M.Sc. (Five Year Integrated) in Computer Science (Artificial Intelligence & Data Science)

Third Semester

Laboratory Record

21-805-0307: DATABASE SYSTEMS LAB

Submitted in partial fulfillment
of the requirements for the award of degree in
Master of Science (Five Year Integrated)
in Computer Science (Artificial Intelligence & Data Science) of
Cochin University of Science and Technology (CUSAT)
Kochi

Submitted by

LENAT THOMAS
(80521012)

DEPARTMENT OF COMPUTER SCIENCE
COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY (CUSAT)
KOCHI-682022

JANUARY 2023

DEPARTMENT OF COMPUTER SCIENCE

COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY (CUSAT)

KOCHI, KERALA-682022

This is to certify that the software laboratory record for ***21-805-0307: DATABASE SYSTEMS LAB*** *is a record of work carried out by* ***LENAT THOMAS (80521012)****, in partial fulfillment of the requirements for the award of degree in* ***Master of Science (Five Year Integrated)*** *in* ***Computer Science (Artificial Intelligence & Data Science)*** *of Cochin University of Science and Technology (CUSAT), Kochi. The lab record has been approved as it satisfies the academic requirements in respect of the second semester laboratory prescribed for the Master of Science (Five Year Integrated) in Computer Science degree.*

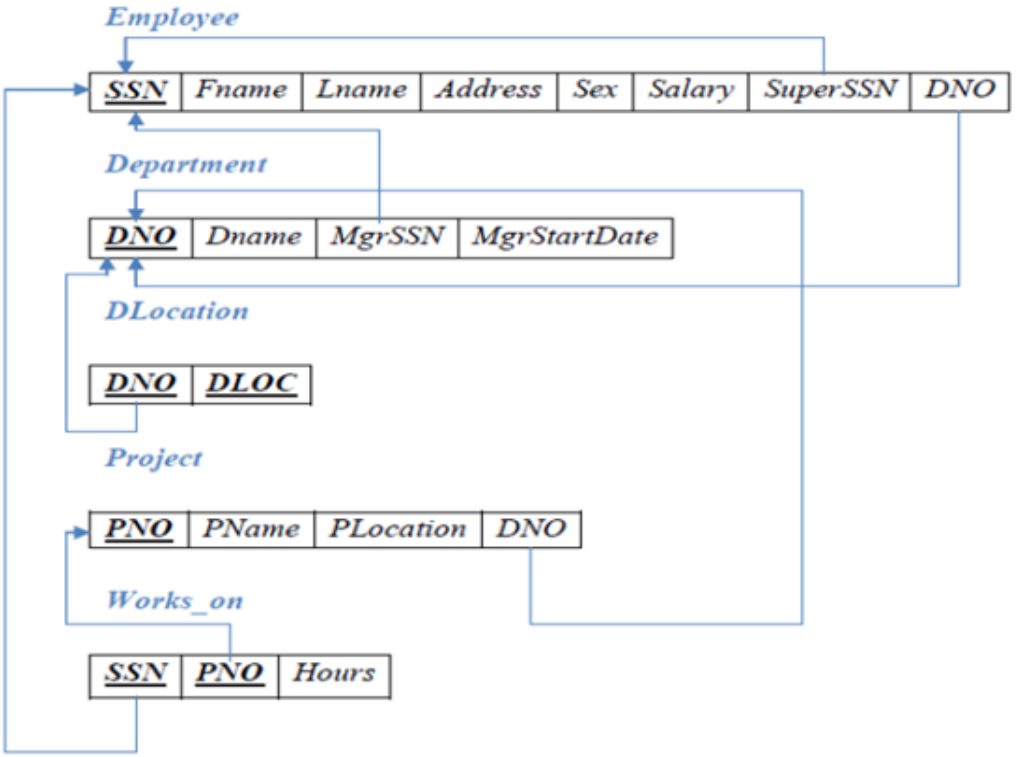
Faculty Member in-charge

Sajmi Salam
Guest Faculty
Department of Computer Science
CUSAT

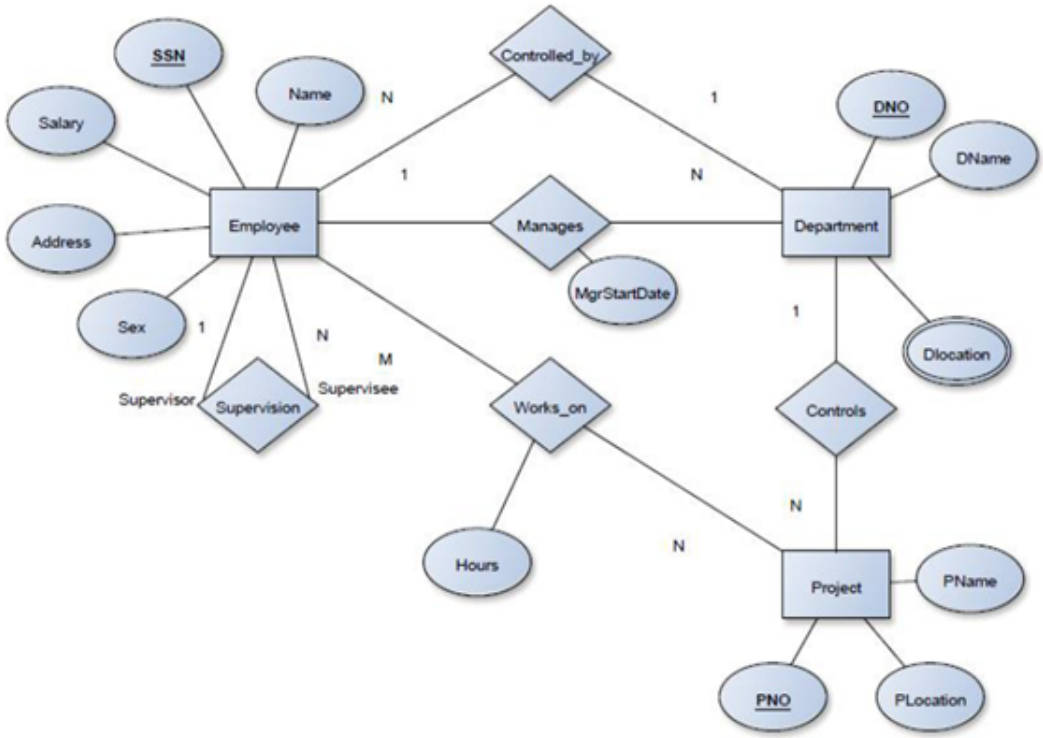
Dr. Philip Samuel
Professor and Head
Department of Computer Science
CUSAT

Table of Contents

Schema Diagram



ER Diagram



DDL Commands

AIM

Develop SQL Queries to execute and verify the Data Definition Language commands and also implement Data Constraints.

Question : 1

Create five tables using constraints like primary key, not null, check, default, null, unique, foreign key as per the above schema

Query

```
create table employee (
    -> ssn varchar(10) primary key,
    -> firstname char(30) not null,
    -> lastname char(30),
    -> Address varchar(100) not null,
    -> sex char(5) not null,
    -> salary int(10) not null,
    -> superssn varchar(10),
    -> dno int(10) not null);

create table department (
    -> dno int(10) primary key,
    -> dname char(30) not null,
    -> mgrssn varchar(10) references employee(ssn),
    -> mgrstartdate date);

alter table employee add foreign key (dno) references department(dno);

create table project(
    -> pno varchar(10) primary key,
    -> pname varchar(100) not null,
    -> plocation char(30) not null,
    -> dno int(10) references department(dno));

create table works_on(
    -> ssn varchar(10) references employee(ssn),
    -> pno varchar(10) references project(pno),
    -> hours int(5) not null);
```

Database Tables

employee

Field	Type	Null	Key	Default	Extra
ssn	varchar(10)	NO	PRI	NULL	
firstname	char(30)	NO		NULL	
lastname	char(30)	YES		NULL	
Address	varchar(100)	NO		NULL	
sex	char(5)	NO		NULL	
salary	int	NO		NULL	
superssn	varchar(10)	YES		NULL	
dno	int	NO		NULL	

department

Field	Type	Null	Key	Default	Extra
dno	int	NO	PRI	NULL	
dname	char(30)	NO		NULL	
mgrssn	varchar(10)	YES		NULL	
mgrstartdate	date	YES		NULL	

dlocation

Field	Type	Null	Key	Default	Extra
dno	int	YES		NULL	
dlocation	char(30)	NO		NULL	

project

Field	Type	Null	Key	Default	Extra

```
| pno       | varchar(10)  | NO   | PRI | NULL    |       |
| pname     | varchar(100) | NO   |     | NULL    |       |
| plocation | char(30)     | NO   |     | NULL    |       |
| dno       | int          | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
```

works_on

```
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| ssn   | varchar(10) | YES  |     | NULL    |       |
| pno   | varchar(10) | YES  |     | NULL    |       |
| hours | int         | NO   |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
```

Question : 2

Add another column Age with datatype integer in employee table

Query

```
alter table employee add column (age int(5));
```

Database Tables

employee

```
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| ssn       | varchar(10)  | NO   | PRI | NULL    |       |
| firstname | char(30)     | NO   |     | NULL    |       |
| lastname  | char(30)     | YES  |     | NULL    |       |
| Address   | varchar(100) | NO   |     | NULL    |       |
| sex       | char(5)      | NO   |     | NULL    |       |
| salary    | int          | NO   |     | NULL    |       |
| superssn  | varchar(10)  | YES  |     | NULL    |       |
| dno       | int          | NO   | MUL | NULL    |       |
```

```
| age       | int          | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
```

Question : 3

Drop a table named Project

Query

```
drop table project;
show tables;
```

Database Tables

```
+-------------------+
| Tables_in_company |
+-------------------+
| department        |
| dlocation         |
| employee          |
| works_on          |
+-------------------+
```

Question : 4

Truncate a table named WORKS ON

Query

```
delete from works_on;
select * from works_on;
```

Database Tables

```
Empty set (0.00 sec)
```

Question : 5

View the structure of the table Department

Query

```
desc department;
```

Database Tables

```
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| dno          | int         | NO   | PRI | NULL    |       |
| dname        | char(30)    | NO   |     | NULL    |       |
| mgrssn       | varchar(10) | YES  |     | NULL    |       |
| mgrstartdate | date        | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
```

DML Commands

AIM

Develop SQL Queries to execute and verify the Data Manipulation Language Commands.

Question : 1

Insert five records in the table as per the above schema.

Query

```
insert into department(dno, dname) values (8052, 'Computer Science');

insert into employee(ssn, firstname, lastname, Address ,sex, salary, dno) values('8052AM',
'Sarath' , 'Palakkal', '8052 7th street Bengaluru' , 'M', 60000 , 8052);

update department set mgrssn = '8052AM', mgrstartdate = '2020-05-28' where dno = 8052;

insert into employee values
    -> ('8052A1', 'Tony', 'Vincent', '636 WC Street Bengaluru', 'M', 34000,
    '8052AM', 8052, 36),
    -> ('8052A2', 'Sreelakshmi', 'Suthi', '36 Pattanam Street Thrissur', 'F', 28000,
    '8052AM', 8052, 25),
    -> ('8052A3', 'Deepthi', 'Shiva', 'Gerudda street Mumbai', 'F', 32000,
    '8052A1', 8052, 27),
    -> ('8052A4', 'Sapthan', 'Sundhar', 'Kamala Street Kolakata', 'M', 20000,
    '8052A2', 8052, 24);

insert into department(dno, dname) values(1001, 'Bussiness Studies');

insert into employee values('e1001', 'George', 'William', '2c Second Street Amurtha',
'M', 45000, '8052AM', 1001 , 25);

insert into employee values('e1002', 'Taran', 'Kumar', 'Nathursha 8304', 'M', 30000,
'8052AM', 1001, 40);

select * from employee;
```

Database Tables

ssn	firstname	lastname	Address	sex	salary	superssn	dno	age
8052A1	Tony	Vincent	636 WC Stre	M	34000	8052AM	8052	36
8052A2	Sreelakshmi	Suthi	36 Pattanam	F	28000	8052AM	8052	25
8052A3	Deepthi	Shiva	Gerudda str	F	32000	8052A1	8052	27
8052A4	Sapthan	Sundhar	Kamala Stre	M	20000	8052A2	8052	24
8052AM	Sarath	Palakkal	8052 7th st	M	60000	NULL	8052	NULL
e1001	George	William	2c Second S	M	45000	8052AM	1001	25
e1001	George	William	2c Second S	M	25000	8052AM	1001	25

Question : 2

Display the entire content of the tables as per the above schema.

Query

```
select * from employee;
```

Database Tables

ssn	firstname	lastname	Address	sex	salary	superssn	dno	age
8052A1	Tony	Vincent	636 WC Stre	M	34000	8052AM	8052	36
8052A2	Sreelakshmi	Suthi	36 Pattanam	F	28000	8052AM	8052	25
8052A3	Deepthi	Shiva	Gerudda str	F	32000	8052A1	8052	27
8052A4	Sapthan	Sundhar	Kamala Stre	M	20000	8052A2	8052	24
8052AM	Sarath	Palakkal	8052 7th st	M	60000	NULL	8052	NULL
e1001	George	William	2c Second S	M	45000	8052AM	1001	25
e1001	George	William	2c Second S	M	25000	8052AM	1001	25

Question : 3

Modify the salary of the employee as 25000 whose SSN is e1001

Query

```
update employee set salary = 25000 where ssn = 'e1001';

select * from employee where ssn = 'e1001';
```

Database Tables

```
+-------+-----------+----------+-------------+-----+--------+----------+------+------+
| ssn   | firstname | lastname | Address     | sex | salary | superssn | dno  | age  |
+-------+-----------+----------+-------------+-----+--------+----------+------+------+
| e1001 | George    | William  | 2c Second St| M   |  25000 | 8052AM   | 1001 |   25 |
+-------+-----------+----------+-------------+-----+--------+----------+------+------+
```

Question : 4

Delete the details of the employee whose SSN is 'e1002'

Query

```
delete from employee where ssn = 'e1002';

select * from employee where snn = 'e1002';
```

Database Tables

```
Empyt set (0.01 sec)
```

DCL Commands

AIM

Develop SQL Queries to implement Data Control Language Commands.

Question : 1

To grant a SELECT permission on employee table to user1.

Query

```
create user me@localhost identified by '1234';

select user from mysql.user;

grant all on company.* to me@localhost;

show grants for me@localhost;
```

Database Tables

```
+------------------+
| user             |
+------------------+
| me               |
| mysql.infoschema |
| mysql.session    |
| mysql.sys        |
| root             |
+------------------+
```

```
+-----------------------------------------------------------+
| Grants for me@localhost                                   |
+-----------------------------------------------------------+
| GRANT USAGE ON *.* TO `me`@`localhost`                    |
| GRANT ALL PRIVILEGES ON `company`.* TO `me`@`localhost`   |
+-----------------------------------------------------------+
```

Question : 2

Revoking privileges to all users in a table.

Query

```
revoke all on company.* from me@localhost;

show grants for me@localhost;
```

Database Tables

```
+---------------------------------------+
| Grants for me@localhost               |
+---------------------------------------+
| GRANT USAGE ON *.* TO ‘me‘@‘localhost‘ |
+---------------------------------------+
```

Group Functions Or Aggregate Functions

AIM

Develop SQL Queries to execute computation on table data with built-in functions.

Question : 1

List the first name of all the employee having 'a' as the the second last character in their name.

Query

```
select firstname from employee where firstname like '_a%';
```

Database Tables

```
+-----------+
| firstname |
+-----------+
| Sapthan   |
| Sarath    |
+-----------+
```

Question : 2

Count the total number of male and female employees in the Employee table.

Query

```
select sex, count(*) as sex from employee group by sex;
```

Database Tables

```
+-----+-----+
| sex | sex |
+-----+-----+
| M   |   4 |
| F   |   2 |
+-----+-----+
```

Question : 3

Calculate the average salary of the female employees.

Query

```
select avg(salary) from employee where sex = 'F';
```

Database Tables

```
+-------------+
| avg(salary) |
+-------------+
|  30000.0000 |
+-------------+
```

Question : 4

Calculate the sum of salaries of male employees.

Query

```
select sum(salary) from employee where sex = 'M';
```

Database Tables

```
+-------------+
| sum(salary) |
+-------------+
|      139000 |
+-------------+
```

Question : 5

Display the maximum and minimum salaries of male employees

Query

```
select max(salary), min(salary) from employee where sex = 'M';
```

Database Tables

```
+-------------+-------------+
| max(salary) | min(salary) |
+-------------+-------------+
|       60000 |       20000 |
+-------------+-------------+
```

Question : 6

Display the details of all employees whose salary between 25000 and 50000

Query

```
select * from employee where salary between 25000 and 50000;
```

Database Tables

```
+--------+-------------+----------+-----------+-----+--------+----------+------+------+
| ssn    | firstname   | lastname | Address   | sex | salary | superssn | dno  | age  |
+--------+-------------+----------+-----------+-----+--------+----------+------+------+
| 8052A1 | Tony        | Vincent  | 636 WC Str| M   |  34000 | 8052AM   | 8052 |   36 |
| 8052A2 | Sreelakshmi | Suthi    | 36 Pattana| F   |  28000 | 8052AM   | 8052 |   25 |
| 8052A3 | Deepthi     | Shiva    | Gerudda st| F   |  32000 | 8052A1   | 8052 |   27 |
| e1001  | George      | William  | 2c Second | M   |  25000 | 8052AM   | 1001 |   25 |
+--------+-------------+----------+-----------+-----+--------+----------+------+------+
```

Question : 7

Display the last name of the employees whose salaries are 30000 or 40000 or 50000

Query

```
select lastname from employee where salary = 30000 or salary = 40000 or salary = 50000;
```

Database Tables

```
Empty set (0.00 sec)
```

Nested Queries

AIM

Develop SQL Queries to implement Nested Queries/ Sub Quesries and Joins

Question : 1

Update the salary by 0.25 times for all the employees whose Plocation is 'Chennai'.

Query

```
update employee, project set salary = salary * 0.25 where
employee.dno = project.dno and plocation = 'Chennai';

select ssn, salary from employee;
```

Database Tables

```
+--------+--------+
| ssn    | salary |
+--------+--------+
| 8052A1 |  34000 |
| 8052A2 |  28000 |
| 8052A3 |  32000 |
| 8052A4 |  20000 |
| 8052AM |  60000 |
| e1001  |    391 |
+--------+--------+
```

Question : 2

To display the name and project location of employees whose working hour is greater than 5

Query

```
select firstname, lastname, plocation from employee, project, works_on where
employee.ssn = works_on.ssn and project.pno = works_on.pno and hours > 5;
```

Database Tables

```
+-----------+----------+-----------+
| firstname | lastname | plocation |
+-----------+----------+-----------+
| George    | William  | Chennai   |
+-----------+----------+-----------+
```

Question : 3

Left join employee table and works on table

Query

```
select * from employee left join works_on on employee.ssn = works_on.ssn;
```

Database Tables

```
+--------+-------------+----------+-----------------------------+-----+--------+-
| ssn    | firstname   | lastname | Address                     | sex | salary |
+--------+-------------+----------+-----------------------------+-----+--------+-
| 8052A1 | Tony        | Vincent  | 636 WC Street Bengaluru     | M   |  34000 |
| 8052A2 | Sreelakshmi | Suthi    | 36 Pattanam Street Thrissur | F   |  28000 |
| 8052A3 | Deepthi     | Shiva    | Gerudda street Mumbai       | F   |  32000 |
| 8052A4 | Sapthan     | Sundhar  | Kamala Street Kolakata      | M   |  20000 |
| e1001  | George      | William  | 2c Second Street Amurtha    | M   |  45000 |
+--------+-------------+----------+-----------------------------+-----+--------+-
```

```
---------+------+------+-------+------+-------+
superssn | dno  | age  | ssn   | pno  | hours |
---------+------+------+-------+------+-------+
8052AM   | 8052 |   36 | NULL  | NULL |  NULL |
8052AM   | 8052 |   25 | NULL  | NULL |  NULL |
```

```
8052A1   | 8052 |   27 | NULL  | NULL |  NULL |
8052A2   | 8052 |   24 | NULL  | NULL |  NULL |
8052AM   | 1001 |   25 | e1001 | 1008 |    10 |
---------+------+------+-------+------+-------+
```

Question : 4

Right join works on table and employee table

Query

```
select * from works_on right join employee on employee.ssn = works_on.ssn;
```

Database Tables

```
+-------+------+-------+--------+-------------+----------+-
| ssn   | pno  | hours | ssn    | firstname   | lastname |
+-------+------+-------+--------+-------------+----------+-
| NULL  | NULL |  NULL | 8052A1 | Tony        | Vincent  |
| NULL  | NULL |  NULL | 8052A2 | Sreelakshmi | Suthi    |
| NULL  | NULL |  NULL | 8052A3 | Deepthi     | Shiva    |
| NULL  | NULL |  NULL | 8052A4 | Sapthan     | Sundhar  |
| e1001 | 1008 |    10 | e1001  | George      | William  |
+-------+------+-------+--------+-------------+----------+-
```

```
---------------------------+-----+--------+----------+------+------+
Address                    | sex | salary | superssn | dno  | age  |
---------------------------+-----+--------+----------+------+------+
636 WC Street Bengaluru    | M   |  34000 | 8052AM   | 8052 |   36 |
36 Pattanam Street Thrissur| F   |  28000 | 8052AM   | 8052 |   25 |
Gerudda street Mumbai      | F   |  32000 | 8052A1   | 8052 |   27 |
Kamala Street Kolakata     | M   |  20000 | 8052A2   | 8052 |   24 |
2c Second Street Amurtha   | M   |  45000 | 8052AM   | 1001 |   25 |
---------------------------+-----+--------+----------+------+------+
```

Question : 5

Full join works on table and employee table

Query

```
select * from works_on full join employee;
```

Database Tables

```
+-------+------+-------+--------+-------------+----------+-
| ssn   | pno  | hours | ssn    | firstname   | lastname |
+-------+------+-------+--------+-------------+----------+-
| e1001 | 1008 |    10 | 8052A1 | Tony        | Vincent  |
| e1001 | 1008 |    10 | 8052A2 | Sreelakshmi | Suthi    |
| e1001 | 1008 |    10 | 8052A3 | Deepthi     | Shiva    |
| e1001 | 1008 |    10 | 8052A4 | Sapthan     | Sundhar  |
| e1001 | 1008 |    10 | e1001  | George      | William  |
+-------+------+-------+--------+-------------+----------+-
```

```
----------------------------+-----+--------+----------+------+------+
Address                     | sex | salary | superssn | dno  | age  |
----------------------------+-----+--------+----------+------+------+
636 WC Street Bengaluru     | M   |  34000 | 8052AM   | 8052 |   36 |
36 Pattanam Street Thrissur | F   |  28000 | 8052AM   | 8052 |   25 |
Gerudda street Mumbai       | F   |  32000 | 8052A1   | 8052 |   27 |
Kamala Street Kolakata      | M   |  20000 | 8052A2   | 8052 |   24 |
2c Second Street Amurtha    | M   |  45000 | 8052AM   | 1001 |   25 |
----------------------------+-----+--------+----------+------+------+
```

Views

AIM

Develop SQL queries for creating and dropping Views.

Question : 1

Create a view VW-emp on employee table

Query

```
create view vw_emp as select * from employee;

desc vw_emp;
```

Database Tables

```
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| ssn       | varchar(10)  | NO   |     | NULL    |       |
| firstname | char(30)     | NO   |     | NULL    |       |
| lastname  | char(30)     | YES  |     | NULL    |       |
| Address   | varchar(100) | NO   |     | NULL    |       |
| sex       | char(5)      | NO   |     | NULL    |       |
| salary    | int          | NO   |     | NULL    |       |
| superssn  | varchar(10)  | YES  |     | NULL    |       |
| dno       | int          | NO   |     | NULL    |       |
| age       | int          | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
```

Question : 2

Create another view VW SSN contains SuperSSN and Dno of female employees

Query

```
create view vw_ssn as select superssn, dno from employee where sex = 'F';

select * from vw_ssn;
```

Database Tables

```
+----------+------+
| superssn | dno  |
+----------+------+
| 8052AM   | 8052 |
| 8052A1   | 8052 |
+----------+------+
```

Question : 3

Update the address of employee to Chennai whose id is e100 in view VW emp

Query

```
update vw_emp set address = 'Chennai' where ssn = 'e1001';

select * from vw_emp where ssn = 'e1001';
```

Database Tables

```
+-------+-----------+----------+---------+-----+--------+----------+------+------+
| ssn   | firstname | lastname | Address | sex | salary | superssn | dno  | age  |
+-------+-----------+----------+---------+-----+--------+----------+------+------+
| e1001 | George    | William  | Chennai | M   |    391 | 8052AM   | 1001 |   25 |
+-------+-----------+----------+---------+-----+--------+----------+------+------+
```

Question : 4

Delete the view VW emp

Query

```
delete from vw_emp;

select * from vw_emp;
```

Database Tables

```
Empty set (0.00 sec)\
```

Functions And Procedures

AIM

Develop PL/SQL program to familiarize with Function and Procedure

Question : 1

Write a PL/SQL function to find factorial of a number

Query

```
set serveroutput on
edit@factorial.sql

create or replace function get_factorial(N int)
return varchar
is
fact int := 1;
begin
for i in 1..N loop
    fact := fact*i;
end loop;
return 'Factorial is ' || fact ;
end;
/
select get_factorial(6) from dual;

@XEfactorial.sql
```

Output

```
Factorial is 720
```

Question : 2

Write a PL/SQL function to find maximum of two numbers

Query

```
set serveroutput on
edit@max.sql

create or replace function maximum(n1 int, n2 int)
return varchar
is
m int := 0;
begin
if n1>n2 then
    m := n1;
else
    m := n2;
end if;
return 'Maximum is ' ||m;
end;
/
select maximum(12, 25) from dual;

@XEmax.sql
```

Output

```
Maximum is 25
```

Question : 3

Query

Write a PL/SQL procedure to print the prime

```
set serveroutput on
edit@prime.sql
```

```
declare
    i number(3);
    j number(3);
begin
dbms_output.Put_line('The prime numbers are:');
dbms_output.new_line;
i := 2;
loop
    j := 2;
    loop
        exit when( ( mod(i, j) = 0 ) or ( j = i ) );
        j := j + 1;
    end loop;
    if( j = i ) then
        dbms_output.Put(i||' ');
    end if;
    i := i + 1;
    exit when i = 50;
end loop;
dbms_output.new_line;
end;
/

@XEprime.sql
```

Output

```
The prime numbers are 2 3 5 7 11 13
```

Question : 4

Write a PL/SQL procedure to display numbers from 1 to 10 using while loop

Query

```
set serveroutput on
```

```
edit@numbers.sql

declare
    i INTEGER := 1;
begin
while i <= 10 loop
    dbms_output.Put_line(i);
    i := i+1;
end loop;
wnd;
/
@XEnumbers.sql
```

Database Tables

```
1
2
3
4
5
6
7
8
9
10
```

Cursor

AIM

Develop PL/SQL program to implement Cursor.

Question : 1

Write a PL/SQL cursor program to update the salary of each employee of department number D001 in the Employee table as per the schema

Query

```
declare cursor c1 is select id,salary from employee where id = 01
for update;
sal number;
begin
for i in c1 loop
    if i.Salary>20000 then
        sal := .10;
    else
        sal := .15;
    end if;
    update employee set salary  = salary+salary*sal where  current of c1;
end loop;
end;
/
```

Output

```
ID                   NAME                 ADDRESS             SALARY     AGE
-------------------- -------------------- -------------- ---------- -------
D001                 Ramcharan            Hyderabad           30800      32

D002                 Diya                 Hyderabad           29000      28

D003                 Shekhar              Delhi               30000      29
```

ID	NAME	ADDRESS	SALARY	AGE
D001	Ramcharan	Hyderabad	31000	32
D002	Diya	Hyderabad	25000	28
D003	Shekhar	Delhi	25000	29

Question : 2

Write a PL/SQL cursor program to retrieve Dno and DName from Department table as per the schema

Query

```
declare cursor c2 is select dno,dname from department;
data1 department.Dno%type;
data2 department.Dname%type;
begin
open c2;
loop
    fetch c2 into data1,data2;
    exit when c2%notfound;
    dbms_output.put_line('Dno : '||data1||':: Dname : '||data2);
end loop;
close c2;
end;
/
```

Output

```
Dno : 1:: Dname : DCS
Dno : 2:: Dname : DDUk
Dno : 3:: Dname : DCA
Dno : 4:: Dname : SOE
```

```
Dno : 5:: Dname : SMS
```

Trigger

AIM

Develop and execute a Trigger before and after Update/Delete/Insert operations on a table

Question : 1

Write PL/SQL trigger program to display the salary differences between the old values and new values in the table employee as per the schema

Query

```
CREATE OR REPLACE TRIGGER display_sal_changes
BEFORE DELETE OR INSERT OR UPDATE ON employee
FOR EACH ROW WHEN (NEW.ID > 0)
DECLARE
    sal_diff number;
BEGIN
    sal_diff := :NEW.salary - :old.salary;
    dbms_output.put line( 'Old salary: '|| :OLD.salary);
   dbms_output.put_line( 'New salary: '|| :NEW. salary);
   dbms_output.put_line('Salary difference: '|| sal_diff);
 END;
/

update employeee set salary = 50000 where ssn = 'e1001';
```

Output

```
+------------+
| Old salary |
+------------+
|    34000   |
|    28000   |
|    32000   |
|    20000   |
|    45000   |
```

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

```
+------------+
| New salary |
+------------+
|    34000   |
|    28000   |
|    32000   |
|    20000   |
|    50000   |
+------------+
```

```
+------------+
| Sal diff |
+----------+
|     0    |
|     0    |
|     0    |
|     0    |
|    5000  |
+----------+
```

Question : 2

Write PL/SQL trigger program to display the hour differences between the old values and new values in the table Workson as per the schema

Query

```
CREATE OR REPLACE TRIGGER display_hour_updates
BEFORE DELETE OR INSERT OR UPDATE ON works_on
FOR EACH ROW
WHEN (NEW.HOURS > 0)
DECLARE
   hour_diff number;
BEGIN
   hour_diff := :NEW.HOURS  - :OLD.HOURS;
   dbms_output.put_line('Old HOURS: ' || :OLD.HOURS);
   dbms_output.put_line('New HOURS: ' || :NEW.HOURS);
```

```
   dbms_output.put_line('HOUR difference: ' || hour_diff);
END;
/

update works_on set hours = 20 where ssn = 'e1001';
```

Output

```
+------------+
|  Old hours |
+------------+
|     10     |
+------------+

+------------+
|  New hours |
+------------+
|     20     |
+------------+

+------------+
|  hour diff |
+------------+
|     10     |
+------------+
```

Transaction Control

AIM

Develop SQL Queries to understand the concept of Transaction Control Language

Question : 1

Creating Check points in the program

Query

```
begin transaction;

savepoint s1;
```

Question : 2

Rollback to a previously created Checkpoint in the program

Query

```
update employee set salary where = 50000 ssn = 'e1001';

select ssn, salary from employee;

rollback to s1;

select ssn, salary from employee;
```

Database Tables

```
+--------+--------+
| ssn    | salary |
+--------+--------+
| 8052A1 |  34000 |
| 8052A2 |  28000 |
| 8052A3 |  32000 |
```

```
| 8052A4 |  20000 |
| e1001  |  50000 |
+--------+--------+
```

```
+--------+--------+
| ssn    | salary |
+--------+--------+
| 8052A1 |  34000 |
| 8052A2 |  28000 |
| 8052A3 |  32000 |
| 8052A4 |  20000 |
| e1001  |  45000 |
+--------+--------+
```

Question : 3

Commit the program

Query

```
commit
```

MongoDB

AIM

Develop program to perform operations in MongoDB

Question : 1

Create a database emp

Query

```
use emp;
```

Database Tables

```
switched to db emp
```

Question : 2

Create new Collection

Query

```
db.createCollection("employee")
show collections;
```

Database Tables

```
employee
```

Question : 3

Check the collection list created and drop collection

Query

```
show collections;
db.employee.drop()
```

Database Tables

```
employee
```

Question : 4

Insert document in selected Collection

Query

```
db.employee.insertOne({
    ssn : 'e1001',
    firstname : 'Raj',
    lastname : 'Kiran',
    sex : 'M',
    Address : 'Bangalore',
    salary : 25000,
    department : 'Department of Computer Science'
})

db.employee.find();
```

Database Tables

```
{
```

```
  acknowledged: true,
  insertedId: ObjectId("63a09189b2a462e8c1bcb151")
}

[
  {
    _id: ObjectId("63a09189b2a462e8c1bcb151"),
    ssn: 'e10001',
    firstname: 'Raj',
    lastname: 'Kiran',
    sex: 'M',
    Address: 'Bangalore',
    salary: 25000,
    department: 'Department of Computer Science'
  }
]
```

Question : 5

To get the list documents in Collection

Query

```
db.employee.find();
[
```

Database Tables

```
[
  {
    _id: ObjectId("63a09189b2a462e8c1bcb151"),
    ssn: 'e10001',
    firstname: 'Raj',
    lastname: 'Kiran',
    sex: 'M',
    Address: 'Bangalore',
```

```
    salary: 25000,
    department: 'Department of Computer Science'
  }
]
```

Question : 6

Update the document in Collection

Query

```
db.employee.updateOne({ssn : 'e10001'}, {$set : {salary : 30000}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Database Tables

```
emp> db.employee.find()
[
  {
    _id: ObjectId("63a09189b2a462e8c1bcb151"),
    ssn: 'e10001',
    firstname: 'Raj',
    lastname: 'Kiran',
    sex: 'M',
    Address: 'Bangalore',
    salary: 30000,
    department: 'Department of Computer Science'
  }
]
```

Question : 7

Delete the document in selected Collection

Query

```
db.employee.deleteOne({ssn : 'e10001'})
```

Database Tables

```
{ acknowledged: true, deletedCount: 1 }
```

Question : 8

Projection using find() method

Query

```
db.employee.find({},{firstname : 1, lastname : 1, salary : 1})
[
  {
    _id: ObjectId("63a09363b2a462e8c1bcb152"),
    firstname: 'Deepthy',
    lastname: 'Varyar',
    salary: 35000
  }
]
```

Question : 9

Drop database emp

Query

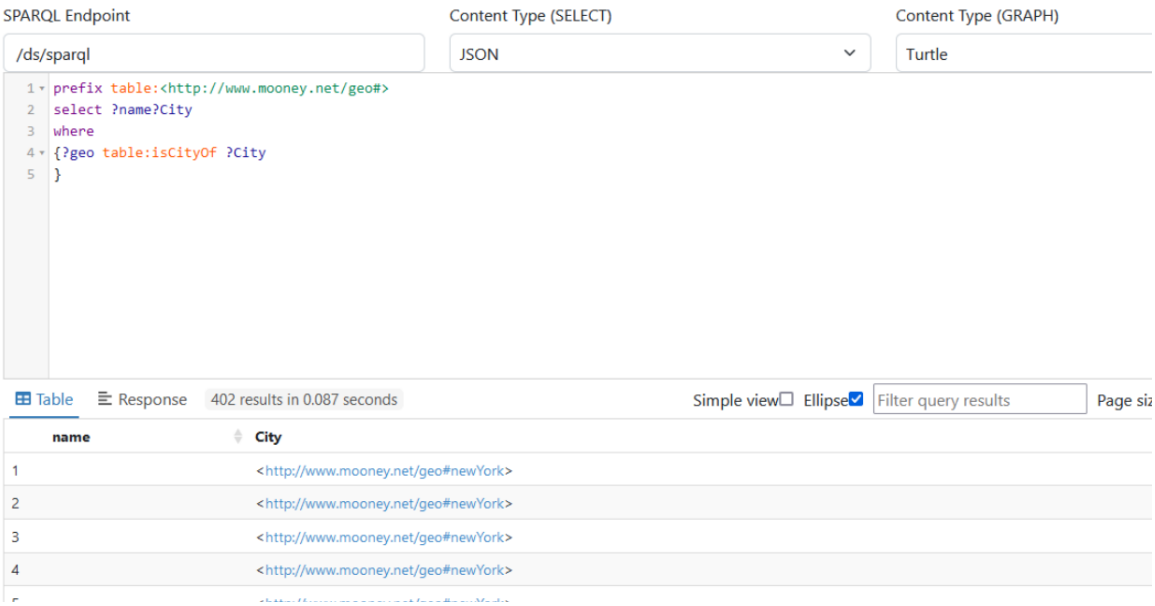
```
db.dropDatabase("emp")
```

GraphQL

AIM

Develop a GraphQL program to perform different operations in created ontology

Output



Database Connectivity

AIM

Develop program to implement Java Database Connectivity

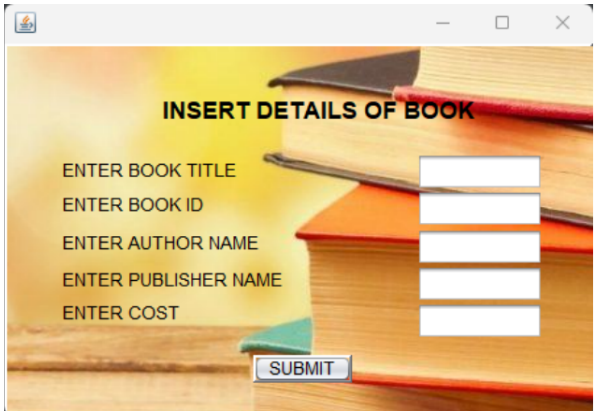
Question : 1

Write a program which connects to an online book database and insert the details of the books in to the database

Program

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
    }
    catch (ClassNotFoundException ex) {
        Logger.getLogger(insert.class.getName()).log(Level.SEVERE, null, ex);
    }
    try (Connection con = DriverManager.getConnection
    ("jdbc:mysql://localhost:3306/book","root","root")) {
        String sql = "insert into books values(?,?,?,?,?)";
        PreparedStatement ps = con.prepareStatement(sql);
        ps.setString(1,jTextField1.getText());
        ps.setString(2,jTextField2.getText());
        ps.setString(3, jTextField3.getText());
        ps.setString(4, jTextField4.getText());
        ps.setInt(5,Integer.parseInt(jTextField5.getText()));
        ps.execute();
        JOptionPane.showMessageDialog(this,"data saved succesfully");
    }
    catch(HeadlessException | NumberFormatException | SQLException e){
        JOptionPane.showMessageDialog(this,e);
    }
}
```

Output



Question : 2

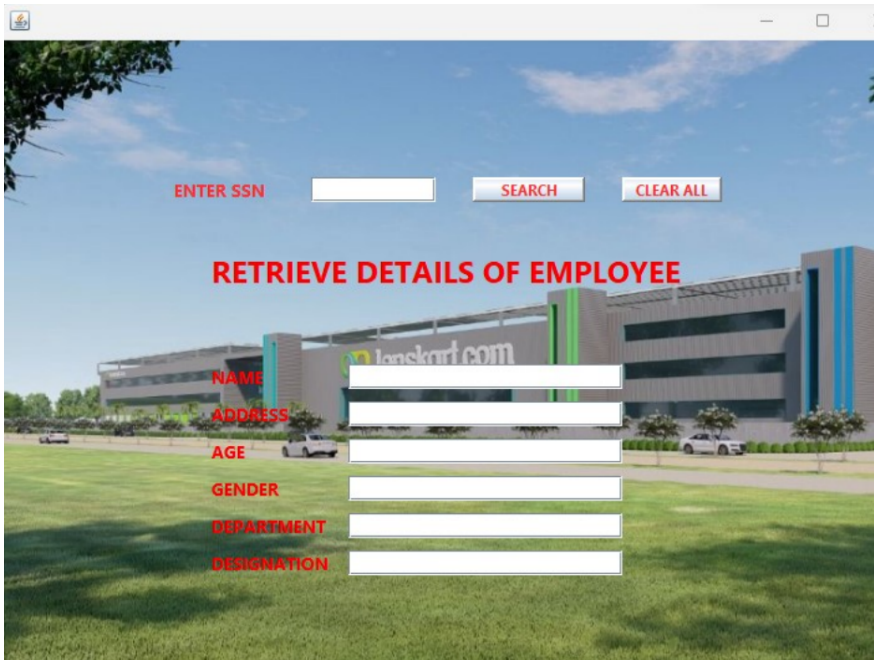
Write a program which connects to an online Employee database and retrieve the details of the employees in the database as per the schema

Program

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
    }
    catch (ClassNotFoundException ex) {
        Logger.getLogger(Retrieve.class.getName()).log(Level.SEVERE, null, ex);
        }
    try (Connection con = DriverManager.getConnection
    ("jdbc:mysql://localhost:3306/company","root","root")) {
        String sql = "select * from employee where SSN = ?";
        PreparedStatement ps = con.prepareStatement(sql);
        ps.setString(1,ssn.getText());
        ResultSet rs = ps.executeQuery();
        if(rs.next()){
            Name.setText(rs.getString("Name"));
            address.setText(rs.getString("Address"));
            Age.setText(rs.getString("Age"));
            gender.setText(rs.getString("Sex"));
        }
        else{
            JOptionPane.showMessageDialog(this,"data Not Found");
        }
    }
    catch(HeadlessException | NumberFormatException | SQLException e){
```

```
            JOptionPane.showMessageDialog(this,e);
        }
}
```

Output



Question : 3

Write a program which connects to an online hospital database and update the details of the patients in the database

Program

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
    }
    catch (ClassNotFoundException ex) {
        Logger.getLogger(Update.class.getName()).log(Level.SEVERE, null, ex);
    }
    try (Connection con = DriverManager.getConnection
    ("jdbc:mysql://localhost:3306/hospital","root","root")) {
        String sql = "select * from patients where phone = ?";
        PreparedStatement ps = con.prepareStatement(sql);
```

```
        ps.setString(1,phone.getText());
        ResultSet rs = ps.executeQuery();
        if(rs.next()){
            String sql2 = "update patients set Name=?,Gender = ?,
            bld_grp = ?,Age = ?,disease = ? where Phone = ?;";
            PreparedStatement ps2 = con.prepareStatement(sql2);
            ps2.setString(1,Name.getText());
            ps2.setString(2, gender.getText());
            ps2.setString(3, bld_grp.getText());
            ps2.setInt(4, Integer.parseInt(Age.getText()));
            ps2.setString(5, disease.getText());
            ps2.setInt(6,Integer.parseInt(phone.getText()));
            ps2.execute();
            JOptionPane.showMessageDialog(this,"Data Updated Succesfully");
        }
        else{
            JOptionPane.showMessageDialog(this,"data Not Found");
        }
    }
    catch(HeadlessException | NumberFormatException | SQLException e){
        JOptionPane.showMessageDialog(this,e);
    }
}
```

Output

Question : 4

Write a program which connects to an online Hotel database and delete the details of the orders from the database

Program

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    String res = jTextField1.getText();
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
    }
    catch (ClassNotFoundException ex) {
        Logger.getLogger(delete_order.class.getName()).log(Level.SEVERE, null, ex);
    }
    try (Connection con = DriverManager.getConnection
    ("jdbc:mysql://localhost:3306/hotel","root","root")) {
        if (res.equals("")){
            JOptionPane.showMessageDialog(this,"Empty Field Not Allowed");
        }
        PreparedStatement statement = (PreparedStatement)con.prepareStatement
        ("select * from orders where reservation_no = ?;");
        statement.setString(1, res);
        ResultSet result = statement.executeQuery();
        if (!(result.next())){
            JOptionPane.showMessageDialog(this,"Data not in the DataBase");
```

```
        }
        else{
            PreparedStatement ps = (PreparedStatement)con.prepareStatement
        ("delete from orders where reservation_no = ?;");
        ps.setString(1,res);
        ps.execute();
        JOptionPane.showMessageDialog(this,"data deleted succesfully");
        con.close();
        statement.close();
        result.close();
        ps.close();
        }
    }
    catch(HeadlessException | NumberFormatException | SQLException e){
        JOptionPane.showMessageDialog(this,e);
    }
}
```

Output

Project

AIM

Develop an Application Software using Java and Mysql for Information Management purpose.

Project Description

A place to store the details of the lands allocated to individuals or groups in a particular area. Whenever a change happens to usage, ownership or topography, the records in the database are changed and the change and its reason are stored in a log history. This helps to monitor each and every land in a particular area. Anything in the land that is different from the records can be proven to be illegal. Additionally the owners of these lands can also view the details of these lands after several security features or others land if they get permissions. If certain details of land are wrong and needs to be updated, the owners of the concerned land can send a letter of a resurvey using the same software

Users And Functionalities

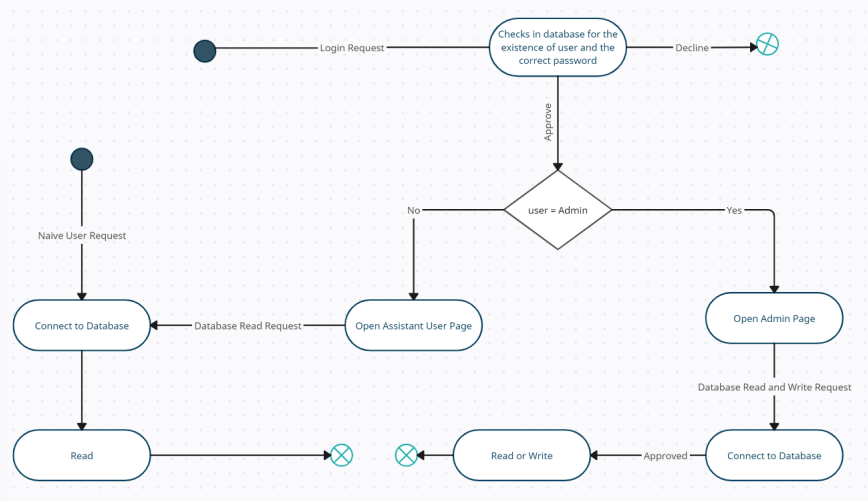
Admin User :-
Expected to be a Village Officer or Taluk Officer. Can add new records, delete records and update records. The change is recorded in change log. Admins are unable to change the history log.

Assistant User :-
Expected to be someone of lower ranking than a Village Officer, an Office Assistance. Has the ability to view any records, but not able edit the records in any way. Even though Office Assistant has the ability to view any record, he is only allowed when the situation demands it.

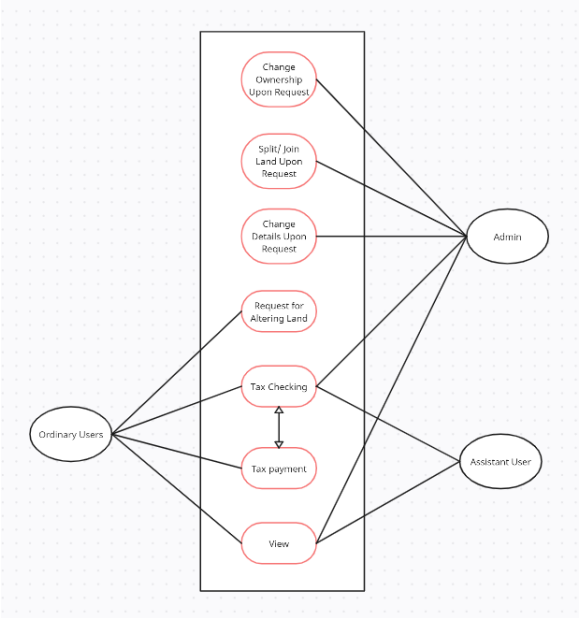
End Users :-
They are the owners of the lands. With the survey number of their land and their aadhaar number they can view the details of their land. With permissions they can also view other lands with the help of Assistant users as an intermediary. The resurvey function is available to End Users.

Reference Design

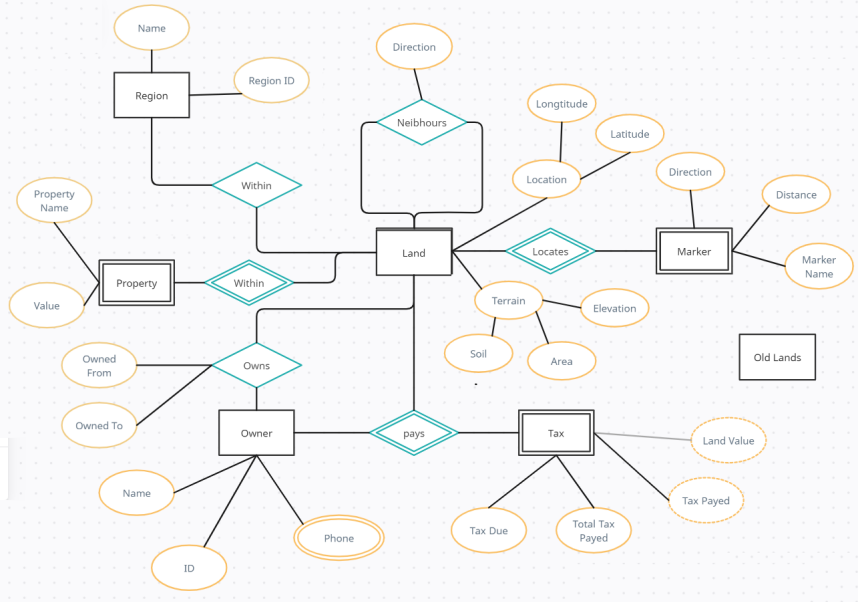
Activity Diagram



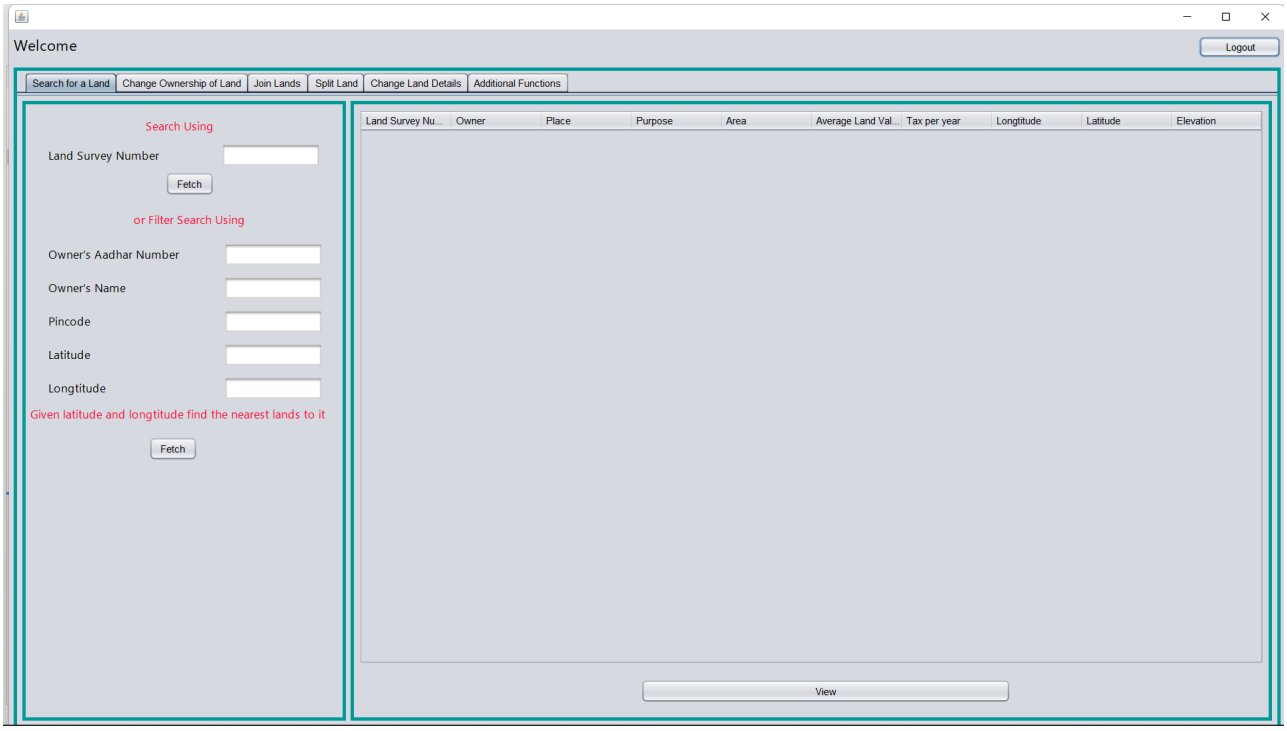
UML Diagram

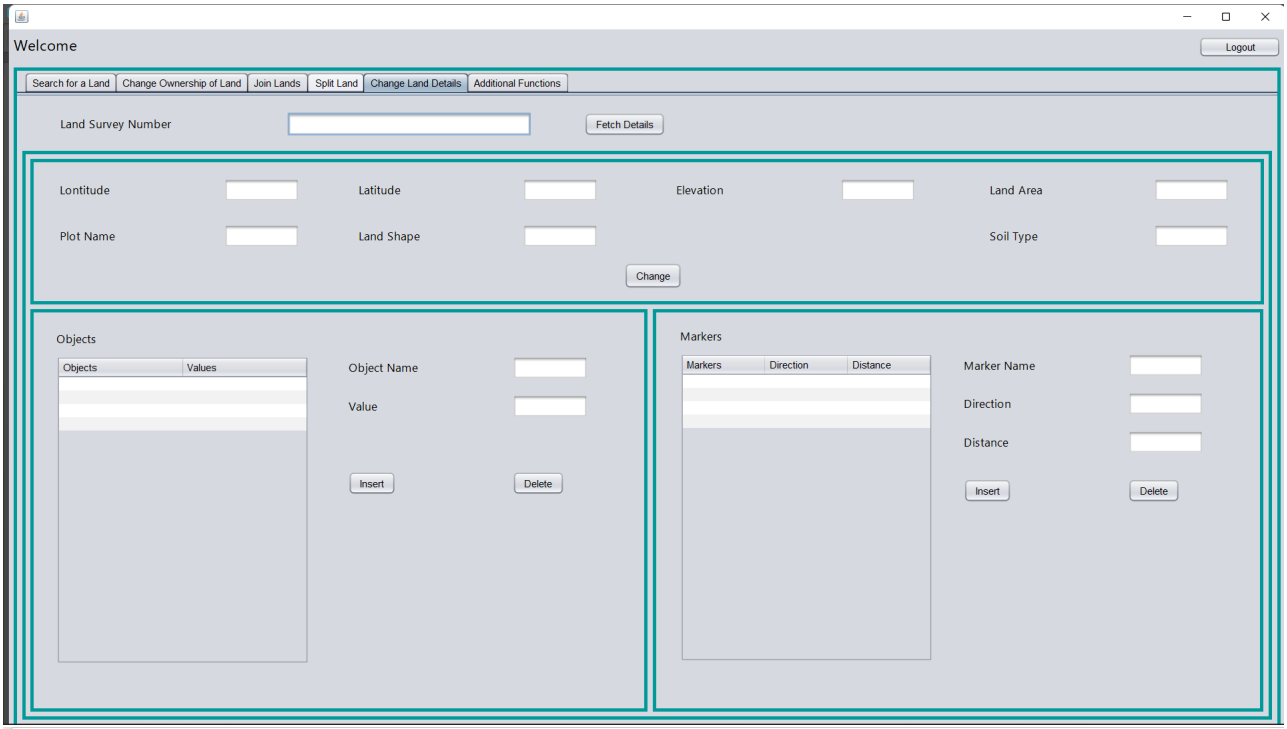


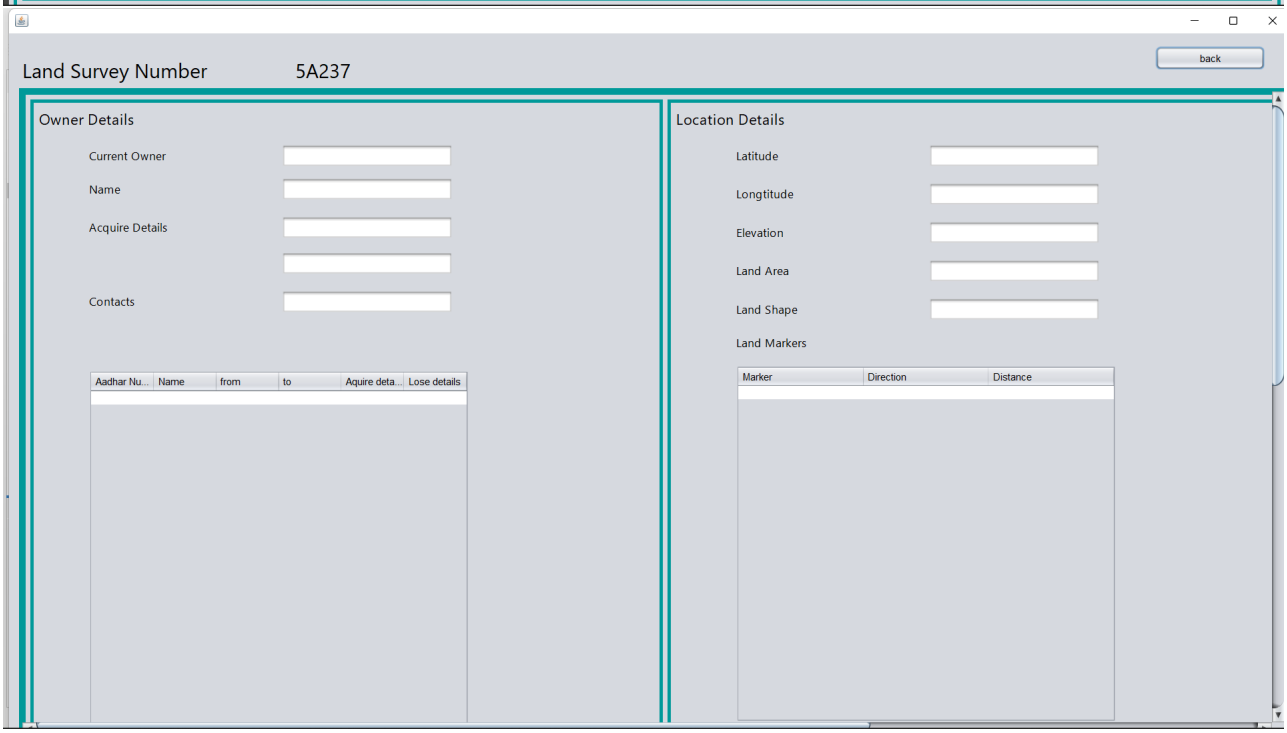
ER Diagram



GUI







Software Tools

java.sql.Connection
java.sql.DriverManager
java.sql.SQLException
javax.swing.JOptionPane
javax.swing.table.DefaultTableModel
java.sql.PreparedStatement

java.sql.ResultSet

Implementation

Login and User identification

```
try {
        Class.forName("com.mysql.jdbc.Driver");
        Connection connection =
        DriverManager.getConnection("jdbc:mysql://localhost:3306/
        landuse", "root", "login");
        PreparedStatement statement = (PreparedStatement) connection.prepareStatement
        ("Select * from users where username = ? and password = ?");
        String usernamestaff = JTextUsernameStaff.getText();
        String passwordstaff = JpasswordStaff.getText();
        statement.setString(1, usernamestaff);
        statement.setString(2, passwordstaff);
        ResultSet result = statement.executeQuery();
        if (result.next()){
            if (result.getString(4).equals("M")){
                System.out.print("Succsefully Logged in as Main staff");
                MainStaff_Page run = new MainStaff_Page
                (usernamestaff, result.getInt(3));
                run.setVisible(true);
            }
        }
        else {
            JOptionPane.showMessageDialog(LoginStaff,"Wrong Username or Password");
        }
        result.close();
        statement.close();
        connection.close();
    }
    catch (SQLException | ClassNotFoundException e) {
        JOptionPane.showMessageDialog(null, e);
    }
```

Searching for entry and adding it to JTable

```
try {
        Class.forName("com.mysql.jdbc.Driver");
        Connection connection =
```

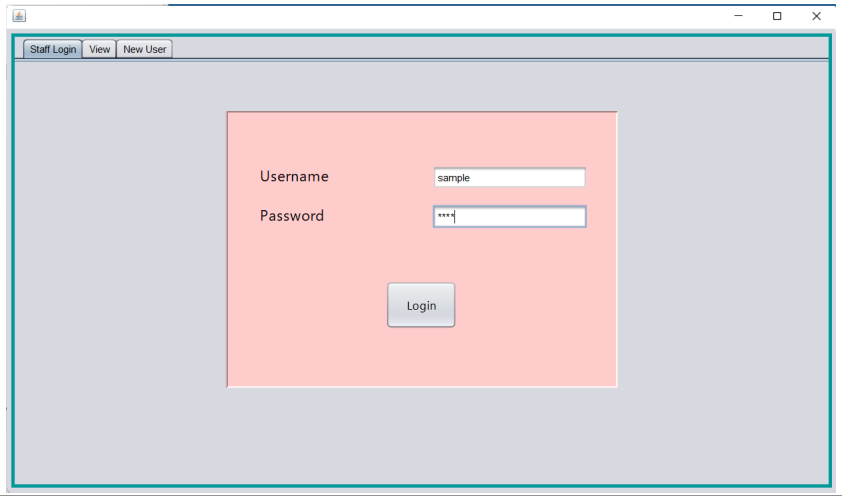
```
        DriverManager.getConnection("jdbc:mysql://localhost:3306/
        landuse", "root", "login");
        PreparedStatement statement = (PreparedStatement)connection.prepareStatement
        ("select * from land_details, land_and_current_owners where
        land_details.land_survey_number =
        land_and_current_owners.land_survey_number and
        land_details.land_survey_number = ?");
        statement.setString(1, Land_Number);
        ResultSet result = statement.executeQuery();
        while (result.next()){
            model.addRow(new Object[]{
                result.getString(1),
                result.getInt(15),
                result.getString(2),
                result.getString(7),
                result.getFloat(8),
                result.getFloat(11),
                result.getFloat(13),
                result.getFloat(4),
                result.getFloat(3),
                result.getFloat(5)
            });
        }
    }
    catch (SQLException | ClassNotFoundException e) {
        JOptionPane.showMessageDialog(null, e);
    }
```

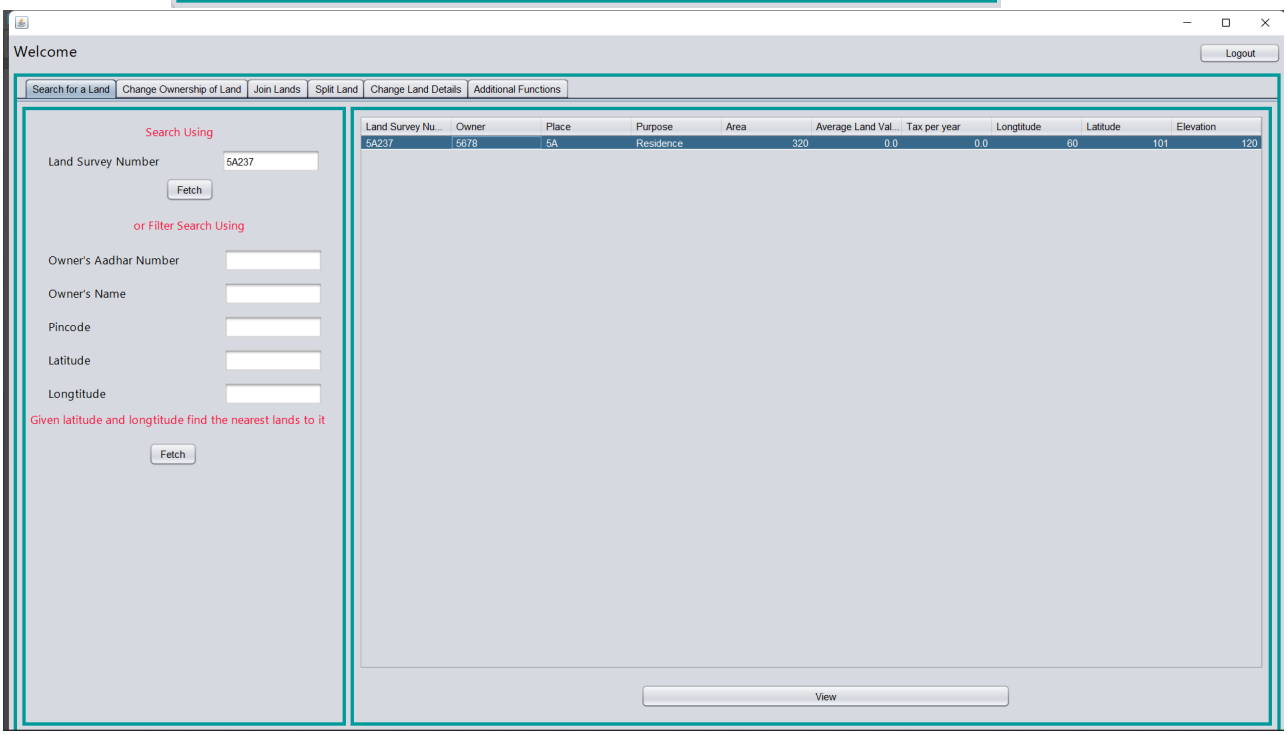
Changing the entries of a row

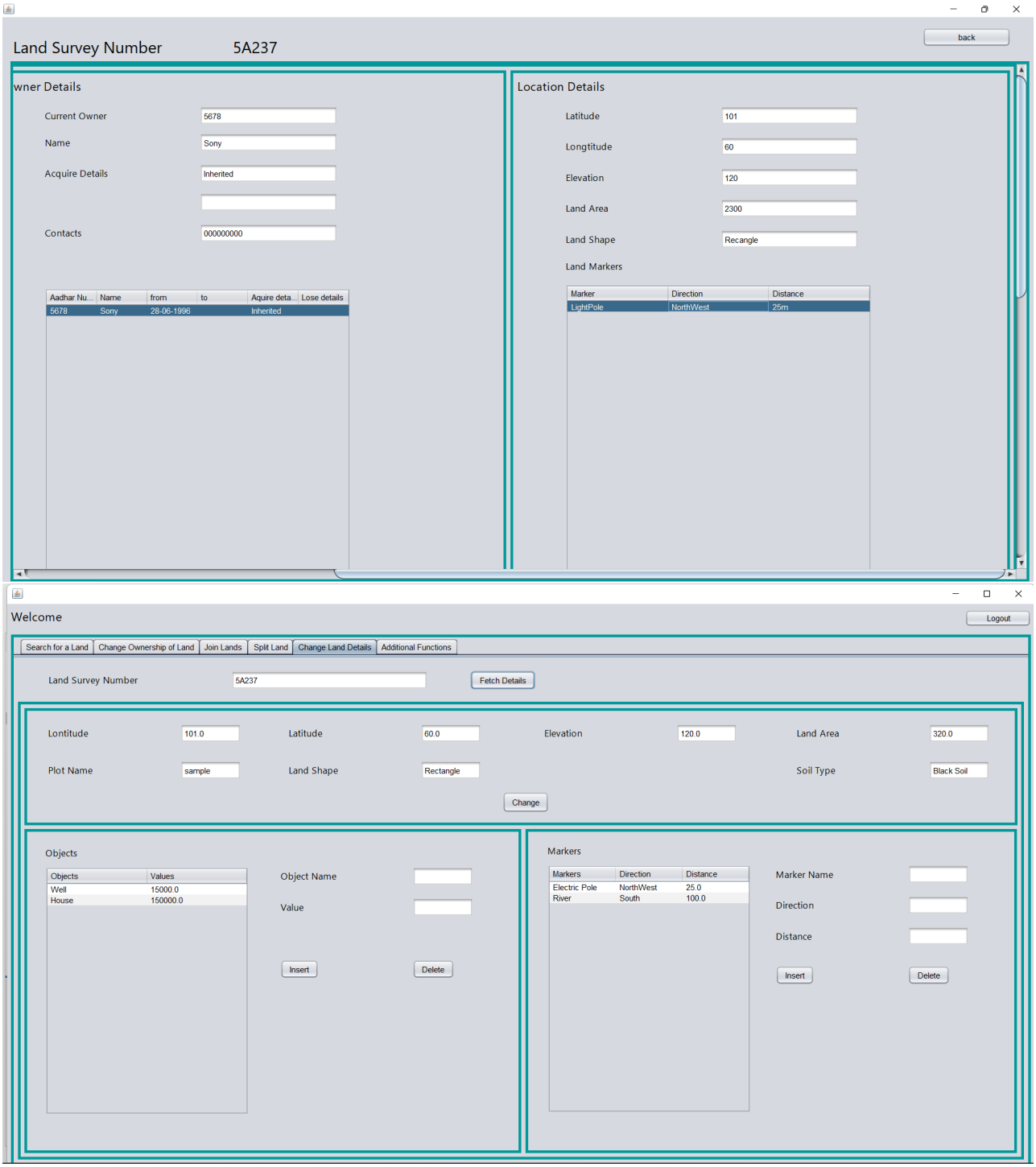
```
try{
        Class.forName("com.mysql.jdbc.Driver");
        Connection connection =
        DriverManager.getConnection("jdbc:mysql://localhost:3306/
        landuse", "root", "login");
        PreparedStatement statement = (PreparedStatement)connection.prepareStatement
        ("update land_details set longtitude = ? , latitude = ? ,
        elevation = ?, land_area = ? , plot_name = ? ,
        land_shape = ? , Soil_type = ? where land_survey_number = ?;");
        statement.setString(1, lontitude);
        statement.setString(2, latitude);
        statement.setString(3, elevation);
```

```
            statement.setString(4, area);
            statement.setString(5, plot);
            statement.setString(6, shape);
            statement.setString(7, soil);
            statement.setString(8, land_number);
            statement.execute();
            JOptionPane.showMessageDialog(this , "Successfully updated the values");
        }
        catch (SQLException | ClassNotFoundException e) {
            JOptionPane.showMessageDialog(null, e);
        }
```

Result And Output







Critical Evaluation

Searching

No	Input	Expected Output	Actual Output
1	Survey Number = '5A237'	Entry 5A237 in table	Entry 5A237 in table
2	Survey Number = '2B236'	Entry 2B236 in table	Nothing added to table
3	Pin code = 555555	All Entries with pin 55555	All Entries with pin 55555
4	Pin code = 5B369	All Entries with pin 5B369	Error

No	Success / Reason for Failure
1	No Entry
2	Success
3	Entry not in the Database

No	Success / Reason for Failure
1	Success
2	No Element with Survey Number = '2B236' in Database
3	Success
4	Characters not allowed in pincode

Updating Values / Fetching Details

No	Input	Expected Output	Actual Output
1	No input	Fetch Some Details	Error Showing no entry
2	Survey Number = '5A237'	Fetch Details of '5A237'	Fetch Details of '5A237'
3	Survey Number = '2B236'	Fetch Details of '2B236'	No Such Entry in Database

Login

No	Input	Expected Output	Actual Output	Success / Reason for
1	username = user,pswd = correct	Login	Login	Success
2	username = notu,pswd = correct	Login	Fail to Login	Given Username not
3	username = user,pswd = wrong	Login	Fail to Login	Wrong Password
4	No username and pswd	Login	No Entry Given	No Entry Given

Conclusion

The Landuse Management System is a Application Software that records the details of lands. Everytime a change happens it is recorded so it is possible to retrieve past data. It also tracks the tax on the land and its payment.

References

MySQL
JAVA Database Connectivity
JAVA Swing