'gugan','asp',1,15000);

Query OK, 1 row affected (0.06 sec)

mysql> insert into employees values(4,'karthik','ap',2,30000);

Query OK, 1 row affected (0.10 sec)

```
mysql> select*from employees;
+-------+---------+------+--------+--------+
| empno | ename   | job  | deptno | salary |
+-------+---------+------+--------+--------+
| 1     | mathi   | ap   | 1      | 10000  |
| 2     | arjun   | asp  | 2      | 15000  |
| 3     | gugan   | asp  | 1      | 15000  |
| 4     | karthik | ap   | 2      | 30000  |
+-------+---------+------+--------+--------+
4 rows in set (0.00 sec)
mysql> start transaction;
Query OK, 0 rows affected (0.14 sec)
mysql> insert into employees values(5,'akalya','asp',1,10000);
Query OK, 1 row affected (0.00 sec)
mysql> select*from employees;
+-------+---------+------+--------+--------+
| empno | ename   | job  | deptno | salary |
+-------+---------+------+--------+--------+
| 1     | mathi   | ap   | 1      | 10000  |
| 2     | arjun   | asp  | 2      | 30000  |
| 3     | gugan   | asp  | 1      | 15000  |
| 4     | karthik | ap   | 2      | 30000  |
| 5     | akalya  | asp  | 1      | 10000  |
```

```
+-------+---------+------+--------+--------+
5 rows in set (0.00 sec)
mysql> savepoint initial;
Query OK, 0 rows affected (0.00 sec)
mysql> insert into employees values(6,'ram','ap',2,20000);
Query OK, 1 row affected (0.00 sec)
mysql> select*from employees;
+-------+---------+------+--------+--------+
| empno | ename   | job  | deptno | salary |
+-------+---------+------+--------+--------+
| 1     | mathi   | ap   | 1      | 10000  |
| 2     | arjun   | asp  | 2      | 30000  |
| 3     | gugan   | asp  | 1      | 15000  |
| 4     | karthik | ap   | 2      | 30000  |
| 5     | akalya  | asp  | 1      | 10000  |
| 6     | ram     | ap   | 2      | 20000  |
+-------+---------+------+--------+--------+
6 rows in set (0.00 sec)
mysql> savepoint name;
Query OK, 0 rows affected (0.00 sec)
mysql> delete from employees where empno=2;
Query OK, 1 row affected (0.04 sec)
mysql> select*from employees;
```

```
+-------+---------+------+--------+--------+
| empno | ename   | job  | deptno | salary |
+-------+---------+------+--------+--------+
| 1     | mathi   | ap   | 1      | 10000  |
| 3     | gugan   | asp  | 1      | 15000  |
| 4     | karthik | ap   | 2      | 30000  |
| 5     | akalya  | asp  | 1      | 10000  |
| 6     | ram     | ap   | 2      | 20000  |
+-------+---------+------+--------+--------+
5 rows in set (0.00 sec)
mysql> savepoint s1;
Query OK, 0 rows affected (0.00 sec)
mysql> rollback to name;
Query OK, 0 rows affected (0.00 sec)
mysql> select*from employees;
+-------+---------+------+--------+--------+
| empno | ename   | job  | deptno | salary |
+-------+---------+------+--------+--------+
| 1     | mathi   | ap   | 1      | 10000  |
| 2     | arjun   | asp  | 2      | 30000  |
| 3     | gugan   | asp  | 1      | 15000  |
| 4     | karthik | ap   | 2      | 30000  |
| 5     | akalya  | asp  | 1      | 10000  |
```

```
| 6    | ram     | ap   | 2      | 20000  |
+-------+---------+------+--------+--------+
6 rows in set (0.00 sec)
mysql> commit;
Query OK, 0 rows affected (0.10 sec)
mysql> select*from employees;
+-------+---------+------+--------+--------+
| empno | ename   | job  | deptno | salary |
+-------+---------+------+--------+--------+
| 1    | mathi   | ap   | 1      | 10000  |
| 2    | arjun   | asp  | 2      | 30000  |
| 3    | gugan   | asp  | 1      | 15000  |
| 4    | karthik | ap   | 2      | 30000  |
| 5    | akalya  | asp  | 1      | 10000  |
| 6    | ram     | ap   | 2      | 20000  |
+-------+---------+------+--------+--------+
6 rows in set (0.00 sec)
```

Result:

Thus the commands were created and executed successfully.

EX NO: 8
DATE:

8.Write SQL Triggers for INSERT ,DELETE, AND UPDATE Operations in a Database table.

Aim :

To write SQL Triggers for INSERT ,DELETE, AND UPDATE Operations in a Database table.

Description :

Step 1: Create the table with its essential attributes.

Step 2: Insert attributes values into the table.

Step 3: Execute different commands and extract information from the table.

Commands:

```
SQL> create table itempls(name varchar(10),id number(5),salary number(30));

Table created.

SQL> insert into itempls values('aaa',14,34000);

1 row created.

SQL> insert into itempls values('bbb',15,20000);

1 row created.

SQL> insert into itempls values('ccc',16,30000);

1 row created.

SQL> select*from itempls;

NAME            ID     SALARY

---------- ---------- ----------

aaa             14      34000
```

```
bbb              15      20000
ccc              16      30000
SQL>   create trigger ittrigg1 before insert or update or delete on itempls for each
row
  2  begin
  3  raise_application_error(-20010,'You cannot do manipulation');
  4  end;
  5  /
Trigger created.
SQL>  insert into itempls values('ddd',17,40000);
insert into itempls values('ddd',17,40000)
            *
ERROR at line 1:
ORA-04098: trigger 'SYSTEM.ITTRIGG' is invalid and failed re-validation
SQL>  delete from itempls where name='aaa';
delete from itempls where name='aaa'
            *
ERROR at line 1:
ORA-04098: trigger 'SYSTEM.ITTRIGG' is invalid and failed re-validation
SQL>  update itempls set id=15 where name='bbb';
update itempls set id=15 where name='bbb'
```

```
      *
ERROR at line 1:
ORA-04098: trigger 'SYSTEM.ITTRIGG' is invalid and failed re-validation
SQL>  drop trigger ittrigg1;
Trigger dropped.
SQL>  create trigger ittriggs4 before insert or update of salary on itempls for each
row
  2   declare
  3   triggsal itempls.salary%type;
  4  begin
  5  select salary into triggsal from itempls where id=14;
  6  if(:new.salary>triggsal or :new.salary<triggsal)then
  7  raise_application_error(-20100,'Salary has not been changed');
  8  end if;
  9  end;
 10  /
Trigger created.
SQL>  insert into itempls values('eee',18,50000);
insert into itempls values('eee',18,50000)
            *ERROR at line 1:
ORA-04098: trigger 'SYSTEM.ITTRIGG' is invalid and failed re-validation
SQL>  update itempls set id=16 where name='ccc';
update itempls set id=16 where name='ccc'
```

*

ERROR at line 1:

ORA-04098: trigger 'SYSTEM.ITTRIGG' is invalid and failed re-validation

Result:

Thus the commands were created and executed successfully.

EX NO: 9
DATE:

9.Create view and Index for Database table with a large number of records

Aim :

To Create view and Index for Database table with a large number of records.

Description :

Step 1: Create the table with its essential attributes.

Step 2: Insert attributes values into the table.

Step 3: Execute different commands and extract information from the table.

Commands:

mysql> create table employees(empno varchar(10),ename varchar(30),job varchar(10),deptno varchar(10),salary varchar(30));

Query OK, 0 rows affected (0.63 sec)

mysql> insert into employees values(1,'mathi','ap',1,10000);

Query OK, 1 row affected (0.23 sec)

mysql> insert into employees values(2,'arjun','asp',2,15000);

Query OK, 1 row affected (0.10 sec)

mysql> insert into employees values(3,'gugan','asp',1,15000);

Query OK, 1 row affected (0.06 sec)

mysql> insert into employees values(4,'karthik','ap',2,30000);

Query OK, 1 row affected (0.10 sec)

mysql> insert into employees values(5,'akalya','asp',1,10000);

Query OK, 1 row affected (0.00 sec)

mysql> insert into employees values(6,'ram','ap',2,20000);

Query OK, 1 row affected (0.00 sec)

mysql> select*from employees;

```
+-------+---------+------+--------+--------+
| empno | ename   | job  | deptno | salary |
+-------+---------+------+--------+--------+
| 1     | mathi   | ap   | 1      | 10000  |
| 2     | arjun   | asp  | 2      | 30000  |
| 3     | gugan   | asp  | 1      | 15000  |
| 4     | karthik | ap   | 2      | 30000  |
| 5     | akalya  | asp  | 1      | 10000  |
| 6     | ram     | ap   | 2      | 20000  |
+-------+---------+------+--------+--------+
```

6 rows in set (0.00 sec)

mysql> create view empview as select*from employees where job='asp';

Query OK, 0 rows affected (0.30 sec)

mysql> select*from empview;

```
+-------+--------+------+--------+--------+
| empno | ename  | job  | deptno | salary |
+-------+--------+------+--------+--------+
| 2     | arjun  | asp  | 2      | 30000  |
| 3     | gugan  | asp  | 1      | 15000  |
| 5     | akalya | asp  | 1      | 10000  |
+-------+--------+------+--------+--------+
```

3 rows in set (0.14 sec)

mysql> create view empview as select*from employees where job='asp';

Query OK, 0 rows affected (0.30 sec)

mysql> select*from empview;

```
+-------+--------+------+--------+--------+
| empno | ename  | job  | deptno | salary |
+-------+--------+------+--------+--------+
| 2     | arjun  | asp  | 2      | 30000  |
| 3     | gugan  | asp  | 1      | 15000  |
| 5     | akalya | asp  | 1      | 10000  |
+-------+--------+------+--------+--------+
```

3 rows in set (0.14 sec)

mysql> create view empview1 as select ename from employees;

Query OK, 0 rows affected (0.32 sec)

mysql> select*from empview1;

```
+---------+
| ename   |
+---------+
| mathi   |
| arjun   |
| gugan   |
| karthik |
| akalya  |
```

| ram |

+---------+

6 rows in set (0.00 sec)

mysql> create view empview2 as select salary from employees;

Query OK, 0 rows affected (0.13 sec)

mysql> select*from empview2;

+--------+

| salary |

+--------+

| 10000 |

| 30000 |

| 15000 |

| 30000 |

| 10000 |

| 20000 |

+--------+

6 rows in set (0.00 sec)

mysql> drop view empview1;

Query OK, 0 rows affected (0.37 sec)

mysql> select*from empview1;

ERROR 1146 (42S02): Table 'kavi2db.empview1' doesn't exist

mysql> create table splr(sname varchar(10),sid varchar(10),scity varchar(20));

Query OK, 0 rows affected (0.87 sec)

mysql> insert into splr values('hcl','01','chennai');

Query OK, 1 row affected (0.29 sec)

mysql> insert into splr values('dell','04','madurai');

Query OK, 1 row affected (0.29 sec)

mysql> insert into splr values('hp','02','kovai');

Query OK, 1 row affected (0.28 sec)

mysql> insert into splr values('lenovo','03','trichy');

Query OK, 1 row affected (0.07 sec)

mysql> select*from splr;

```
+--------+------+---------+
| sname  | sid  | scity   |
+--------+------+---------+
| hcl    | 01   | chennai |
| dell   | 04   | madurai |
| hp     | 02   | kovai   |
| lenovo | 03   | trichy  |
+--------+------+---------+
```

4 rows in set (0.00 sec)

mysql> create index sp on splr(sid);

Query OK, 0 rows affected, 1 warning (0.46 sec)

Records: 0 Duplicates: 0 Warnings: 1

mysql> create index sp3 on splr(sname,sid);

Query OK, 0 rows affected, 1 warning (0.20 sec)

Records: 0 Duplicates: 0 Warnings: 1

mysql> drop index sp on splr;

Query OK, 0 rows affected (0.31 sec)

Records: 0 Duplicates: 0 Warnings: 0

mysql> drop index sp3 on splr;

Query OK, 0 rows affected (0.10 sec)

Records: 0 Duplicates: 0 Warnings: 0

Result:

Thus the commands were created and executed successfully.

EX NO: 10
DATE:

10.XML database and validate it using XML schema Aim

Creating XML database and validate it using XML schema.

Procedure

XML

- Xml (eXtensible Markup Language) is a mark up language.
- XML is designed to store and transport data.
- Xml was released in late 90's. it was created to provide an easy to use and store self describing data.
- XML became a W3C Recommendation on February 10, 1998.
- XML is not a replacement for HTML.
- XML is designed to be self-descriptive.
- XML is designed to carry data, not to display data.
- XML tags are not predefined. You must define your own tags. XML is platform independent and language independent.

XML Schema

XML Schema is commonly known as XML Schema Definition (XSD). It is used to describe and validate the structure and the content of XML data. XML schema defines the elements, attributes and data types. Schema element supports Namespaces. It is similar to a database schema that describes the data in a database.

How to validate XML against XSD in java:

Java XML Validation API can be used to validate XML against an XSD. javax.xml.validation.Validator class is used in this program to validate xml file against xsd file.Here are the sample XSD and XML files used.

Employee.xsd

```
<?xml version="1.0" encoding="UTF-8"?>

<schema xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.journaldev.com/Employee"

xmlns:empns="http://www.journaldev.com/Employee"
elementFormDefault="qualified">
```

<element name="empRequest" type="empns:empRequest"></element>

<element name="empResponse" type="empns:empResponse"></element>

<complexType name="empRequest">

<sequence>

<element name="id" type="int"></element>

</sequence>

</complexType>

<complexType name="empResponse">

<sequence>

<element name="id" type="int"></element>

<element name="role" type="string"></elemen>

<element name="fullName" type="string"></element>

</sequence>

</complexType
> </schema>

Notice that above XSD contains two root element and namespace also, I have created two sample XML file from XSD.

Employee Request.xml

<?xml version="1.0" encoding="UTF-8"?>

<empns:empRequest xmlns:empns="http://www.journaldev.com/Employee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.journaldev.com/Employee Employee.xsd ">

EmployeeResponse.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<empns:empResponse xmlns:empns="http://www.journaldev.com/Employee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://www.journaldev.com/Employee Employee.xsd ">

<empns:id>1</empns:id>

<empns:role>Developer</empns:role>

<empns:fullName>Pankaj Kumar</empns:fullName>

</empns:empResponse>
```

Here is another XML file that doesn't confirms to the Employee.xsd

employee.xml

```
<?xml version="1.0"?>

<Employee>

<name>Pankaj</name>

<age>29</age>

<role>Java Developer</role>

<gender>Male</gender>

</Employee>
```

Here is the program that is used to validate all three XML files against the XSD. The validateXMLSchema method takes XSD and XML file as argument and return true if validation is successful or else returns false.

XMLValidation.java

```
package com.journaldev.xml;

import java.io.File;

import java.io.IOException;
```

```
javax.xml.XMLConstants; import
javax.xml.transform.stream.Stream
Source; import
javax.xml.validation.Schema;
import
javax.xml.validation.SchemaFactor
y; import
javax.xml.validation.Validator;
import org.xml.sax.SAXException;
public class XMLValidation {
public static void main(String[]
args) {

System.out.println("EmployeeRequest.xml validates against Employee.xsd? "+validateXMLSchema("Employee.xsd", "EmployeeRequest.xml"));
System.out.println("EmployeeResponse.xml validates against Employee.xsd? "+validateXMLSchema("Employee.xsd", "EmployeeResponse.xml"));
System.out.println("employee.xml validates against Employee.xsd?

"+validateXMLSchema("Employee.xsd", "employee.xml"));

}

public static boolean validateXMLSchema(String xsdPath,
String xmlPath){ try {

SchemaFactory factory =

SchemaFactory.newInstance(XMLConstants.W3C_XML_SCHEMA_NS_URI);

Schema schema = factory.newSchema(new
File(xsdPath)); Validator validator =
schema.newValidator(); validator.validate(new
StreamSource(new File(xmlPath)));

} catch (IOException | SAXException e) {
```

```
System.out.println("Exception:
"+e.getMessage()); return false; }
return true;

}

}
```

Output of the above program is:

EmployeeRequest.xml validates against Employee.xsd? true

EmployeeResponse.xml validates against Employee.xsd? true Exception: cvc-elt.1: Cannot find the declaration of element 'Employee'. employee.xml validates against Employee.xsd? false

RESULT:

Thus the XML database created and validates it using XML schema.

EX NO: 11
DATE:

11. NoSQL Database Tools

Aim:

To create the document, columns and graphs based on data by using NoSQL tools.

Procedure :

Document database

A document database (also known as a document-oriented database or a document store) is a database that stores information in documents.

A document is a record in a document database. A document typically stores information about one object and any of its related metadata.

Documents store data in field-value pairs. The values can be a variety of types and structures, including strings, numbers, dates, arrays, or objects. Documents can be stored in formats like JSON, BSON, and XML.
Below is a JSON document that stores information about a user named Tom.

```
{
    "_id": 1,
    "first_name": "Tom",
    "email": "tom@example.com",
    "cell": "765-555-5555",
    "likes": [
      "fashion",
      "spas",
      "shopping"
    ],
    "businesses": [
      {
```

```
      "name": "Entertainment 1080",
      "partner": "Jean",
      "status": "Bankrupt",
      "date_founded": {
        "$date": "2012-05-19T04:00:00Z"
      }
    },
    {
      "name": "Swag for Tweens",
      "date_founded": {
        "$date": "2012-11-01T04:00:00Z"
      }
    }
  ]
}
```

Collections

A collection is a group of documents. Collections typically store documents that have similar contents.

Continuing with the example above, the document with information about Tom could be stored in a collection named users. More documents could be added to the users collection in order to store information about other users. For example, the document below that stores information about Donna could be added to the users collection.

```
{
  "_id": 2,
```

```
  "first_name": "Donna",
  "email": "donna@example.com",
  "spouse": "Joe",
  "likes
    ":    [
    "spa
    s",
    "shopping",
    "live tweeting"
  ],
  "businesses": [
    {
      "name": "Castle Realty",
      "status": "Thriving",
      "date_founded": {
        "$date": "2013-11-21T04:00:00Z"
      }
    }
  ]
}
```


Columnar Data Model of NoSQL :

In Columnar Data Model instead of organizing information into rows, it does in columns. This makes them function the same way that tables work in relational

databases. This type of data model is much more flexible obviously because it is a type of NoSQL database.
The below example will help in understanding the Columnar data model:

Row-Oriented Table:

S.No.	Name	Course	Branch	ID
01.	Tanmay	B-Tech	Computer	2
02.	Abhishek	B-Tech	Electronics	5
03.	Samriddha	B-Tech	IT	7
S.No.	**Name**	**Course**	**Branch**	**ID**
04.	Aditi	B-Tech	E & TC	8

Column – Oriented Table:

S.No.	Name	ID
01.	Tanmay	2
02.	Abhishek	5
03.	Samriddha	7
04.	Aditi	8

Graph Database on NoSQL

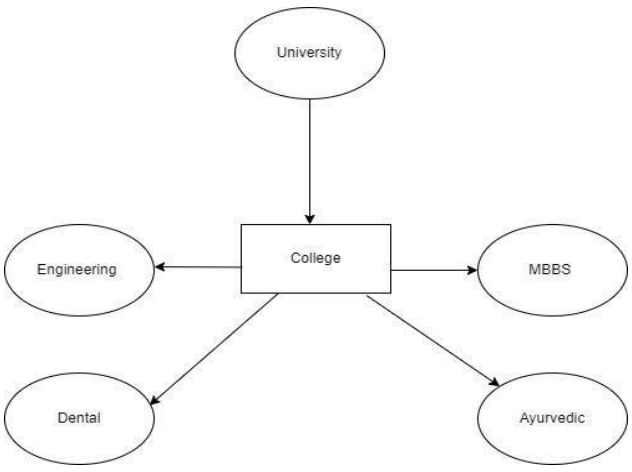
Graph Based Data Model in NoSQL is a type of Data Model which tries to focus on building the relationship between data elements. As the name suggests Graph-Based Data Model, each element here is stored as a node, and the association between these elements is often known as Links. Association is stored directly as these are the first-class elements of the data model. These data models give us a conceptual view of the data.

These are the data models which are based on topographical network structure. Obviously, in graph theory, we have terms like Nodes, edges, and properties, let's see what it means here in the Graph-Based data model.

Nodes: These are the instances of data that represent objects which is to be tracked.

Edges: As we already know edges represent relationships between nodes. **Properties:** It represents information associated with nodes.

The below image represents Nodes with properties from relationships represented by edges.



Result :

Thus we studied the various NoSQL database tools to create document, column and graph successfully.

EX NO: 12
DATE:

12. Simple GUI Application (Displaying Student Mark List) Aim:

Write a program in Java to create Displaying student mark listusing JSP and Databases (three tire architecture).

Definition, usage and procedure

Three tier architecture is a very common architecture. A three tier architecture is typically split into a presentation or GUI tier, an application logic tier, and a data tier.

Presentation tier encapsulates the presentation logic required to serve clients. A JSP in the presentation tier intercepts client requests, manages logons, sessions, accesses the business services, and finally constructs a response, which gets delivered to client.

Business tier provides the business services. This tier contains the business logic and the business data. All the business logic is centralized into this tier as opposed to 2-tier systems where the business logic is scattered between the front end and the backend. The benefit of having a centralized business tier is that same business logic can support different types of clients like browser, WAP (Wireless Application Protocol) client. In our exercise we will use servlet as business tier.

Data Tier

Data tier is used by the databases

JSP

Java Server Pages (JSP) is a server-side programming technology that enables the creation of dynamic, platform-independent method for building Web-based applications. JSP have access to the entire family of Java APIs, including the JDBC API to access enterprise databases

Servlet

A **servlet** is a small Java program that runs within a Web server. **Servlets** receive and respond to requests from Web clients, usually across HTTP, the HyperText Transfer Protocol

Client:

Step1: In index.html on the client side declare the contents that you like to transfer to the server using html form and input type tags.
Step2: create a submit button and close all the included tags.
Servlet:
Step 1: Import all necessary packages

Step 2: Define a class that extends servlet

Step 3: In the doPost() method, do the following:

i) Set the content type of the response to "text/html" ii) connect with the database which has the student marklist

iii) query the data to the database

Step 4: Display the student marklist

First Create database as **db8** in that create table as **mark** with the following field

create table mark(rno varchar(20),name1 varchar(20),m1 varchar(20),m2 varchar(20),m3 varchar(20),m4 varchar(20),m5 varchar(20),m6 varchar(20)) insert into mark values('100','mohammed','90','90','90','90','90','90')

select * from mark

index.html

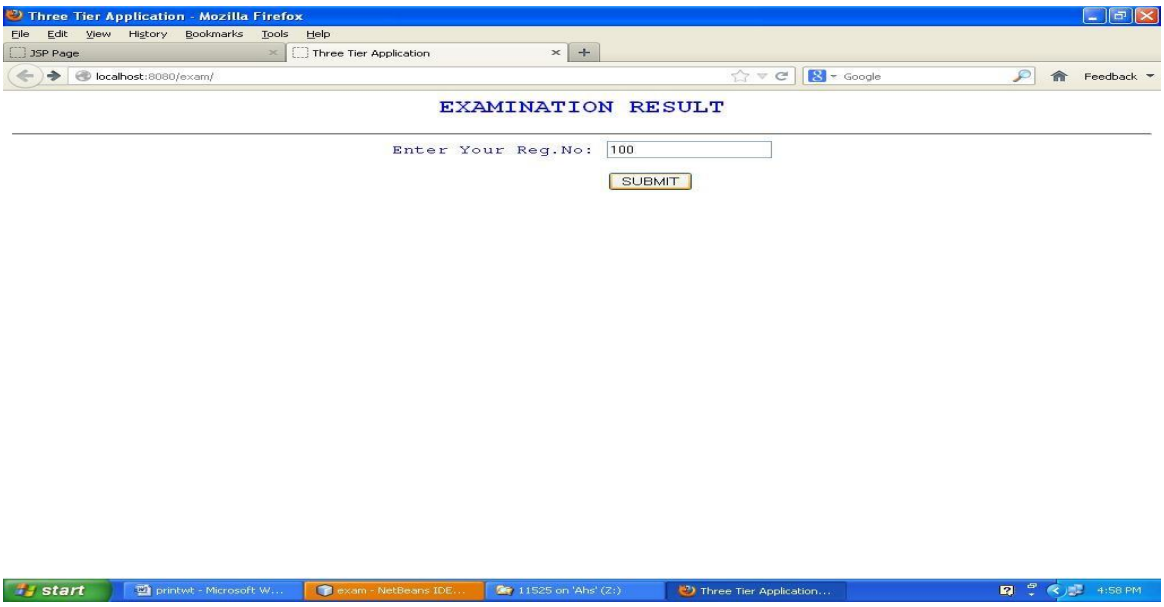
```
<head>

<title>Three Tier Application</title>

<style type="text/css">

body{color:blue;font-family:courier;text-align:center}

</style
>
</head
>

<body>
```

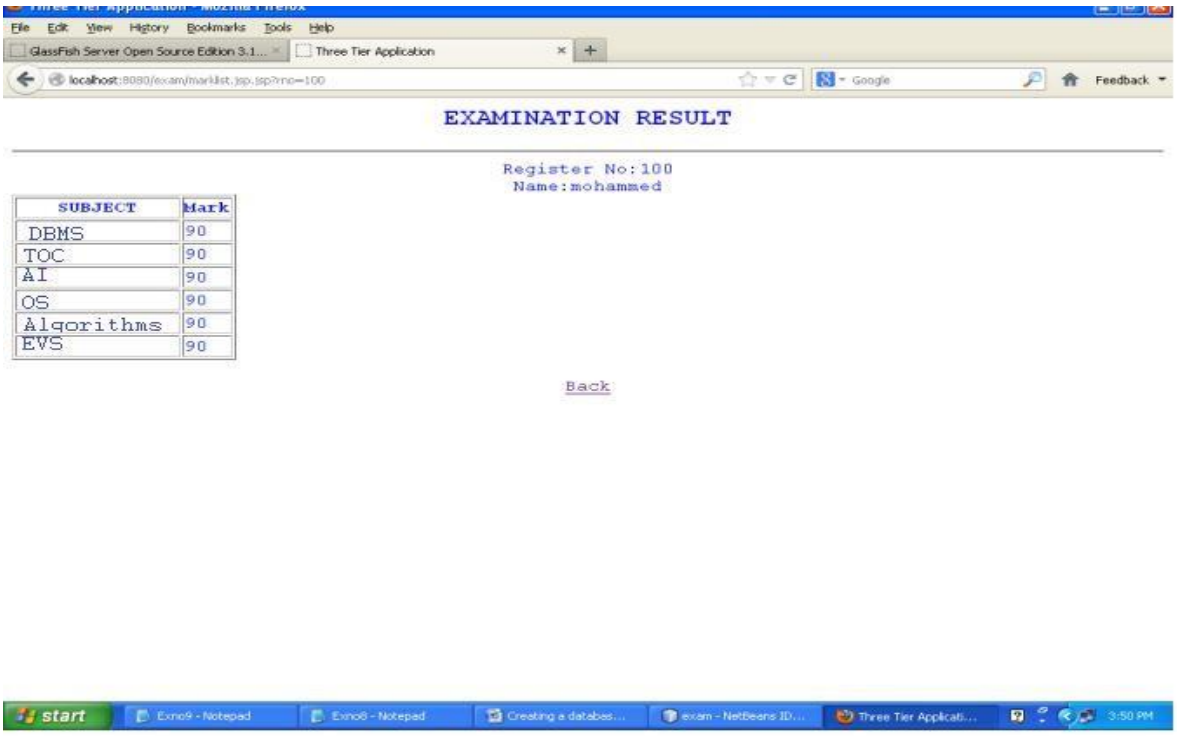
```
<h2>EXAMINATION RESULT</h2><hr/>

<form name="f1" method="GET"
action="marklist.jsp"> Enter Your Reg.No:

<input type="text" name="rno"/><br/><br/>

      

    

<input type="submit" value="SUBMIT"/>

</form>

</body>

</html>

<html>

Marklist.jsp

<%@page import="java.util.Properties"%>

<%@page contentType="text/html" language="java" import="java.sql.*"%>

<html>

<head>

<title>Three Tier Application</title>

<style type="text/css">

body{color:blue;font-family:courier;text-align:center}

</style
>
</head
>

<body>
```

```
<h2>EXAMINATION RESULT</h2><hr/>

<%

String str=request.getParameter("rno");

Class.forName("org.apache.derby.jdbc.
ClientDriver"); Properties p=new
Properties(); p.put("user","root");

p.put("password","root");

Connection
con=DriverManager.getConnection("jdbc:derby://localhost:1527/db8",p);

Statement stmt=con.createStatement();

ResultSet rs=stmt.executeQuery( " Select * FROM mark WHERE rno
='"+str+"'"); while(rs.next())

{

%>

Register No:<%=rs.getString(1)%><br/>

Name:<%=rs.getString(2)%><br/>

<table border="1">

<th>SUBJECT</th><th>Mark</th>

<tr><td>DBMS</td><td><%=rs.getString(3)%></td></tr>

<tr><td>TOC</td><td><%=rs.getString(4)%></td></tr>

<tr><td>AI</td><td><%=rs.getString(5)%></td></tr>

<tr><td>OS</td><td><%=rs.getString(6)%></td></tr>

<tr><td>Algorithms</td><td><%=rs.getString(7)%></td></tr>

<tr><td>EVS</td><td><%=rs.getString(8)%></td></tr>
```

```
</table>
<%
}
%>
<br/>
<a href="index.jsp">Back</a>
</body>
</html>
```

Output





Result :

Thus the program is created for displaying examination result and successfully verified

EX NO: 13
DATE:

13. Case Study of Cop Friendly App – Eseva

Aim:

To case Study of Cop Friendly APP –Eseva

About Case study

The Case Study is all about Smart Cop, a mobile application or tool developed by Sapio Analytics along with Dinosys Infotech that places a strong emphasis on combining existing interdisciplinary information, improving analysis techniques, and developing more efficient police strategies. For example, the DataDriven Approach to Crime Scene Management is a technology that specializes in handling evidence collected at a crime scene, and its notification feature aids in pulling data from various media. Escape routes and the location of the incident could also be tracked. Also, describes how the police can effectively manage their limited resources with this new data-driven and AI-based policing

Purpose of this Case Study

The purpose of the case study is to demonstrate how using the SMART COP tool developed by Sapio Analytics along with Dinosys Infotech when used on a daily basis in policing can help police agencies to figure out what works in crime reduction and crime prevention initiatives, as well as lead to much more effective decisions, faster actions, which can lead to better policing, higher impact on citizens, and improved citizen satisfaction. It's to demonstrate the value of such efficacious, efficient, and cost effective law enforcement strategies and tactics based on data and analytics, especially when it's aided by Sapio Analytics' exclusive systems.

Background of the Product

Sapio Analytics along with Dinosys Infotech is providing training consultations and analytical solutions backed by Artificial Intelligence to the enforcement bodies in matters related to cyber-crime, suspect profiling, predictive policing, and so on. The Smart Cop tool developed by Sapio Analytics along with Dinosys is an essential guarantee to modernize the police to manage local security. It will also emphasize the new exigencies of the police management system. It can likewise be utilized for mappingcrime which can monitor high-crime locations. With this, police can monitor very closely and have a real-time pulse on criminal activities. The mounted Smart Cop App will enhance the balance, security, sustenance, and scalability of the existing department. The application facilitates an integrated policing management system along with an information processing platform that

will empower frontline Police Officers who are on a beat towards achieving Global Policing Standards

Problem Statement

India is known as the world's largest democratic country, little is known about how it polices such a vast, complex, and unpredictable country. As a result, police officers encounter challenges and barriers in carrying out their duties on a daily basis. Some of the problems faced are

The police leadership has not placed a high priority on using technology to deliver services to citizens.

- Investigations are being delayed due to a lack of collaboration between internal divisions.
- The training standards are quite inadequate and do not account for the use of new technology.
- There is a significant disparity between the rate at which crimes are committed and the rate at which FIRs are filed.
- The workload is one of the key causes of police inefficiency once again.

Solution to the Problem

The fundamental function of police forces is to uphold and investigate crimes, safeguard the safety of citizens, and enforce laws. In a large and populous country like India, police forces must be well equipped to fulfill their duties effectively. As a result, the police force must adjust to changing conditions. Police modernization has been needed for data protection, counterterrorism/insurgency, and reliance on technology for policing. This necessitates more investment in technological advancement and modernization. So, to solve the problem in a modern way Sapio Analytics Came up with Smart Cop which aims to train the law enforcement agencies on emerging technology, mitigating cybercrimes, enhancing their skill set as well as capacity building so that they are combatready in a real-time basis. The Smart Cop is an essential guarantee to modernize the police to manage local security. It will also emphasize the new exigencies of the police management system.

Benefits of the App

The main aim of Smart Cop is to digitize the whole functioning of beats. In the due process, we facilitated the beat police with one application and have integrated the IT applications and databases which are in line with the beat system, and converged them on to Smart Cop

- Effectively utilizing Information and Communication Technology (ICT) traversing from EGovernance to M-Governance
- Empowering and delegating the frontline Beat Police Officers for demonstrating quick & smart decision making
- Delivering Services 'Anytime & Anywhere' for faster response to the Citizens Seamless
  integration of different application functionalities through Single-Sign-On services

- Efficient identification and tracking of suspects, quicker resolution of cases, and increased rate of convictions for the offenders
- Proactively preventing crimes through real-time intelligence inputs and analysis relating to crimes and criminals
- Encouraging transparency and accountability in every police personnel

Conclusive Summary

The case study shows how as the rate of crime rises; the utilization of existing Artificial Intelligence algorithms is proving to be extremely beneficial. The tool developed by Sapio Analytics along with Dinosys Infotech- SMART COP sets a promising example. To a considerable extent, Smart Cop aids in the prediction of crime as well as the criminal. Artificial intelligence has the potential to become a permanent element of the criminal justice system. Technological reforms are required to accomplish the vision of SMART policing, it's important to train the police for new challenges, and strengthen their investigative and emergency response capabilities. This will eventually increase public confidence in the police force's effectiveness and its ability to serve efficiently. The police force must be eager to bring a change and adopt new-age technologies and systems into the realm of law enforcement for it to be more proactive than reactive.

INVENTORY MANAGEMENT SYSTEM PROBLEM STATEMENT:

INVENTORY MANAGEMENT SYSTEM is a realtimeapplication used in the merchant’s day to day system. This is a database to store the transaction that takes places between the Manufacturer, Dealer and the Shop Keeper that includes stock inward and stock outward with reference to the dealer. Here we assume our self as the Dealer and proceed with the transaction as follows:

The Manufacturer is the producer of the items and it contains the necessary information
of the item such as price per item, Date of manufacture, best before use, Number of Item available and their Company Address.

The Dealer is the secondary source of an Item and he purchases Item from the manufacturer by requesting the required Item with its corresponding Company Name and the Number of Items required. The Dealer is only responsible for distribution of the Item to the Retailers in the Town or City.

The Shop Keeper or Retailer is the one who is prime source for selling items in the market. The customers get Item from the Shop Keeper and not directly from the Manufacturer or the Dealer.

The Stock is the database used in our System which records all transactions that takes
place between the Manufacturer and the Dealer and the Dealer and the Retailer. **CODING:**

FORM1

```
Dim db As Database

Dim rs As Recordset

Private Sub Command1_Click()

Form3.Show

End Sub

Private Sub Command2_Click()

Form4.Show

End Sub

Private Sub Command3_Click()

Form5.Show
```

```
End Sub

Private Sub Command4_Click()

End

End Sub

Private Sub Form_Load()

Set db = OpenDatabase("D:\prj789\invent\INVENTORY.MDB")

Set rs = db.OpenRecordset("SYSTEM")

End Sub
```

FORM2

```
Dim db As Database Dim rs As Recordset
Dim i As Integer

Private Sub Command1_Click()

Form1.Show

End Sub

Private Sub Command2_Click()
rs.MoveFirst

For i = 1 To rs.RecordCount
  If rs(0) = Text1.Text Then
```

```
   rs.Edit

  If Text7.Text = "" Then

      MsgBox "enter the no of items ordered"
      Else rs(6) = Text7.Text rs(7) = rs(5) * rs(6) rs(4) = rs(4) + Val(Text7.Text)
      Text8.Text = rs(7)
      Text5.Text = rs(4) Text4.Text = rs(3) rs.Update
      GoTo l1
      End If
      End If rs.MoveNext Next i l1: End Sub
Private Sub Command3_Click()

Text1.Text = ""

Text2.Text = ""

Text3.Text = ""

Text4.Text = ""

Text5.Text = ""

Text6.Text = ""

Text7.Text = ""

Text8.Te
xt   =   ""
End Sub

Private                    Sub
Command4_Click()
rs.AddNew

rs(0)        =
Text1.Text
rs(1)        =
Text2.Text
rs(2)        =
Text3.Text
```

```
rs(3)        =
Text4.Text
rs(4)        =
Text5.Text
rs(5)        =
Text6.Text
rs(6)        =
Text7.Text
rs(7)        =
Text8.Text
rs.Update
End Sub

Private Sub Form_Load()

Set db = OpenDatabase("D:\prj789\invent\INVENTORY.MDB")

Set rs = db.OpenRecordset("SYSTEM")

End Sub

Private                Sub
List1_Click()
Text1.Text = List1.Text
rs.MoveFirst

For i = 1 To rs.RecordCount

If   rs(0)   =   Text1.Text
  Then   Text2.Text   =
  rs(1)

   Text3.Text = rs(2)

   Text4.Text = rs(3)

   Text5.Text = rs(4)

   Text6.Text = rs(5)
```

```
   Text7.Text = ""

      Text8.Text = "" End If rs.MoveNext Next i
      End Sub
FORM3

Dim db As Database

Dim    rs    As
Recordset
Dim    i     As
Integer

Private                    Sub
Command1_Click()
rs.MoveFirst

For i = 1 To rs.RecordCount

   If rs(0) = Text1.Text Then

         rs.Edit

         If Text4.Text = "" Then

         MsgBox "Enter the no of items needed"

      Else rs(6) = Text4.Text If rs(6) <= rs(4) Then rs(7) = rs(5) * rs(6) rs(4) =
      rs(4) - Val(Text4.Text)
      Text2.Text = rs(4)
      Text5.Text = rs(7)
      Else
      MsgBox " ITEM NOT SUFFICIENT"
      End If rs.Update GoTo l1
      End If
      End If rs.MoveNext
      Next i l1: End Sub
      Private Sub Command2_Click()
      Form1.Show
End Sub
```

```
Private Sub Command3_Click()

Text1.Text = ""

Text2.Text = ""

Text3.Text = ""

Text4.Text = ""

Text5.Te
xt   =   ""
End Sub

Private Sub Form_Load()

Set db = OpenDatabase("D:\prj789\invent\INVENTORY.MDB")

Set rs = db.OpenRecordset("SYSTEM")

End Sub

Private                Sub
List2_Click()
Text1.Text = List2.Text
rs.MoveFirst

For i = 1 To rs.RecordCount

If rs(0) = Text1.Text Then

   Text2.Text = rs(4)

   Text3.Text = rs(5)

      Text4.Text = ""
      Text5.Text = "" End If rs.MoveNext Next i
      End Sub
FORM4

Dim db As Database
```

```
Dim rs As
Recordset
Dim r, i As
Integer

Private Sub Command1_Click()

Form1.
Show
End
Sub

Private Sub Form_Load()

Set db = OpenDatabase("D:\prj789\invent\INVENTORY.MDB")

Set rs = db.OpenRecordset("SYSTEM")

MSFlexGrid1.FixedRows = 0
MSFlexGrid1.FixedCols = 0


r = 0

MSFlexGrid1.ColWidth(0) = 2000

MSFlexGrid1.ColWidth(1) = 2000

MSFlexGrid1.ColWidth(2) = 2000

MSFlexGrid1.ColWidth(3) = 1700

MSFlexGrid1.ColWidth(4) = 1750

MSFlexGrid1.ColWidth(5) = 1650

'MSFlexGrid1.ForeColor = "GREEN"

MSFlexGrid1.TextMatrix(0, 0) = "COMPANY NAME"

MSFlexGrid1.TextMatrix(0, 1) = "COMPANY ADDRESS"
```

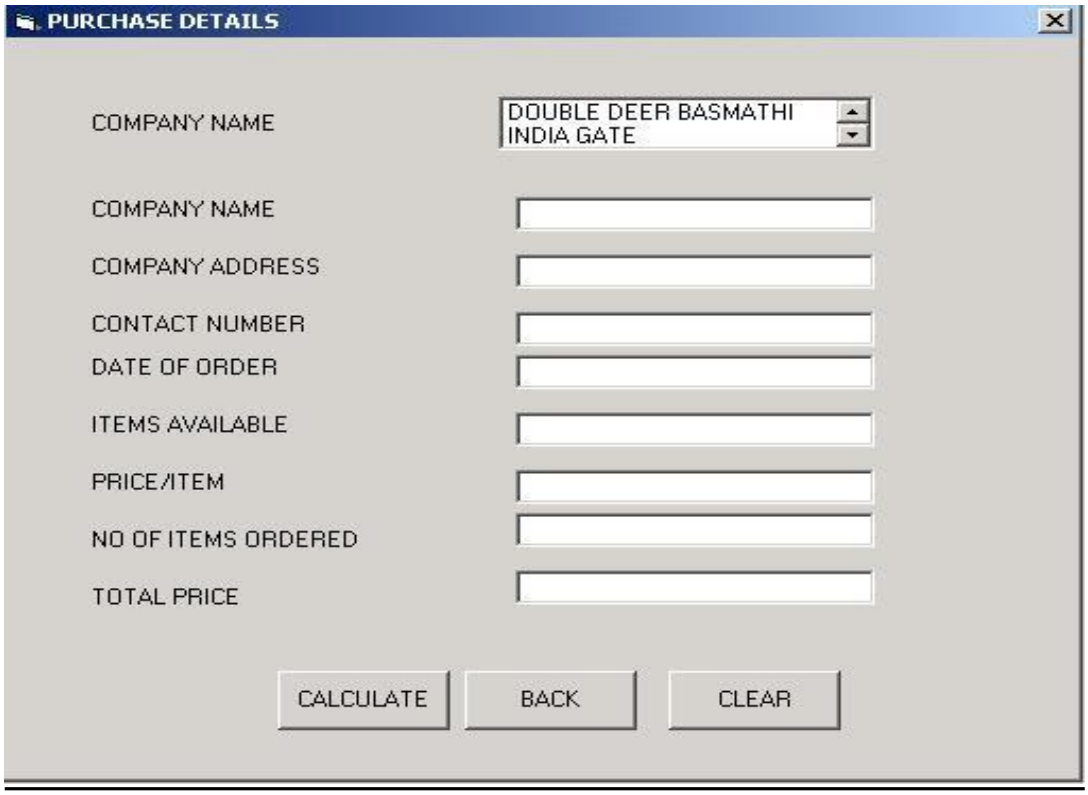
```
MSFlexGrid1.TextMatrix(0, 2) = "CONTACT NUMBER"
MSFlexGrid1.TextMatrix(0, 3) = "DATE OF ORDER"
MSFlexGrid1.TextMatrix(0, 4) = "ITEMS AVAILABLE"
MSFlexGrid1.TextMatrix(0, 5) = "PRICE/ITEM" rs.MoveFirst
r = 1
Do Until rs.EOF
MSFlexGrid1.FixedRows = r
MSFlexGrid1.FixedCols = 0
MSFlexGrid1.Text = rs(0)
MSFlexGrid1.FixedRows = r
MSFlexGrid1.FixedCols = 1
MSFlexGrid1.Text = rs(1)
MSFlexGrid1.FixedRows = r
MSFlexGrid1.FixedCols = 2
MSFlexGrid1.Text = rs(2)
MSFlexGrid1.FixedRows = r
MSFlexGrid1.FixedCols = 3
MSFlexGrid1.Text = rs(3)
MSFlexGrid1.FixedRows = r
MSFlexGrid1.FixedCols = 4
MSFlexGrid1.Text = rs(4)
MSFlexGrid1.FixedRows = r
```

```
MSFlexGrid1.FixedCols = 5

MSFlexGrid1.Text = rs(5)

MSFlexGrid1.FixedRows = r

MSFlexGrid1.FixedCols = 6

'MSFlexGrid1.Text = rs(6)

'MSFlexGrid1.FixedRows = r

'MSFlexGrid1.FixedCols = 7

'MSFlexGrid1.Text = rs(7)

r = r + 1

rs.Move
Next
Loop

End Sub
```

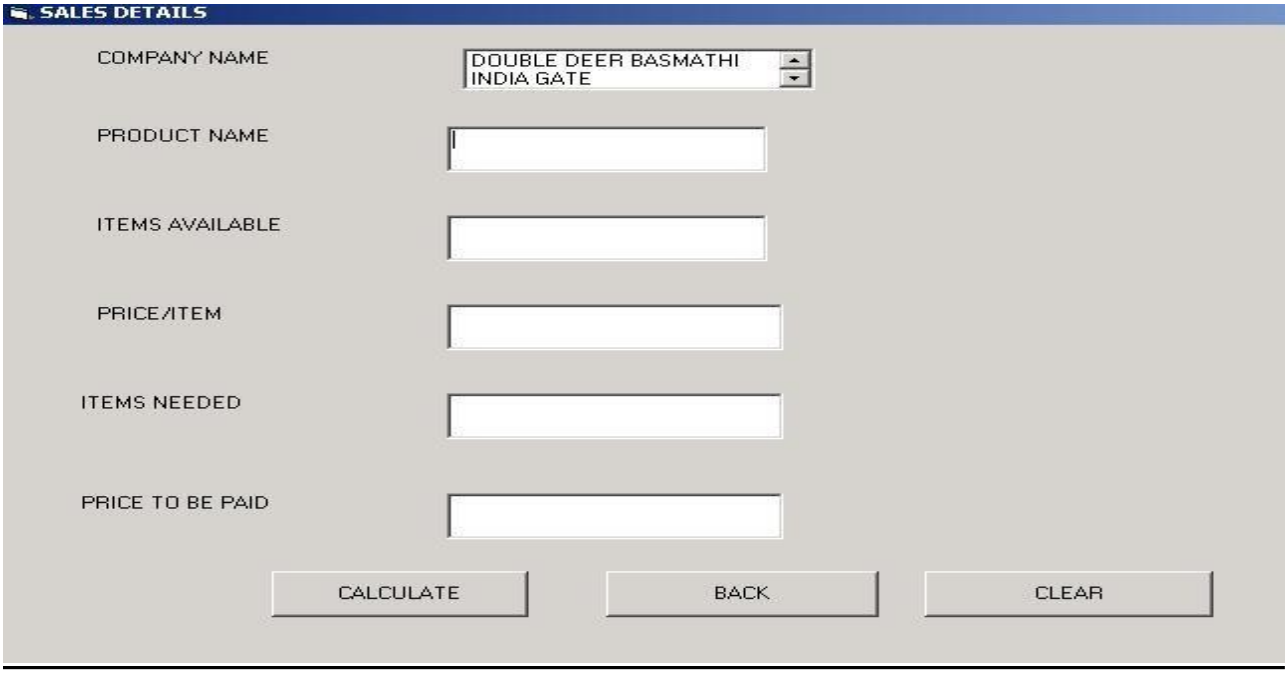
FORMS

FORM1 : MAIN MENU

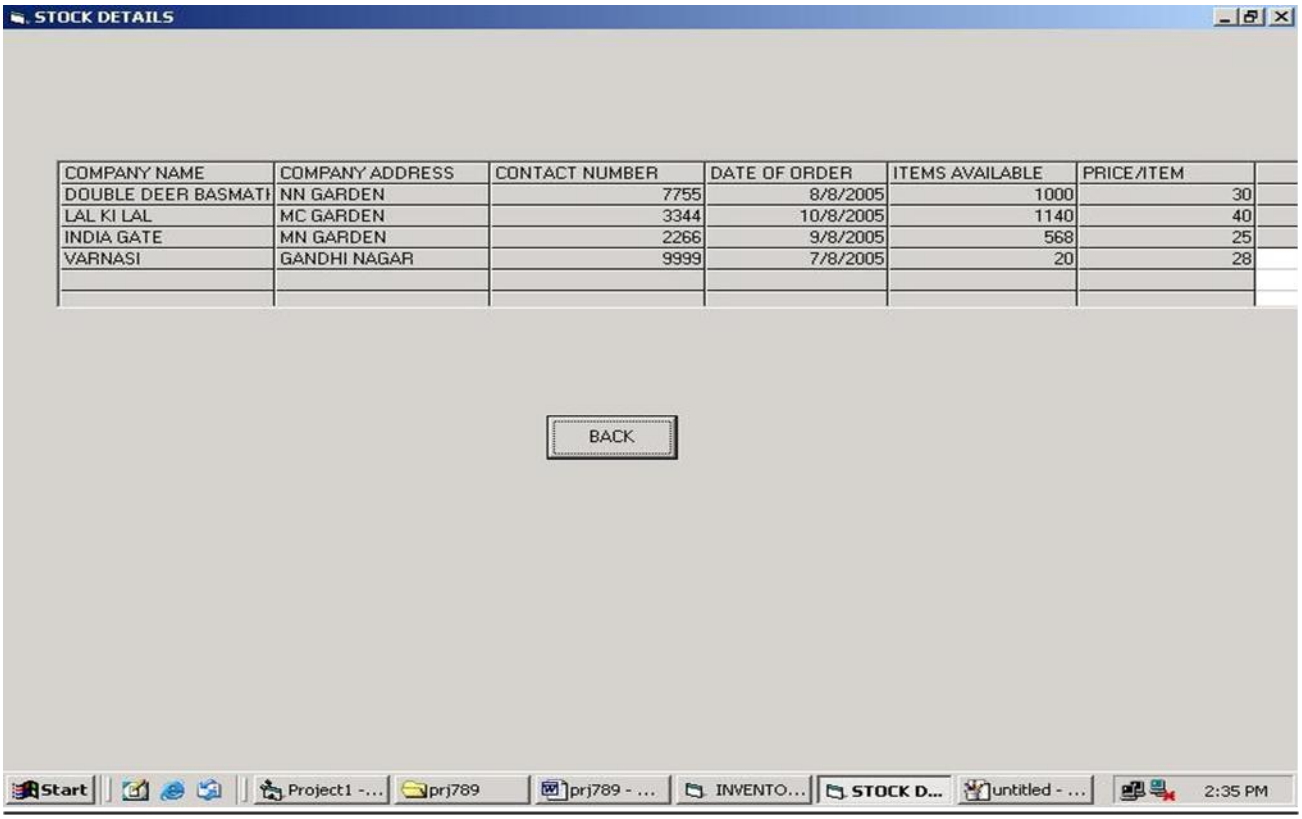
FORM2 PURCHASE DETAILS



FORM3 : SALES DETAILS



FORM4 :STOCK DETAILS


STOCK DETAILS
COMPANY NAME
COMPANY ADDRESS
CONTACT NUMBER
DATE OF ORDER
ITEMS AVAILABLE
PRICE/ITEM
DOUBLE DEER BASMATI
NN GARDEN
7755
8/8/2005
1000
30
LAL KI LAL
MC GARDEN
3344
10/8/2005
1140
40
INDIA GATE
MN GARDEN
2266
9/8/2005
568
25
VARNASI
GANDHI NAGAR
9999
7/8/2005
20
28
BACK
Start
Project1 -...
prj789
prj789 - ...
INVENTO...
STOCK D...
untitled - ...
2:35 PM