MAWLANA BHASHANI
SCIENCE AND TECHNOLOGY UNIVERSITY

Class Assignment

Assignment Name : Database Problem Solutions in Oracle
Assignment No : 01
Course Code : CSE-3208
Course Title : Relational Database Management System Lab
Date of Submission : 15 - 09 - 2018

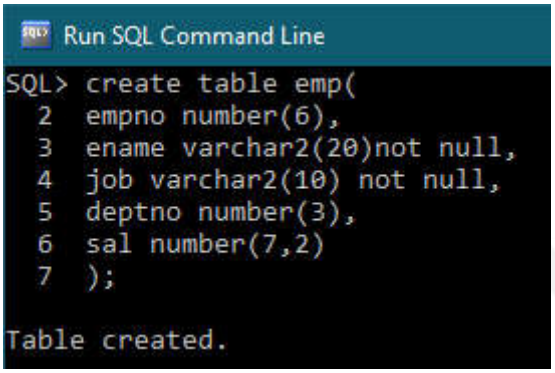
Submitted by	Submitted to
Name : MD. ASHEF SHAHRIOR **ID** : CE-15007 **Batch** : 3rd YEAR 2nd SEMESTER **Department** : COMPUTER SCIENCE AND ENGINEERING	MD. MAHFUZ REZA Asst. Professor Dept. Of Computer Science & Engineering MAWLANA BHASHANI SCIENCE AND TECHNOLOGY UNIVERSITY

Exercise Number: 1

Title of the Exercise: DATA DEFINITION LANGUAGE (DDL) COMMANDS

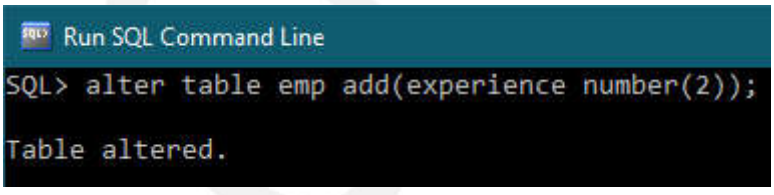
Q1. Create a table called EMP with the following structure.
Name Type
--------- --------------------
EMPNO NUMBER(6)
ENAME VARCHAR2(20)
JOB VARCHAR2(10)
DEPTNO NUMBER(3)
SAL NUMBER(7,2)
Allow NULL for all columns except ename and job.

Ans.



Q2: Add a column experience to the emp table. experience numeric null allowed.

Ans. alter table emp add(experience number(2));



Q3: Modify the column width of the job field of emp table.

Ans. alter table emp modify(job varchar2(12));

```
SQL> alter table emp modify(job varchar2(12));

Table altered.

SQL> desc emp;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------
 EMPNO                                              NUMBER(6)
 ENAME                                     NOT NULL VARCHAR2(10)
 JOB                                       NOT NULL VARCHAR2(12)
 DEPTNO                                             NUMBER(3)
 SAL                                                NUMBER(7,2)
 EXPERIENCE                                         NUMBER(2)
```

Q4: Create dept table with the following structure.
Name Type

DEPTNO NUMBER(2)
DNAME VARCHAR2(10)
LOC VARCHAR2(10)
Deptno as the primarykey

Ans.

Run SQL Command Line

```
SQL> create table dept(
  2  deptno number(2) primary key,
  3  dname varchar2(10),
  4  loc varchar2(10)
  5  );

Table created.
```

Q5: create the emp1 table with ename and empno, add constraints to check the empno value while entering (i.e) empno > 100.

Ans.

Run SQL Command Line

```
SQL> create table emp1(
  2  ename varchar2(10),
  3  empno number(6) constraint ch check(empno>100)
  4  );

Table created.
```

Q6: drop a column experience to the emp table.

Ans. alter table emp drop column experience;

```
SQL> alter table emp drop column experience;

Table altered.

SQL> desc emp;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------
 EMPNO                                              NUMBER(6)
 ENAME                                     NOT NULL VARCHAR2(10)
 JOB                                       NOT NULL VARCHAR2(12)
 DEPTNO                                             NUMBER(3)
 SAL                                                NUMBER(7,2)
```

Q7: Truncate the emp table and drop the dept table.

Ans. drop table dept;

```
Run SQL Command Line

SQL> truncate table emp;

Table truncated.
```

Exercise Number: 2

Title of the Exercise: DATA MANIPULATION LANGUAGE (DML) COMMANDS

Q1: Insert a single record into dept table.

Ans.

```
Run SQL Command Line
SQL> insert into dept values (1,'CSE','MBSTU');

1 row created.
```

Q2: Insert more than a record into emp table using a single insert command.

Ans. insert into emp values(&empno, '&ename', '&job', &deptno, &sal);

```
SQL> insert into emp values(&empno,'&ename','&job',&deptno,&sal);
Enter value for empno: 1
Enter value for ename: Mahi
Enter value for job: Coder
Enter value for deptno: 1
Enter value for sal: 50000
old   1: insert into emp values(&empno,'&ename','&job',&deptno,&sal)
new   1: insert into emp values(1,'Mahi','Coder',1,50000)

1 row created.

SQL> insert into emp values(2,'Sobuj','Tester',2,42000);

1 row created.
```

Q3: Update the emp table to set the salary of all employees to Rs15000/- who are working as ASP.

Ans. update emp set sal=45000 where job= 'Tester';

```
SQL> select * from emp;

     EMPNO ENAME                JOB             DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         1 Mahi                 Coder                 1      50000
         2 Sobuj                Tester                2      42000
         3 Rafi                 Tester                1      42000

SQL> update emp set sal=45000 where job='Tester';

2 rows updated.

SQL> select * from emp;

     EMPNO ENAME                JOB             DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         1 Mahi                 Coder                 1      50000
         2 Sobuj                Tester                2      45000
         3 Rafi                 Tester                1      45000
```

Q4: Create a pseudo table employee with the same structure as the table emp and insert rows into the table using select clauses.

Ans. create table employee as select * from emp;

```
SQL> create table employee as select * from emp;

Table created.

SQL> desc employee;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMPNO                                              NUMBER(6)
 ENAME                                     NOT NULL VARCHAR2(20)
 JOB                                       NOT NULL VARCHAR2(12)
 DEPTNO                                             NUMBER(3)
 SAL                                                NUMBER(7,2)

SQL> select * from employee;

     EMPNO ENAME                JOB             DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         1 Mahi                 Coder                1      50000
         2 Sobuj                Tester               2      45000
         3 Rafi                 Tester               1      45000
```

Q5: select employee name, job from the emp table.

Ans. select ename, job from emp;

```
SQL> select ename, job from emp;

ENAME                JOB
-------------------- ------------
Mahi                 Coder
Sobuj                Tester
Rafi                 Tester
```

Q6: Delete only those who are working as lecturer.

Ans. delete from emp where job='lect';

```
SQL> select * from emp;

     EMPNO ENAME                JOB             DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         1 Mahi                 Coder                1      50000
         2 Sobuj                Tester               2      45000
         3 Rafi                 Tester               1      45000
         4 Robi                 lect                 1      35000

SQL> delete from emp where job='lect';

1 row deleted.

SQL> select * from emp;

     EMPNO ENAME                JOB             DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         1 Mahi                 Coder                1      50000
         2 Sobuj                Tester               2      45000
         3 Rafi                 Tester               1      45000
```

Q7: List the records in the emp table orderby salary in ascending order.

Ans. select * from emp order by sal asc;

```
SQL> select * from emp order by sal asc;

     EMPNO ENAME                JOB              DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         2 Sobuj                Tester                2      45000
         3 Rafi                 Tester                1      45000
         1 Mahi                 Coder                 1      50000
```

Q8: List the records in the emp table orderby salary in descending order.
Ans. select * from emp order by sal desc;

```
SQL>  select * from emp order by sal desc;

     EMPNO ENAME                JOB              DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         1 Mahi                 Coder                 1      50000
         3 Rafi                 Tester                1      45000
         2 Sobuj                Tester                2      45000
```

Q9: Display only those employees whose deptno is 2.
Ans. select * from emp where deptno=2;

```
SQL> select * from emp where deptno=2;

     EMPNO ENAME                JOB              DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         2 Sobuj                Tester                2      45000
```

Q10: Display deptno from the table employee avoiding the duplicated values.
Ans. select distinct deptno from emp;

```
SQL> select distinct deptno from emp;

    DEPTNO
----------
         1
         2
```

Exercise Number: 3

Title of the Exercise: DATA CONTROL LANGUAGE (DCL), TRANSACTION CONTROL LANGUAGE (TCL) COMMANDS

Consider the following tables namely "DEPARTMENTS" and "EMPLOYEES"
Their schemas are as follows ,
Departments (dept _no , dept_ name , dept_location);
Employees (emp_id , emp_name , emp_salary);

Q1: Develop a query to grant all privileges of employees table into departments table.
Ans. grant all on employees to departments;

```
SQL> Grant all on employees to departments;
Grant all on employees to departments
                          *
ERROR at line 1:
ORA-01917: user or role 'DEPARTMENTS' does not exist
```

Q2: Develop a query to grant some privileges of employees table into departments table.
Ans. grant select, update , insert on departments to departments with grant option;

```
SQL> grant select, update , insert on departments to departments with grant option;
grant select, update , insert on departments to departments with grant option
                                                *
ERROR at line 1:
ORA-01917: user or role 'DEPARTMENTS' does not exist
```

Q3: Develop a query to revoke all privileges of employees table from departments table
Ans. revoke all on employees from departments;

```
SQL> revoke all on employees from departments;
revoke all on employees from departments
                         *
ERROR at line 1:
ORA-01917: user or role 'DEPARTMENTS' does not exist
```

Q4: Develop a query to revoke some privileges of employees table from departments table
Ans. revoke select, update , insert on departments from departments;

```
SQL> revoke select, update , insert on departments from departments;
revoke select, update , insert on departments from departments
                                                 *
ERROR at line 1:
ORA-01917: user or role 'DEPARTMENTS' does not exist
```

Q5: Write a query to implement the save point
Ans. savepoint s1;

Q6: Write a query to implement the rollback
Ans. rollback to s1;

Run SQL Command Line

```
SQL> savepoint s1;

Savepoint created.

SQL> select * from emp;

     EMPNO ENAME                JOB              DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         1 Mahi                 Coder                 1      50000
         2 Sobuj                Tester                2      45000
         3 Rafi                 Tester                1      45000
         5 Akalya               AP                    1      10000

SQL> INSERT INTO EMP VALUES(6,'Akshay','ASP',2,10000);

1 row created.

SQL> select * from emp;

     EMPNO ENAME                JOB              DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         1 Mahi                 Coder                 1      50000
         2 Sobuj                Tester                2      45000
         3 Rafi                 Tester                1      45000
         5 Akalya               AP                    1      10000
         6 Akshay               ASP                   2      10000

SQL> rollback to s1;

Rollback complete.

SQL> select * from emp;

     EMPNO ENAME                JOB              DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         1 Mahi                 Coder                 1      50000
         2 Sobuj                Tester                2      45000
         3 Rafi                 Tester                1      45000
         5 Akalya               AP                    1      10000
```

Q6: Write a query to implement the commit
Ans. COMMIT;

```
Run SQL Command Line

Commit complete.

SQL> _
```

Exercise Number: 4
Title of the Exercise: IN BUILT FUNCTIONS

Q1: Display all the details of the records whose employee name starts with 'A'.

Ans. select * from emp where ename like 'A%';

```
SQL>  select * from emp where ename like 'A%';

     EMPNO ENAME                JOB             DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         5 Aranna               AP                    1      10000
```

Q2: Display all the details of the records whose employee name does not starts with 'A'.
Ans. select * from emp where ename not line 'A%';

```
SQL>  select * from emp where ename not like 'A%';

     EMPNO ENAME                JOB             DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         1 Mahi                 Coder                 1      50000
         2 Sobuj                Tester                2      45000
         3 Rafi                 Tester                1      45000
```

Q3: Display the rows whose salary ranges from 30000 to 40000.
Ans. select * from emp where sal between 30000 and 40000;

```
SQL> select * from emp where sal between 30000 and 40000;

     EMPNO ENAME                JOB             DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         5 Aranna               AP                    1      32000
         4 Nila                 ASP                   2      40000
```

Q4: Calculate the total and average salary amount of the emp table.

Ans. select sum(sal),avg(sal) from emp;

```
SQL> select sum(sal),avg(sal) from emp;

  SUM(SAL)   AVG(SAL)
---------- ----------
    209000      41800
```

Q5: Count the total records in the emp table.

Ans. select count(*) from emp;

```
     EMPNO ENAME                JOB              DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         1 Mahi                 Coder                 1      50000
         2 Sobuj                Tester                2      42000
         3 Rafi                 Tester                1      45000
         5 Aranna               AP                    1      32000
         4 Nila                 ASP                   2      40000

SQL> select count(*) from emp;

  COUNT(*)
----------
         5
```

Q6: Determine the max and min salary and rename the column as max_salary and min_salary.

Ans. select max(sal) as max_salary, min(sal) as min_salary from emp;

```
SQL>  select max(sal) as max_salary, min(sal) as min_salary from emp;

MAX_SALARY MIN_SALARY
---------- ----------
     50000      32000
```

Q7: Display the month between “1-jun-10” and “1-aug-10” in full.

Ans. select months_between ('1-jun-2010','1-aug-2010') from dual;

```
SQL> select months_between ('1-jun-2010','1-aug-2010') from dual;

MONTHS_BETWEEN('1-JUN-2010','1-AUG-2010')
-----------------------------------------
                                       -2
```

Q8: Display the last day of that month in —05-Oct-09‖.

Ans. select last_day ('1-jun-2009') from dual;

```
SQL> select last_day ('1-jun-2009') from dual;

LAST_DAY(
---------
30-JUN-09
```

Q9: Find how many job titles are available in employee table.
Ans. select count(job) from emp;

```
SQL> select count(job) from emp;

COUNT(JOB)
----------
         5
```

Q10: What is the difference between maximum and minimum salaries of employees in the organization?
Ans. select max(sal), min(sal) from emp;

```
SQL> select max(sal), min(sal) from emp;

  MAX(SAL)   MIN(SAL)
---------- ----------
     50000      32000
```

Exercise Number: 5

Title of the Exercise: NESTED QUERIES AND JOIN QUERIES

Q1: Display all employee names and salary whose salary is greater than minimum salary of the company and job title starts with 'M'.

Ans. select ename, sal from emp where sal>(select min(sal) from emp where job like 'A%');

```
SQL> select ename,sal from emp where sal>(select min(sal) from emp where job like 'A%'
);

ENAME                       SAL
-------------------- ----------
Mahi                      50000
Sobuj                     42000
Rafi                      45000
Nila                      40000
```

Q2: Issue a query to find all the employees who work in the same job as Sobuj.

Ans. select ename from emp where job=(select job from emp where ename='Sobuj');

```
SQL> select ename from emp where job=(select job from emp where ename='Sobuj');

ENAME
--------------------
Sobuj
Rafi
```

Q3: Issue a query to display information about employees who earn more than any employee in dept 2.

Ans: select * from emp where sal>(select max(sal) from emp where empno=2);

```
SQL>  select * from emp where sal>(select max(sal) from emp where empno=2);

     EMPNO ENAME                JOB              DEPTNO        SAL
---------- -------------------- ------------ ---------- ----------
         1 Mahi                 Coder                 1      50000
         3 Rafi                 Tester                1      45000
```

Q4: Display the employee details, departments that the departments are same in both the emp and dept.

Ans. select * from emp,dept where emp.deptno=dept.deptno;

```
SQL> select * from emp,dept where emp.deptno=dept.deptno;

     EMPNO ENAME      JOB                DEPTNO        SAL     DEPTNO DNAME      LOC
---------- ---------- --------------- ---------- ---------- ---------- ---------- ----------
         1 Mahi       Coder                   1      50000          1 Accounting New York
         2 Sobuj      Tester                  2      42000          2 Research   Dallas
         3 Rafi       Tester                  1      45000          1 Accounting New York
         4 Nila       ASP                     2      40000          2 Research   Dallas
         5 Aranna     AP                      1      32000          1 Accounting New York
```

Q5: Display the employee details, departments that the departments are not same in both the emp and dept.

Ans. select * from emp,dept where emp.deptno!=dept.deptno;

```
SQL> select * from emp,dept where emp.deptno!=dept.deptno;

     EMPNO ENAME      JOB                DEPTNO        SAL     DEPTNO DNAME      LOC
---------- ---------- --------------- ---------- ---------- ---------- ---------- ----------
         2 Sobuj      Tester                  2      42000          1 Accounting New York
         4 Nila       ASP                     2      40000          1 Accounting New York
         1 Mahi       Coder                   1      50000          2 Research   Dallas
         3 Rafi       Tester                  1      45000          2 Research   Dallas
         5 Aranna     AP                      1      32000          2 Research   Dallas
         1 Mahi       Coder                   1      50000         30 Sales      Chicago
         2 Sobuj      Tester                  2      42000         30 Sales      Chicago
         3 Rafi       Tester                  1      45000         30 Sales      Chicago
         4 Nila       ASP                     2      40000         30 Sales      Chicago
         5 Aranna     AP                      1      32000         30 Sales      Chicago
         1 Mahi       Coder                   1      50000         40 Operations Boston

     EMPNO ENAME      JOB                DEPTNO        SAL     DEPTNO DNAME      LOC
---------- ---------- --------------- ---------- ---------- ---------- ---------- ----------
         2 Sobuj      Tester                  2      42000         40 Operations Boston
         3 Rafi       Tester                  1      45000         40 Operations Boston
         4 Nila       ASP                     2      40000         40 Operations Boston
         5 Aranna     AP                      1      32000         40 Operations Boston

15 rows selected.
```

```
SQL> select * from stud1;

     REGNO NAME            MARK2      MARK3 RESULT
---------- ---------- ---------- ---------- ----------
       101 john               89         80 pass
       102 Raja               70         80 pass
       103 Sharin             70         90 pass
       104 Sam                90         95 pass
```

```
SQL> select * from stud2;

NAME       GRADE
---------- -----
john       s
raj        s
sam        a
sharin     a
```

Q6: Display the Student name and grade by implementing a left outer join.
Ans. select stud1.name,grade from stud1 left outer join stud2 on stud1.name=stud2.name;

```
SQL> select stud1.name,grade from stud1 left outer join stud2 on stud1.name=stud2.name;

NAME       GRADE
---------- -----
john       s
raj        s
sam        a
sharin     a
```

Q7: Display the Student name, register no, and result by implementing a right outer join.
Ans. select stud1.name, regno, result from stud1 right outer join stud2 on stud1.name = stud2.name;

```
SQL> select stud1.name, regno, result from stud1 right outer join stud2 on stud1.name =
  2  stud2.name;

NAME            REGNO RESULT
---------- ---------- ----------
john               101 pass
raj                102 pass
sharin             103 pass
sam                104 pass
```

Q8: Display the Student name, register no by implementing a full outer join.
Ans. select stud1.name, regno from stud1 full outer join stud2 on stud1.name= stud2.name;

```
SQL> select stud1.name, regno from stud1 full outer join stud2 on stud1.name= stud2.name;

NAME            REGNO
---------- ----------
john               101
raj                102
sam                104
sharin             103
```

Q9: Write a query to display their employee names.
Ans. select distinct ename from emp x, dept y where x.deptno=y.deptno;

```
SQL> select distinct ename from emp x, dept y where x.deptno=y.deptno;

ENAME
----------
Nila
Rafi
Sobuj
Mahi
Aranna
```

Q10: Display the details of those who draw the salary greater than the average salary.
Ans. select distinct * from emp x where x.sal >= (select avg(sal) from emp);

```
SQL> select distinct * from emp x where x.sal >= (select avg(sal) from emp);

     EMPNO ENAME      JOB                 DEPTNO        SAL
---------- ---------- --------------- ---------- ----------
         1 Mahi       Coder                    1      50000
         2 Sobuj      Tester                   2      42000
         3 Rafi       Tester                   1      45000

SQL> (select avg(sal) from emp);

  AVG(SAL)
----------
     41800
```

Exercise Number: 6

Title of the Exercise: SET OPERATORS

Q1: Display all the dept numbers available with the dept and emp tables avoiding duplicates.

Ans. select deptno from emp union select deptno from dept;

```
SQL> select deptno from emp union select deptno from dept;

    DEPTNO
----------
         1
         2
        30
        40
```

Q2: Display all the dept numbers available with the dept and emp tables.

Ans. select deptno from emp union all select deptno from dept;

```
SQL> select deptno from emp union all select deptno from dept;

    DEPTNO
----------
         1
         2
         1
         2
         1
         1
         2
        30
        40

9 rows selected.
```

Q3: **Display all the dept numbers available in emp and not in dept tables and vice versa.**

Ans. select deptno from emp minus select deptno from dept;
select deptno from dept minus select deptno from emp;

```
SQL>  select deptno from emp minus select deptno from dept;

no rows selected

SQL> select deptno from dept minus select deptno from emp;

    DEPTNO
----------
        30
        40
```

Exercise Number: 7

Title of the Exercise: VIEWS

Q1: The organization wants to display only the details of the employees those who are ASP. (Horizontal portioning)

Ans. create view empview as select * from emp where job='ASP';
select * from empview;

```
SQL> create view empview as select * from emp where job='ASP';

View created.

SQL> select * from empview;

     EMPNO ENAME      JOB                 DEPTNO        SAL
---------- ---------- --------------- ---------- ----------
         4 Nila       ASP                      2      40000
         5 Aranna     ASP                      1      32000
```

Q2: The organization wants to display only the details like empno, empname, deptno, deptname of the employees. (Vertical portioning)

Ans. create view empview1 as select ename,sal from emp;

```
SQL> create view empview1 as select ename,sal from emp;

View created.

SQL> select * from empview1;

ENAME             SAL
---------- ----------
Mahi            50000
Sobuj           42000
Rafi            45000
Nila            40000
Aranna          32000
```

Q3: Display all the views generated.
Ans: select * from tab;

Exercise Number: 8

Title of the Exercise: CONTROL STRUCTURE

Q1: write a pl/sql program to swap two numbers

Ans.

```
SQL> declare
  2  a number(10);
  3  b number(10);
  4  c number(10);
  5  begin
  6  dbms_output.put_line('THE PREV VALUES OF A AND B WERE');
  7  dbms_output.put_line(a);
  8  dbms_output.put_line(b);
  9  a:=&a;
 10  b:=&b;
 11  c:=a;
 12  a:=b;
 13  b:=c;
 14  dbms_output.put_line('THE VALUES OF A AND B ARE');
 15  dbms_output.put_line(a);
 16  dbms_output.put_line(b);
 17  end;
 18  /
Enter value for a: 43
old   9: a:=&a;
new   9: a:=43;
Enter value for b: 87
old  10: b:=&b;
new  10: b:=87;
THE PREV VALUES OF A AND B WERE
THE VALUES OF A AND B ARE
87
43

PL/SQL procedure successfully completed.
```

Q2: Write a pl/sql program to find the largest of three numbers.

Ans.

```
SQL> create or replace procedure maxi(a number, b number, c number) as
  2  begin
  3  if a>b then
  4  if a>c then
  5  dbms_output.put_line('biggest is: ' || a);
  6  else dbms_output.put_line('biggest is: ' || c);
  7  end if;
  8  elsif b>c then
  9  dbms_output.put_line('biggest is: ' || b);
 10  else dbms_output.put_line('biggest is: ' || c);
 11  end if;
 12  end;
 13  /

Procedure created.

SQL> begin
  2  maxi(2,1,3);
  3  end;
  4  /
biggest is: 3

PL/SQL procedure successfully completed.
```

Q3: write a pl/sql program to find the total and average of 6 subjects and display the grade.
Ans.

Query:

```
SQL> declare
  2  java number(10);
  3  dbms number(10);
  4  co number(10);
  5  se number(10);
  6  es number(10);
  7  ppl number(10);
  8  total number(10);
  9  avgs number(10);
 10  per number(10);
 11  begin
 12  dbms_output.put_line('ENTER THE MARKS');
 13  java:=&java;
 14  dbms:=&dbms;
 15  co:=&co;
 16  se:=&se;
 17  es:=&es;
 18  ppl:=&ppl;
 19  total:=(java+dbms+co+se+es+ppl);
 20  per:=(total/600)*100;
 21  if java<50 or dbms<50 or co<50 or se<50 or es<50 or ppl<50 then
 22  dbms_output.put_line('FAIL');
 23  elsif per>75 then
 24  dbms_output.put_line('GRADE A');
 25  elsif per>65 and per<75 then
 26  dbms_output.put_line('GRADE B');
 27  elsif per>55 and per<65 then
 28  dbms_output.put_line('GRADE C');
 29  else
 30  dbms_output.put_line('INVALID INPUT');
 31  end if;
 32  dbms_output.put_line('PERCENTAGE IS '||per);
 33  dbms_output.put_line('TOTAL IS '||total);
 34  end;
 35  /
```

Output:

```
Enter value for java: 80
old  13: java:=&java;
new  13: java:=80;
Enter value for dbms: 70
old  14: dbms:=&dbms;
new  14: dbms:=70;
Enter value for co: 89
old  15: co:=&co;
new  15: co:=89;
Enter value for se: 72
old  16: se:=&se;
new  16: se:=72;
Enter value for es: 76
old  17: es:=&es;
new  17: es:=76;
Enter value for ppl: 71
old  18: ppl:=&ppl;
new  18: ppl:=71;
ENTER THE MARKS
GRADE A
PERCENTAGE IS 76
TOTAL IS 458

PL/SQL procedure successfully completed.
```

Q4: Write a pl/sql program to find the sum of digits in a given number.

Ans.

```
SQL> declare
  2  a number;
  3  d number:=0;
  4  sum1 number:=0;
  5  begin
  6  a:=&a;
  7  while a>0
  8  loop
  9  d:=mod(a,10);
 10  sum1:=sum1+d;
 11  a:=trunc(a/10);
 12  end loop;
 13  dbms_output.put_line('sum is'|| sum1);
 14  end;
 15  /
Enter value for a: 678
old   6: a:=&a;
new   6: a:=678;
sum is21

PL/SQL procedure successfully completed.
```

Q5: write a pl/sql program to display the number in reverse order
Ans.

```
SQL> declare
  2  a number;
  3  rev number;
  4  d number;
  5  begin
  6  a:=&a;
  7  rev:=0;
  8  while a>0
  9  loop
 10  d:=mod(a,10);
 11  rev:=(rev*10)+d;
 12  a:=trunc(a/10);
 13  end loop;
 14  dbms_output.put_line('no is '|| rev);
 15  end;
 16  /
Enter value for a: 3265981
old   6: a:=&a;
new   6: a:=3265981;
no is 1895623

PL/SQL procedure successfully completed.
```

Q6: Write a PL / SQL program to check whether the given number is prime or not.
Ans.

```
SQL> declare
  2  a number; c number:=0; i number;
  3  begin
  4  a:=&a;
  5  for i in 1..a
  6  loop
  7  if mod(a,i)=0 then
  8  c:=c+1;
  9  end if;
 10  end loop;
 11  if c=2 then
 12  dbms_output.put_line(a ||' is a prime number');
 13  else
 14  dbms_output.put_line(a ||' is not a prime number');
 15  end if;
 16  end;
 17  /
Enter value for a: 23
old   4: a:=&a;
new   4: a:=23;
23 is a prime number

PL/SQL procedure successfully completed.
```

```
SQL> declare
  2  a number; c number:=0; i number;
  3  begin
  4  a:=&a;
  5  for i in 1..a
  6  loop
  7  if mod(a,i)=0 then
  8  c:=c+1;
  9  end if;
 10  end loop;
 11  if c=2 then
 12  dbms_output.put_line(a ||' is a prime number');
 13  else
 14  dbms_output.put_line(a ||' is not a prime number');
 15  end if;
 16  end;
 17  /
Enter value for a: 27
old   4: a:=&a;
new   4: a:=27;
27 is not a prime number

PL/SQL procedure successfully completed.
```

Q7: Write a PL/SQL program to find the factorial of a given number.
Ans.

```
SQL> declare
  2  n number;f number:=1;
  3  begin
  4  n:=&n;
  5  for i in 1..n
  6  loop
  7  f:=f*i;
  8  end loop;
  9  dbms_output.put_line('the factorial is '|| f);
 10  end;
 11  /
Enter value for n: 13
old   4: n:=&n;
new   4: n:=13;
the factorial is 6227020800

PL/SQL procedure successfully completed.
```

Q8: write a pl/sql code block to calculate the area of a circle for a value of radius varying from 3 to 7. Store the radius and the corresponding values of calculated area in an empty table named areas, consisting of two columns radius & area.

Ans.

```
SQL> declare
  2  pi constant number(4,2):=3.14;
  3  radius number(5):=3; area number(6,2);
  4  begin
  5  while radius<7
  6  loop
  7  area:=pi*power(radius,2);
  8  insert into areas values(radius,area);
  9  radius:=radius+1;
 10  end loop;
 11  end;
 12  /

PL/SQL procedure successfully completed.

SQL> select * from areas;

    RADIUS       AREA
---------- ----------
         3      28.26
         4      50.24
         5       78.5
         6     113.04
```

Q9: write a PL/SQL code block that will accept an account number from the user, check if the users balance is less than minimum balance, only then deduct rs.100/- from the balance. This process is fired on the acct table.

Ans.

```
SQL> declare
  2  mano number(5);
  3  mcb number(6,2);
  4  minibal constant number(7,2):=1000.00;
  5  fine number(6,2):=100.00;
  6  begin
  7  mano:=&mano;
  8  select cur_bal into mcb from acct where acctno=mano;
  9  if mcb<minibal then
 10  update acct set cur_bal=cur_bal-fine where acctno=mano;
 11  end if;
 12  end;
 13  /
Enter value for mano: 855
old   7: mano:=&mano;
new   7: mano:=855;

PL/SQL procedure successfully completed.
```

Before query execution:

```
SQL> select * from acct;

NAME          CUR_BAL     ACCTNO
---------- ---------- ----------
sirius          10000        777
john             1000        765
sam               500        855
peter             800        353
```

After query execution:

```
SQL> select * from acct;

NAME          CUR_BAL     ACCTNO
---------- ---------- ----------
sirius          10000        777
john             1000        765
sam               400        855
peter             800        353
```

Exercise Number: 9

Title of the Exercise: PROCEDURE AND FUNCTION

Q1: Write a procedure to calculate total for the all the students and pass regno, mark1, & mark2 as arguments.

Ans.

Procedure:

```
SQL> create or replace procedure
  2  cal(sno number,mark1 number,mark2 number)
  3  is
  4  tot number(5);
  5  begin
  6  tot:=mark1+mark2;
  7  update itstudent2 set total=tot where regno=sno;
  8  end;
  9  /

Procedure created.
```

Executing procedure:

```
SQL> declare
  2  cursor c1 is select * from itstudent2;
  3  rec itstudent2 % rowtype;
  4  begin
  5  open c1;
  6  loop
  7  fetch c1 into rec;
  8  exit when c1 % notfound;
  9  cal(rec.regno,rec.mark1,rec.mark2);
 10  end loop;
 11  close c1;
 12  end;
 13  /

PL/SQL procedure successfully completed.
```

Before procedure call:

```
SQL> select * from itstudent2;

    REGNO NAME          MARK1      MARK2      TOTAL
--------- --------- ---------- ---------- ----------
      110 ash               99        100
      112 shila             95         90
```

After procedure call:

```
SQL> select * from itstudent2;

    REGNO NAME          MARK1      MARK2      TOTAL
--------- --------- ---------- ---------- ----------
      110 ash               99        100        199
      112 shila             95         90        185
```

Q2: Write a PL/SQL procedure called MULTI_TABLE that takes two numbers as parameter and displays the multiplication of the first parameter till the second parameter.
Ans.

Procedure:

```
SQL> create or replace procedure
  2  MULTI_TABLE (a number, b number) as
  3  mul number;
  4  begin
  5  for i in 1..b
  6  loop
  7  mul:=a*i;
  8  dbms_output.put_line(a || '*' || i || '=' || mul);
  9  end loop;
 10  end;
 11  /

Procedure created.
```

Executing procedure:

```
SQL> begin
  2       MULTI_TABLE(9,12);
  3  end;
  4  /
9*1=9
9*2=18
9*3=27
9*4=36
9*5=45
9*6=54
9*7=63
9*8=72
9*9=81
9*10=90
9*11=99
9*12=108

PL/SQL procedure successfully completed.
```

Q3: Consider the EMPLOYEE (EMPNO, SALARY, ENAME) Table.
Write a procedure raise_sal which increases the salary of an employee. It accepts an employee number and salary increase amount. It uses the employee number to find the current salary from the EMPLOYEE table and update the salary.

Ans.

Procedure:

```
SQL> create or replace procedure
  2  raiseSal(roll int, salInc int)
  3  as
  4  begin
  5  update employees
  6  set Salary=Salary+salInc where ID=roll;
  7  dbms_output.put_line('Update Successful for ID '||roll);
  8  end;
  9  /

Procedure created.
```

Before Procedure Call:

```
SQL> select * from employees;

NAME                 ID     SALARY
---------- ---------- ----------
B                     1        300
M                     2        200
S                     3        400
X                     4        700
```

After Procedure Call:

```
SQL> select * from employees;

NAME                 ID     SALARY
---------- ---------- ----------
B                     1        350
M                     2        200
S                     3        400
X                     4        700
```

Q4: Write a PL/SQL function CheckDiv that takes two numbers as arguments and returns the values 1 if the first argument passed to it is divisible by the second argument, else will return the value 0;
Ans.

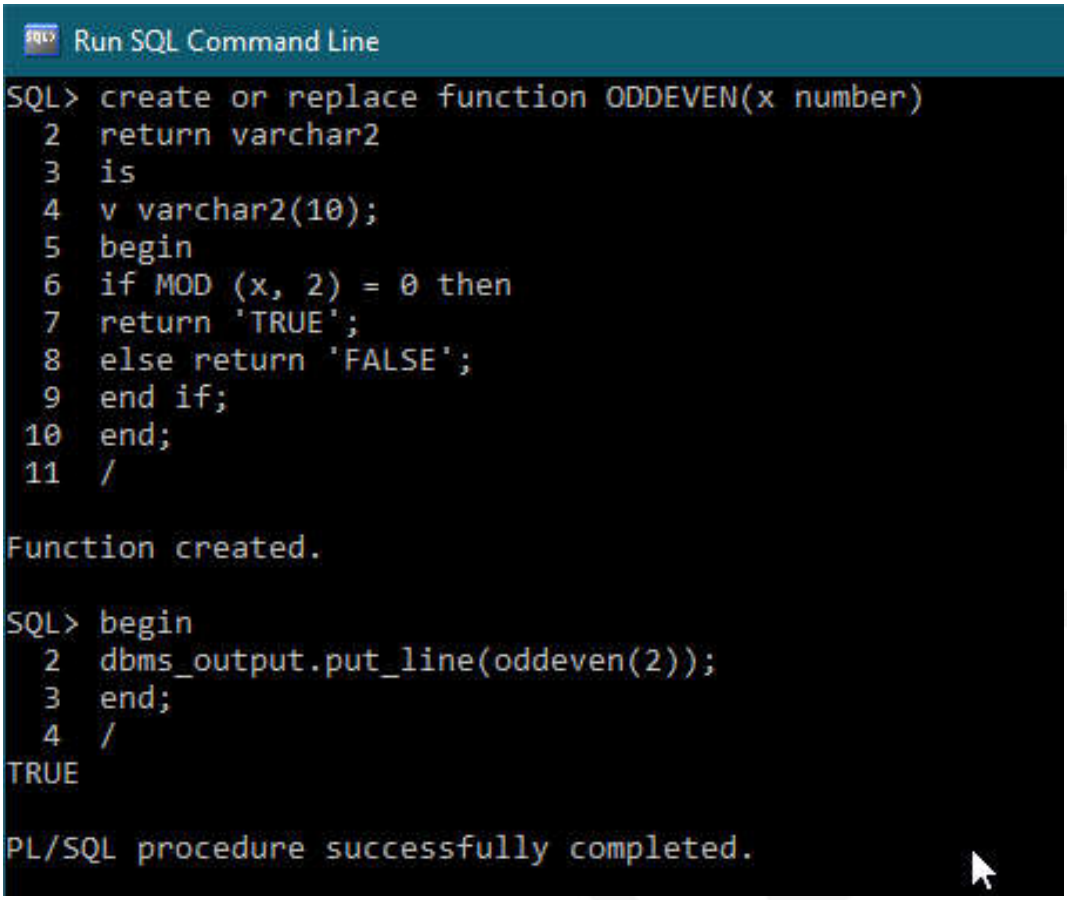
```
Run SQL Command Line
SQL> create or replace function CheckDiv(m number,n number)
  2  return number
  3  is
  4  begin
  5  if MOD (m,n) = 0 then
  6  return 1;
  7  else return 0;
  8  end if;
  9  end;
 10  /

Function created.

SQL> begin
  2  dbms_output.put_line(checkdiv(12,3));
  3  end;
  4  /
1

PL/SQL procedure successfully completed.
```

Q5: Write a PL/SQL function called POW that takes two numbers as argument and return the value of the first number raised to the power of the second.
Ans.

```
SQL> create or replace function POWW(m number,n number)
  2  return number
  3  is
  4  begin
  5  return power(m,n);
  6  end;
  7  /

Function created.

SQL> begin
  2  dbms_output.put_line(poww(8,3));
  3  end;
  4  /
512

PL/SQL procedure successfully completed.
```

Q6: Write a PL/SQL function ODDEVEN to return value TRUE if the number passed to it is EVEN else will return FALSE.

Ans.



Exercise Number: 10

Title of the Exercise: TRIGGER

Q1: Create a trigger that insert current user into a username column of an existing table.

Ans.

```
SQL> create table itstudent4(name varchar2(15),username varchar2(15));

Table created.

SQL> create or replace trigger inuser before insert
  2  on itstudent4
  3  for each row
  4  declare
  5  name varchar2(20);
  6  begin
  7  select user into name from dual;
  8  :new.username:=name;
  9  end;
 10  /

Trigger created.
```

```
SQL> insert into itstudent4 values('ashef','ashahrior');

1 row created.

SQL> insert into itstudent4 values('ashfaq','ashfakmiya');

1 row created.

SQL> select * from itstudent4;

NAME            USERNAME
--------------- ---------------
ashef           SYSTEM
ashfaq          SYSTEM
```

Q2: Create a Simple Trigger that does not allow Insert Update and Delete Operations on the Table.

Ans.

```
SQL> create table items(ename varchar2(10), eid int, salary int);

Table created.

SQL> create trigger barrier
  2  before insert or update or delete
  3  on items
  4  for each row
  5  begin
  6  raise_application_error(-20010,'Manipulation not allowed');
  7  end;
  8  /

Trigger created.

SQL> insert into items values('xyz',11,1000);
insert into items values('xyz',11,1000)
            *
ERROR at line 1:
ORA-20010: Manipulation not allowed
ORA-06512: at "SYSTEM.BARRIER", line 2
ORA-04088: error during execution of trigger 'SYSTEM.BARRIER'
```

Q3: Create a Trigger that raises a User Defined Error Message and does not allow updating and Insertion.

Ans.

```
SQL> create trigger barrier2 before
  2  insert or update on items
  3  for each row
  4  begin
  5  raise_application_error(-20100,'Sorry! No modification allowed');
  6  end;
  7  /

Trigger created.

SQL> insert into items values('xyz',11,1000);
insert into items values('xyz',11,1000)
            *
ERROR at line 1:
ORA-20100: Sorry! No modification allowed
ORA-06512: at "SYSTEM.BARRIER2", line 2
ORA-04088: error during execution of trigger 'SYSTEM.BARRIER2'
```

Q4: develop a query to Drop the Created Trigger.
Ans.

```
SQL> drop trigger barrier;

Trigger dropped.

SQL> drop trigger barrier2;

Trigger dropped.
```

Exercise Number: 15

Title of the Exercise: DATABASE DESIGN AND IMPLEMENTATION

The following is a student information management system GUI application.

Tools for development:

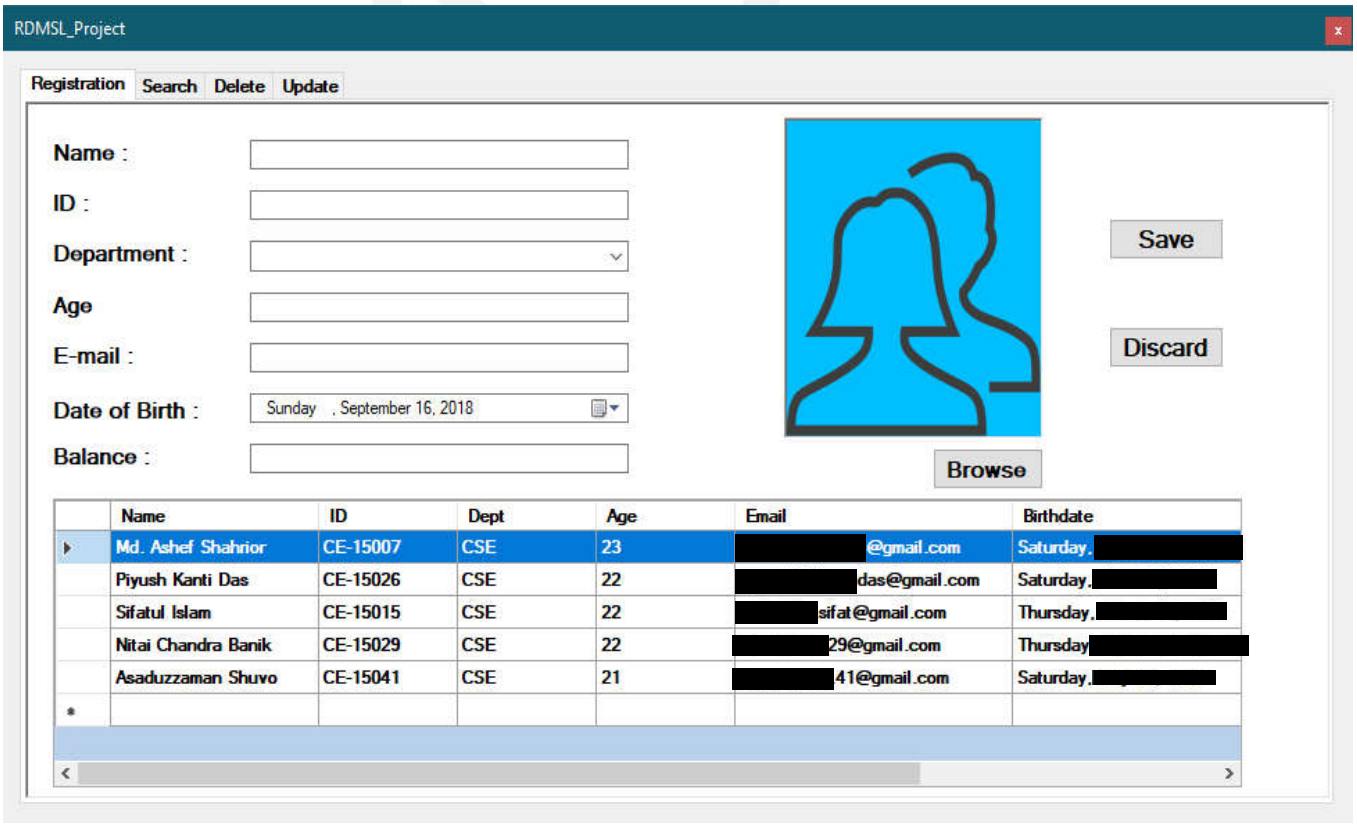
- C#
- Visual Studio 2015
- SQL Server Management Studio 2014

Features:

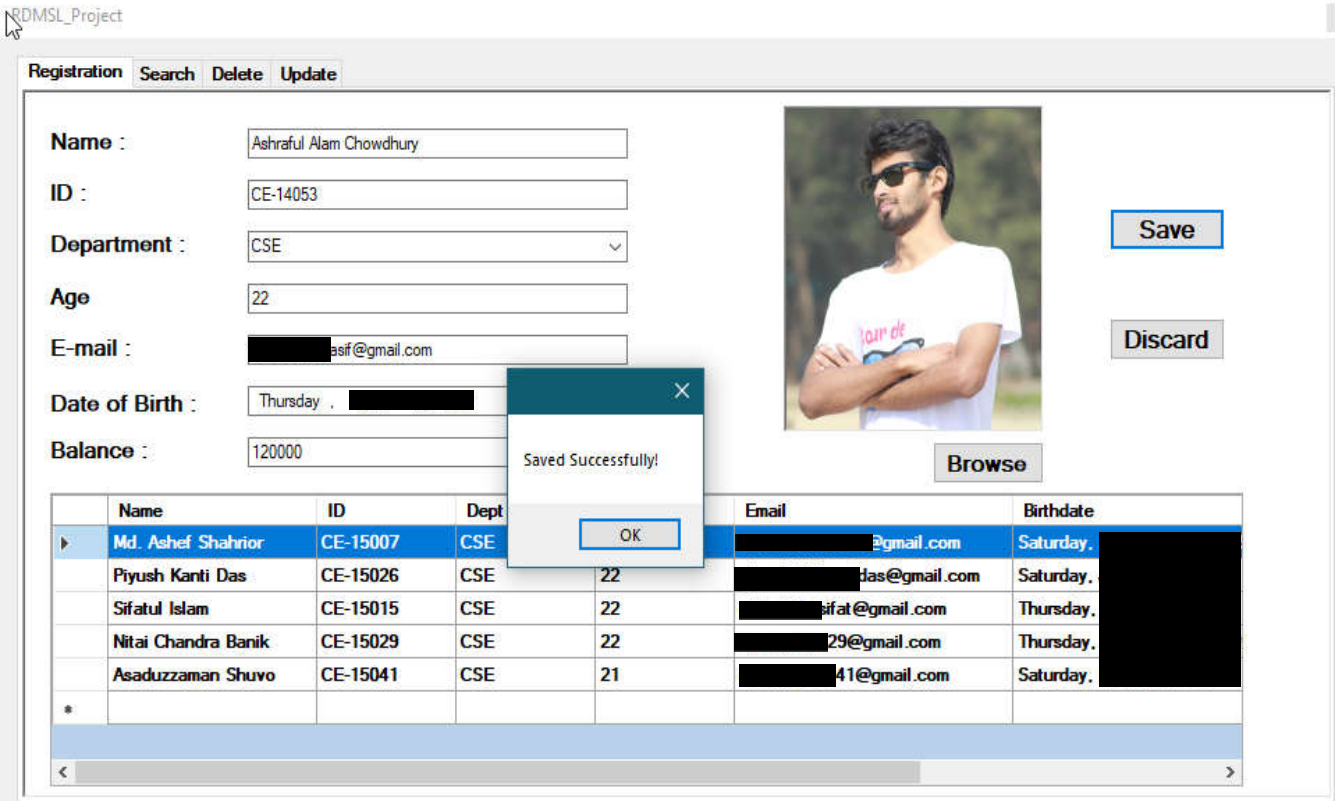
- Data registration into database
- Registered data into PDF format printability
- Search capability for information into database
- Delete information from database
- Update information of database

Walk through of the application: There is only a single form with multiple tabs on it performing different actions on the database table with the help of the application.

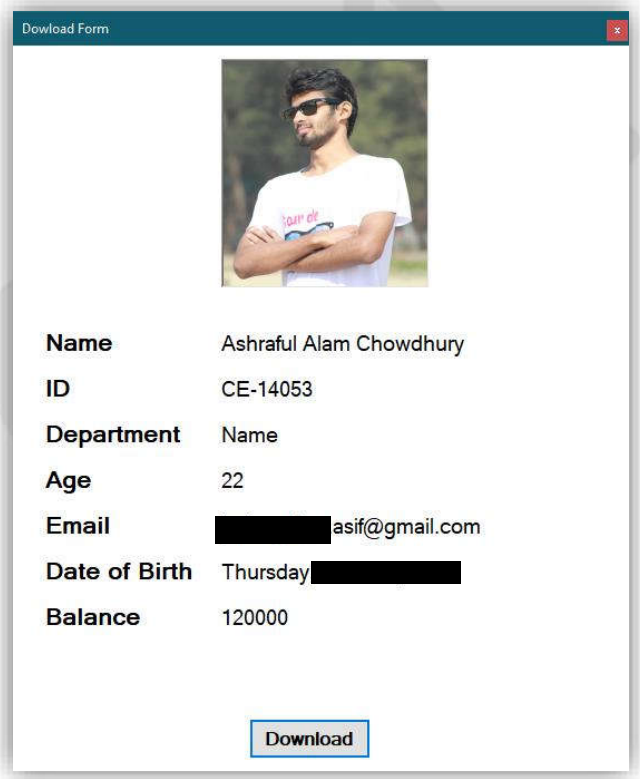
❖ **Student Registration Tab:**
General view -



Data insertion –



Data print –



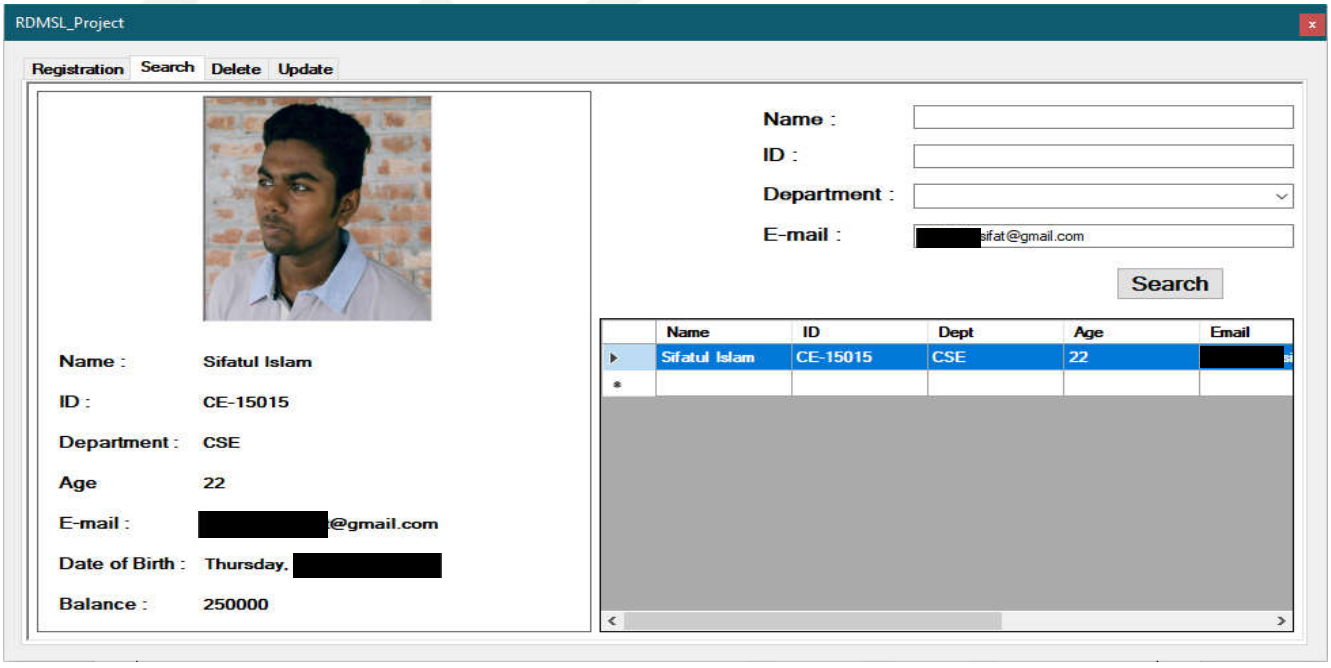
Stored procedure for data insertion –

```
CREATE procedure [dbo].[REGISTER]
  @nameP varchar(50),
  @idP varchar(50),
  @deptP varchar(50),
  @ageP int,
  @emailP varchar(max) = null,
  @bdateP varchar(max),
  @balP int,
  @imgP image = null
 as
 set nocount on
begin
insert into [dbo].[Student_Info]
 ( Name, ID, Dept, Age, Email, Birthdate, Balance, Photo)
 select
 @nameP, @idP, @deptP, @ageP, @emailP, @bdateP, @balP, @imgP
 end
```

Data grid view of the database –

Name	ID	Dept	Age	Email	Birthdate
Md. Ashef Shahrior	CE-15007	CSE	23	[redacted]1994@gmail.com	Saturday, [redacted]
Piyush Kanti Das	CE-15026	CSE	22	[redacted]antidas@gmail.com	Saturday, [redacted]
Sifatul Islam	CE-15015	CSE	22	[redacted]fat@gmail.com	Thursday, [redacted]
Ashraful Alam Chowdhury	CE-14053	CSE	22	[redacted].asif@gmail.com	Thursday, [redacted]
Nitai Chandra Banik	CE-15029	CSE	22	[redacted]29@gmail.com	Thursday, [redacted]
Asaduzzaman Shuvo	CE-15041	CSE	21	[redacted]41@gmail.com	Saturday, [redacted]

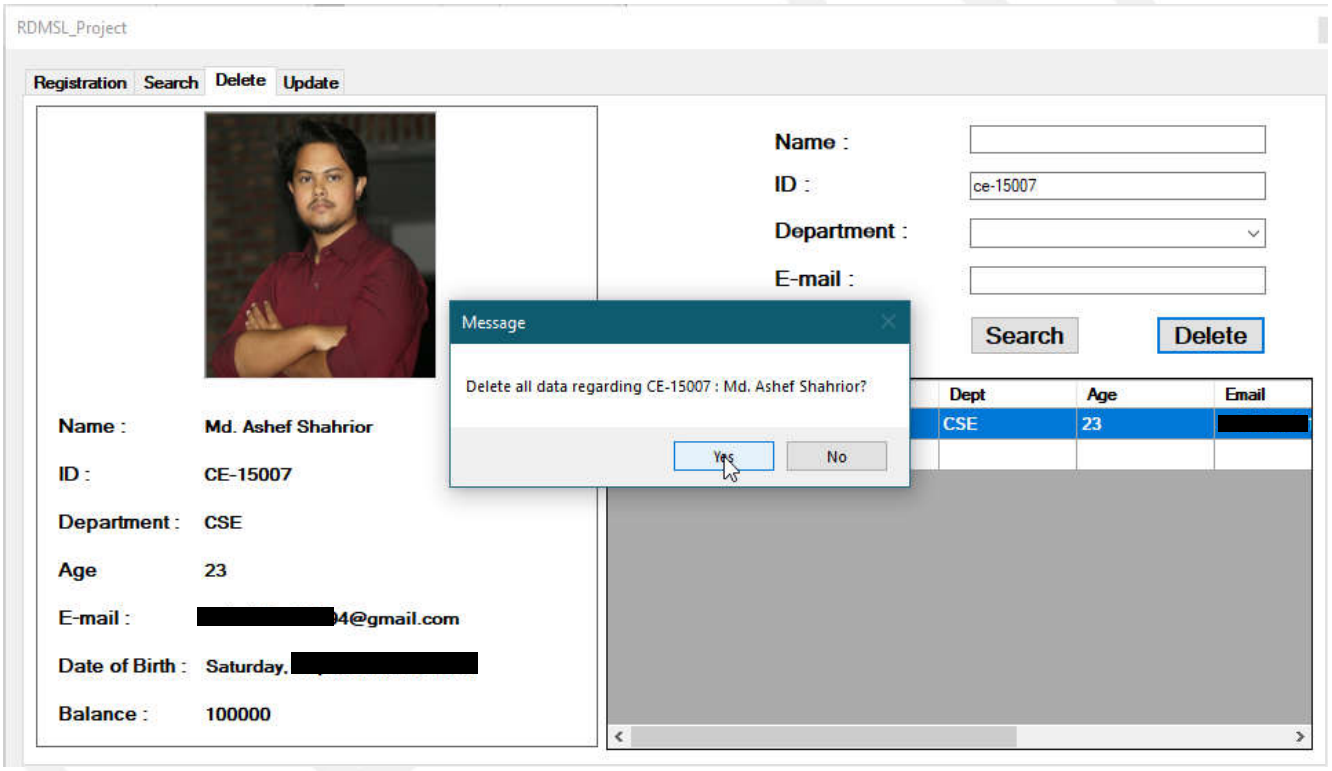
❖ **Student information Search Tab:**



Stored procedure for searching data –

```
CREATE procedure [dbo].[Search]
  @nameP varchar(50) = null,
  @idP varchar(50) = null,
  @deptP varchar(50) = null,
  @emailP varchar(max) = null
 as
 set nocount on
begin
select * from [dbo].[Student_Info]
 as c where c.Name like @nameP or c.ID = @idP or c.Dept = @deptP or c.Email = @emailP
 end
```

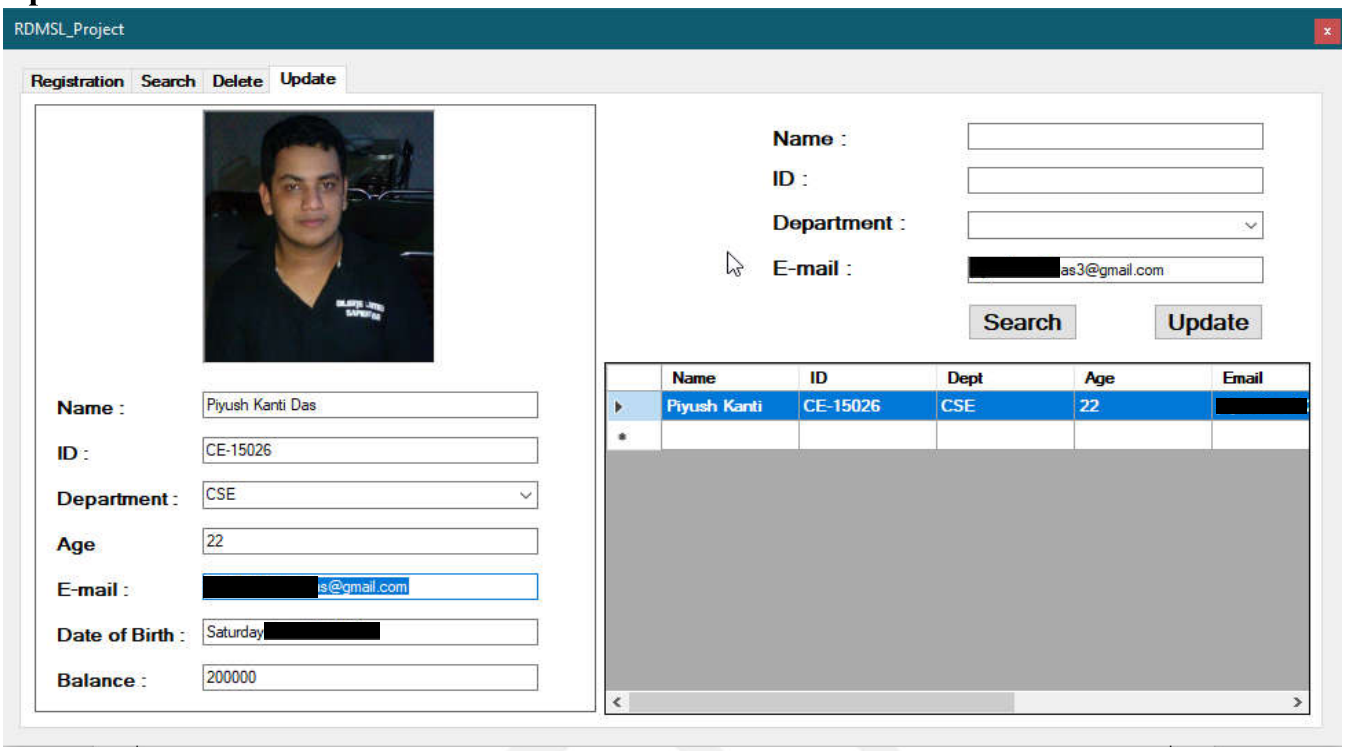
❖ Delete information Tab:



Stored procedure for data deletion –

```
CREATE procedure [dbo].[DELETE_DATA]
  @idP varchar(50)
 as
 set nocount on
begin
delete from [dbo].[Student_Info]
 where ID = @idP
 end
```

❖ Update information Tab:



Store procedure for updating data table –

```
CREATE procedure [dbo].[UPDATE_DATA]
  @idP varchar(50),

  @nameP1 varchar(50),
  @idP1 varchar(50),
  @deptP1 varchar(50),
  @ageP1 int,
  @emailP1 varchar(max),
  @bdateP1 varchar(max),
  @balP1 int
 as
 set nocount on
 begin
 update [dbo].[Student_Info]
 set Name = @nameP1, ID = @idP1, Dept = @deptP1, Age = @ageP1,
 Email = @emailP1, Birthdate = @bdateP1, Balance = @balP1
 where ID = @idP
 end
```