《数据库原理和应用》

课程设计报告

Course Project of Database Principles and Applications

学 号：191764145
中文姓名：艾琳
英文姓名：Aanchal Upreti

南京航空航天大学
国际教育学院

INTRODUCTION

Course description: Describe the nature, academic status, and aims of the course (theory, ability and technique)

1. Course nature and academic status

This course project is for undergraduate students. Understand the role of SQL in the development of applications over relational databases, and use SQL language skillfully. Students will learn how to manipulate data objects (create, store, retrieve, and modify data) and how to develop small applications involving menu design, screen design, report design using Oracle DBMS.

2. Course aims (theory, ability and technique)

This course project covers the following techniques.

- Create/drop tables (including constraints and table population with data) and indexes.
- Updating and removing data.
- Write single table queries using all common clauses (DISTINCT, WHERE, ORDER BY) that include the selection of data based on logical operators, lists, data ranges, character matching, unknown values, and with expressions.
- Write queries with aggregate functions.
- Write JOIN queries and associated subqueries.
- Write queries with string, mathematical, conversion, and date/time functions.
- Create/drop views, and query these database objects.
- Develop small applications involving menu design, screen design, report design using Oracle DBMS(Pro*C/C++ or ODBC).

Requirements for courses; ability and knowledge in advance

- Programming Language
- Discrete Mathematics
- Data Structures

Course structure explanation:

Make clear the necessary parts, optional parts, distribution of hours. Courses with experiments or practice are expected to explain hours needed, content, scheme and functions.

Experiment 1: SQL data definition and data insertion

1. CREATE TABLE

The database schema consists of the three relations, whose schemas are:

S (Sno, Sname, Sgender, Sage, Sdept) // students(SID, name, gender, age, department)
SC (Sno, Cno, Grade) //Course(SID, CID, grade)
C (Cno, Cname, Cpno, Ccredit) //courses (CID, course name, prerequisite courses, credit)

```
SQL> CREATE TABLE S
  2  (
  3  Sno NUMBER(10),
  4  Sname VARCHAR(40),
  5  Ssex CHAR(6),
  6  Sage NUMBER(2),
  7  Sdept VARCHAR(40),
  8  PRIMARY KEY (Sno)
  9  );

Table created.
```

```
SQL> DESC S;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SNO                                       NOT NULL NUMBER(10)
 SNAME                                              VARCHAR2(40)
 SSEX                                               CHAR(6)
 SAGE                                               NUMBER(2)
 SDEPT                                              VARCHAR2(40)

SQL> CREATE TABLE C
  2  (
  3  Cno NUMBER(10),
  4  Cname CHAR(20) NOT NULL,
  5  Cpno NUMBER(5),
  6  Ccredit NUMBER(2),
  7  PRIMARY KEY (Cno)
  8  );

SQL> DESC C
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CNO                                       NOT NULL NUMBER(10)
 CNAME                                     NOT NULL CHAR(20)
 CPNO                                               NUMBER(5)
 CCREDIT                                            NUMBER(2)

SQL> CREATE TABLE SC
  2  (
  3  Sno NUMBER(10),
  4  Cno NUMBER(10),
  5  GRADE NUMBER(3),
  6  PRIMARY KEY (Sno,Cno)
  7  );

SQL> desc sc
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SNO                                       NOT NULL NUMBER(10)
 CNO                                       NOT NULL NUMBER(10)
 GRADE                                              NUMBER(3)
```

2. DROP TABLE, ALTER TABLE, CREATE INDEX, DROP INDEX and INSERT statement to enter data.

Drop Table

```
SQL> CREATE TABLE skuy
  2  (
  3  Sno NUMBER(10),
  4  Sname VARCHAR(30),
  5  Sage NUMBER(2)
  6  );

Table created.

SQL> DROP TABLE skuy;

Table dropped.
```

Alter Table

- **Alter ADD**

```
SQL> ALTER TABLE S ADD (Major CHAR(30));

Table altered.

SQL> DESC S
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SNO                                       NOT NULL NUMBER(10)
 SNAME                                              VARCHAR2(40)
 SSEX                                               CHAR(6)
 SAGE                                               NUMBER(2)
 SDEPT                                              VARCHAR2(40)
 MAJOR                                              CHAR(30)
```

- **Alter Modify**

```
SQL> ALTER TABLE S MODIFY (Major CHAR(10));

Table altered.

SQL> DESC S
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SNO                                       NOT NULL NUMBER(10)
 SNAME                                              VARCHAR2(40)
 SSEX                                               CHAR(6)
 SAGE                                               NUMBER(2)
 SDEPT                                              VARCHAR2(40)
 MAJOR                                              CHAR(10)
```

- **Alter RENAME COLUMN**

```
SQL> ALTER TABLE S RENAME COLUMN Major TO MJR;

Table altered.

SQL> DESC S
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SNO                                       NOT NULL NUMBER(10)
 SNAME                                              VARCHAR2(40)
 SSEX                                               CHAR(6)
 SAGE                                               NUMBER(2)
 SDEPT                                              VARCHAR2(40)
 MJR                                                CHAR(10)
```

- **Alter DROP**

```
SQL> ALTER TABLE S DROP COLUMN MJR;

Table altered.

SQL> DESC S
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SNO                                       NOT NULL NUMBER(10)
 SNAME                                              VARCHAR2(40)
 SSEX                                               CHAR(6)
 SAGE                                               NUMBER(2)
 SDEPT                                              VARCHAR2(40)
```

Create Index

```
SQL> CREATE INDEX SAGE_INDEX ON S(Sage);

Index created.
```

Drop index

```
SQL> DROP INDEX SAGE_INDEX;

Index dropped.
```

Insert Data to Table S

```
SQL> INSERT INTO s VALUES (191764101, ‘Abdul’, ‘male’, 21, ‘Software’);
1 row created.

SQL> INSERT INTO s VALUES (191764102, ‘Arnold’, ‘male’, 21, ‘Software’);
1 row created.

SQL> INSERT INTO s VALUES (191764103, ‘Belinda’, ‘female’, 22, ‘Software’);
1 row created.

SQL> INSERT INTO s VALUES (191764104, ‘Clarissa’, ‘female’, 27, ‘Civil’);
1 row created.

SQL> INSERT INTO s VALUES (191764105, ‘Doris’, ‘female’, 24, ‘Art’);
1 row created.

SQL> INSERT INTO s VALUES (191764106, ‘Ediancuk’, ‘male’, 19, ‘Art’);
1 row created.

SQL> INSERT INTO s VALUES (191764107, ‘Gamblis’, ‘male’, 19, ‘Civil’);
1 row created.

SQL> INSERT INTO s VALUES (191764108, ‘Eueue’, ‘male’, 24, ‘Civil’);
1 row created.


SQL> SELECT * FROM S;

SNO      SNAME                          SSEX      SAGE  SDEPT
---------- ---------------------------------------- ------ -------------------------------------------------
191764101   Abdi                      male       21       Software

191764102   Aryan                     male       21       Software

191764103  Bella                    female      22        Software

191764104  Clarke                    female      27       Civil

191764105   Dummy                         female      24  Art

191764106  Ediancuk                    male        19      Art

191764107  Gwenn                     male        19       Civil

191764108   Eliora                      male        24      Civil

8 rows selected.
```

Insert Data To Table C

```
INSERT INTO c VALUES (001, ‘AI', 0, 3);
1 row created.

SQL> INSERT INTO c VALUES (002, ‘DATABASE', 0, 3);
1 row created.

SQL> INSERT INTO c VALUES (00, ‘DATA STRUCTURE', 0, 2);
1 row created.

SQL> INSERT INTO c VALUES (004, ‘DISCRETE MATH', 0, 2);
1 row created.

SQL> INSERT INTO c VALUES (005, ‘C++', 0, 4);
1 row created.

SQL> INSERT INTO c VALUES (006, ‘Probability', 0, 3);
1 row created.

SQL> INSERT INTO c VALUES (007, ‘Classical Art', 0, 4);
1 row created.


SQL> SELECT *FROM C;

     CNO          CNAME                 CPNO                CCREDIT
---------- ------------------- ---------- ----------
       1             AI                    0                3
       2        DATABASE                   0                3
       0   DATA STRUCTURE                  0                2
       4    DISCRETE MATH                  0                2
       5             C++                   0                4
       6             Probability           0                3
       7             Classical Art         0                4

7 rows selected.
```

Insert Data To Table SC

```
SQL> INSERT INTO sc VALUES (191764101, 001, 73);
1 row created.

SQL> INSERT INTO sc VALUES (191764101, 002, 80);
1 row created.

SQL> INSERT INTO sc VALUES (191764101, 00, 80);
1 row created.

SQL> INSERT INTO sc VALUES (191764102, 001, 62);
1 row created.

SQL> INSERT INTO sc VALUES (191764102, 002, 82);
1 row created.

SQL> INSERT INTO sc VALUES (191764102, 00, 78);
1 row created.

SQL> INSERT INTO sc VALUES (191764103, 001, 85);
1 row created.

SQL> INSERT INTO sc VALUES (191764103, 002, 93);
1 row created.

SQL> INSERT INTO sc VALUES (191764103, 00, 84);
1 row created.

SQL> INSERT INTO sc VALUES (191764101, 005, 78);
1 row created.

SQL> INSERT INTO sc VALUES (191764102, 005, 70);
1 row created.

SQL> INSERT INTO sc VALUES (191764103, 005, 86);
1 row created.

SQL> INSERT INTO sc VALUES (191764104, 004, 80);
1 row created.

SQL> INSERT INTO sc VALUES (191764104, 006, 80);
1 row created.

SQL> INSERT INTO sc VALUES (191764105, 007, 95);
1 row created.

SQL> INSERT INTO sc VALUES (191764106, 007, 100);
1 row created.

SQL> INSERT INTO sc VALUES (191764107, 004, 32);
1 row created.

SQL> INSERT INTO sc VALUES (191764107, 006, 66);
1 row created.

SQL> INSERT INTO sc VALUES (191764108, 004, 78);
```

```
1 row created.

SQL> INSERT INTO sc VALUES (191764108, 006, 52);
1 row created.

SQL> SELECT *FROM SC;

       SNO        CNO      GRADE
---------- ---------- ----------
 191764101          1         73
 191764101          2         80
 191764101          0         80
 191764102          1         62
 191764102          2         82
 191764102          0         78
 191764103          1         85
 191764103          2         93
 191764103          0         84
 191764101          5         78
 191764102          5         70
 191764103          5         86
 191764104          4         80
 191764104          6         80
 191764105          7         95
 191764106          7        100
 191764107          4         32
 191764107          6         66
 191764108          4         78
 191764108          6         52

20 rows selected.
```

Experiment 2: Data Queries

1. Find the SIDs and names of all students who have enrolled a number 1 course.

```
SQL> SELECT S.Sno, Sname
  2  FROM S, SC
  3  WHERE S.Sno = SC.Sno AND Cno = 1
  4  ORDER BY Sno;

SNO        SNAME
---------- ----------------------------------------
 191764101 Abdulah
 191764102 Aryan
 191764103 Bella
```

2. Find the SIDs and names of all students who have enrolled a data structure course.

```
SQL> SELECT S.Sno, Sname
  2  FROM S, SC, C
  3  WHERE S.Sno = SC.Sno AND SC.Cno = C.Cno AND c.cname = 'DATA STRUCTURE';

      SNO                   SNAME
---------- ----------------------------------------
 191764101                  Abdullah
 191764102                  Aryan
 191764103                  Bella
```

3. Find the SIDs and names of all students who have not enrolled a number 1 course.

```
SQL> SELECT S.Sno, Sname
  2  FROM S
  3  WHERE Sno not in
  4  (SELECT Sno
  5  FROM SC
  6  WHERE Cno=1)
  7  ORDER BY S.Sno;

      SNO                   SNAME
---------- ----------------------------------------
 191764104                  Clarke
 191764105                  Dummy
 191764106                  Ediancuk
 191764107                  Gwenn
 191764108                  Eliora
```

4. Find the names of all students who have enrolled all courses.

```
SQL> SELECT Sname
  2  FROM S
  3  WHERE NOT EXISTS (SELECT * FROM C
  4  WHERE NOT EXISTS (SELECT * FROM SC
  5  WHERE SC.Sno=S.Sno and SC.Cno=C.Cno));
```

no rows selected (Because there is no student who take all the courses).

5. Find the SIDs and average grade of all students who have passed all courses except for a number 1 course, and the query results are sorted in descending order by average grade.

```
SQL> SELECT Sno, avg(grade)
  2  FROM SC
  3  WHERE SC.grade >= 60 AND Sno in
  4  (SELECT Sno FROM SC WHERE Cno <> 1)
  5  GROUP BY Sno
  6  ORDER BY avg(grade) DESC;
```

```
      SNO           AVG(GRADE)
---------- ----------
 191764106       100
 191764105        95
 191764103        87
 191764104        80
 191764108        78
 191764101      77,75
 191764102        73
 191764107        66
```

6. Find the name of student who has enrolled a database course with the second highest score.

```
SQL> SELECT Sname
  2  FROM S
  3  WHERE Sno in
  4  (SELECT Sno
  5  FROM SC, C
  6  WHERE Cname = 'DATABASE' AND SC.Cno=C.Cno
  7  AND grade = (SELECT MAX(grade)
  8  FROM SC, C
  9  WHERE Cname = 'DATABASE' AND SC.Cno=C.Cno
 10  AND grade not in (SELECT MAX(grade)
 11  FROM SC, C
 12  WHERE Cname = 'DATABASE' AND SC.Cno = C.Cno)));


SNAME
---------------
Arnold
```

7. Find the names of all students who have enrolled at least 3 courses with 3 credits and whose score of each course enrolled is 80 or more.

```
SQL> CREATE VIEW murid(Sno, Cno, grade)AS
  2  SELECT Sno, SC.Cno, grade
  3  FROM SC, C
  4  WHERE grade>=80 AND Ccredit=3 AND SC.Cno=C.Cno;

View created.

SQL> SELECT Sname
  2  FROM S
  3  WHERE Sno IN
  4  (SELECT Sno
  5  FROM murid
  6  GROUP BY Sno HAVING COUNT(*) >= 3);
```

8. Find the SIDs of students whose number of courses taken is unique.

```
SQL> SELECT Sno
  2  FROM SC
  3  GROUP BY Sno HAVING COUNT(*)=1;

       SNO
----------
 191764105
 191764106
```

9. Use SELECT statement to do the queries of various kinds of WHERE conditions

-
  ```
  SQL> SELECT *FROM SC
  2  WHERE grade BETWEEN 75 AND 80;
  ```

```
       SNO        CNO      GRADE
---------- ---------- ----------
 191764101          2         80
 191764101          0         80
 191764102          0         78
 191764101          5         78
 191764104          4         80
 191764104          6         80
 191764108          4         78
```

-
  ```
  SQL> SELECT Sname
  2  FROM S
  3  WHERE Sname LIKE 'E%';
  ```

```
SNAME
----------------------------------------
Ediancuk
Eueue
```

-
  ```
  SQL> SELECT Sname, SC.Cno
   2  FROM S, SC
   3  WHERE Cno=6 AND SC.Sno=S.Sno;
  ```

```
SNAME                                           CNO
---------------------------------------- ----------
Clarissa                                          6
Gamblis                                           6
Eueue                                             6
```

-
  ```
  SQL> SELECT Sname
  2  FROM S
  3  WHERE sname in (SELECT sname FROM
  4  (SELECT sname FROM s INNER JOIN sc
  5  ON s.sno = sc.sno WHERE grade < 90)
  6  GROUP BY sname
  7  HAVING COUNT(sname) >= 2);
  ```

```
SNAME
----------------------------------------
Abdullah
Aryan
Bella
Clarke
Gwenn
Eliora
```

Experiment 3: Modification and deletion of data

1. All grades except for null value in a number 1 course are to be increased by 10 percent.

```
SQL> UPDATE SC
  2  SET grade = grade * (1+0.1)
  3  WHERE Cno = 1 AND grade is not NULL;

3 rows updated.
SQL> SELECT * FROM SC;

       SNO        CNO      GRADE
---------- ---------- ----------
 191764101          1         80
 191764101          2         80
 191764101          0         80
 191764102          1         68
 191764102          2         82
 191764102          0         78
 191764103          1         94
 191764103          2         93
 191764103          0         84
 191764101          5         78
 191764102          5         70
 191764103          5         86
 191764104          4         80
 191764104          6         80
 191764105          7         95
 191764106          7        100
 191764107          4         32
 191764107          6         66
 191764108          4         78
 191764108          6         52
```

2. Delete all records enrolled at data structure course in the SC table.

```
SQL> DELETE FROM SC
  2  WHERE Cno in
  3  (SELECT Cno
  4  FROM C
  5  WHERE Cname = 'DATA STRUCTURE');

3 rows deleted.

SQL> SELECT * FROM SC;

       SNO        CNO      GRADE
---------- ---------- ----------
 191764101          1         80
 191764101          2         80
 191764102          1         68
 191764102          2         82
 191764103          1         94
 191764103          2         93
 191764101          5         78
 191764102          5         70
 191764103          5         86
 191764104          4         80
 191764104          6         80
 191764105         7        95
 191764106         7       100
 191764107         4        32
 191764107         6        66
 191764108         4        78
 191764108         6        52
```

3. Delete all records in the SC and S table whose student number is 95002.

```
SQL> DELETE FROM SC
  2  WHERE Sno = 95002;

0 rows deleted.

SQL> DELETE FROM S
  2  WHERE Sno = 95002;

0 rows deleted.
```

Experiment 4: The operation of view.

1. **Define a view of male students whose attributes include SID, student name, course name and grade enrolled.**

```
SQL> CREATE VIEW cowok(Sno, Sname, Cname, grade)
  2  AS SELECT S.Sno, Sname, Cname, grade
  3  FROM S, SC, C
  4  WHERE S.Sno=SC.Sno AND Ssex='male' AND C.Cno=SC.Cno;

View created.

SQL> SELECT * FROM cowok;

       SNO SNAME                    CNAME                 GRADE
---------- -------------------------------------- --------------------
191764102 Aryan                    AI                    68

191764101 Abdullah                        AI                    80

191764102 Aryan                    DATABASE              82

191764101 Abdullah                        DATABASE              80

191764108 Eliora                   DISCRETE MATH         78

191764107 Gwenn                    DISCRETE MATH         32

191764102 Aryan                    C++                   70

191764101 Abdullah                        C++                   78

191764108 Eliora                   Probability           52

191764107 Gwenn                    Probability           66

191764106 Ediancuk                 Classical Art         100
```

2. **Find the SIDs and names of all students in the previous view with average grade 80 and more.**

```
SQL> SELECT Sno, Sname
  2  FROM S
  3  WHERE Sno IN
  4  (SELECT Sno
  5  FROM cowok
  6  GROUP BY Sno
  7  HAVING AVG(grade)>80);

       SNO SNAME
---------- ---------------------------------------
 191764106 Ediancuk
```

Experiment 5: library function and access control

1. **Calculate the number of courses taken and average grade for each student whose grade is not null.**

```
SQL> SELECT S.Sno, Sname, COUNT(Cno), AVG(grade)
  2  FROM S INNER JOIN SC ON S.Sno = SC.Sno
  3  WHERE grade IS NOT NULL
  4  GROUP BY S.Sno, Sname
  5  ORDER BY S.Sno;

       SNO SNAME                                COUNT(CNO)   AVG(GRADE)
---------- ------------------------------------ ---------- ----------
 191764101 Abdullah                                      3            79,3333333
 191764102 Aryan                             3            73,3333333
 191764103 Bella                             3            91
 191764104 Clarke                            2            80
 191764105 Dummy                                         1            95
 191764106 Ediancuk                          1            100
 191764107 Gwenn                             2            49
 191764108 Eliora                            2             65
```

2. **Using the GRANT statement, grant various privileges on the base tables of S, SC and C to other users.**

```
SQL> GRANT SELECT, INSERT, UPDATE
  2  ON S
  3  TO SYSTEM;
Grant succeeded.

SQL> GRANT SELECT, INSERT, UPDATE
  2  ON C
  3  TO SYSTEM;
Grant succeeded.

SQL> GRANT ALL
  2  ON SC
  3  TO PUBLIC;
Grant succeeded.
```

3. **After successful completion of the experiment, withdraw the base tables and views having been created.**

```
SQL> DROP TABLE S;
Table dropped.

SQL> DROP TABLE C
  2  ;
Table dropped.
```

```
SQL> DROP TABLE SC;
Table dropped.

SQL> DROP VIEW murid;
View dropped.

SQL> DROP VIEW cowok;
View dropped.
```

Experiment 6: comprehensive experiment: the implementation of a small management information system.

Experimentation Medium

- Operating System - Windows 10
- Database - Oracle Database 11g
- IDE - Visual Studio C++ 2019

Running Steps

1. Installing Oracle Database 11g
2. Installing Visual Studio C++ 2019
3. Configuring Visual Studio
   ❖ Specifying Oracle Pro*C Directory
   - ORACLE_BASE\ORACLE_HOME\BIN
   ❖ Specifying Oracle Pro*C Header Files
   - ORACLE_BASE\ORACLE_HOME\PRECOMP\PUBLIC
   - ORACLE_BASE\ORACLE_HOME\OCI\INCLUDE
   ❖ Specifying Oracle Pro*C Library (Inside Project)
   - ORACLE_BASE\ORACLE_HOME\PRECOMP\LIB\orasql11.lib
   - ORACLE_BASE\ORACLE_HOME\PRECOMP\LIB\MSVC\orasqx11.lib
4. Creating .pc Files
5. Compiling .pc Files Using Pro*C Compiler Via Command Prompt

Program source code

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <conio.h>
#include <sqlca.h>

void loginMenu(char*, char*, char*);
void mainMenu();

void tableCreate();
void tableInsert();
void tableUpdate();
void tableDelete();
void tableQuery();
void tableDisp();
void queryInfo();
void quits();

EXEC SQL BEGIN DECLARE SECTION;

char sno[11];
char sname[32];
char sgender[8];
char sage[4];
char sdept[32];

char cno[4];
char cname[22];
char cpno[4];
char ccredit[5];

char grade[5];

char temps1[11];
char temps2[5];

EXEC SQL END DECLARE SECTION;

int main()
{
	EXEC SQL BEGIN DECLARE SECTION;

	char user[10], pass[10], server[10];

	EXEC SQL END DECLARE SECTION;

	loginMenu(user, pass, server);
	mainMenu();

	return 0;
}
```

```
void loginMenu(char* user, char* pass, char* server)
{


	printf("\n");
	printf("---------------------------------Login Menu----------------------------------");
	printf("\n____________________________________________________________________________
__________");

	printf("\nUsername ");
	fgets(user, 10, stdin);
	user[strcspn(user, "\n")] = 0;

	printf("\nPassword ");
	fgets(pass, 10, stdin);
	pass[strcspn(pass, "\n")] = 0;

	printf("\nServer ");
	fgets(server, 10, stdin);
	server[strcspn(server, "\n")] = 0;

	EXEC SQL CONNECT :user IDENTIFIED BY :pass USING :server;

	if (sqlca.sqlcode == 0)
	{
		printf("\n");
		printf("Connecting Success \nUser %s Connected ", user);
	}
	else
	{
		printf("%.*s\n", sqlca.sqlerrm.sqlerrml, sqlca.sqlerrm.sqlerrmc);

		exit(-1);
	}

	while (getchar() != '\n');
}

void mainMenu()
{
	system("cls");

	int input;

	printf("\n\n");
	printf("  ---------------------------------MAIN MENU---------------------------------\n");
	printf("  ---------------------------------------------------------------------------                    \n");
	printf("                    1917641 Class Directory                                                     \n");
	printf("  ---------------------------------------------------------------------------                    \n");
	printf("                      1. Create  Table                 \n");
	printf("                      2. Insert  Table                 \n");
	printf("                      3. Update  Table         \n");
	printf("                      4. Delete  Table                 \n");
	printf("                      5. Query   Table                 \n");
	printf("                      6. Display Table         \n");
```

```
        printf("                    7. Show Information   \n");
        printf("                    8. Exit - Saved               \n");
        printf("                    0. Exit                  \n");
        printf("                    ------------------            \n");
        printf("                  Input Command :  \n\n");

        scanf_s("%d", &input);

                switch (input)
                {
                        case 1: tableCreate();     break;
                        case 2: tableInsert();     break;
                        case 3: tableUpdate();     break;
                        case 4: tableDelete();     break;
                        case 5: tableQuery();      break;
                        case 6: tableDisp();       break;
                        case 7: queryInfo();       break;
                        case 8: quits();           break;
                }
}


void tableCreate()
{
        EXEC SQL CREATE TABLE S
        (
                sno      NUMBER(9),
                sname    VARCHAR2(30),
                sgender  CHAR(6),
                sage     NUMBER(2),
                sdept    VARCHAR2(30),

                PRIMARY KEY(sno)
        );

        if (sqlca.sqlcode == 0)
                printf("Student Table Created \n");
        else
                printf("Student Table Creation Failed     \n");

        EXEC SQL CREATE TABLE SC
        (
                sno      NUMBER(9),
                cno      NUMBER(2),
                grade    NUMBER(3),

                PRIMARY KEY(sno, cno)
        );

        if (sqlca.sqlcode == 0)
                printf("Student-Course Table Created \n");
        else
                printf("Student-Course Table Creation Failed     \n");

        EXEC SQL CREATE TABLE C
```

```
	(
		cno       NUMBER(2),
		cname   CHAR(20)  NOT NULL,
		cpno     NUMBER(2),
		ccredit  NUMBER(2, 1),

		PRIMARY KEY(cno)
	);

	if (sqlca.sqlcode == 0)
		printf("Course Table Created \n");
	else
		printf("Course Table Creation Failed      \n");

	while (getchar() != '\n');
	while (getchar() != '\n');
	mainMenu();
}




void tableInsert()
{
	system("cls");

	int inputIns;

	printf("---------------------------------Insert Table---------------------------------                        \n");
	printf("1. Course            \n");
	printf("2. Stdent             \n");
	printf("3. Student-Course            \n");
	printf("\nInput Command :  ");

	scanf_s("%d", &inputIns);

	switch (inputIns)
	{
		case 1:
		{
			system("cls");

			while (getchar() != '\n');

			printf("Input Course Number                  \n");
			fgets(cno, 4, stdin);
			cno[strcspn(cno, "\n")] = 0;

			printf("Input Course Name          \n");
			fgets(cname, 22, stdin);
			cname[strcspn(cname, "\n")] = 0;

			printf("Input Course Prerequisite             \n");
			fgets(cpno, 4, stdin);
```

```
        cpno[strcspn(cpno, "\n")] = 0;

        printf("Input Course Credit        \n");
        fgets(ccredit, 5, stdin);
        ccredit[strcspn(ccredit, "\n")] = 0;

        EXEC SQL INSERT INTO C(cno, cname, cpno, ccredit)
                        VALUES(:cno, :cname, :cpno, :ccredit);

        if (sqlca.sqlcode == 0)
                printf("Insertion Successful        \n");
        else
                printf("Insertion Failed                        \n");

        while (getchar() != '\n');

        break;
}
case 2:
{
        system("cls");

        while (getchar() != '\n');

        printf("Input Student Number                \n");
        fgets(sno, 11, stdin);
        sno[strcspn(sno, "\n")] = 0;

        printf("Input Student Name        \n");
        fgets(sname, 32, stdin);
        sname[strcspn(sname, "\n")] = 0;

        printf("Input Student Gender      \n");
        fgets(sgender, 8, stdin);
        sgender[strcspn(sgender, "\n")] = 0;

        printf("Input Student Age            \n");
        fgets(sage, 4, stdin);
        sage[strcspn(sage, "\n")] = 0;

        printf("Input Student Department            \n");
        fgets(sdept, 32, stdin);
        sdept[strcspn(sdept, "\n")] = 0;

        EXEC SQL INSERT INTO S(sno, sname, sgender, sage, sdept)
                VALUES(:sno, :sname, :sgender, :sage, :sdept);

        if (sqlca.sqlcode == 0)
                printf("Insertion Successful        \n");
        else
                printf("Insertion Failed                        \n");

        while (getchar() != '\n');

        break;
```

```
		}
		case 3:
		{
			system("cls");

			while (getchar() != '\n');

			printf("Input Student Number     \n");
			fgets(sno, 11, stdin);
			sno[strcspn(sno, "\n")] = 0;

			printf("Input Course Number      \n");
			fgets(cno, 4, stdin);
			cno[strcspn(cno, "\n")] = 0;

			printf("Input Grade                              \n");
			fgets(grade, 5, stdin);
			grade[strcspn(grade, "\n")] = 0;

			EXEC SQL INSERT INTO SC(sno, cno, grade) VALUES(:sno, :cno, :grade);

			if (sqlca.sqlcode == 0)
				printf("Insertion Successful \n");
			else
				printf("Insertion Failed     \n");

			while (getchar() != '\n');

			break;
		}
	}

	mainMenu();
}

void tableUpdate()
{
	system("cls");

	int inputUp;

	printf("---------------------------------Update Table---------------------------------          \n");
	printf("1. Table Course        \n");
	printf("2. Table Student       \n");
	printf("3. Table Student-Course \n");
	printf("\nYour Option? ");

	scanf_s("%d", &inputUp);

	EXEC SQL WHENEVER NOT FOUND DO break;

	switch (inputUp)
	{
		case 1:
		{
        system("cls");

        while (getchar() != '\n');

        printf("Enter Course Number     \n");
        fgets(temps2, 4, stdin);
        temps2[strcspn(temps2, "\n")] = 0;

        printf("\n");
        printf("Input Remaining Data                \n");
        printf("Input Course Name       \n");
        fgets(cname, 22, stdin);
        cname[strcspn(cname, "\n")] = 0;

        printf("Input Course Prerequisite           \n");
        fgets(cpno, 4, stdin);
        cpno[strcspn(cpno, "\n")] = 0;

        printf("Input Course Credit     \n");
        fgets(ccredit, 5, stdin);
        ccredit[strcspn(ccredit, "\n")] = 0;

        EXEC SQL UPDATE C SET cname = :cname, cpno = :cpno, ccredit = :ccredit
        WHERE cno = :temps2;

        if (sqlca.sqlcode == 0)
                printf("Update Successful       \n");
        else
                printf("Update Failed           \n");

        break;
}
case 2:
{
        system("cls");

        while (getchar() != '\n');

        printf("Enter Student Number                \n");
        fgets(temps1, 11, stdin);
        temps1[strcspn(temps1, "\n")] = 0;

        printf("\n");
        printf("Input Remaining Data                \n");
        printf("Input Student Name      \n");
        fgets(sname, 32, stdin);
        sname[strcspn(sname, "\n")] = 0;

        printf("Input Student Gender    \n");
        fgets(sgender, 8, stdin);
        sgender[strcspn(sgender, "\n")] = 0;

        printf("Input Student Age       \n");
        fgets(sage, 4, stdin);
        sage[strcspn(sage, "\n")] = 0;
```

```
                        printf("Input Student Department            \n");
                        fgets(sdept, 32, stdin);
                        sdept[strcspn(sdept, "\n")] = 0;

                        EXEC SQL UPDATE S SET sname = :sname, sgender = :sgender, sage
= :sage,
                        sdept = :sdept WHERE sno = :temps1;

                        if (sqlca.sqlcode == 0)
                                printf("Update Successful          \n");
                        else
                                printf("Update Failed              \n");

                        break;
                }
                case 3:
                {
                        system("cls");

                        while (getchar() != '\n');

                        printf("Enter Student Number    \n");
                        fgets(temps1, 11, stdin);
                        temps1[strcspn(temps1, "\n")] = 0;

                        printf("Enter Course Number     \n");
                        fgets(temps2, 4, stdin);
                        temps2[strcspn(temps2, "\n")] = 0;

                        printf("\n");
                        printf("Input Remaining Data    \n");
                        printf("Input New Grade         \n");
                        fgets(grade, 5, stdin);
                        grade[strcspn(grade, "\n")] = 0;

                        EXEC SQL UPDATE SC SET grade = :grade WHERE sno = :temps1 AND
cno = :temps2;

                        if (sqlca.sqlcode == 0)
                                printf("Update Successful          \n");
                        else
                                printf("Update Failed              \n");

                        break;
                }
        }

        while (getchar() != '\n');
        mainMenu();
}

void tableDelete()
{
        system("cls");
```

```
int inputDel;
int iterates;

printf("----------------------------------Delete Row----------------------------------        \n");
printf("1. Table Course             \n");
printf("2. Table Student            \n");
printf("3. Table Student-Course  \n");
printf("\nYour Option? ");

scanf_s("%d", &inputDel);

EXEC SQL WHENEVER NOT FOUND DO printf("Table Not Found \n");

switch (inputDel)
{
        case 1:
        {
                system("cls");

                printf("Choose Data Identification          \n");
                printf("1. Course Number           \n");
                printf("2. Course Name             \n");
                printf("\nYour Option :  ");

                scanf_s("%d", &iterates);

                if (iterates == 1)
                {
                        while (getchar() != '\n');

                        printf("\nInput Course Number   \n");
                        fgets(cno, 4, stdin);
                        cno[strcspn(cno, "\n")] = 0;

                        EXEC SQL DELETE FROM C WHERE cno = :cno;

                        if (sqlca.sqlcode == 0)
                                printf("Deletion Successful        \n");
                        else
                                printf("Deletion Failed                         \n");
                }
                else if (iterates == 2)
                {
                        while (getchar() != '\n');

                        printf("\nInput Course Name       \n");
                        fgets(cname, 22, stdin);
                        cname[strcspn(cname, "\n")] = 0;

                        EXEC SQL DELETE FROM C WHERE cname = :cname;

                        if (sqlca.sqlcode == 0)
                                printf("Deletion Successful        \n");
                        else
                                printf("Deletion Failed                 \n");
                }

                break;
        }
        case 2:
        {
                system("cls");

                printf("Choose Data Identification      \n");
                printf("1. Student Number       \n");
                printf("2. Student Name         \n");
                printf("\nYour Option :  ");

                scanf_s("%d", &iterates);

                if (iterates == 1)
                {
                        while (getchar() != '\n');

                        printf("\nInput Student Number          \n");
                        fgets(sno, 11, stdin);
                        sno[strcspn(sno, "\n")] = 0;

                        EXEC SQL DELETE FROM S WHERE sno = :sno;

                        if (sqlca.sqlcode == 0)
                                printf("Deletion Successful     \n");
                        else
                                printf("Deletion Failed                 \n");
                }
                else if (iterates == 2)
                {
                        while (getchar() != '\n');

                        printf("\nInput Student Name    \n");
                        fgets(sname, 32, stdin);
                        sname[strcspn(sname, "\n")] = 0;

                        EXEC SQL DELETE FROM S WHERE sname = :sname;

                        if (sqlca.sqlcode == 0)
                                printf("Deletion Successful     \n");
                        else
                                printf("Deletion Failed                 \n");
                }

                break;
        }
        case 3:
        {
                system("cls");

                while (getchar() != '\n');
```

```
                    printf("Enter Student Number  \n");
                    fgets(sno, 11, stdin);
                    sno[strcspn(sno, "\n")] = 0;

                    printf("Enter Course Number  \n");
                    fgets(cno, 4, stdin);
                    cno[strcspn(cno, "\n")] = 0;

                    EXEC SQL DELETE FROM SC WHERE sno = :sno AND cno = :cno;

                    if (sqlca.sqlcode == 0)
                              printf("Deletion Successful         \n");
                    else
                              printf("Deletion Failed                             \n");

                    break;
          }
     }

     while (getchar() != '\n');
     mainMenu();
}

void tableQuery()
{
     system("cls");

     int inputQry, puts, sets;

     printf("---------------------------------Query---------------------------------                    \n");
     printf("1. Table Course              \n");
     printf("2. Table Student             \n");
     printf("3. Table Student-Course  \n");
     printf("\nYour Option? ");

     scanf_s("%d", &inputQry);

     switch (inputQry)
     {
          case 1:
          {
                    system("cls");

                    printf("Query Regarding :          \n");
                    printf("1. Course Credit \n");
                    printf("2. Course Preq    \n");
                    printf("\nYour Option :  ");

                    scanf_s("%d", &puts);

                    if (puts == 1)
                    {
                              system("cls");

                              printf("Query Credit By :          \n");
printf("1. Exact   Amount                      \n");
printf("2. Between Amount         \n");
printf("3. Lower   Than              \n");
printf("4. Higher  Than              \n");
printf("\nYour Option :  ");

scanf_s("%d", &sets);

if (sets == 1)
{
        system("cls");

        while (getchar() != '\n');

        printf("Enter Credit Amount \n");
        fgets(temps1, 5, stdin);
        temps1[strcspn(temps1, "\n")] = 0;

        EXEC SQL DECLARE TEMPCC1 CURSOR FOR SELECT
cno,                cname, cpno, ccredit FROM C WHERE ccredit
= :temps1;
        EXEC SQL OPEN TEMPCC1;
        EXEC SQL WHENEVER NOT FOUND DO break;

        printf("\nEmpty If No Data Found\n");

        for (;;)
        {
                EXEC SQL FETCH TEMPCC1
                                INTO :cno, :cname, :cpno, :ccredit;

                printf("\n");
                printf("Course Number              %s\n", cno);
                printf("Course Name                %s\n", cname);
                printf("Course Prerequisite        %s\n", cpno);
                printf("Course Credit                        %s\n",
ccredit);
        }

        EXEC SQL CLOSE TEMPCC1;
}
else if (sets == 2)
{
        system("cls");

        while (getchar() != '\n');

        printf("Enter Lowest  Amount \n");
        fgets(temps1, 5, stdin);
        temps1[strcspn(temps1, "\n")] = 0;

        printf("Enter Highest Amount \n");
        fgets(temps2, 5, stdin);
        temps2[strcspn(temps2, "\n")] = 0;
```

```
                                    EXEC SQL DECLARE TEMPCC2 CURSOR FOR SELECT
cno,                                        cname, cpno, ccredit FROM C WHERE ccredit
BETWEEN :temps1                                             AND :temps2;
                                    EXEC SQL OPEN TEMPCC2;
                                    EXEC SQL WHENEVER NOT FOUND DO break;

                                    printf("\nEmpty If No Data Found\n");

                                    for (;;)
                                    {
                                            EXEC SQL FETCH TEMPCC2
                                                    INTO :cno, :cname, :cpno, :ccredit;

                                            printf("\n");
                                            printf("Course Number           %s\n", cno);
                                            printf("Course Name             %s\n", cname);
                                            printf("Course Prerequisite     %s\n", cpno);
                                            printf("Course Credit                   %s\n",
ccredit);
                                    }

                                    EXEC SQL CLOSE TEMPCC2;
                            }
                            else if (sets == 3)
                            {
                                    system("cls");

                                    while (getchar() != '\n');

                                    printf("Enter Credit Amount \n");
                                    fgets(ccredit, 5, stdin);
                                    ccredit[strcspn(ccredit, "\n")] = 0;

                                    EXEC SQL DECLARE TEMPCC3 CURSOR FOR SELECT
cno,                                        cname, cpno, ccredit FROM C WHERE ccredit
< :ccredit;
                                    EXEC SQL OPEN TEMPCC3;
                                    EXEC SQL WHENEVER NOT FOUND DO break;

                                    printf("\nEmpty If No Data Found\n");

                                    for (;;)
                                    {
                                            EXEC SQL FETCH TEMPCC3
                                                    INTO :cno, :cname, :cpno, :ccredit;

                                            printf("\n");
                                            printf("Course Number           %s\n", cno);
                                            printf("Course Name             %s\n", cname);
                                            printf("Course Prerequisite     %s\n", cpno);
                                            printf("Course Credit                   %s\n",
ccredit);
                                    }

                                    EXEC SQL CLOSE TEMPCC3;
```

```
                        }
                        else if (sets == 4)
                        {
                                system("cls");

                                while (getchar() != '\n');

                                printf("Enter Credit Amount \n");
                                fgets(ccredit, 5, stdin);
                                ccredit[strcspn(ccredit, "\n")] = 0;

                                EXEC SQL DECLARE TEMPCC4 CURSOR FOR SELECT
cno,                                    cname, cpno, ccredit FROM C WHERE
ccredit > :ccredit;
                                EXEC SQL OPEN TEMPCC4;
                                EXEC SQL WHENEVER NOT FOUND DO break;

                                printf("\nEmpty If No Data Found\n");

                                for (;;)
                                {
                                        EXEC SQL FETCH TEMPCC4
                                                        INTO :cno, :cname, :cpno, :ccredit;

                                        printf("\n");
                                        printf("Course Number           %s\n", cno);
                                        printf("Course Name             %s\n", cname);
                                        printf("Course Prerequisite     %s\n", cpno);
                                        printf("Course Credit                   %s\n",
ccredit);
                                }

                                EXEC SQL CLOSE TEMPCC4;
                        }

                }
                else if (puts == 2)
                {
                        system("cls");

                        while (getchar() != '\n');

                        printf("Input Prerequisite Course \n");
                        fgets(cpno, 4, stdin);
                        cpno[strcspn(cpno, "\n")] = 0;

                        EXEC SQL DECLARE TEMPCP CURSOR FOR SELECT cno,
cname, cpno,                            ccredit FROM C WHERE cpno = :cpno;
                        EXEC SQL OPEN TEMPCP;
                        EXEC SQL WHENEVER NOT FOUND DO break;

                        printf("\nEmpty If No Data Found\n");

                        for (;;)
                        {
```

```
					EXEC SQL FETCH TEMPCP
INTO :cno, :cname, :cpno, :ccredit;

					printf("\n");
					printf("Course Number		%s\n", cno);
					printf("Course Name		%s\n", cname);
					printf("Course Prerequisite	%s\n", cpno);
					printf("Course Credit			%s\n", ccredit);
				}

				EXEC SQL CLOSE TEMPCP;
			}

			break;
		}
		case 2:
		{
			system("cls");

			printf("Query Regarding Student Age \n");
			printf("1. Exact   Age     \n");
			printf("2. Between Age		\n");
			printf("3. Lower   Than  \n");
			printf("4. Higher  Than  \n");
			printf("\nYour Option :  ");

			scanf_s("%d", &puts);

			if (puts == 1)
			{
				system("cls");

				while (getchar() != '\n');

				printf("Enter Exact Age \n");
				fgets(temps2, 4, stdin);
				temps2[strcspn(temps2, "\n")] = 0;

				EXEC SQL DECLARE TEMPS1 CURSOR FOR SELECT sno,
sname, sgender,				sage, sdept FROM S WHERE sage = :temps2;
				EXEC SQL OPEN TEMPS1;
				EXEC SQL WHENEVER NOT FOUND DO break;

				printf("\nEmpty If No Data Found\n");

				for (;;)
				{
					EXEC SQL FETCH TEMPS1
INTO :sno, :sname, :sgender, :sage, :sdept;

					printf("\n");
					printf("Student Number		%s\n", sno);
					printf("Student Name	%s\n", sname);
					printf("Student Gender	%s\n", sgender);
					printf("Student Age	%s\n", sage);
```

```
                                        printf("Student Dept      %s\n", sdept);
                                }

                                EXEC SQL CLOSE TEMPS1;
                        }
                        else if (puts == 2)
                        {

                                system("cls");

                                while (getchar() != '\n');

                                printf("Enter Low  Age \n");
                                fgets(temps2, 4, stdin);
                                temps2[strcspn(temps2, "\n")] = 0;

                                printf("Enter High Age \n");
                                fgets(temps1, 4, stdin);
                                temps1[strcspn(temps1, "\n")] = 0;

                                EXEC SQL DECLARE TEMPS2 CURSOR FOR SELECT sno,
sname, sgender,                                 sage, sdept FROM S WHERE sage
BETWEEN :temps2 AND :temps1;
                                EXEC SQL OPEN TEMPS2;
                                EXEC SQL WHENEVER NOT FOUND DO break;

                                printf("\nEmpty If No Data Found\n");

                                for (;;)
                                {
                                        EXEC SQL FETCH TEMPS2
INTO :sno, :sname, :sgender, :sage, :sdept;

                                        printf("\n");
                                        printf("Student Number            %s\n", sno);
                                        printf("Student Name     %s\n", sname);
                                        printf("Student Gender  %s\n", sgender);
                                        printf("Student Age       %s\n", sage);
                                        printf("Student Dept      %s\n", sdept);
                                }

                                EXEC SQL CLOSE TEMPS2;
                        }
                        else if (puts == 3)
                        {
                                system("cls");

                                while (getchar() != '\n');

                                printf("Enter Exact Age \n");
                                fgets(sage, 4, stdin);
                                sage[strcspn(sage, "\n")] = 0;

                                EXEC SQL DECLARE TEMPS3 CURSOR FOR SELECT sno,
sname, sgender,                                 sage, sdept FROM S WHERE sage < :sage;
```

```
                                        EXEC SQL OPEN TEMPS3;
                                        EXEC SQL WHENEVER NOT FOUND DO break;

                                        printf("\nEmpty If No Data Found\n");

                                        for (;;)
                                        {
                                                EXEC SQL FETCH TEMPS3
INTO :sno, :sname, :sgender, :sage, :sdept;

                                                printf("\n");
                                                printf("Student Number          %s\n", sno);
                                                printf("Student Name    %s\n", sname);
                                                printf("Student Gender  %s\n", sgender);
                                                printf("Student Age     %s\n", sage);
                                                printf("Student Dept    %s\n", sdept);
                                        }

                                        EXEC SQL CLOSE TEMPS3;
                                }
                                else if (puts == 4)
                                {
                                        system("cls");

                                        while (getchar() != '\n');

                                        printf("Enter Exact Age \n");
                                        fgets(sage, 4, stdin);
                                        sage[strcspn(sage, "\n")] = 0;

                                        EXEC SQL DECLARE TEMPS4 CURSOR FOR SELECT sno,
sname, sgender,                                         sage, sdept FROM S WHERE sage > :sage;
                                        EXEC SQL OPEN TEMPS4;
                                        EXEC SQL WHENEVER NOT FOUND DO break;

                                        printf("\nEmpty If No Data Found\n");

                                        for (;;)
                                        {
                                                EXEC SQL FETCH TEMPS4
INTO :sno, :sname, :sgender, :sage, :sdept;

                                                printf("\n");
                                                printf("Student Number          %s\n", sno);
                                                printf("Student Name    %s\n", sname);
                                                printf("Student Gender  %s\n", sgender);
                                                printf("Student Age     %s\n", sage);
                                                printf("Student Dept    %s\n", sdept);
                                        }

                                        EXEC SQL CLOSE TEMPS4;
                                }

                                break;
                        }
```

```
case 3:
{
        system("cls");

        printf("Query Regarding :                    \n");
        printf("1. Course Taken Amount               \n");
        printf("2. Grade                             \n");
        printf("\nYour Option : ");

        scanf_s("%d", &puts);

        if (puts == 1)
        {
                system("cls");

                while (getchar() != '\n');

                printf("Input Course Taken Amount \n");
                fgets(temps2, 4, stdin);
                temps2[strcspn(temps2, "\n")] = 0;

                EXEC SQL DECLARE TEMPSCE CURSOR FOR SELECT sno
FROM SC                                 GROUP BY sno HAVING COUNT(distinct cno)
= :temps2;
                EXEC SQL OPEN TEMPSCE;
                EXEC SQL WHENEVER NOT FOUND DO break;

                printf("\nEmpty If No Data Found\n\n");

                for (;;)
                {
                        EXEC SQL FETCH TEMPSCE INTO :sno;

                        printf("Student Number %s\n", sno);
                }

                EXEC SQL CLOSE TEMPSCE;
        }
        else if (puts == 2)
        {
                system("cls");

                printf("Query Regarding Grade \n");
                printf("1. Exact   Grade             \n");
                printf("2. Between Grade             \n");
                printf("3. Lower  Than   \n");
                printf("4. Higher Than   \n");
                printf("\nYour Option : ");

                scanf_s("%d", &sets);

                if (sets == 1)
                {
                        system("cls");
```

```
                    while (getchar() != '\n');

                    printf("Enter Exact Grade \n");
                    fgets(grade, 5, stdin);
                    grade[strcspn(grade, "\n")] = 0;

                    EXEC SQL DECLARE TEMPSCG1 CURSOR FOR SELECT
sno, cno,                   grade FROM SC WHERE grade = :grade;
                    EXEC SQL OPEN TEMPSCG1;
                    EXEC SQL WHENEVER NOT FOUND DO break;

                    printf("\nEmpty If No Data Found\n");

                    for (;;)
                    {
                        EXEC SQL FETCH TEMPSCG1
INTO :sno, :cno, :grade;

                        printf("\n");
                        printf("Student Number          %s\n", sno);
                        printf("Course  Number          %s\n", cno);
                        printf("Grade                   %s\n", grade);
                    }

                    EXEC SQL CLOSE TEMPSCG1;
                }
                else if (sets == 2)
                {
                    system("cls");

                    while (getchar() != '\n');

                    printf("Enter Lower  Grade \n");
                    fgets(temps1, 5, stdin);
                    temps1[strcspn(temps1, "\n")] = 0;

                    printf("Enter Higher Grade \n");
                    fgets(temps2, 5, stdin);
                    temps2[strcspn(temps2, "\n")] = 0;

                    EXEC SQL DECLARE TEMPSCG2 CURSOR FOR SELECT
sno, cno,                   grade FROM SC WHERE grade BETWEEN :temps1
AND :temps2;
                    EXEC SQL OPEN TEMPSCG2;
                    EXEC SQL WHENEVER NOT FOUND DO break;

                    printf("\nEmpty If No Data Found\n");

                    for (;;)
                    {
                        EXEC SQL FETCH TEMPSCG2
INTO :sno, :cno, :grade;

                        printf("\n");
                        printf("Student Number              %s\n", sno);
```

```
                                        printf("Course  Number              %s\n", cno);
                                        printf("Grade                       %s\n", grade);
                                }

                                EXEC SQL CLOSE TEMPSCG2;
                        }
                        else if (sets == 3)
                        {
                                system("cls");

                                while (getchar() != '\n');

                                printf("Enter Exact Grade \n");
                                fgets(grade, 5, stdin);
                                grade[strcspn(grade, "\n")] = 0;

                                EXEC SQL DECLARE TEMPSCG3 CURSOR FOR SELECT
sno, cno,                               grade FROM SC WHERE grade < :grade;
                                EXEC SQL OPEN TEMPSCG3;
                                EXEC SQL WHENEVER NOT FOUND DO break;

                                printf("\nEmpty If No Data Found\n");

                                for (;;)
                                {
                                        EXEC SQL FETCH TEMPSCG3
INTO :sno, :cno, :grade;

                                        printf("\n");
                                        printf("Student Number              %s\n", sno);
                                        printf("Course  Number              %s\n", cno);
                                        printf("Grade                       %s\n", grade);
                                }

                                EXEC SQL CLOSE TEMPSCG3;
                        }
                        else if (sets == 4)
                        {
                                system("cls");

                                while (getchar() != '\n');

                                printf("Enter Exact Grade \n");
                                fgets(grade, 5, stdin);
                                grade[strcspn(grade, "\n")] = 0;

                                EXEC SQL DECLARE TEMPSCG4 CURSOR FOR SELECT
sno, cno, grade FROM SC WHERE grade > :grade;
                                EXEC SQL OPEN TEMPSCG4;
                                EXEC SQL WHENEVER NOT FOUND DO break;

                                printf("\nEmpty If No Data Found\n");

                                for (;;)
                                {
```

```
                                                EXEC SQL FETCH TEMPSCG4
INTO :sno, :cno, :grade;

                                                printf("\n");
                                                printf("Student Number          %s\n", sno);
                                                printf("Course  Number          %s\n", cno);
                                                printf("Grade                   %s\n", grade);
                                        }

                                        EXEC SQL CLOSE TEMPSCG4;
                                }
                        }

                        break;
                }
        }

        while (getchar() != '\n');
        mainMenu();
}

void tableDisp()
{
        system("cls");

        int inputDisp;

        printf("---------------------------------Table Display--------------------------------- \n");
        printf("1. Table Course             \n");
        printf("2. Table Student            \n");
        printf("3. Table Student-Course  \n");
        printf("\nYour Option :  ");

        scanf_s("%d", &inputDisp);

        switch (inputDisp)
        {
                case 1:
                {
                        EXEC SQL DECLARE TEMP1 CURSOR FOR SELECT cno, cname, cpno,
ccredit FROM C;
                        EXEC SQL OPEN TEMP1;
                        EXEC SQL WHENEVER NOT FOUND DO break;

                        printf("\nEmpty If No Data Found\n");

                        for (;;)
                        {
                                EXEC SQL FETCH TEMP1 INTO :cno, :cname, :cpno, :ccredit;

                                printf("\n");
                                printf("Course Number           %s\n", cno);
                                printf("Course Name             %s\n", cname);
                                printf("Course Prerequisite     %s\n", cpno);
                                printf("Course Credit                   %s\n", ccredit);
```

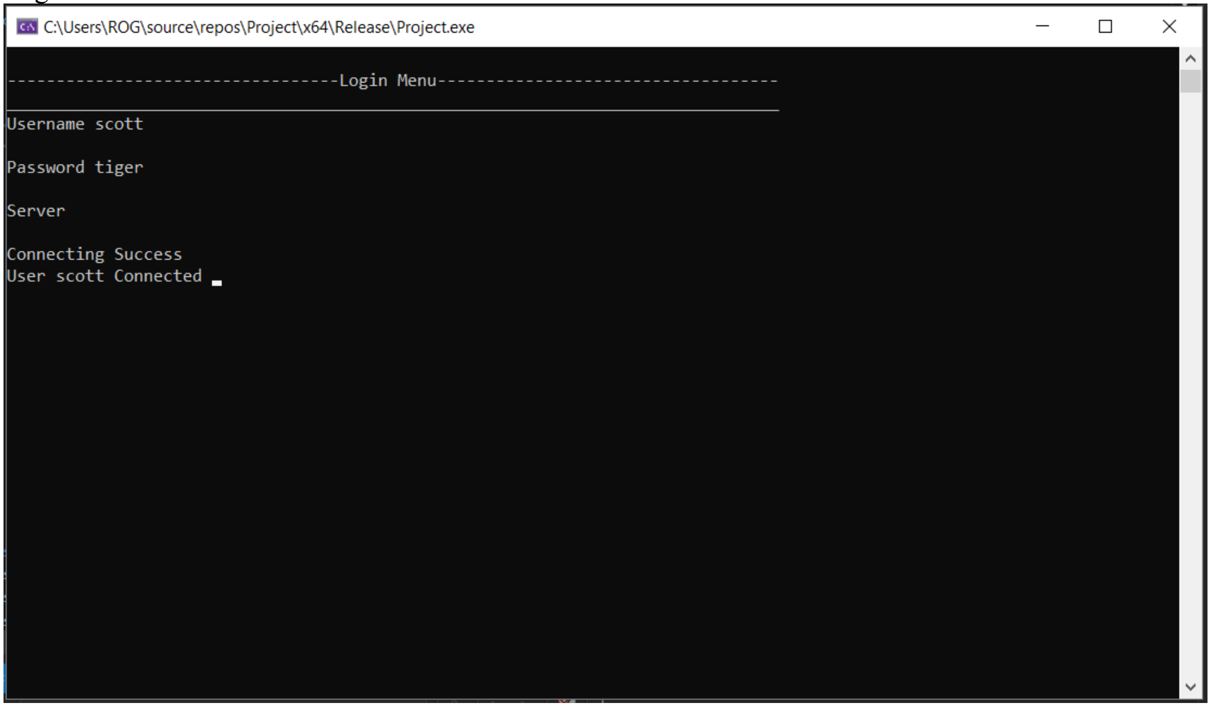
```
			}

			EXEC SQL CLOSE TEMP1;

			break;
		}
		case 2:
		{
			EXEC SQL DECLARE TEMP2 CURSOR FOR SELECT sno, sname, sgender,
sage, sdept FROM S;
			EXEC SQL OPEN TEMP2;
			EXEC SQL WHENEVER NOT FOUND DO break;

			printf("\nEmpty If No Data Found\n");

			for (;;)
			{
				EXEC SQL FETCH TEMP2
INTO :sno, :sname, :sgender, :sage, :sdept;

				printf("\n");
				printf("Student Number		%s\n", sno);
				printf("Student Name	%s\n", sname);
				printf("Student Gender	%s\n", sgender);
				printf("Student Age	%s\n", sage);
				printf("Student Dept	%s\n", sdept);
			}

			EXEC SQL CLOSE TEMP2;

			break;
		}
		case 3:
		{
			EXEC SQL DECLARE TEMP3 CURSOR FOR SELECT sno, cno, grade
FROM SC;
			EXEC SQL OPEN TEMP3;
			EXEC SQL WHENEVER NOT FOUND DO break;

			printf("\nEmpty If No Data Found\n");

			for (;;)
			{
				EXEC SQL FETCH TEMP3 INTO :sno, :cno, :grade;

				printf("\n");
				printf("Student Number		%s\n", sno);
				printf("Course  Number		%s\n", cno);
				printf("Student Grade	%s\n", grade);
			}

			EXEC SQL CLOSE TEMP3;

			break;
		}
```

```
	}

	while (getchar() != '\n');
	while (getchar() != '\n');
	mainMenu();
}

void queryInfo()
{
	system("cls");

	int inputInf, puts;

	printf("---------------------------------Select Table Show Information---------------------------------
	\n");
	printf("1. Table Course                    \n");
	printf("2. Table Student                   \n");
	printf("3. Table Student-Course            \n");
	printf("\nYour Option :  ");

	scanf_s("%d", &inputInf);

	switch (inputInf)
	{
		case 1:
		{
			system("cls");

			printf("Show Information From :            \n");
			printf("1. Course Number                   \n");
			printf("2. Course Name                     \n");
			printf("\nYour Option :  ");

			scanf_s("%d", &puts);

			if (puts == 1)
			{
				system("cls");

				while (getchar() != '\n');

				printf("Enter Course Number \n");
				fgets(cno, 4, stdin);
				cno[strcspn(cno, "\n")] = 0;

				EXEC SQL DECLARE TEMP4 CURSOR FOR SELECT cno, cname,
cpno,					ccredit FROM C WHERE cno = :cno;
				EXEC SQL OPEN TEMP4;
				EXEC SQL WHENEVER NOT FOUND DO break;

				printf("\nEmpty If No Data Found\n");

				for (;;)
				{
					EXEC SQL FETCH TEMP4
INTO :cno, :cname, :cpno, :ccredit;

                                        printf("\n");
                                        printf("Course Number              %s\n", cno);
                                        printf("Course Name                %s\n", cname);
                                        printf("Course Prerequisite        %s\n", cpno);
                                        printf("Course Credit                          %s\n", ccredit);
                                }

                                EXEC SQL CLOSE TEMP4;
                        }
                        else if (puts == 2)
                        {
                                system("cls");

                                while (getchar() != '\n');

                                printf("Input Course Name   \n");
                                fgets(cname, 22, stdin);
                                cname[strcspn(cname, "\n")] = 0;

                                EXEC SQL DECLARE TEMP5 CURSOR FOR SELECT cno, cname,
cpno,                                   ccredit FROM C WHERE cname = :cname;
                                EXEC SQL OPEN TEMP5;
                                EXEC SQL WHENEVER NOT FOUND DO break;

                                printf("\nEmpty If No Data Found\n");

                                for (;;)
                                {
                                        EXEC SQL FETCH TEMP5
INTO :cno, :cname, :cpno, :ccredit;

                                        printf("\n");
                                        printf("Course Number              %s\n", cno);
                                        printf("Course Name                %s\n", cname);
                                        printf("Course Prerequisite        %s\n", cpno);
                                        printf("Course Credit                          %s\n", ccredit);
                                }

                                EXEC SQL CLOSE TEMP5;
                        }

                        break;
                }
                case 2:
                {
                        system("cls");

                        printf("Show Information From? :          \n");
                        printf("1. Student Number                 \n");
                        printf("2. Student Name                   \n");
                        printf("3. Student Gender                 \n");
                        printf("4. Student Dept                   \n");
                        printf("\nYour Option :  ");
                        scanf_s("%d", &puts);

                        if (puts == 1)
                        {
                                system("cls");

                                while (getchar() != '\n');

                                printf("Enter Student Number :    \n");
                                fgets(sno, 11, stdin);
                                sno[strcspn(sno, "\n")] = 0;

                                EXEC SQL DECLARE TEMP6 CURSOR FOR SELECT sno, sname,
sgender,                                        sage, sdept FROM S WHERE sno = :sno;
                                EXEC SQL OPEN TEMP6;
                                EXEC SQL WHENEVER NOT FOUND DO break;

                                printf("\nEmpty If No Data Found\n");

                                for (;;)
                                {
                                        EXEC SQL FETCH TEMP6
INTO :sno, :sname, :sgender, :sage, :sdept;

                                        printf("\n");
                                        printf("Student Number          %s\n", sno);
                                        printf("Student Name    %s\n", sname);
                                        printf("Student Gender  %s\n", sgender);
                                        printf("Student Age     %s\n", sage);
                                        printf("Student Dept    %s\n", sdept);
                                }

                                EXEC SQL CLOSE TEMP6;
                        }
                        else if (puts == 2)
                        {
                                system("cls");

                                while (getchar() != '\n');

                                printf("Enter Student Name :     \n");
                                fgets(sname, 32, stdin);
                                sname[strcspn(sname, "\n")] = 0;

                                EXEC SQL DECLARE TEMP7 CURSOR FOR SELECT sno, sname,
sgender,                                        sage, sdept FROM S WHERE sname = :sname;
                                EXEC SQL OPEN TEMP7;
                                EXEC SQL WHENEVER NOT FOUND DO break;

                                printf("\nEmpty If No Data Found\n");

                                for (;;)
                                {
                                        EXEC SQL FETCH TEMP7
INTO :sno, :sname, :sgender, :sage, :sdept;

                                        printf("\n");
                                        printf("Student Number          %s\n", sno);
                                        printf("Student Name     %s\n", sname);
                                        printf("Student Gender  %s\n", sgender);
                                        printf("Student Age       %s\n", sage);
                                        printf("Student Dept      %s\n", sdept);

                                }

                                EXEC SQL CLOSE TEMP7;
                        }
                        else if (puts == 3)
                        {
                                system("cls");

                                while (getchar() != '\n');

                                printf("Input Student Gender :   \n");
                                fgets(sgender, 8, stdin);
                                sgender[strcspn(sgender, "\n")] = 0;

                                EXEC SQL DECLARE TEMP8 CURSOR FOR SELECT sno, sname,
sgender,                                        sage, sdept FROM S WHERE sgender = :sgender;
                                EXEC SQL OPEN TEMP8;
                                EXEC SQL WHENEVER NOT FOUND DO break;

                                printf("\nEmpty If No Data Found\n");

                                for (;;)
                                {
                                        EXEC SQL FETCH TEMP8
INTO :sno, :sname, :sgender, :sage, :sdept;

                                        printf("\n");
                                        printf("Student Number          %s\n", sno);
                                        printf("Student Name     %s\n", sname);
                                        printf("Student Gender  %s\n", sgender);
                                        printf("Student Age       %s\n", sage);
                                        printf("Student Dept      %s\n", sdept);
                                }

                                EXEC SQL CLOSE TEMP8;
                        }
                        else if (puts == 4)
                        {
                                system("cls");

                                while (getchar() != '\n');

                                printf("Input Student Department :  \n");
                                fgets(sdept, 32, stdin);
                                sdept[strcspn(sdept, "\n")] = 0;
```

```
                                        EXEC SQL DECLARE TEMP9 CURSOR FOR SELECT sno, sname,
sgender,                                                    sage, sdept FROM S WHERE sdept = :sdept;
                                        EXEC SQL OPEN TEMP9;
                                        EXEC SQL WHENEVER NOT FOUND DO break;

                                        printf("\nEmpty If No Data Found\n");

                                        for (;;)
                                        {
                                                  EXEC SQL FETCH TEMP9
INTO :sno, :sname, :sgender, :sage, :sdept;

                                                  printf("\n");
                                                  printf("Student Number            %s\n", sno);
                                                  printf("Student Name     %s\n", sname);
                                                  printf("Student Gender  %s\n", sgender);
                                                  printf("Student Age        %s\n", sage);
                                                  printf("Student Dept      %s\n", sdept);
                                        }

                                        EXEC SQL CLOSE TEMP9;
                              }

                              break;
                    }
                    case 3:
                    {
                              system("cls");

                              printf("Show Information From :  \n");
                              printf("1. Student Number \n");
                              printf("2. Course  Number \n");
                              printf("\nYour Option :  ");

                              scanf_s("%d", &puts);

                              if (puts == 1)
                              {
                                        system("cls");

                                        while (getchar() != '\n');

                                        printf("Enter Student Number :  \n");
                                        fgets(sno, 11, stdin);
                                        sno[strcspn(sno, "\n")] = 0;

                                        EXEC SQL DECLARE TEMP10 CURSOR FOR SELECT sno, cno,
grade                                             FROM SC WHERE sno = :sno;
                                        EXEC SQL OPEN TEMP10;
                                        EXEC SQL WHENEVER NOT FOUND DO break;

                                        printf("\nEmpty If No Data Found\n");

                                        for (;;)
                                        {
```

```
                                        EXEC SQL FETCH TEMP10 INTO :sno, :cno, :grade;

                                        printf("\n");
                                        printf("Student Number              %s\n", sno);
                                        printf("Course  Number              %s\n", cno);
                                        printf("Student Grade     %s\n", grade);
                                }

                                EXEC SQL CLOSE TEMP10;
                        }
                        else if (puts == 2)
                        {
                                system("cls");

                                while (getchar() != '\n');

                                printf("Enter Course Number : \n");
                                fgets(cno, 4, stdin);
                                cno[strcspn(cno, "\n")] = 0;

                                EXEC SQL DECLARE TEMP11 CURSOR FOR SELECT sno, cno,
grade                                   FROM SC WHERE cno = :cno;
                                EXEC SQL OPEN TEMP11;
                                EXEC SQL WHENEVER NOT FOUND DO break;

                                printf("\nEmpty If No Data Found\n");

                                for (;;)
                                {
                                        EXEC SQL FETCH TEMP11 INTO :sno, :cno, :grade;

                                        printf("\n");
                                        printf("Student Number              %s\n", sno);
                                        printf("Course  Number              %s\n", cno);
                                        printf("Student Grade     %s\n", grade);
                                }

                                EXEC SQL CLOSE TEMP11;
                        }

                        break;
                }
        }

        while (getchar() != '\n');
        mainMenu();
}

void quits()
{
        EXEC SQL COMMIT WORK RELEASE;

        exit(0);
}
```

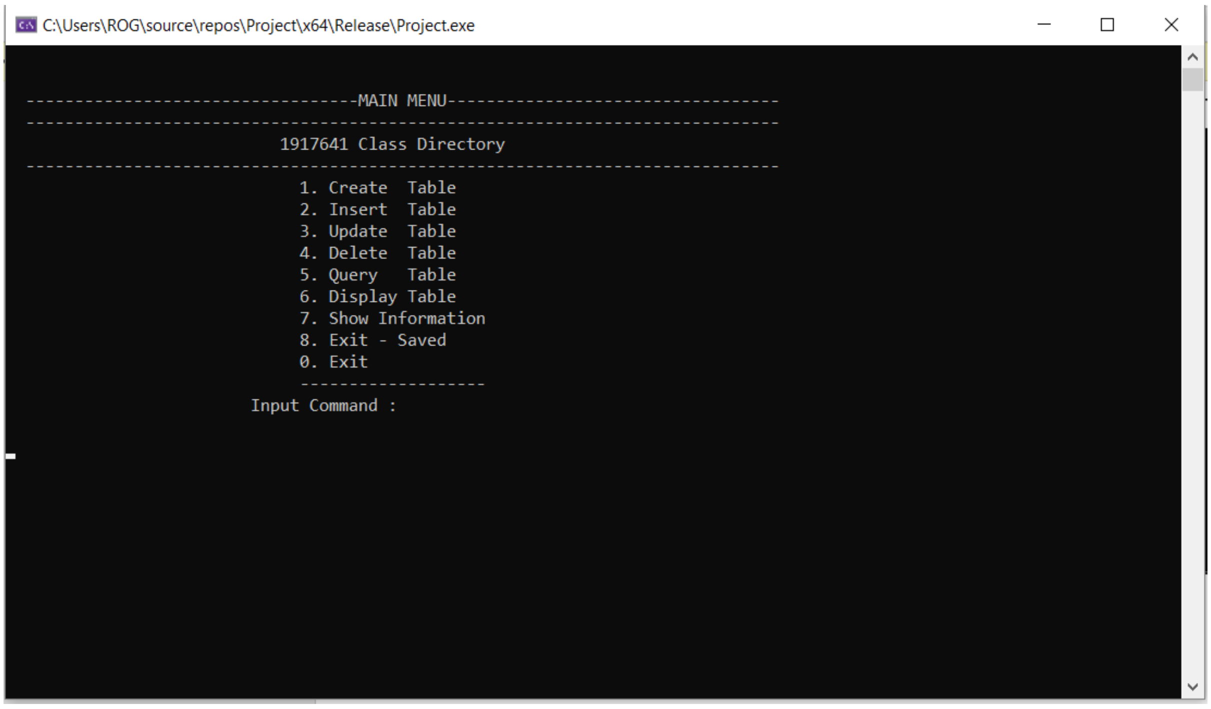
Login Menu



C:\Users\ROG\source\repos\Project\x64\Release\Project.exe

```
---------------------------------Login Menu----------------------------------
_____________________________________________________________________________
Username scott

Password tiger

Server

Connecting Success
User scott Connected
```

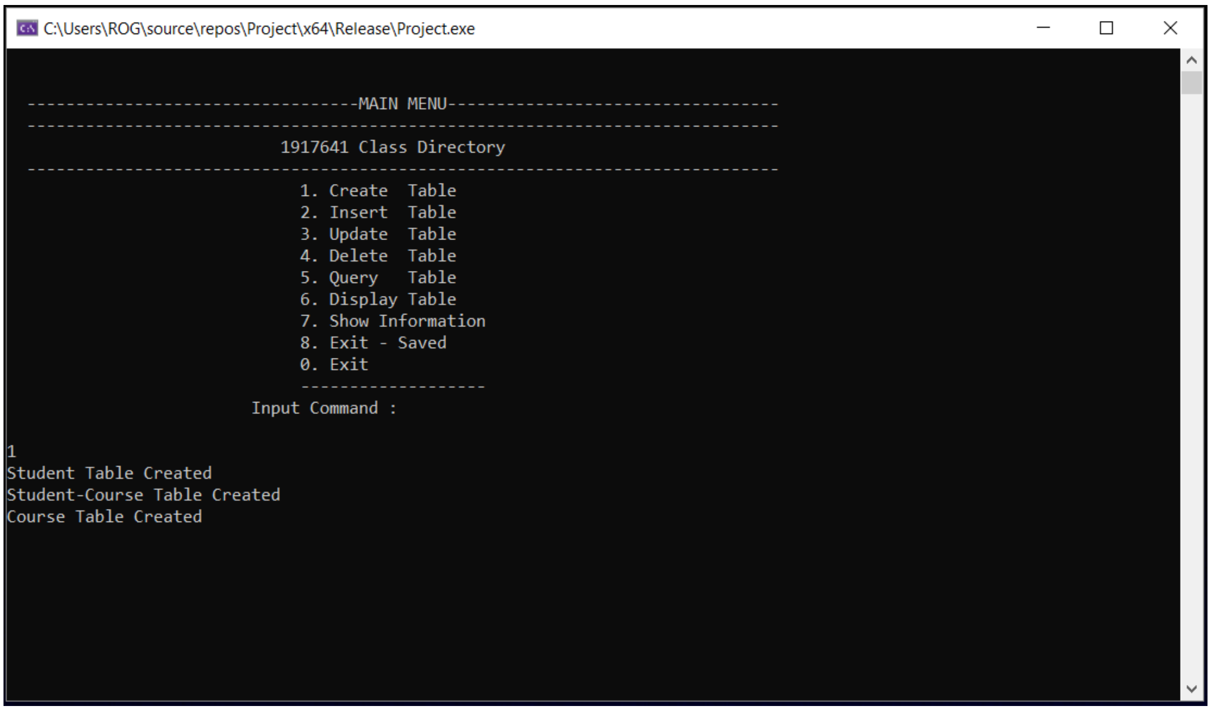
Main Menu



C:\Users\ROG\source\repos\Project\x64\Release\Project.exe

```
  ---------------------------------MAIN MENU--------------------------------
  --------------------------------------------------------------------------
                          1917641 Class Directory
  --------------------------------------------------------------------------
                            1. Create  Table
                            2. Insert  Table
                            3. Update  Table
                            4. Delete  Table
                            5. Query   Table
                            6. Display Table
                            7. Show Information
                            8. Exit - Saved
                            0. Exit
                            -------------------
                         Input Command :
```

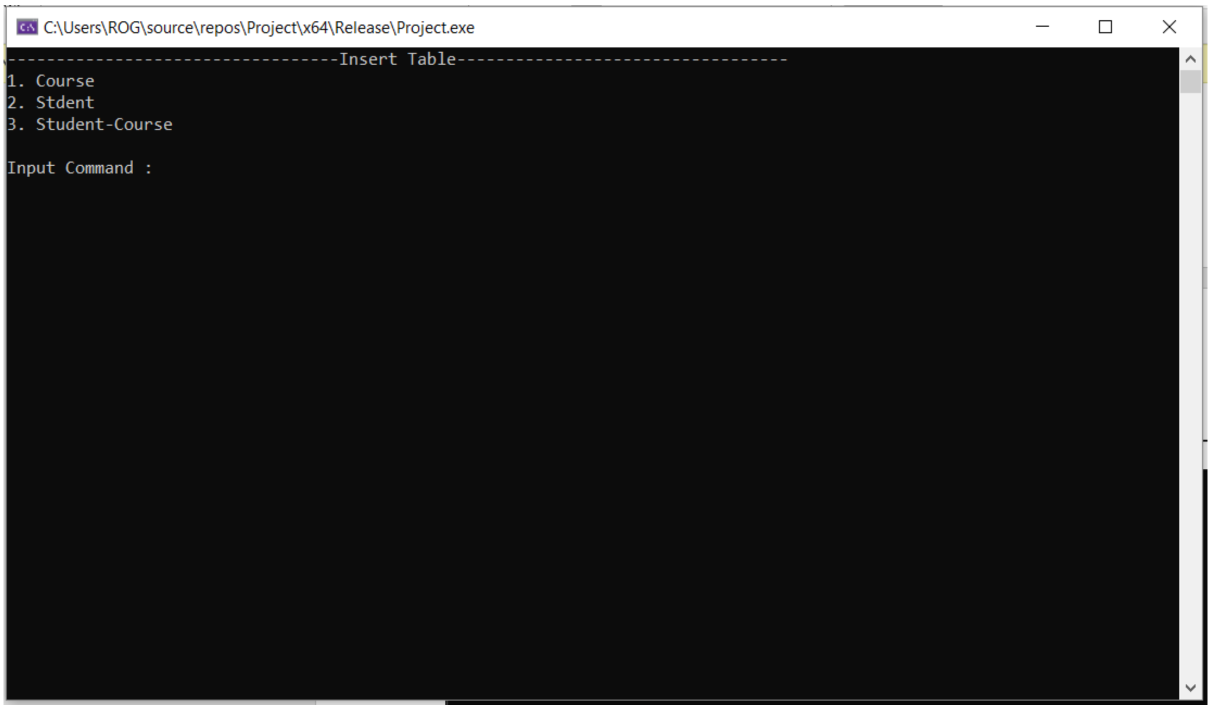
Create Table



C:\Users\ROG\source\repos\Project\x64\Release\Project.exe

```
  --------------------------------MAIN MENU--------------------------------
  ------------------------------------------------------------------------
                          1917641 Class Directory
  ------------------------------------------------------------------------
                            1. Create  Table
                            2. Insert  Table
                            3. Update  Table
                            4. Delete  Table
                            5. Query   Table
                            6. Display Table
                            7. Show Information
                            8. Exit - Saved
                            0. Exit
                            -------------------
                         Input Command :

1
Student Table Created
Student-Course Table Created
Course Table Created
```

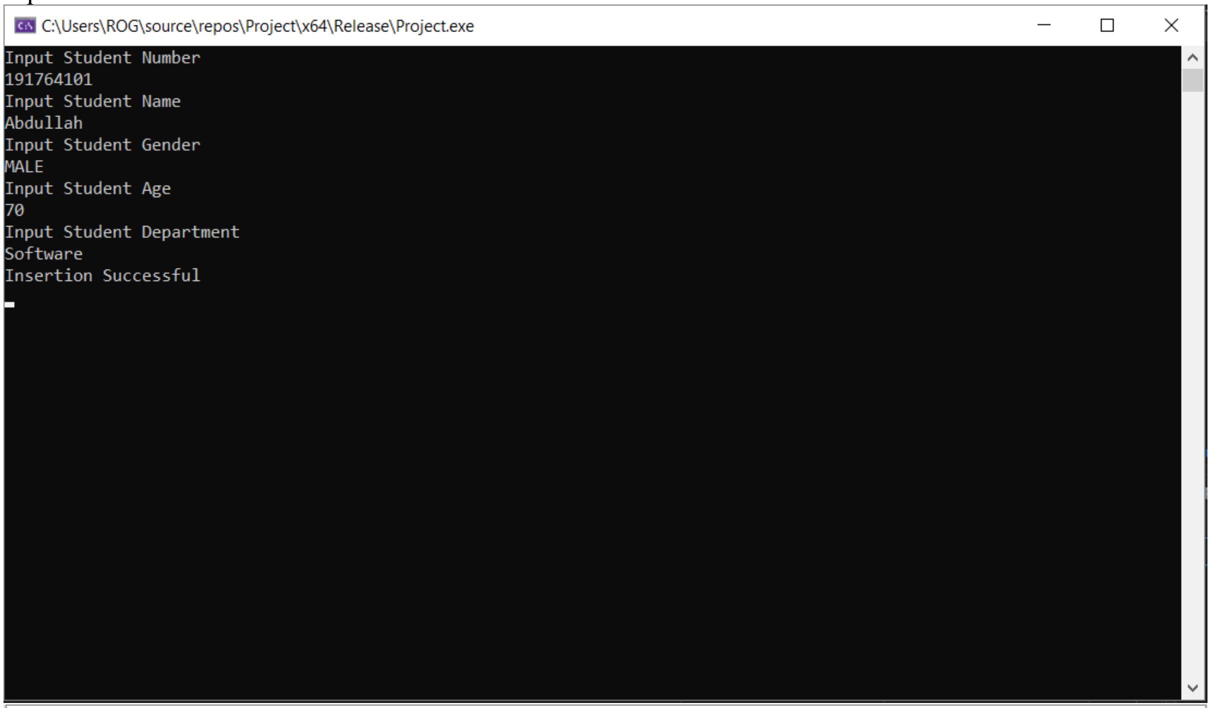
Insert Menu



C:\Users\ROG\source\repos\Project\x64\Release\Project.exe

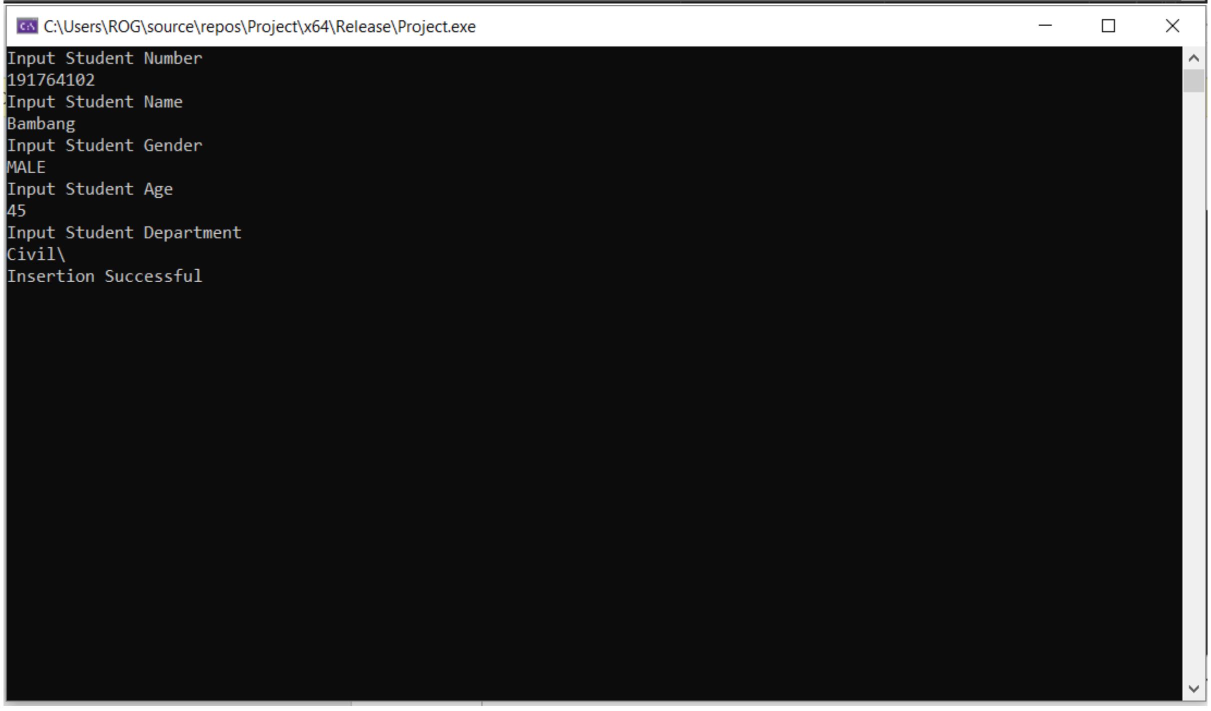
```
---------------------------------Insert Table--------------------------------
1. Course
2. Stdent
3. Student-Course

Input Command :
```

Input Student Table



C:\Users\ROG\source\repos\Project\x64\Release\Project.exe

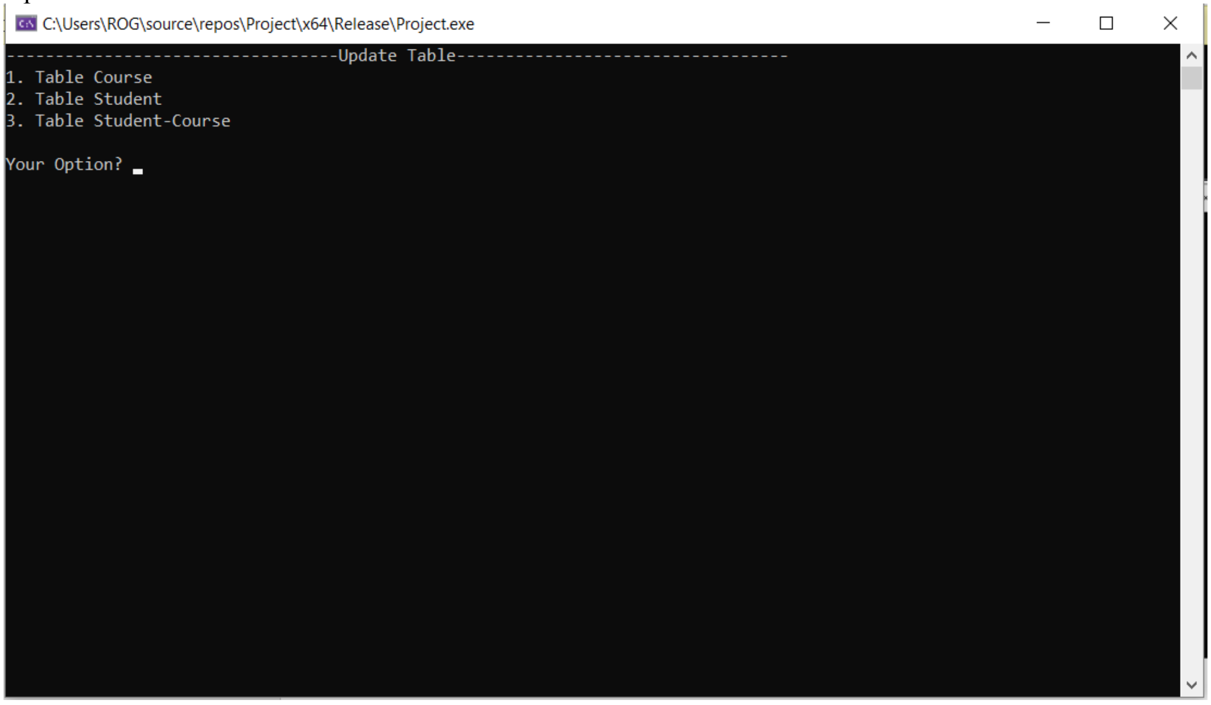
```
Input Student Number
191764101
Input Student Name
Abdullah
Input Student Gender
MALE
Input Student Age
70
Input Student Department
Software
Insertion Successful
```



C:\Users\ROG\source\repos\Project\x64\Release\Project.exe

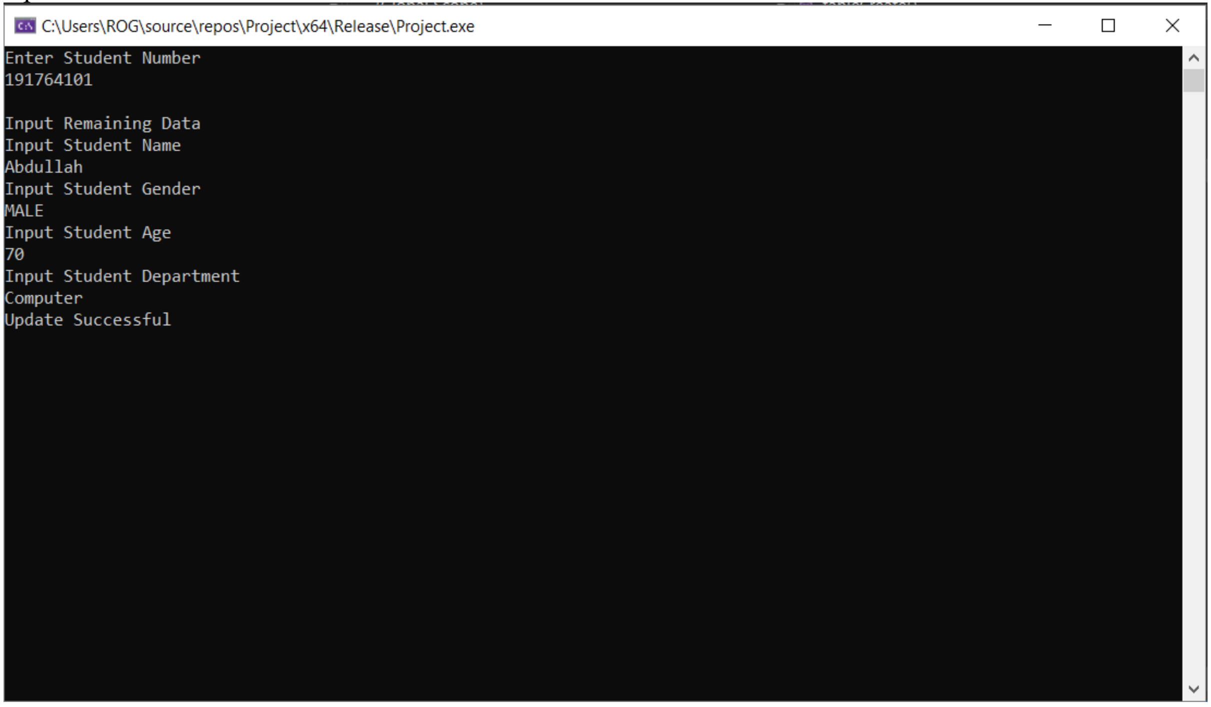
```
Input Student Number
191764102
Input Student Name
Bambang
Input Student Gender
MALE
Input Student Age
45
Input Student Department
Civil\
Insertion Successful
```

Update Table Menu



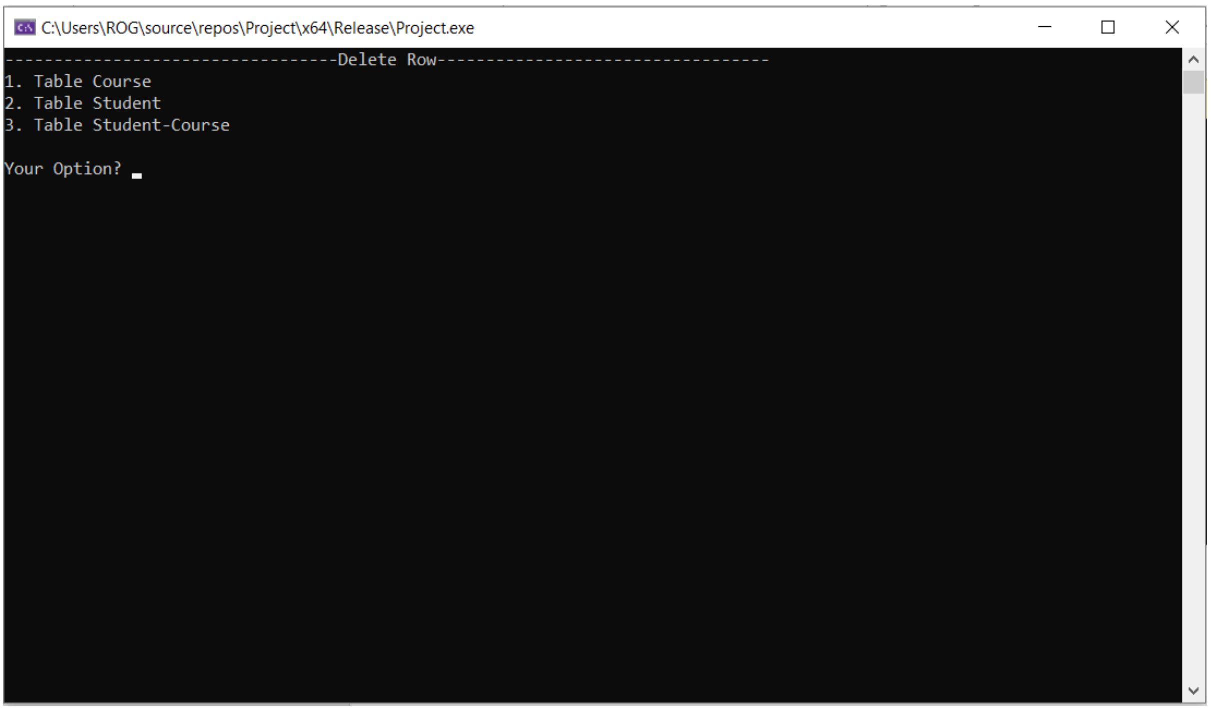
```
C:\Users\ROG\source\repos\Project\x64\Release\Project.exe
---------------------------------Update Table---------------------------------
1. Table Course
2. Table Student
3. Table Student-Course

Your Option? _
```

Update Student Info



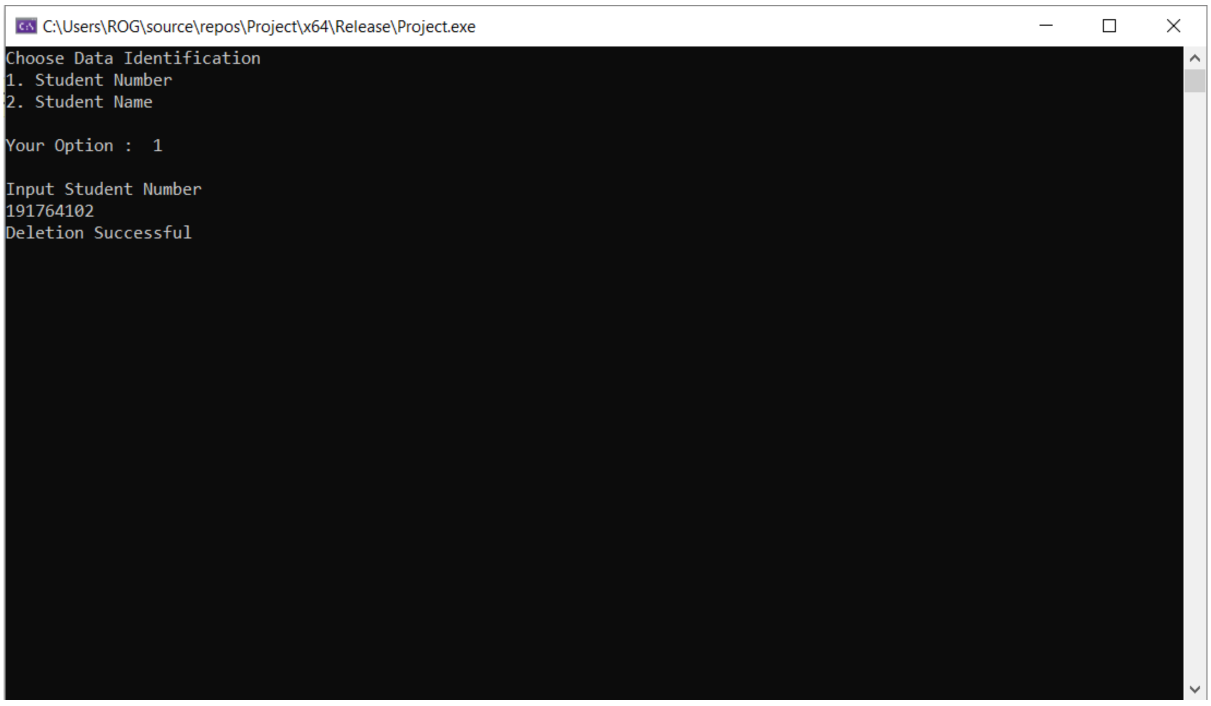
```
C:\Users\ROG\source\repos\Project\x64\Release\Project.exe
Enter Student Number
191764101

Input Remaining Data
Input Student Name
Abdullah
Input Student Gender
MALE
Input Student Age
70
Input Student Department
Computer
Update Successful
```

Delete Table Menu



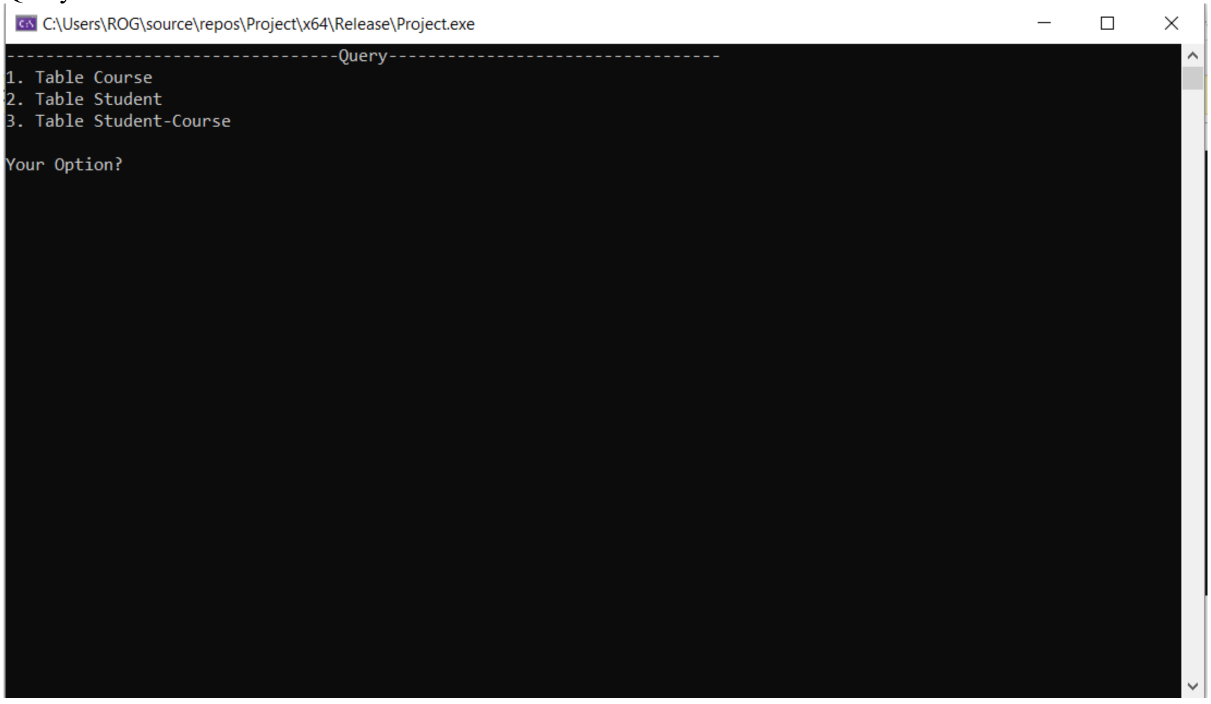
```
----------------------------------Delete Row---------------------------------
1. Table Course
2. Table Student
3. Table Student-Course

Your Option? _
```

Delete Student Table



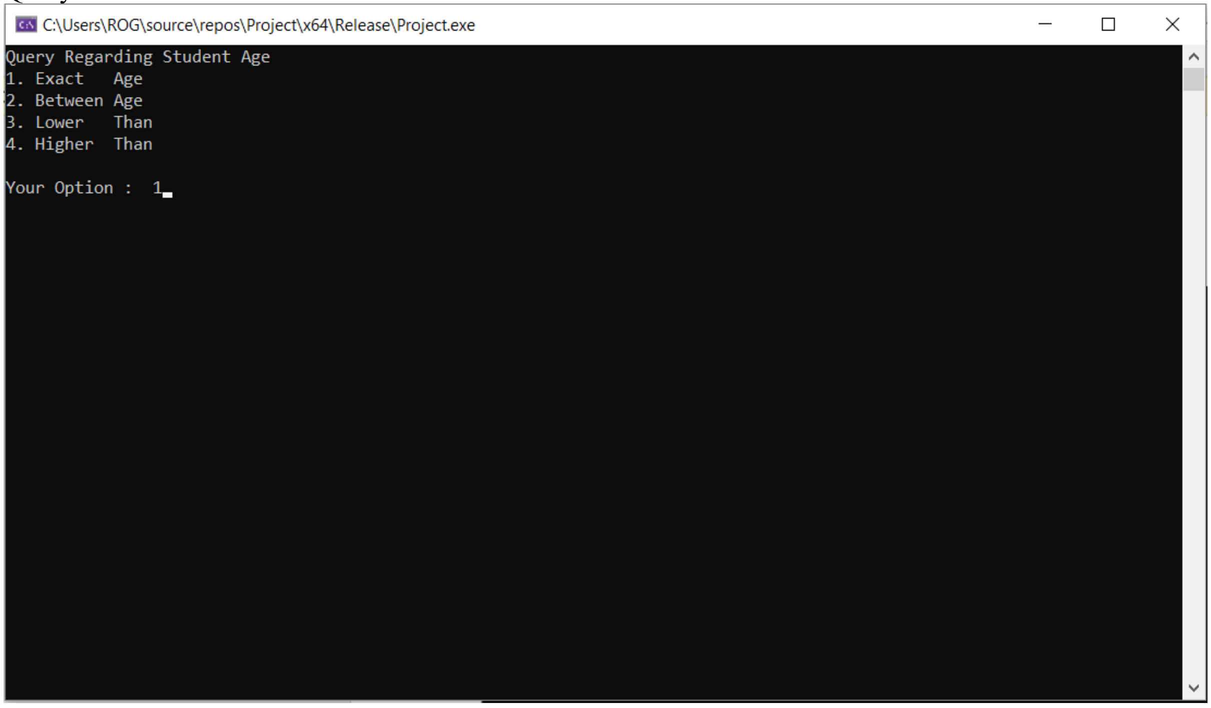
```
Choose Data Identification
1. Student Number
2. Student Name

Your Option :  1

Input Student Number
191764102
Deletion Successful
```

Query Menu

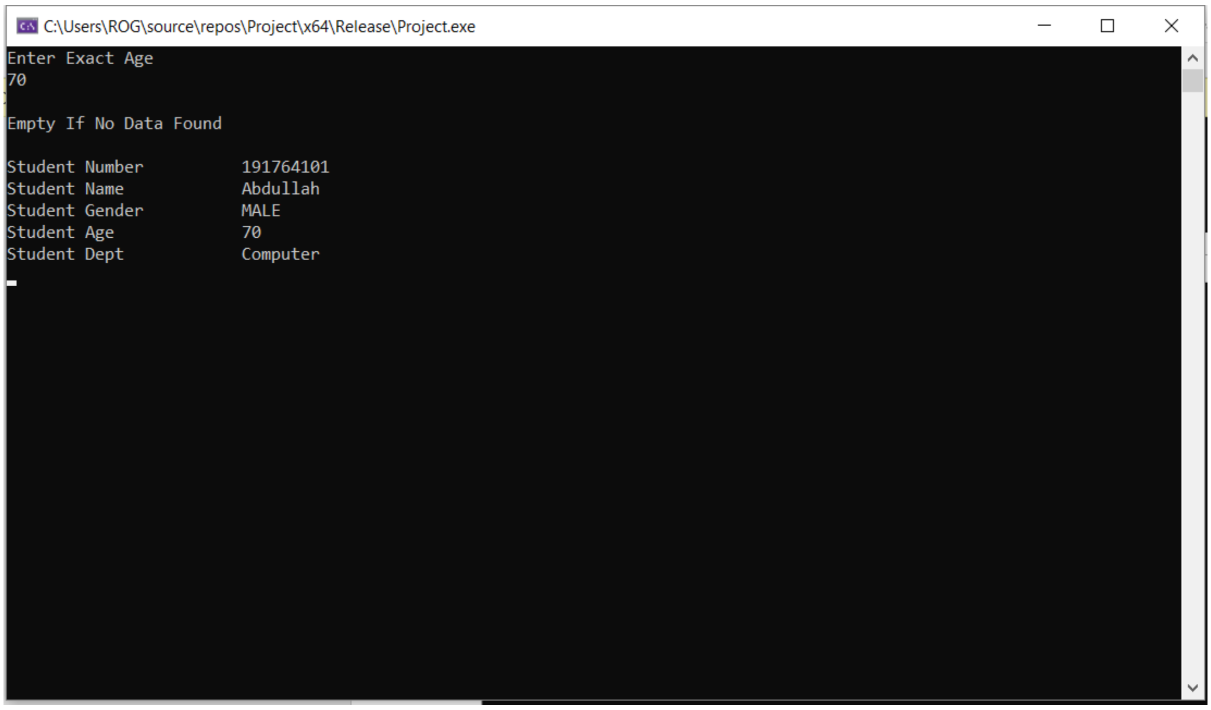


```
C:\Users\ROG\source\repos\Project\x64\Release\Project.exe
----------------------------------Query----------------------------------
1. Table Course
2. Table Student
3. Table Student-Course

Your Option?
```

Query Student Table

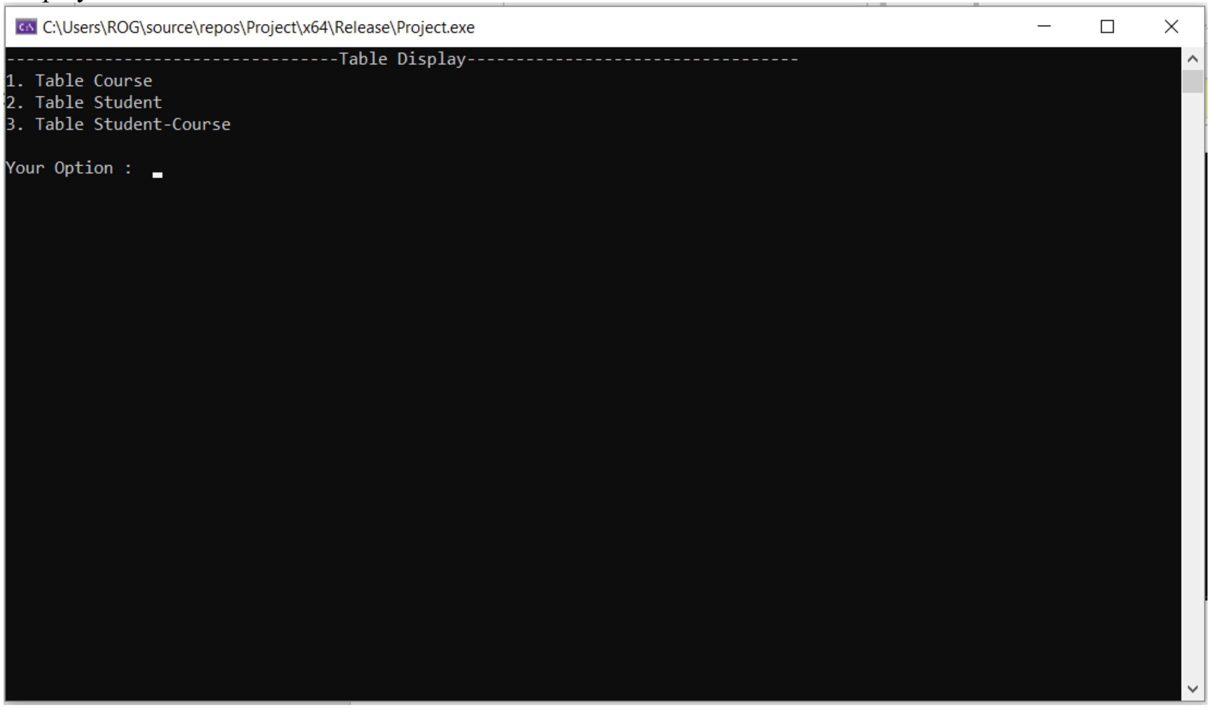


```
C:\Users\ROG\source\repos\Project\x64\Release\Project.exe
Query Regarding Student Age
1. Exact   Age
2. Between Age
3. Lower   Than
4. Higher  Than

Your Option :  1
```



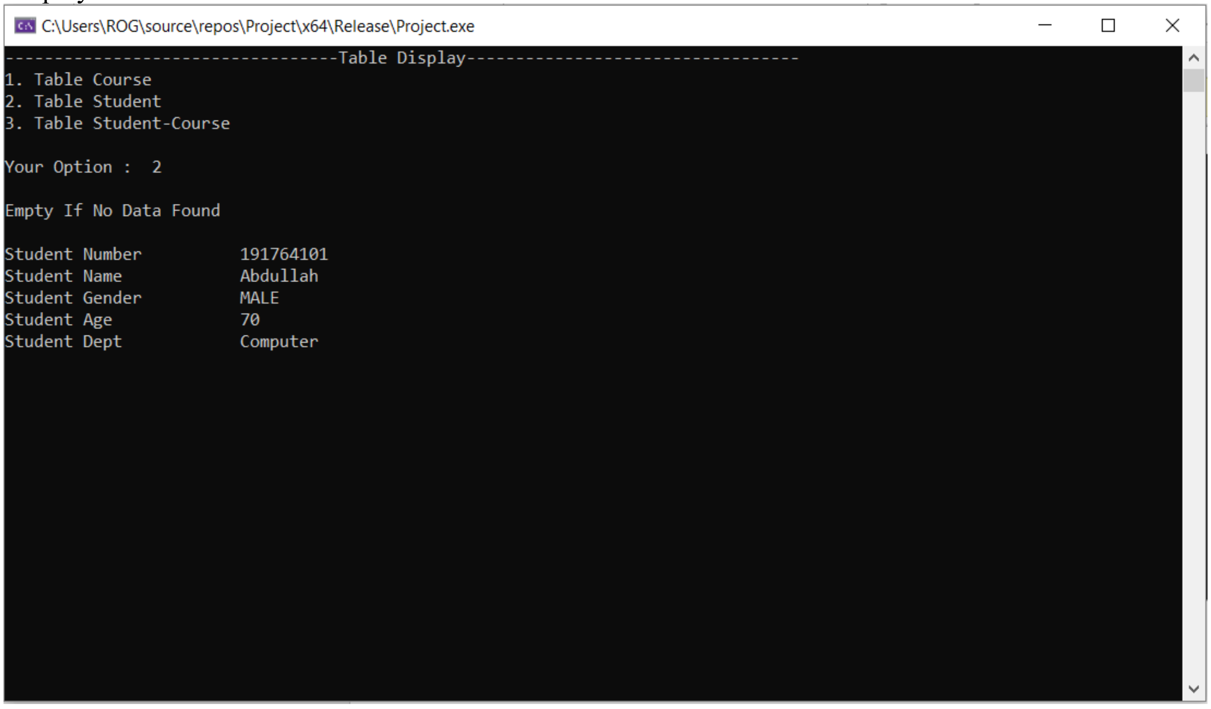
```
Enter Exact Age
70

Empty If No Data Found

Student Number          191764101
Student Name            Abdullah
Student Gender          MALE
Student Age             70
Student Dept            Computer
```

Display Table



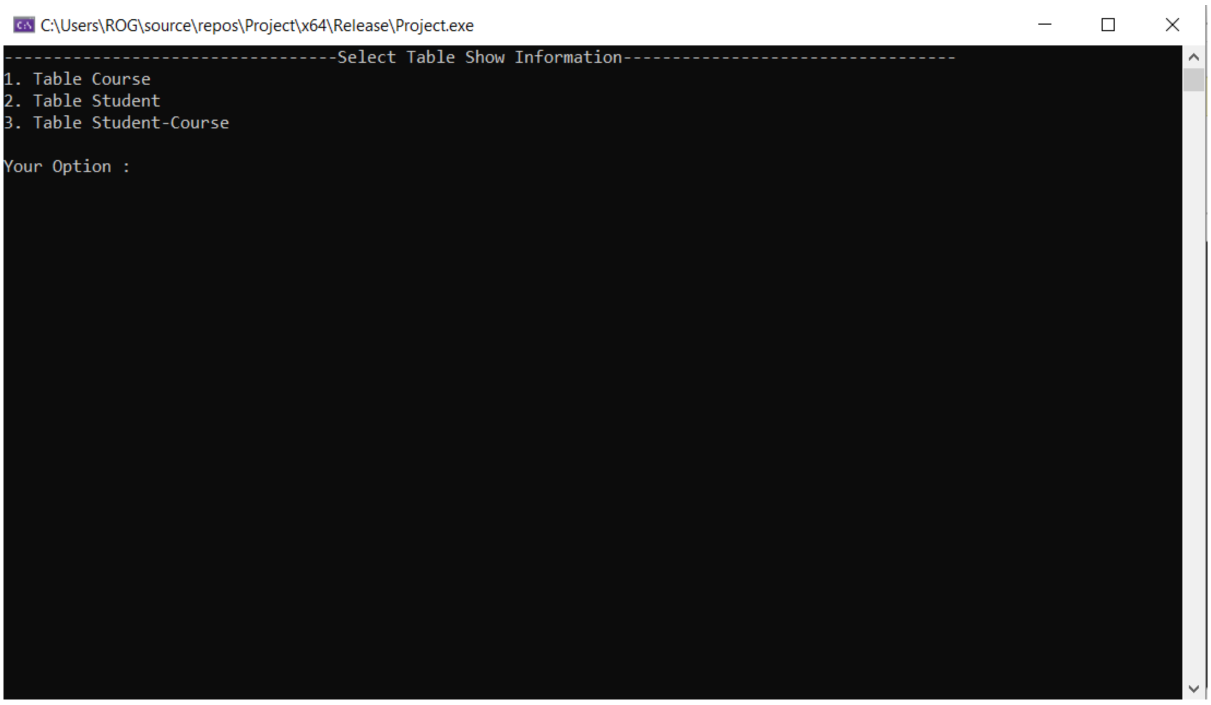
```
----------------------------------Table Display----------------------------------
1. Table Course
2. Table Student
3. Table Student-Course

Your Option :
```

Display Student Table



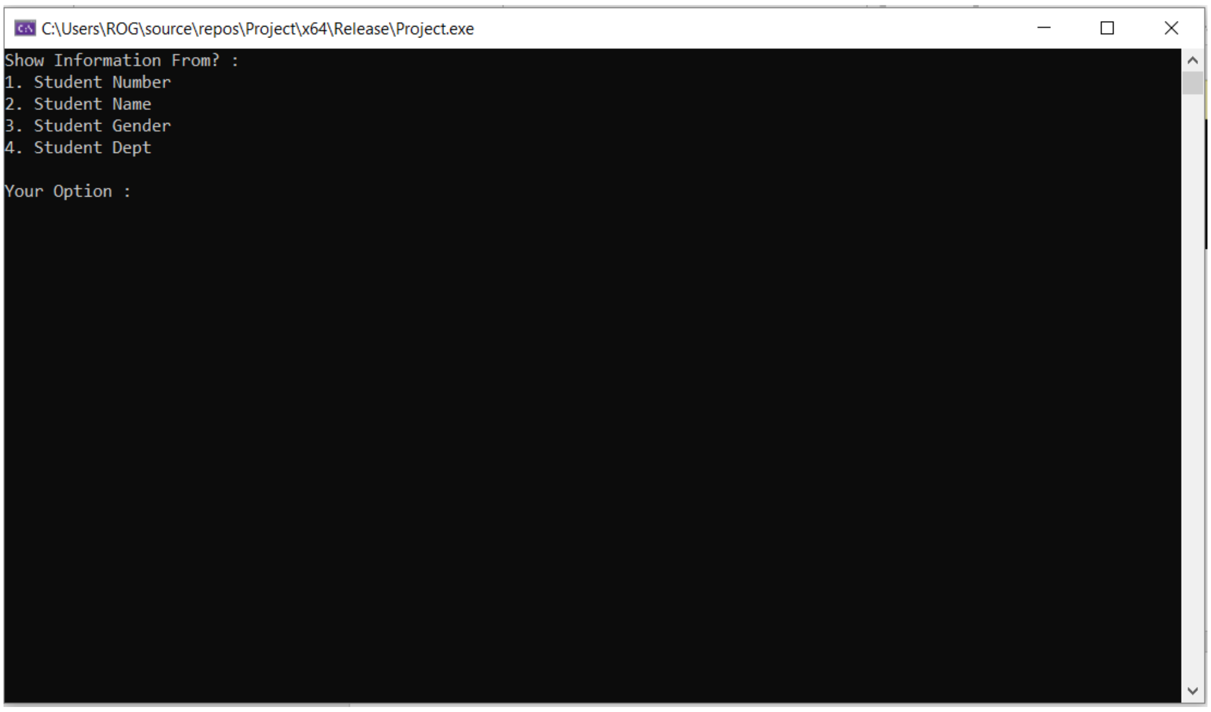
```
C:\Users\ROG\source\repos\Project\x64\Release\Project.exe
----------------------------------Table Display----------------------------------
1. Table Course
2. Table Student
3. Table Student-Course

Your Option :  2

Empty If No Data Found

Student Number          191764101
Student Name            Abdullah
Student Gender          MALE
Student Age             70
Student Dept            Computer
```

Show Information Menu



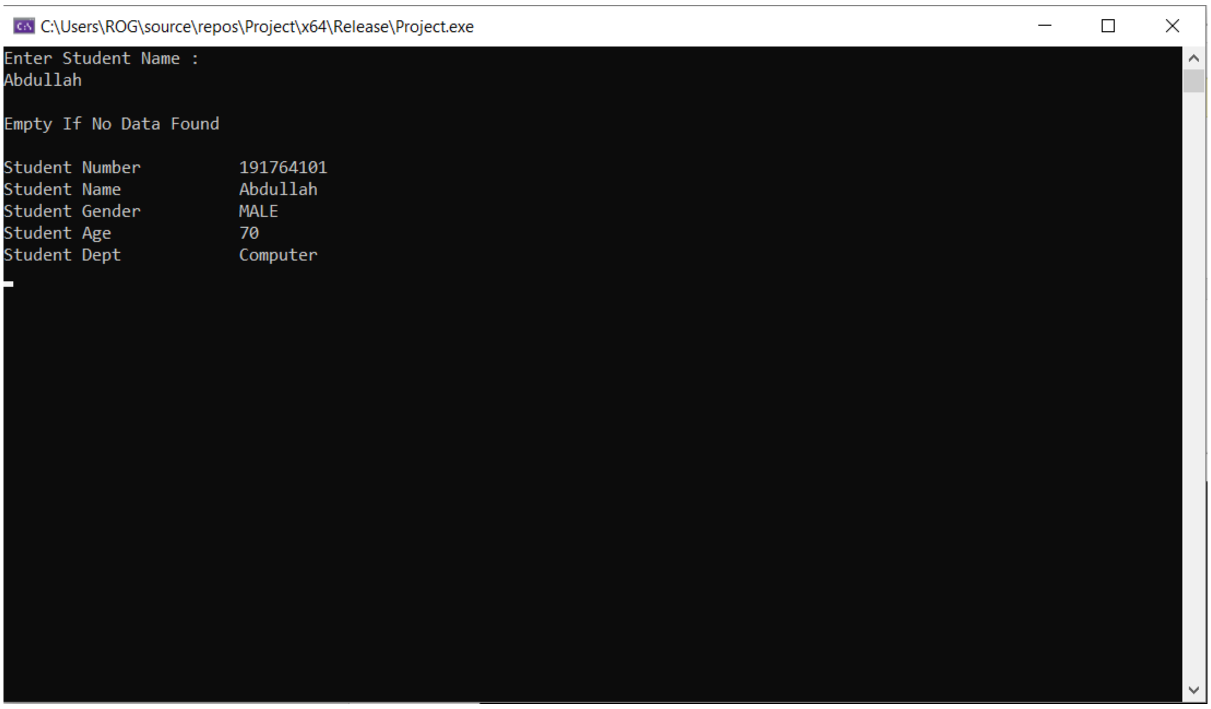
```
C:\Users\ROG\source\repos\Project\x64\Release\Project.exe
----------------------------------Select Table Show Information----------------------------------
1. Table Course
2. Table Student
3. Table Student-Course

Your Option :
```

Show Information Student Table



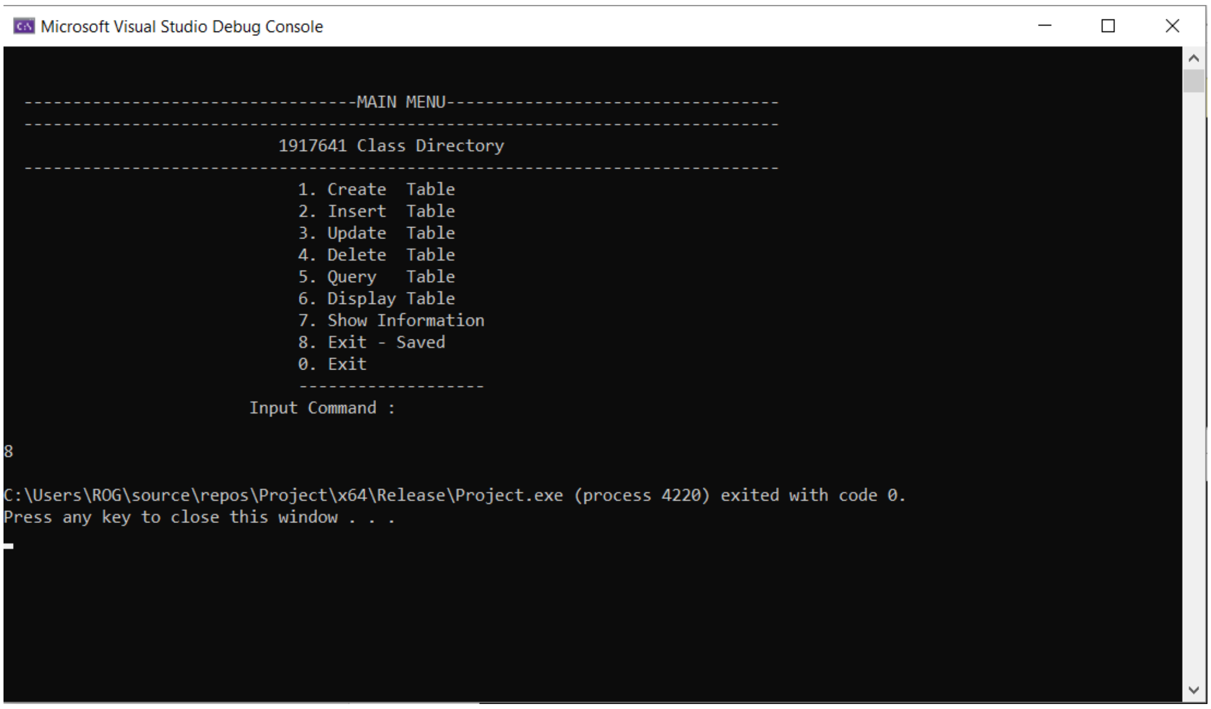
```
Show Information From? :
1. Student Number
2. Student Name
3. Student Gender
4. Student Dept

Your Option :
```

Information of Student Table Based on Student Name



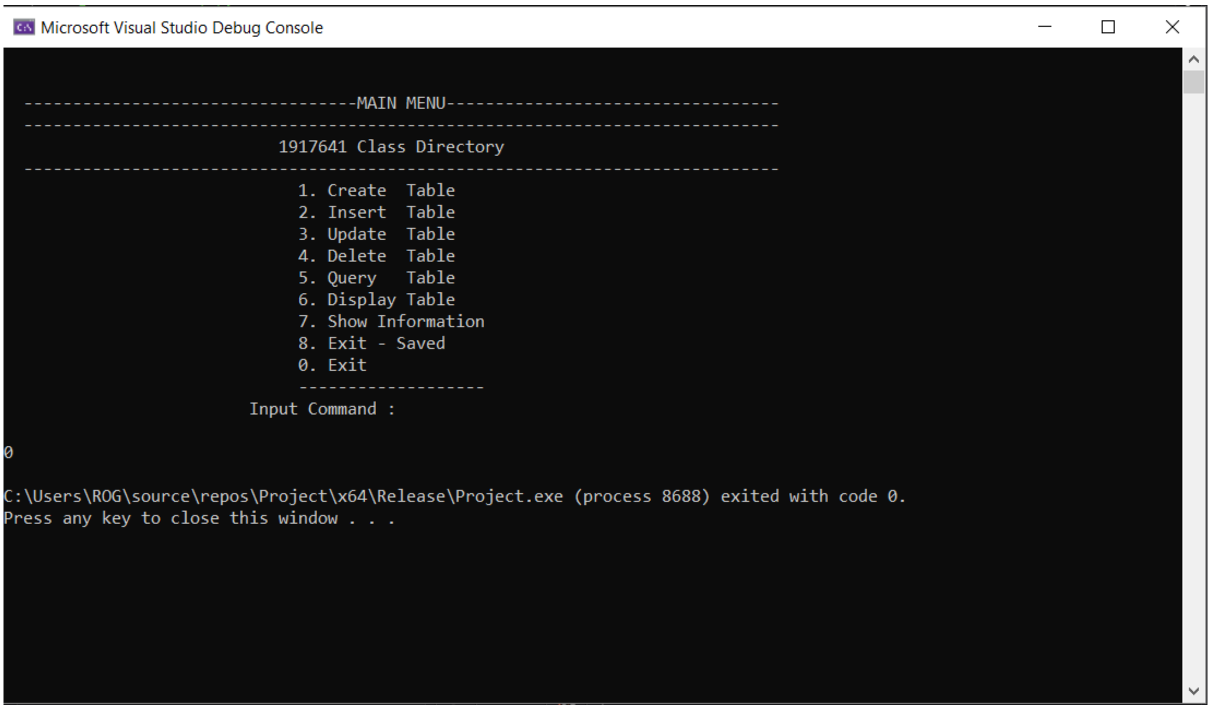
```
Enter Student Name :
Abdullah

Empty If No Data Found

Student Number          191764101
Student Name            Abdullah
Student Gender          MALE
Student Age             70
Student Dept            Computer
```

Exit – Save Menu



```
 --------------------------------MAIN MENU--------------------------------
 ---------------------------------------------------------------------------
                          1917641 Class Directory
 ---------------------------------------------------------------------------
                            1. Create  Table
                            2. Insert  Table
                            3. Update  Table
                            4. Delete  Table
                            5. Query   Table
                            6. Display Table
                            7. Show Information
                            8. Exit - Saved
                            0. Exit
                            -------------------
                       Input Command :

8

C:\Users\ROG\source\repos\Project\x64\Release\Project.exe (process 4220) exited with code 0.
Press any key to close this window . . .
```

Exit



```
 --------------------------------MAIN MENU--------------------------------
 ---------------------------------------------------------------------------
                          1917641 Class Directory
 ---------------------------------------------------------------------------
                            1. Create  Table
                            2. Insert  Table
                            3. Update  Table
                            4. Delete  Table
                            5. Query   Table
                            6. Display Table
                            7. Show Information
                            8. Exit - Saved
                            0. Exit
                            -------------------
                       Input Command :

0

C:\Users\ROG\source\repos\Project\x64\Release\Project.exe (process 8688) exited with code 0.
Press any key to close this window . . .
```