Module Code & Module Title

CC6001NT – Advance Database System Development

Assessment Weightage & Type

40% Coursework

Year and Semester

2021-22 Autumn

Student Name: Nischal Rai

London Met ID: 19031772

College ID: NP05CP4A190096

Assignment Due Date: 03/23/2022

Assignment Submission Date: 03/23/2022

Module Leader: Mr. Narendra Karn

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded

Acknowledgement

I'd like to thank Mr. Narendra Karn, Advanced Database Systems Development module teachers, for their ongoing guidance and valuable feedback on my coursework. I am extremely grateful for their guidance in completing my overall coursework. I'd also like to thank Itahari International College for providing the resources needed to keep the coursework moving forward.

Table of Contents

1. Introduction 1

2. ERD Before Normalization 2

3. Normalization 3

3.1. Figure 1 3

UNF 3

1NF 3

2NF 4

3NF 4

3.2. Figure 2 5

UNF 5

1NF 5

2NF 5

3NF 6

Assumption and Integration 7

Assumption 7

Integration 8

4. ER Diagram 9

5. Data Dictionary 10

Department Table 10

Grade Table 10

Assignment Table 10

Student Table 10

Address Table 11

Payment Table 11

Module Table 11
Student_Details Table 12
Teacher Table 12
Module_Teacher Table 12
Teacher_Address Table 13
Student_Payment Table 13
Module_Student Table 13
6. Generation of Database 14
6.1. Creating and connecting user 14
6.2. Connecting with database 15
6.3. Creating student table 16
6.4. Creating grade table 17
6.5. Creating assignment table 18
6.6. Creating department table 19
6.7. Creating address table 20
6.8. Creating payment table 21
6.9. Creating module table 22
6.10. Creating student_details table 23
6.11. Creating module_student table 23
6.12. Creating teacher table 24
6.13. Creating module_teacher table 25
6.14. Creating teacher_address table 25
6.15. Creating student_payment table 26
6.16. Altering all the tables that contain foreign keys 27
6.17. Inserting the values in student table 27

6.18. Inserting the values in grade table 28
6.19. Inserting into values of assignment table 29
6.20. Inserting values into department table 30
6.21. Inserting values into address table 31
6.22. Inserting values into Payment 32
6.23. Inserting values into module 33
6.24. Inserting values into student_details table 34
6.25. Inserting values into module_student table 35
6.26. Inserting values into teacher table 36
6.27. Inserting values into module_teacher table 37
6.28. Inserting values into teacher_address table 38
6.29. Inserting values into student_payment table 39
6.30. Showing the values of student table 40
6.31. Showing the values of grade table 40
6.32. Showing the values of assignment table 41
6.33. Showing the values of department table 42
6.34. Showing the values of address table 43
6.35. Showing the values of payment table 44
6.36. Showing the values of module table 44
6.37. Showing the values of student_details table 44
6.38. Showing the values of module_student table 45
6.39. Showing the values of teacher table 45
6.40. Showing the values of module_teacher table 46
6.41. Showing the values of teacher_address table 46
6.42. Showing the values of student_payment table 47

7. Implementation of Web-based Database Application 48
7.1. Connecting database 48
7.2. Basic form 49
7.2.1. Student Details 49
7.2.2. Department Details 50
7.2.3. Teacher Details 51
7.2.4. Address Details 52
7.2.5. Module Details 53
7.3. Complex Form 54
7.3.1. Teacher – Module Mapping form 54
7.3.2. Student Fee Payment Form 55
7.3.3. Student – Assignment Form 56
8. Testing 57
8.1. Inserting into Student Table 57
8.2. Editing in Student Table 59
8.3. Deleting from Student Table 61
8.4. Inserting into Department Table 63
8.5. Editing value from Department 65
8.6. Deleting Value form Department 67
8.7. Inserting into Teacher 69
8.8. Editing Values of Teacher 71
8.9. Deleting Values from Teacher 73
8.10. Inserting Values to Address 75
8.11. Editing Values of Address 77
8.12. Deleting Values from Address 79

8.13. Inserting Values to Module 81

8.14. Editing Values of Module 83

8.15. Deleting Values from Module 85

8.16. Teacher – Module Mapping Form Test 87

8.17. Student Fee Payment Form Test 88

8.18. Student – Assignment Form Test 89

8.19. Error handling 1 (Using False foreign Key for Department_ID in teacher) 90

9. User Manual 95

10. Conclusion 97

11. References 98

Table of Figures

Figure 1: Entity Relationship Diagram 9
Figure 2: Creating and connecting user 14
Figure 3:Connecting with user and database 15
Figure 4: Creating Student Table 16
Figure 5: Creating Grade Table 17
Figure 6: Creating Assignment Table 18
Figure 7: Creating Department Table 19
Figure 8: Creating Address Table 20
Figure 9: Creating Payment Table 21
Figure 10: Creating Module Table 22
Figure 11: Creating Student_ Details Table 23
Figure 12: Creating Module Student Table 23
Figure 13: Creating Teacher Table 24
Figure 14: Creating Module_Teacher Table 25
Figure 15: Creating Teacher_Address Table 25
Figure 16: Creating Student_Payment Table 26
Figure 17: Altering Tables that contain FOREIGN KEYS 27
Figure 18: Inserting Values in Student Table 27
Figure 19: Inserting Values in Grade Table 28
Figure 20: Inserting Values in Assignment Table 29
Figure 21: Inserting Values in Department Table 30
Figure 22: Inserting values in Address Table 31
Figure 23: Inserting Values in Payment Table 32
Figure 24: Inserting Values in Module Table 33
Figure 25: Inserting Values in Student_Details Table 34
Figure 26: Inserting Values in Module_Student Table 35
Figure 27: Inserting Values in Teacher Table 36
Figure 28: Inserting Values in Module_Teacher Table 37
Figure 29: Inserting Values in Teacher Address Table 38
Figure 30: Inserting Values in Student_Payment Table 39

Figure 31: Showing the Values of Student_Address Table 40
Figure 32: Showing the Values of Grade Table..... 40
Figure 33: Showing the Values of Assignment Table 41
Figure 34: Showing the Values of Department Table 42
Figure 35: Showing the Values of Address Table 43
Figure 36: Showing the Values of Payment Table 44
Figure 37: Showing the Values of Modul Table..... 44
Figure 38: Showing the Values of Student Table 44
Figure 39: Showing the Values of Module_Student Table..... 45
Figure 40: Showing the Values of Teacher Table 45
Figure 41: Showing the Values of Module_Teacher 46
Figure 42: Showing the Values of Teacher Address 46
Figure 43: Showing the Values of Student_Payment 47
Figure 44: Connecting Database 48
Figure 45: Basic Form of Student Table..... 49
Figure 46: Basic Form of Department Table 50
Figure 47: Basic Form of Teacher Table 51
Figure 48: Basic Form of Address Table 52
Figure 49: Basic Form of Module Table..... 53
Figure 50: Complex Form of Teacher - Module Mapping 54
Figure 51: Query for Teacher - Module Mapping 54
Figure 52: Complex Form of Student Fee Payment Form..... 55
Figure 53: Query for Student Fee Payment 55
Figure 54: Complex Form of Student – Assignment Form 56
Figure 55:Query for Student-Assignment..... 56
Figure 56: Before Inserting New Student 57
Figure 57: After Inserting New Student..... 58
Figure 58: Before Editing value to Student 59
Figure 59: After Editing Values to Student 60
Figure 60: Before Deleting from Student 61
Figure 61: After Deleting from Student..... 62

Figure 62: Before Inerting new Department 63
Figure 63: After Inserting new Department 64
Figure 64: Before Editing the value of department 65
Figure 65: After Editing the Value of Department 66
Figure 66: Before Deleting the Value from Department 67
Figure 67: After Deleting the Value from the Department 68
Figure 68: Before Inserting new Teacher 69
Figure 69: After Inserting new Teacher 70
Figure 70: Before Editing in Teacher 71
Figure 71: After Editing in Teacher 72
Figure 72: Before Deleting Value of Teacher 73
Figure 73: After Deleting the Value in Teacher 74
Figure 74: Before Inserting new Address 75
Figure 75: After Inserting New Address 76
Figure 76: Before Editing the Value of Address 77
Figure 77: After Editing the Address 78
Figure 78: Before Deleting the Value from Address 79
Figure 79: After Deleting the Value from Address 80
Figure 80: Before Inserting new Module 81
Figure 81: After Inserting new Module 82
Figure 82: Before Editing values of Module 83
Figure 83: After Editing the Value in Module 84
Figure 84: Before Deleting the Value from Module 85
Figure 85: After Deleting the Value from Module 86
Figure 86: Before the drop-down value is changed in Teacher-Module Mapping 87
Figure 87: After the drop-down value is changed in Teacher-Module Mapping 87
Figure 88: Part 1 of error 90
Figure 89: Part 2 of error (an error message) 91
Figure 90: Fixing error Part 1 91
Figure 91: Fixing error Part 2 92
Figure 92: Checking the error 93

Table of Tables

Table 1: ERD Before Normalization 2
Table 2: Data Dictionary for Department Table 10
Table 3:Data Dictionary for Grade Table 10
Table 4:Data Dictionary for Assignment Table 10
Table 5:Data Dictionary for Student Table 10
Table 6:Data Dictionary for Address Table 11
Table 7:Data Dictionary for Payment Table 11
Table 8:Data Dictionary for Module Table 11
Table 9:Data Dictionary for Student_Details Table 12
Table 10:Data Dictionary for Teacher Table 12
Table 11:Data Dictionary for Module_Teacher Table 12
Table 12:Data Dictionary for Teacher_Address Table 13
Table 13:Data Dictionary for Student_Payment Table 13
Table 14: Data Dictionary for Module_Student Table 13
Table 15: Test Case 1 57
Table 16: Test Case 2 59
Table 17: Test Case 3 61
Table 18: Test 4 63
Table 19: Test Case 5 65
Table 20: Test Case 6 67
Table 21: Test Case 7 69
Table 22: Test Case 8 71
Table 23: Test Case 9 73
Table 24: Test Case 10 75
Table 25: Test Case 11 77
Table 26: Test Case 12 79
Table 27: Test Case 13 81
Table 28: Test Case 14 83
Table 29: Test Case 15 85
Table 30: Test Case 16 87

Table 31: Test Case 17 88
Table 32: Before the drop-down value is changed in Student Fee Payment Form 88
Table 33: After the drop-down value is changed in Student Fee Payment Form 88
Table 34: Test Case 18 89
Table 35: Before the drop-down value is changed in Student – Assignment Form 89
Table 36: After the drop-down value is changed in Student – Assignment Form 89
Table 37: Test Case 19 90
Table 38: Test 19 after fixing the error 92
Table 39: Fixed 94
Table 40: Home Page 95
Table 41: Student Form 95
Table 42: Back to Home Page 96
Table 43: Teacher - Module Mapping Form is active 96

1. Introduction

ABC College has several different of departments. Some departments are involved in student exam activities such as assignment and results, while others are involved in student fee management. For the online management of ABC college departments, a web-based application is proposed, and all records must be stored in a database. The goal of this project is to create a web-based application for ABC college. The application is expected to help the college continue to keep moving forward by allowing the college to track their students and management in real time.

Various technologies were used to create the software, including SQL Developer, SQL Data Modeler, and Visual Studio (ASP.NET with C#). SQL Developer is a free graphical user interface that allows you to perform various database tasks quickly and efficiently. SQL developer is used in the course to create tables, insert sample data into the database, and test run various queries. SQL Data Modeler is a free graphical tool for performing data modeling tasks (Hotka, 2006). It can be used to create, browse, and edit models such as logical, relational, physical, and so on. SQL Data Modeler is used in the course to create ERD and generate the DDL script for the database. The application's C# code (ASP.NET) is written in Visual Studio.

Coursework documentation includes the initial and final ER Diagram, Normalization, data dictionary of tables, DDL and DML Scripts. The project also includes the conversion of a database to a web-based application, as well as all necessary testing and an application user manual.

2. ERD Before Normalization

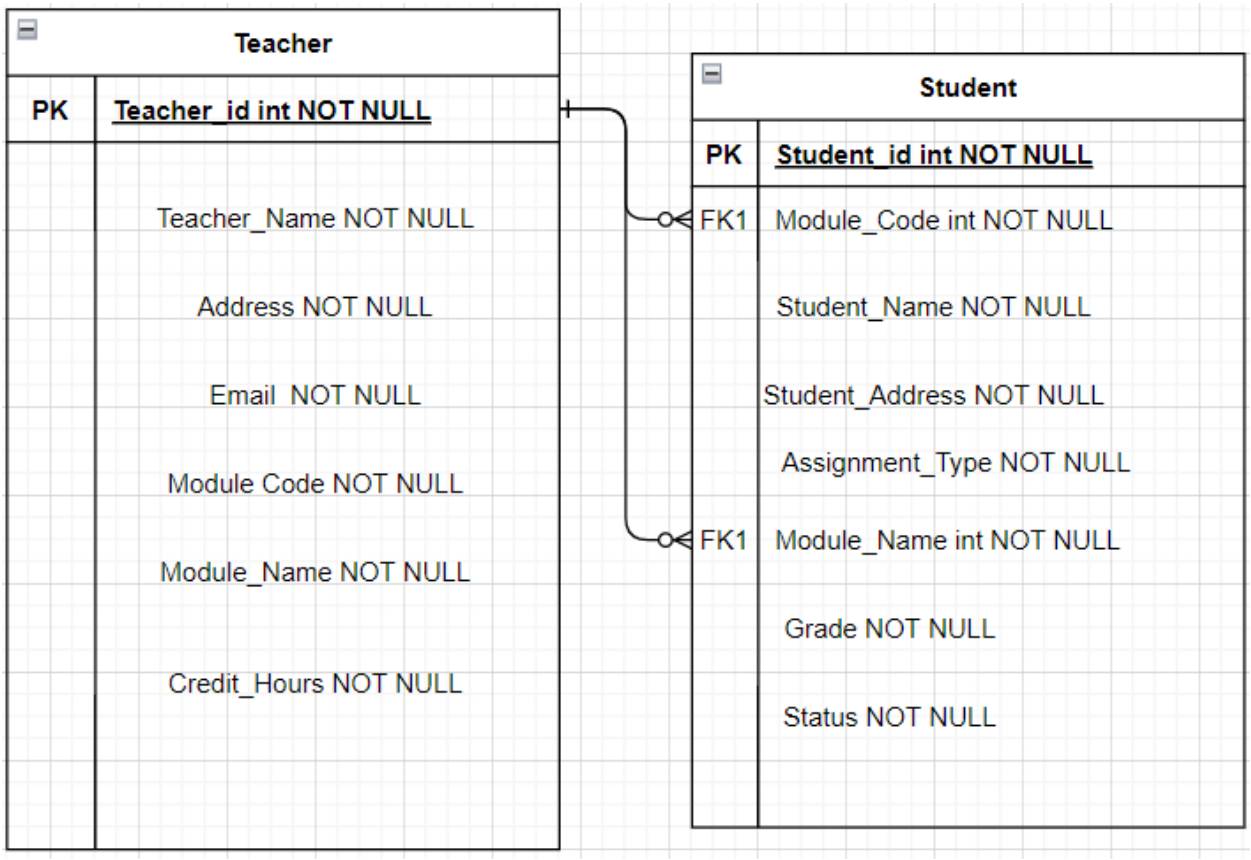


Table 1: ERD Before Normalization

3. Normalization

Normalization is a database design technique that reduce data redundancy and remove undesirable characteristics such as Insertion, Update, and Deletion Anomalies. Normalization rules divide larger tables into smaller tables and use relationships to connect them. The goal of SQL normalization is to remove redundant (repetitive) data and ensure that data is stored logically (Powell, 2006).

Underlined attributes are primary keys, while * (Asterisk symbol) attributes are foreign keys. Attributes with an underline and an asterisk are both primary and foreign keys.

3.1. Figure 1

UNF

The UNF includes all of the attributes, which are identified by the primary key (S.N./Teacher_ID). The repeating attributes are enclosed in curly braces ('{}'). Repeating attributes such as Teacher ID, Name, and so on are enclosed in curly braces ('{}') and the UNF of the first table is given below:

Teacher (Teacher_ID, Teacher_Name, {Address}, Email, {Module_Code, Module_Name, Credit_Hours})

1NF

In the First Normal Form, the repeating columns are divided into several entities. Each entity has one or more unique identifiers, also known as primary keys. The entity-entity relationship is formed by referencing the entity with a foreign key.

Teacher (Teacher_ID, Teacher_Name, Email)

Teacher_Address (Teacher_ID*, Address)

Module_Teacher (Teacher_ID*, Module_Code, Module_Name, Credit_Hours)

2NF

The 2NF tables are obtained by removing the partial dependencies. When a portion of the composite primary key provides non-key attributes, partial dependency exists. The Partial Dependency is fixed by creating a new table with the part of the primary key and making reference non-key attributes.

Since Module_Code → Module_Name, Credit_Hours [Partial Dependency]

Since Address_ID → Address [Partial Dependency]

The above dependency is separated by a new table in 2NF:

Teacher (Teacher_ID, Teacher_Name, Email)

Teacher_Address (Teacher_ID*, Address)

Module_Teacher (Teacher_ID*, Module_Code*)

Module (Module_Code, Module_Name, Credit_Hours)

3NF

There is no transitive dependency because there is no indirect relationship between the non-key attributes, and the tables created in 2NF are the final tables in 3NF.

Teacher (Teacher_ID, Teacher_Name, Email)

Teacher_Address (Teacher_ID*, Address)

Module_Teacher (Teacher_ID*, Module_Code*)

Module (Module_Code, Module_Name, Credit_Hours)

3.2. Figure 2

UNF

The UNF includes all of the attributes, which are identified by the primary key (Student_ID). The repeating attributes are enclosed in curly braces ('{}'). Repeating attributes such as Module_Code, Module_Name, and so on are enclosed in curly braces ('{}') and the UNF of the first table is given below:

Student (Student_ID, Student Name, Student Address, {Module_Code, Module_Name, {Assignment_Type, Grade, Status}})

1NF

In the First Normal Form, the repeating columns are divided into several entities. Each entity has one or more unique identifiers, also known as primary keys. The entity-entity relationship is formed by referencing the entity with a foreign key.

Student (<u>Student_ID</u>, Student_Name, Student_Address)

Module (<u>Module_Code</u>, Module_Name)

Assignment (<u>Assignment_ID</u>, <u>ModuleCode</u>*, <u>Student_ID</u>*, Assignment_Type, Grade, Status)

2NF

The 2NF tables are obtained by removing the partial dependencies. When a portion of the composite primary key provides non-key attributes, partial dependency exists. The Partial Dependency is fixed by creating a new table with the part of the primary key and making reference non-key attributes.

Student (<u>Student_ID</u>, Student_Name, Student_Address)

Student_Assignment (<u>Student_ID</u>*, <u>Assignment_ID</u>*)

Module (<u>Module_Code</u>, Module_Name)

Assignment (<u>Assignment_ID</u>, ModuleCode*, Assignment_Type, Grade, Status)

3NF

We get the 3NF tables by removing the Transitive Dependency. When one non-key attribute provides the value of another non-key attribute, this is referred to as transitive dependence. There is an indirect relationship between two non-essential characteristics. Transitive Dependency is only possible if a table contains more than one non-key attribute.

Student (Student_ID, Student_Name, Student_Address)

In Module, Assignment_ID → Assignment_Type [No Column]

Assignment_ID → Grade → Status

Assignment (<u>Assignment_ID</u>, ModuleCode*, Assignment_Type, Grade*)

Grade (<u>Grade</u>, Status)

Assumption and Integration

Assumption

As shown told in the case study, ABC College has a variety of departments. Some departments conduct and manage student examinations, assignments, and results, while others manage student fees records. Only students who have paid the college fee and have an attendance rate of 80% or higher will be eligible to give module assignments/examinations. Throughout the journey, a student studies various types of modules. A college allocates one teacher to one or many modules. After graduation, a student can work as a teacher at a college. Based on the study of this case the follow tables can be assumed.

Department (Department_ID, Department_Name)

Module_Student (Module_Code*, Student_ID*, Attendance)

Payment (Payment_ID, Payment, Payment_Date)

Student_Payment (Payment_id*, Student_id*)

Address (Address_ID, Address_Name)

Student_Details (Assignment_ID, Module_Code, Student_ID, Grade)

Integration

Integrating all tables together:

Teacher (Teacher_ID, Teacher_Name, Email)

Teacher_Address (Teacher_ID*, Address)

Module_Teacher (Teacher_ID*, Module_Code*)

Module (Module_Code, Module_Name, Credit_Hours)

Student (Student_ID, Student_Name, Student_Address)

Assignment (Assignment_ID, ModuleCode*, Assignment_Type, Grade*)

Grade (Grade, Status)

Department (Department_ID, Department_Name)

Module_Student (Module_Code*, Student_ID*, Attendance)

Payment (Payment_ID, Payment, Payment_Date)

Student_Payment (Payment_id*, Student_id*)

Address (Address_ID, Address_Name

Student_Details (Assignment_ID, Module_Code, Student_ID, Grade)

4. ER Diagram

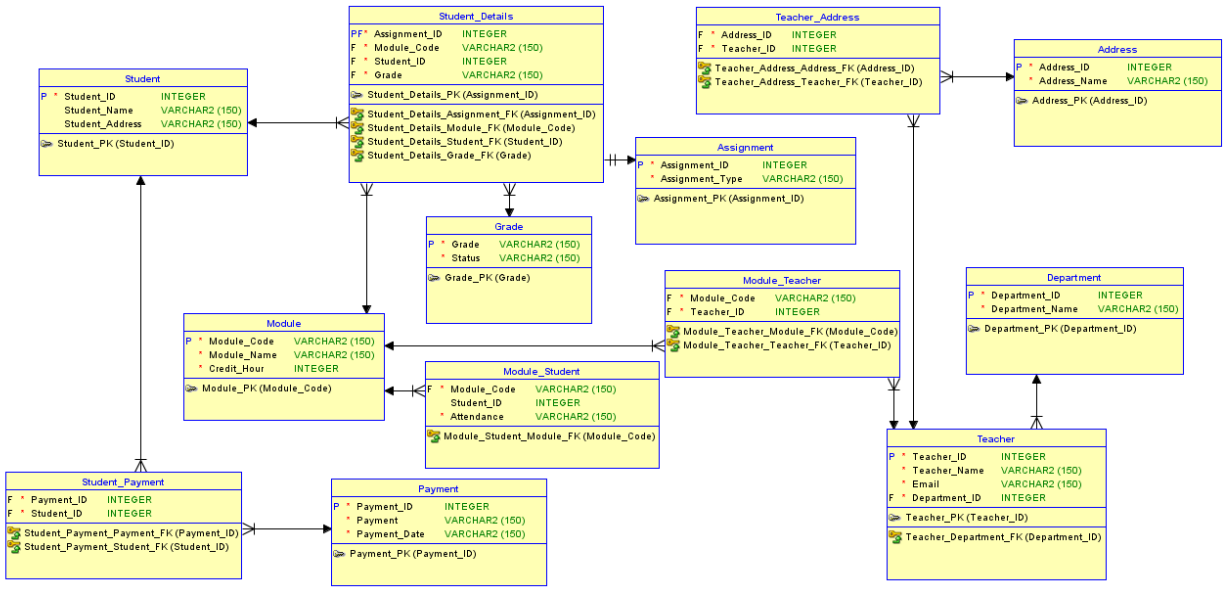


Figure 1: Entity Relationship Diagram

5. Data Dictionary

Department Table

S.N.	Field Name	Data Type	Size	Constraints	Description
1.	Department_ID	INTEGER	4 bytes	PRIMARY KEY	Unique key to identify Department
2.	Department_Name	VARCHAR	150	NOT NULL	Name of Department

Table 2: Data Dictionary for Department Table

Grade Table

S.N.	Field Name	Data Type	Size	Constraints	Description
1.	Grade	VARCHAR	150	PRIMARY KEY	Unique key to identify Grade
2.	Stats	VARCHAR	150	NOT NULL	Stats of Grade

Table 3:Data Dictionary for Grade Table

Assignment Table

S.N.	Field Name	Data Type	Size	Constraints	Description
1.	Assignment_ID	Int	4 byes	PRIMARY KEY	Unique key to identify Assignment
2.	Assignment_Type	VARCHAR	150	NOT NULL	Types of assignment

Table 4:Data Dictionary for Assignment Table

Student Table

S.N.	Field Name	Data Type	Size	Constraints	Description
1.	Student_ID	INTEGER	4 bytes	PRIMARY KEY	Unique key to identify Student
2.	Student_Name	VARCHAR	150	NOT NULL	Name of Student
3.	Student_Address	VARCHAR	150	NOT NULL	Address of Student

Table 5:Data Dictionary for Student Table

Address Table

S.N.	Field Name	Data Type	Size	Constraints	Description
1.	Address_ID	Int	4 byes	PRIMARY KEY	Unique key to identify Address
2.	Address_Name	VARCHAR	150	NOT NULL	Name of address

Table 6:Data Dictionary for Address Table

Payment Table

S.N.	Field Name	Data Type	Size	Constraints	Description
1.	Payment_ID	INTEGER	4 bytes	PRIMARY KEY	Unique key to identify Payment
2.	Payment	VARCHAR	150	NOT NULL	Amount of payment
3.	Payment_Date	VARCHAR	150	NOT NULL	Date of payment

Table 7:Data Dictionary for Payment Table

Module Table

S.N.	Field Name	Data Type	Size	Constraints	Description
1.	Module_Code	VARCHAR	150	PRIMARY KEY	Unique key to identify Module
2.	Module_Name	VARCHAR	150	NOT NULL	Name of Module
3.	Credit_Hours	VARCHAR	150	NOT NULL	Time of Modules in hour

Table 8:Data Dictionary for Module Table

Student_Details Table

S.N.	Field Name	Data Type	Size	Constraints	Description
1.	Assignment_ID	INTEGER	4 bytes	FOREIGN KEY	Unique key to identify Assignment
2.	Module_Code	VARCHAR	150	FOREIGN KEY	Unique key to identify Module
3.	Student_ID	INTEGER	4 bytes	FOREIGN KEY	Unique key to identify Student
4.	Grade	VARCHAR	150	FOREIGN KEY	Unique key to identify Grade

Table 9:Data Dictionary for Student_Details Table

Teacher Table

S.N.	Field Name	Data Type	Size	Constraints	Description
1.	Teacher_ID	INTEGER	4 bytes	PRIMARY KEY	Unique key to identify Teacher
2.	Teacher_Name	VARCHAR	150	NOT NULL	Name of teacher
3.	Email	VARCHAR	150	NOT NULL	Email of teacher
4.	Department_ID	INTEFGER	4 bytes	FOREIGN KEY	Unique key to identify Department

Table 10:Data Dictionary for Teacher Table

Module_Teacher Table

S.N.	Field Name	Data Type	Size	Constraints	Description
1.	Module_Code	VARCHAR	150	FOREIGN KEY	Unique key to identify Module
2.	Teacher_ID	INTEGER	4 bytes	FOREIGN KEY	Unique key to identify Teacher

Table 11:Data Dictionary for Module_Teacher Table

Teacher_Address Table

S.N.	Field Name	Data Type	Size	Constraints	Description
1.	Address_ID	INTEGER	4 byes	FOREIGN KEY	Unique key to identify Address
2.	Teacher_ID	INTEGER	4 bytes	FOREIGN KEY	Unique key to identify Teacher

Table 12:Data Dictionary for Teacher_Address Table

Student_Payment Table

S.N.	Field Name	Data Type	Size	Constraints	Description
1.	Payment_ID	INTEGER	150	FOREIGN KEY	Unique key to identify Payment
2.	Student_ID	INTEGER	4 bytes	FOREIGN KEY	Unique key to identify Student

Table 13:Data Dictionary for Student_Payment Table

Module_Student Table

S.N.	Field Name	Data Type	Size	Constraints	Description
1.	Module_Code	VARCHAR	150	FOREIGN KEY	Unique key to identify Module
2.	Student_ID	INTEGER	4 bytes	FOREIGN KEY	Unique key to identify Student
3.	Attendance	VARCHAR	150	NOT NULL	Attendance of Student

Table 14: Data Dictionary for Module_Student Table

6. Generation of Database

6.1. Creating and connecting user

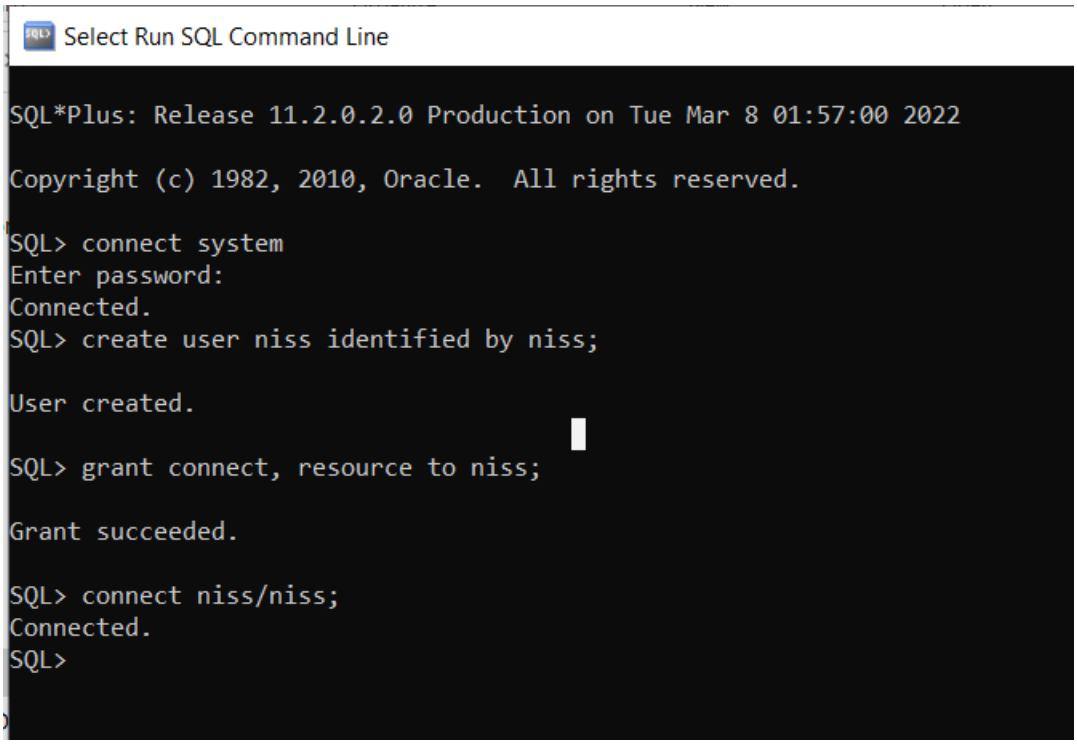


Figure 2: Creating and connecting user

6.2. Connecting with database

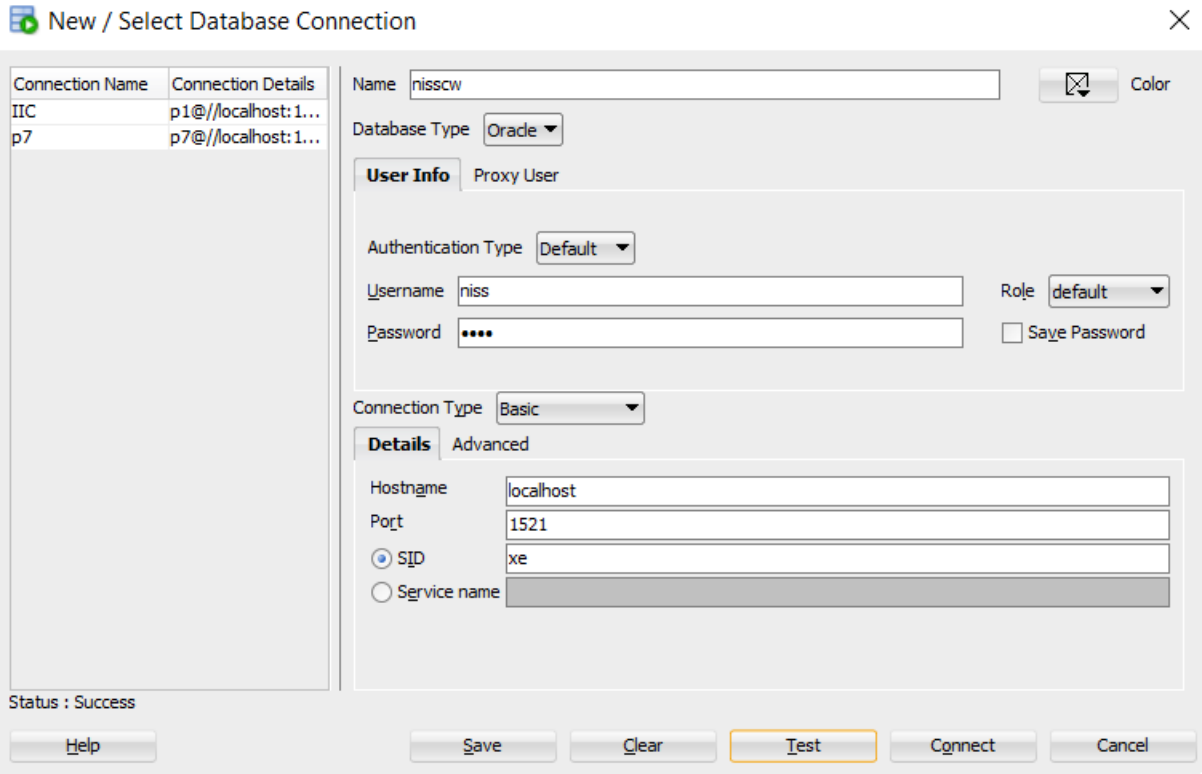


Figure 3:Connecting with user and database

6.3. Creating student table

```
CREATE TABLE student (
    student_id       INTEGER NOT NULL,
    student_name     VARCHAR2(150),
    student_address  VARCHAR2(150)
);

ALTER TABLE student ADD CONSTRAINT student_pk PRIMARY KEY ( student_id );
```

Script Output

Task completed in 0.1 seconds

```
Table STUDENT created.


Table STUDENT altered.
```

Figure 4: Creating Student Table

6.4. Creating grade table

Worksheet | Query Builder

```
CREATE TABLE grade (
    grade   VARCHAR2(150) NOT NULL,
    status  VARCHAR2(150) NOT NULL
);

ALTER TABLE grade ADD CONSTRAINT grade_pk PRIMARY KEY ( grade );
```

Script Output

Task completed in 0.076 seconds

```
Table GRADE created.


Table GRADE altered.
```

Figure 5: Creating Grade Table

6.5. Creating assignment table

Worksheet | Query Builder

```
CREATE TABLE assignment (
    assignment_id    INTEGER NOT NULL,
    assignment_type  VARCHAR2(150) NOT NULL
);

ALTER TABLE assignment ADD CONSTRAINT assignment_pk PRIMARY KEY ( assignment_id );
```

Script Output

Task completed in 0.078 seconds

```
Table ASSIGNMENT created.


Table ASSIGNMENT altered.
```

Figure 6: Creating Assignment Table

6.6. Creating department table

Worksheet | Query Builder

```
CREATE TABLE department (
    department_id     INTEGER NOT NULL,
    department_name   VARCHAR2(150) NOT NULL
);

ALTER TABLE department ADD CONSTRAINT department_pk PRIMARY KEY ( department_id );
```

Script Output

Task completed in 0.082 seconds

```
Table DEPARTMENT created.


Table DEPARTMENT altered.
```

Figure 7: Creating Department Table

6.7. Creating address table

Worksheet | Query Builder

```
CREATE TABLE address (
    address_id    INTEGER NOT NULL,
    address_name  VARCHAR2(150) NOT NULL
);

ALTER TABLE address ADD CONSTRAINT address_pk PRIMARY KEY ( address_id );
```

Script Output

Task completed in 0.095 seconds

```
Table ADDRESS created.


Table ADDRESS altered.
```

Figure 8: Creating Address Table

6.8. Creating payment table

Worksheet | Query Builder

```
CREATE TABLE payment (
    payment_id     INTEGER NOT NULL,
    payment        VARCHAR2(150) NOT NULL,
    payment_date   VARCHAR2(150) NOT NULL
);

ALTER TABLE payment ADD CONSTRAINT payment_pk PRIMARY KEY ( payment_id );
```

Script Output

Task completed in 0.077 seconds

```
Table PAYMENT created.


Table PAYMENT altered.
```

Figure 9: Creating Payment Table

6.9. Creating module table

Worksheet | Query Builder

```
CREATE TABLE module (
    module_code  VARCHAR2(150) NOT NULL,
    module_name  VARCHAR2(150) NOT NULL,
    credit_hour  INTEGER NOT NULL
);

ALTER TABLE module ADD CONSTRAINT module_pk PRIMARY KEY ( module_code );
```

Script Output | Task completed in 0.077 seconds

```
Table MODULE created.


Table MODULE altered.
```

Figure 10: Creating Module Table

6.10. Creating student_details table

Worksheet | Query Builder

```
CREATE TABLE student_details (
    assignment_id  INTEGER NOT NULL,
    module_code    VARCHAR2(150) NOT NULL,
    student_id     INTEGER NOT NULL,
    grade          VARCHAR2(150) NOT NULL
);

ALTER TABLE student_details ADD CONSTRAINT student_details_pk PRIMARY KEY ( assignment_id );
```

Script Output | Task completed in 0.078 seconds

```
Table STUDENT_DETAILS created.


Table STUDENT_DETAILS altered.
```

Figure 11: Creating Student_ Details Table

6.11. Creating module_student table

Worksheet | Query Builder

```
CREATE TABLE module_student (
    module_code  VARCHAR2(150) NOT NULL,
    student_id   INTEGER,
    attendance   VARCHAR2(150) NOT NULL
);
```

Script Output | Task completed in 0.049 seconds

```
Table MODULE_STUDENT created.
```

Figure 12: Creating Module Student Table

6.12. Creating teacher table

Worksheet | Query Builder

```
CREATE TABLE teacher (
    teacher_id      INTEGER NOT NULL,
    teacher_name    VARCHAR2(150) NOT NULL,
    email           VARCHAR2(150) NOT NULL,
    department_id   INTEGER NOT NULL
);

ALTER TABLE teacher ADD CONSTRAINT teacher_pk PRIMARY KEY ( teacher_id );
```

Script Output

Task completed in 0.078 seconds

```
Table TEACHER created.


Table TEACHER altered.
```

Figure 13: Creating Teacher Table

6.13. Creating module_teacher table

Worksheet | Query Builder

```
CREATE TABLE module_teacher (
    module_code  VARCHAR2(150) NOT NULL,
    teacher_id   INTEGER NOT NULL
);
```

Script Output | Task completed in 0.048 seconds

```
Table MODULE_TEACHER created.
```

Figure 14: Creating Module_Teacher Table

6.14. Creating teacher_address table

Worksheet | Query Builder

```
CREATE TABLE teacher_address (
    address_id  INTEGER NOT NULL,
    teacher_id  INTEGER NOT NULL
);
```

Script Output | Task completed in 0.059 seconds

```
Table TEACHER_ADDRESS created.
```

Figure 15: Creating Teacher_Address Table

6.15. Creating student_payment table

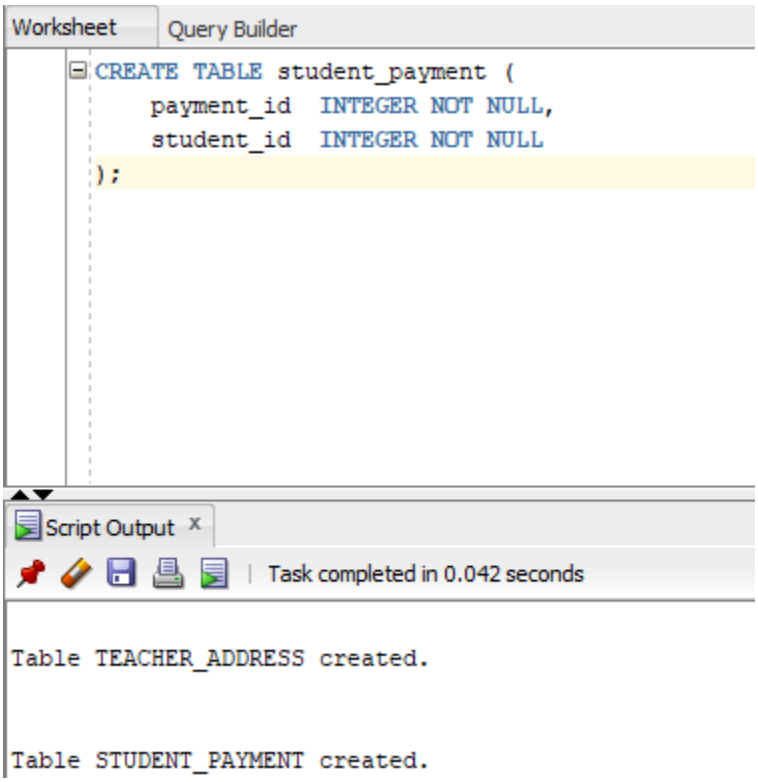


Figure 16: Creating Student_Payment Table

6.16. Altering all the tables that contain foreign keys

Worksheet | Query Builder

```
        REFERENCES student ( student_id );

ALTER TABLE teacher_address
    ADD CONSTRAINT teacher_address_address_fk FOREIGN KEY ( address_id )
        REFERENCES address ( address_id );

ALTER TABLE teacher_address
    ADD CONSTRAINT teacher_address_teacher_fk FOREIGN KEY ( teacher_id )
        REFERENCES teacher ( teacher_id );

ALTER TABLE teacher
    ADD CONSTRAINT teacher_department_fk FOREIGN KEY ( department_id )
        REFERENCES department ( department_id );
```

Script Output | Task completed in 0.198 seconds

```
Table STUDENT_DETAILS altered.

Table STUDENT_DETAILS altered.

Table STUDENT_DETAILS altered.

Table STUDENT_PAYMENT altered.

Table STUDENT_PAYMENT altered.
```

Figure 17: Altering Tables that contain FOREIGN KEYS

6.17. Inserting the values in student table

IIC.sql | Welcome Page | nisscw | nisscw~1 | STUDENT

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Sort.. | Filter:

	STUDENT_ID	STUDENT_NAME	STUDENT_ADDRESS
1	1	Nischal Rai	Dharan
2	2	Tonny Limbu	Itahari
3	3	Tsui Tamang	Birthnagar
4	4	Garnet Chettri	Damak
5	5	Rose Magar	Dulahari

Messages - Log

```
INSERT INTO "NISS"."STUDENT" (STUDENT_ID, STUDENT_NAME, STUDENT_ADDRESS) VALUES ('1', 'Nischal Rai', 'Dharan')
INSERT INTO "NISS"."STUDENT" (STUDENT_ID, STUDENT_NAME, STUDENT_ADDRESS) VALUES ('2', 'Tonny Limbu', 'Itahari')
INSERT INTO "NISS"."STUDENT" (STUDENT_ID, STUDENT_NAME, STUDENT_ADDRESS) VALUES ('3', 'Tsui Tamang', 'Birthnagar')
INSERT INTO "NISS"."STUDENT" (STUDENT_ID, STUDENT_NAME, STUDENT_ADDRESS) VALUES ('4', 'Garnet Chettri', 'Damak')
INSERT INTO "NISS"."STUDENT" (STUDENT_ID, STUDENT_NAME, STUDENT_ADDRESS) VALUES ('5', 'Rose Magar', 'Dulahari')

Commit Successful
```

Figure 18: Inserting Values in Student Table

6.18. Inserting the values in grade table

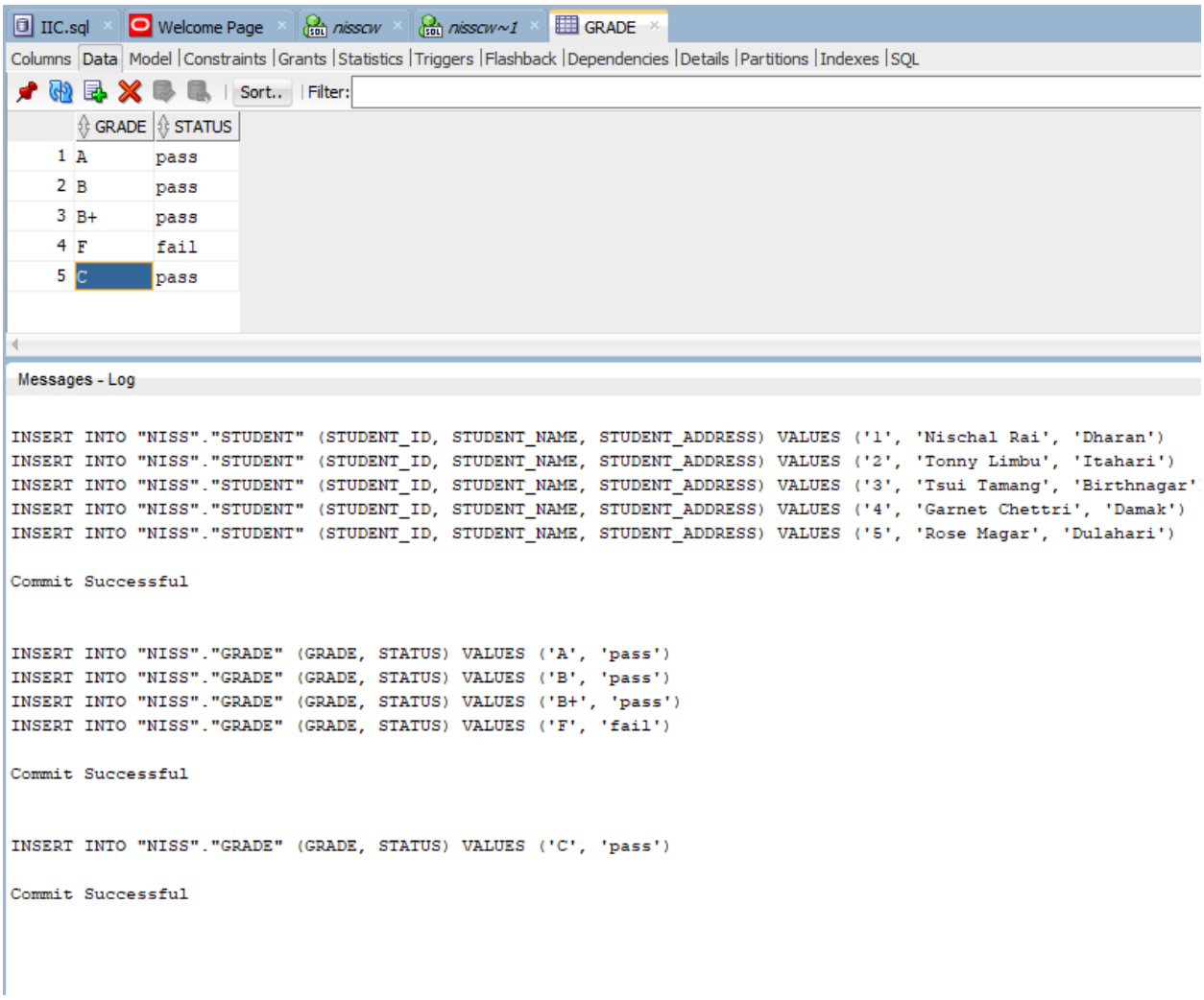


Figure 19: Inserting Values in Grade Table

6.19. Inserting into values of assignment table

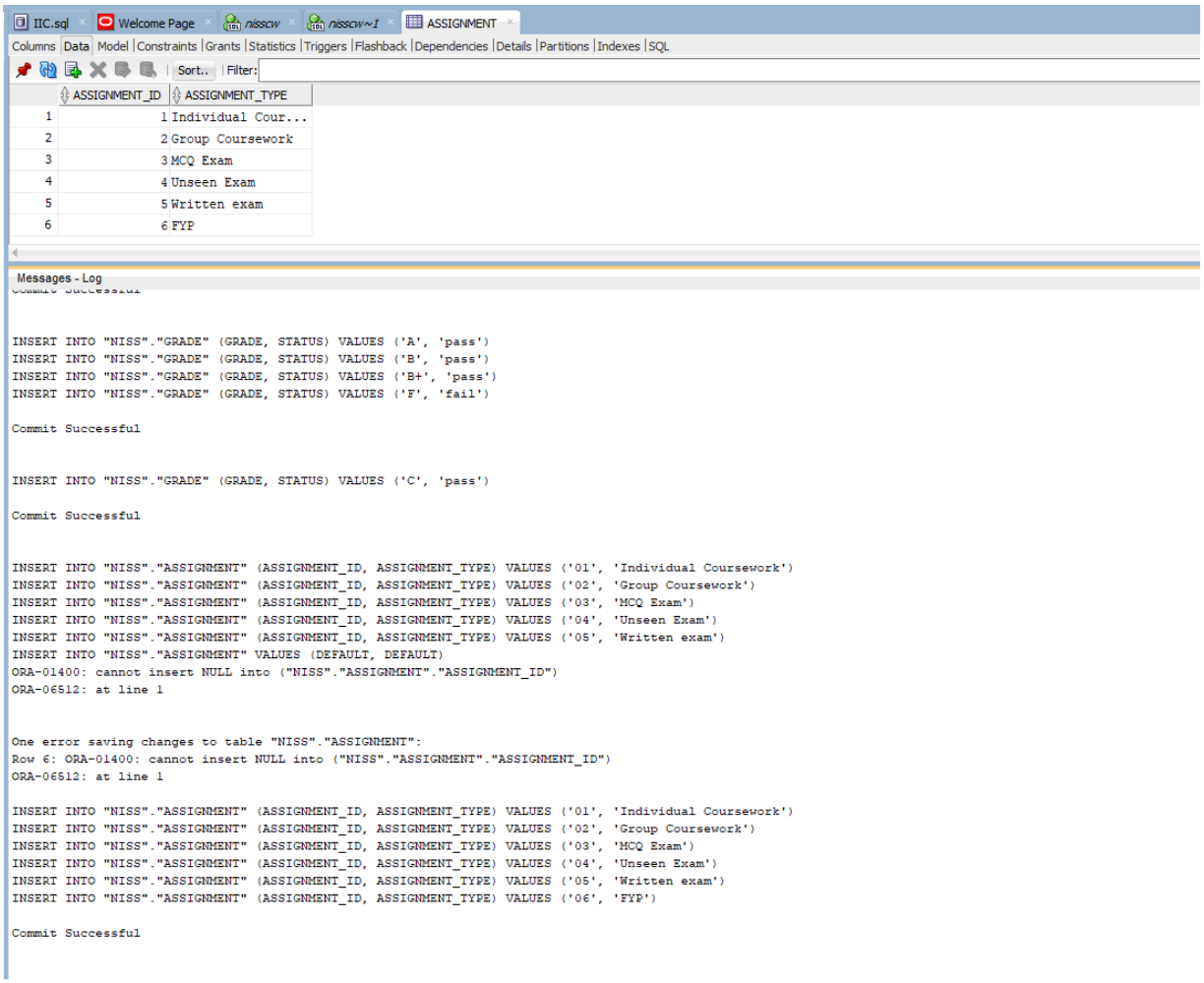


Figure 20: Inserting Values in Assignment Table

6.20. Inserting values into department table

IIC.sql | Welcome Page | nisscw | nisscw~1 | DEPARTMENT

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Sort.. | Filter:

	DEPARTMENT_ID	DEPARTMENT_NAME
1	1	Academics
2	2	SSD
3	3	RTE
4	4	Finance

Messages - Log

```
INSERT INTO "NISS"."GRADE" (GRADE, STATUS) VALUES ('C', 'pass')

Commit Successful


INSERT INTO "NISS"."ASSIGNMENT" (ASSIGNMENT_ID, ASSIGNMENT_TYPE) VALUES ('01', 'Individual Coursework')
INSERT INTO "NISS"."ASSIGNMENT" (ASSIGNMENT_ID, ASSIGNMENT_TYPE) VALUES ('02', 'Group Coursework')
INSERT INTO "NISS"."ASSIGNMENT" (ASSIGNMENT_ID, ASSIGNMENT_TYPE) VALUES ('03', 'MCQ Exam')
INSERT INTO "NISS"."ASSIGNMENT" (ASSIGNMENT_ID, ASSIGNMENT_TYPE) VALUES ('04', 'Unseen Exam')
INSERT INTO "NISS"."ASSIGNMENT" (ASSIGNMENT_ID, ASSIGNMENT_TYPE) VALUES ('05', 'Written exam')
INSERT INTO "NISS"."ASSIGNMENT" VALUES (DEFAULT, DEFAULT)
ORA-01400: cannot insert NULL into ("NISS"."ASSIGNMENT"."ASSIGNMENT_ID")
ORA-06512: at line 1


One error saving changes to table "NISS"."ASSIGNMENT":
Row 6: ORA-01400: cannot insert NULL into ("NISS"."ASSIGNMENT"."ASSIGNMENT_ID")
ORA-06512: at line 1

INSERT INTO "NISS"."ASSIGNMENT" (ASSIGNMENT_ID, ASSIGNMENT_TYPE) VALUES ('01', 'Individual Coursework')
INSERT INTO "NISS"."ASSIGNMENT" (ASSIGNMENT_ID, ASSIGNMENT_TYPE) VALUES ('02', 'Group Coursework')
INSERT INTO "NISS"."ASSIGNMENT" (ASSIGNMENT_ID, ASSIGNMENT_TYPE) VALUES ('03', 'MCQ Exam')
INSERT INTO "NISS"."ASSIGNMENT" (ASSIGNMENT_ID, ASSIGNMENT_TYPE) VALUES ('04', 'Unseen Exam')
INSERT INTO "NISS"."ASSIGNMENT" (ASSIGNMENT_ID, ASSIGNMENT_TYPE) VALUES ('05', 'Written exam')
INSERT INTO "NISS"."ASSIGNMENT" (ASSIGNMENT_ID, ASSIGNMENT_TYPE) VALUES ('06', 'FYP')

Commit Successful


INSERT INTO "NISS"."DEPARTMENT" (DEPARTMENT_ID, DEPARTMENT_NAME) VALUES ('001', 'Academics')
INSERT INTO "NISS"."DEPARTMENT" (DEPARTMENT_ID, DEPARTMENT_NAME) VALUES ('002', 'SSD')
INSERT INTO "NISS"."DEPARTMENT" (DEPARTMENT_ID, DEPARTMENT_NAME) VALUES ('003', 'RTE')
INSERT INTO "NISS"."DEPARTMENT" (DEPARTMENT_ID, DEPARTMENT_NAME) VALUES ('004', 'Finance')

Commit Successful
```

Figure 21: Inserting Values in Department Table

6.21. Inserting values into address table

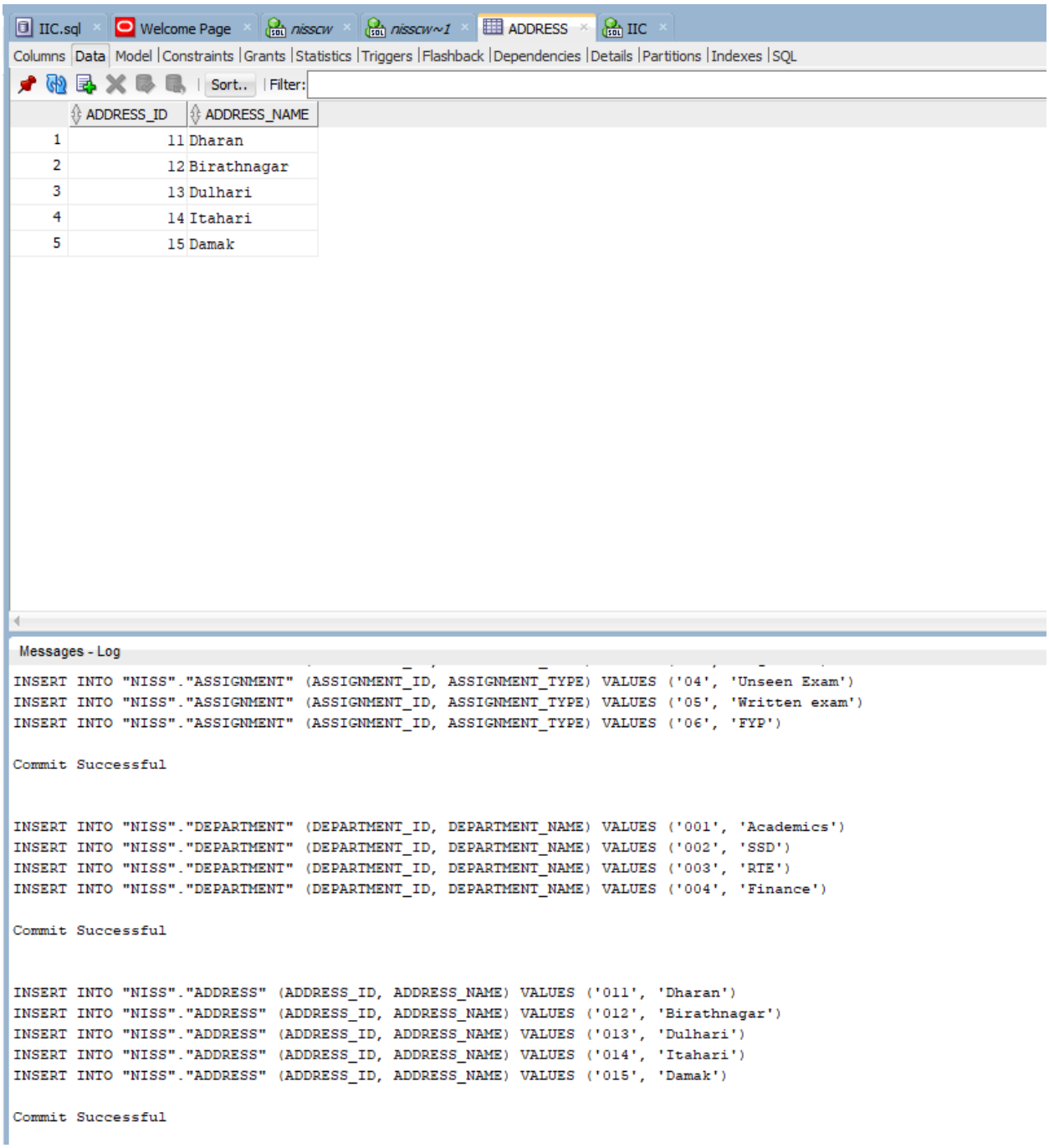


Figure 22: Inserting values in Address Table

6.22. Inserting values into Payment

	PAYMENT_ID	PAYMENT	PAYMENT_DATE
1	21	20000	01/01/2022
2	22	30000	02/02/2022
3	23	40000	03.03/2022

Messages - Log

```
INSERT INTO "NISS"."DEPARTMENT" (DEPARTMENT_ID, DEPARTMENT_NAME) VALUES ('001', 'Academics')
INSERT INTO "NISS"."DEPARTMENT" (DEPARTMENT_ID, DEPARTMENT_NAME) VALUES ('002', 'SSD')
INSERT INTO "NISS"."DEPARTMENT" (DEPARTMENT_ID, DEPARTMENT_NAME) VALUES ('003', 'RTE')
INSERT INTO "NISS"."DEPARTMENT" (DEPARTMENT_ID, DEPARTMENT_NAME) VALUES ('004', 'Finance')

Commit Successful


INSERT INTO "NISS"."ADDRESS" (ADDRESS_ID, ADDRESS_NAME) VALUES ('011', 'Dharan')
INSERT INTO "NISS"."ADDRESS" (ADDRESS_ID, ADDRESS_NAME) VALUES ('012', 'Birathnagar')
INSERT INTO "NISS"."ADDRESS" (ADDRESS_ID, ADDRESS_NAME) VALUES ('013', 'Dulhari')
INSERT INTO "NISS"."ADDRESS" (ADDRESS_ID, ADDRESS_NAME) VALUES ('014', 'Itahari')
INSERT INTO "NISS"."ADDRESS" (ADDRESS_ID, ADDRESS_NAME) VALUES ('015', 'Damak')

Commit Successful


INSERT INTO "NISS"."PAYMENT" (PAYMENT_ID, PAYMENT, PAYMENT_DATE) VALUES ('021', '20000', '01/01/2022')
INSERT INTO "NISS"."PAYMENT" (PAYMENT_ID, PAYMENT, PAYMENT_DATE) VALUES ('022', '30000', '02/02/2022')
INSERT INTO "NISS"."PAYMENT" (PAYMENT_ID, PAYMENT, PAYMENT_DATE) VALUES ('023', '40000', '03.03/2022')

Commit Successful
```

Figure 23: Inserting Values in Payment Table

6.23. Inserting values into module

IIC.sql | Welcome Page | nisscw | nisscw~1 | MODULE | IIC

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Sort.. | Filter:

	MODULE_CODE	MODULE_NAME	CREDIT_HOUR
1	CS01	Database	60
2	CS02	Java	65
3	CS03	WebPage de...	40
4	CS04	AI	60
5	CS05	Software E...	90

Messages - Log

```
INSERT INTO "NISS"."ADDRESS" (ADDRESS_ID, ADDRESS_NAME) VALUES ('012', 'Birathnagar')
INSERT INTO "NISS"."ADDRESS" (ADDRESS_ID, ADDRESS_NAME) VALUES ('013', 'Dulhari')
INSERT INTO "NISS"."ADDRESS" (ADDRESS_ID, ADDRESS_NAME) VALUES ('014', 'Itahari')
INSERT INTO "NISS"."ADDRESS" (ADDRESS_ID, ADDRESS_NAME) VALUES ('015', 'Damak')

Commit Successful


INSERT INTO "NISS"."PAYMENT" (PAYMENT_ID, PAYMENT, PAYMENT_DATE) VALUES ('021', '20000', '01/01/2022')
INSERT INTO "NISS"."PAYMENT" (PAYMENT_ID, PAYMENT, PAYMENT_DATE) VALUES ('022', '30000', '02/02/2022')
INSERT INTO "NISS"."PAYMENT" (PAYMENT_ID, PAYMENT, PAYMENT_DATE) VALUES ('023', '40000', '03.03/2022')

Commit Successful


INSERT INTO "NISS"."MODULE" (MODULE_CODE, MODULE_NAME, CREDIT_HOUR) VALUES ('CS01', 'Database', '60')
INSERT INTO "NISS"."MODULE" (MODULE_CODE, MODULE_NAME, CREDIT_HOUR) VALUES ('CS02', 'Java', '65')
INSERT INTO "NISS"."MODULE" (MODULE_CODE, MODULE_NAME, CREDIT_HOUR) VALUES ('CS03', 'WebPage designing', '40')
INSERT INTO "NISS"."MODULE" (MODULE_CODE, MODULE_NAME, CREDIT_HOUR) VALUES ('CS04', 'AI', '60')
INSERT INTO "NISS"."MODULE" (MODULE_CODE, MODULE_NAME, CREDIT_HOUR) VALUES ('CS05', 'Software Engineering', '90')

Commit Successful
```

Figure 24: Inserting Values in Module Table

6.24. Inserting values into student_details table

IIC.sql | Welcome Page | nisscw | nisscw~1 | STUDENT_DETAILS | IIC

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Sort.. | Filter:

	ASSIGNMENT_ID	MODULE_CODE	STUDENT_ID	GRADE
1	1	CS01	1	A
2	2	CS02	2	F
3	3	CS03	3	B
4	4	CS04	4	B+
5	5	CS05	5	F

Messages - Log

```
INSERT INTO "NISS"."PAYMENT" (PAYMENT_ID, PAYMENT, PAYMENT_DATE) VALUES ('022', '30000', '02/02/2022')
INSERT INTO "NISS"."PAYMENT" (PAYMENT_ID, PAYMENT, PAYMENT_DATE) VALUES ('023', '40000', '03.03/2022')

Commit Successful


INSERT INTO "NISS"."MODULE" (MODULE_CODE, MODULE_NAME, CREDIT_HOUR) VALUES ('CS01', 'Database', '60')
INSERT INTO "NISS"."MODULE" (MODULE_CODE, MODULE_NAME, CREDIT_HOUR) VALUES ('CS02', 'Java', '65')
INSERT INTO "NISS"."MODULE" (MODULE_CODE, MODULE_NAME, CREDIT_HOUR) VALUES ('CS03', 'WebPage designing', '40')
INSERT INTO "NISS"."MODULE" (MODULE_CODE, MODULE_NAME, CREDIT_HOUR) VALUES ('CS04', 'AI', '60')
INSERT INTO "NISS"."MODULE" (MODULE_CODE, MODULE_NAME, CREDIT_HOUR) VALUES ('CS05', 'Software Engineering', '90')

Commit Successful


INSERT INTO "NISS"."STUDENT_DETAILS" (ASSIGNMENT_ID, MODULE_CODE, STUDENT_ID, GRADE) VALUES ('01', 'CS01', '1', 'A')
INSERT INTO "NISS"."STUDENT_DETAILS" (ASSIGNMENT_ID, MODULE_CODE, STUDENT_ID, GRADE) VALUES ('02', 'CS02', '2', 'F')
INSERT INTO "NISS"."STUDENT_DETAILS" (ASSIGNMENT_ID, MODULE_CODE, STUDENT_ID, GRADE) VALUES ('03', 'CS03', '3', 'B')
INSERT INTO "NISS"."STUDENT_DETAILS" (ASSIGNMENT_ID, MODULE_CODE, STUDENT_ID, GRADE) VALUES ('04', 'CS04', '4', 'B+')
INSERT INTO "NISS"."STUDENT_DETAILS" (ASSIGNMENT_ID, MODULE_CODE, STUDENT_ID, GRADE) VALUES ('05', 'CS05', '5', 'F')

Commit Successful
```

Figure 25: Inserting Values in Student_Details Table

6.25. Inserting values into module_student table

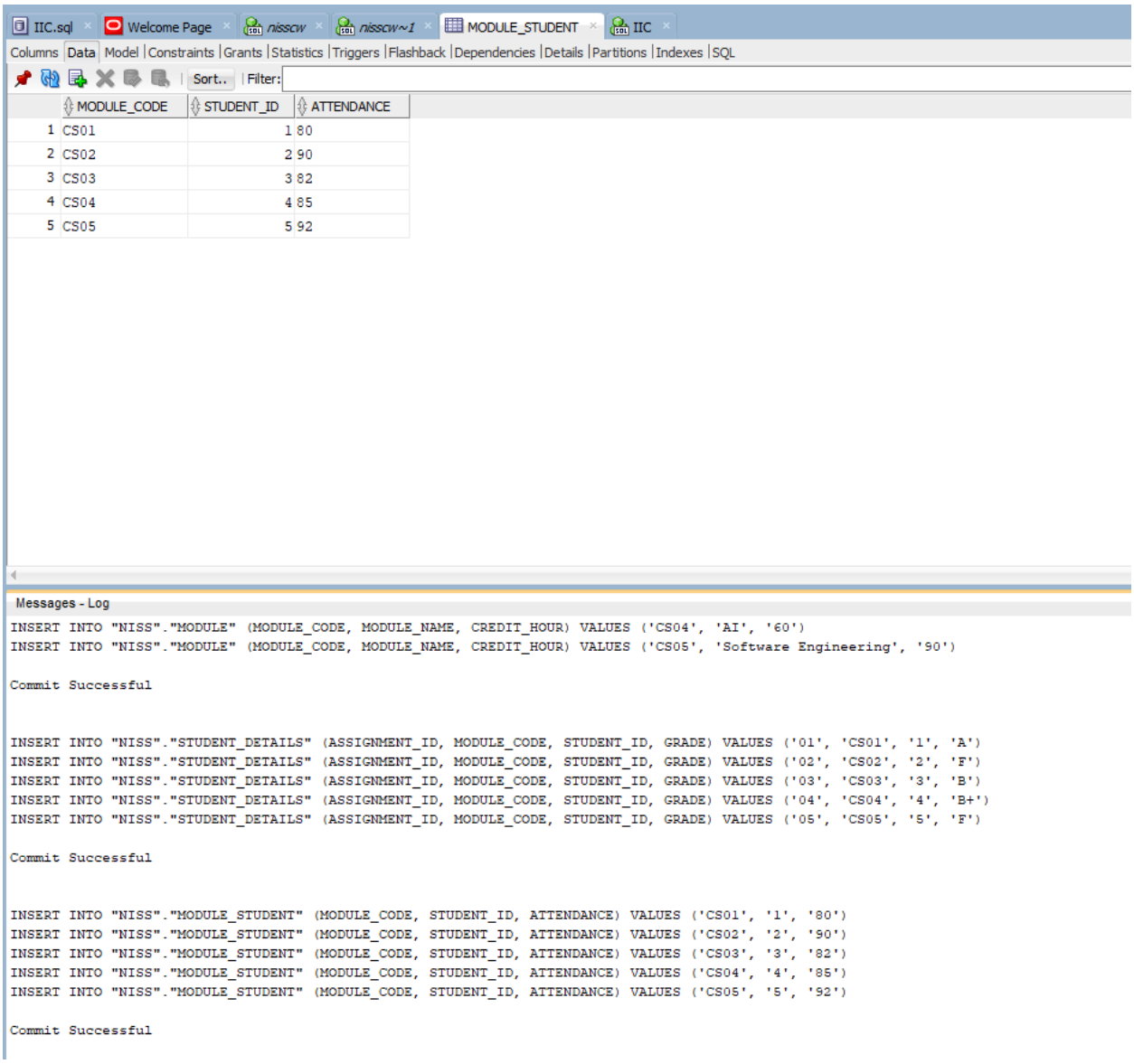
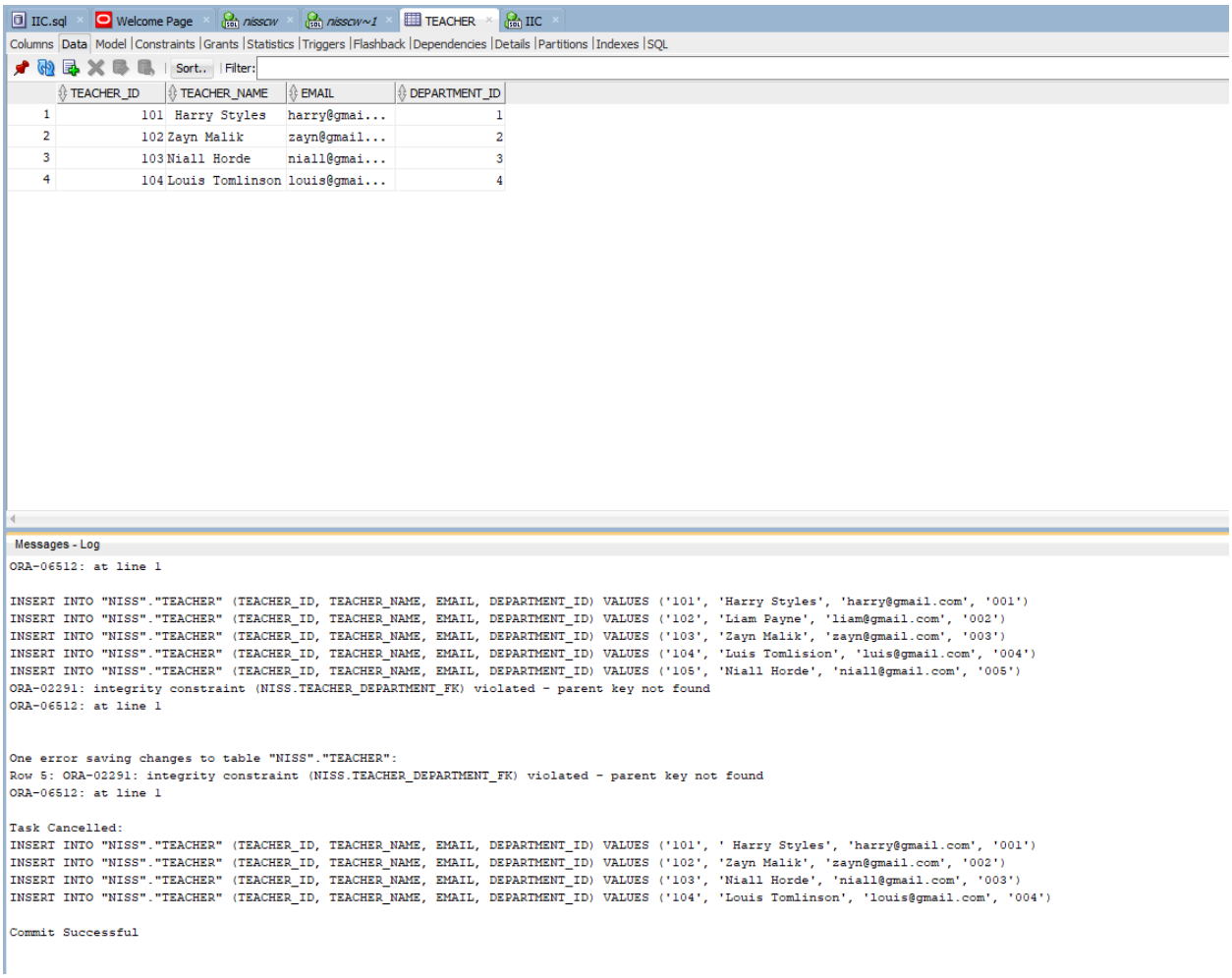


Figure 26: Inserting Values in Module_Student Table

6.26. Inserting values into teacher table



IIC.sql | Welcome Page | nisscw | nisscw~1 | TEACHER | IIC

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Sort.. | Filter:

	TEACHER_ID	TEACHER_NAME	EMAIL	DEPARTMENT_ID
1	101	Harry Styles	harry@gmai...	1
2	102	Zayn Malik	zayn@gmail...	2
3	103	Niall Horde	niall@gmai...	3
4	104	Louis Tomlinson	louis@gmai...	4

Messages - Log

```
ORA-06512: at line 1

INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('101', 'Harry Styles', 'harry@gmail.com', '001')
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('102', 'Liam Payne', 'liam@gmail.com', '002')
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('103', 'Zayn Malik', 'zayn@gmail.com', '003')
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('104', 'Luis Tomlision', 'luis@gmail.com', '004')
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('105', 'Niall Horde', 'niall@gmail.com', '005')
ORA-02291: integrity constraint (NISS.TEACHER_DEPARTMENT_FK) violated - parent key not found
ORA-06512: at line 1


One error saving changes to table "NISS"."TEACHER":
Row 5: ORA-02291: integrity constraint (NISS.TEACHER_DEPARTMENT_FK) violated - parent key not found
ORA-06512: at line 1

Task Cancelled:
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('101', ' Harry Styles', 'harry@gmail.com', '001')
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('102', 'Zayn Malik', 'zayn@gmail.com', '002')
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('103', 'Niall Horde', 'niall@gmail.com', '003')
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('104', 'Louis Tomlinson', 'louis@gmail.com', '004')

Commit Successful
```

Figure 27: Inserting Values in Teacher Table

6.27. Inserting values into module_teacher table

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Sort.. | Filter:

	MODULE_CODE	TEACHER_ID
1	CS01	101
2	CS02	102
3	CS03	103
4	CS04	104
5	CS05	101

Messages - Log

```
One error saving changes to table "NISS"."TEACHER":
Row 5: ORA-02291: integrity constraint (NISS.TEACHER_DEPARTMENT_FK) violated - parent key not found
ORA-06512: at line 1

Task Cancelled:
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('101', ' Harry Styles', 'harry@gmail.com', '001')
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('102', 'Zayn Malik', 'zayn@gmail.com', '002')
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('103', 'Niall Horde', 'niall@gmail.com', '003')
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('104', 'Louis Tomlinson', 'louis@gmail.com', '004')

Commit Successful


INSERT INTO "NISS"."MODULE_TEACHER" (MODULE_CODE, TEACHER_ID) VALUES ('CS01', '101')
INSERT INTO "NISS"."MODULE_TEACHER" (MODULE_CODE, TEACHER_ID) VALUES ('CS02', '102')
INSERT INTO "NISS"."MODULE_TEACHER" (MODULE_CODE, TEACHER_ID) VALUES ('CS03', '103')
INSERT INTO "NISS"."MODULE_TEACHER" (MODULE_CODE, TEACHER_ID) VALUES ('CS04', '104')
INSERT INTO "NISS"."MODULE_TEACHER" (MODULE_CODE, TEACHER_ID) VALUES ('CS05', '101')

Commit Successful
```

Figure 28: Inserting Values in Module_Teacher Table

6.28. Inserting values into teacher_address table

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Sort.. | Filter:

	ADDRESS_ID	TEACHER_ID
1	11	101
2	12	102
3	13	104
4	14	103

Messages - Log

```
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('102', 'Zayn Malik', 'zayn@gmail.com', '002')
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('103', 'Niall Horde', 'niall@gmail.com', '003')
INSERT INTO "NISS"."TEACHER" (TEACHER_ID, TEACHER_NAME, EMAIL, DEPARTMENT_ID) VALUES ('104', 'Louis Tomlinson', 'louis@gmail.com', '004')

Commit Successful


INSERT INTO "NISS"."MODULE_TEACHER" (MODULE_CODE, TEACHER_ID) VALUES ('CS01', '101')
INSERT INTO "NISS"."MODULE_TEACHER" (MODULE_CODE, TEACHER_ID) VALUES ('CS02', '102')
INSERT INTO "NISS"."MODULE_TEACHER" (MODULE_CODE, TEACHER_ID) VALUES ('CS03', '103')
INSERT INTO "NISS"."MODULE_TEACHER" (MODULE_CODE, TEACHER_ID) VALUES ('CS04', '104')
INSERT INTO "NISS"."MODULE_TEACHER" (MODULE_CODE, TEACHER_ID) VALUES ('CS05', '101')

Commit Successful


INSERT INTO "NISS"."TEACHER_ADDRESS" (ADDRESS_ID, TEACHER_ID) VALUES ('011', '101')
INSERT INTO "NISS"."TEACHER_ADDRESS" (ADDRESS_ID, TEACHER_ID) VALUES ('012', '102')
INSERT INTO "NISS"."TEACHER_ADDRESS" (ADDRESS_ID, TEACHER_ID) VALUES ('013', '104')
INSERT INTO "NISS"."TEACHER_ADDRESS" (ADDRESS_ID, TEACHER_ID) VALUES ('014', '103')

Commit Successful
```

Figure 29: Inserting Values in Teacher Address Table

6.29. Inserting values into student_payment table

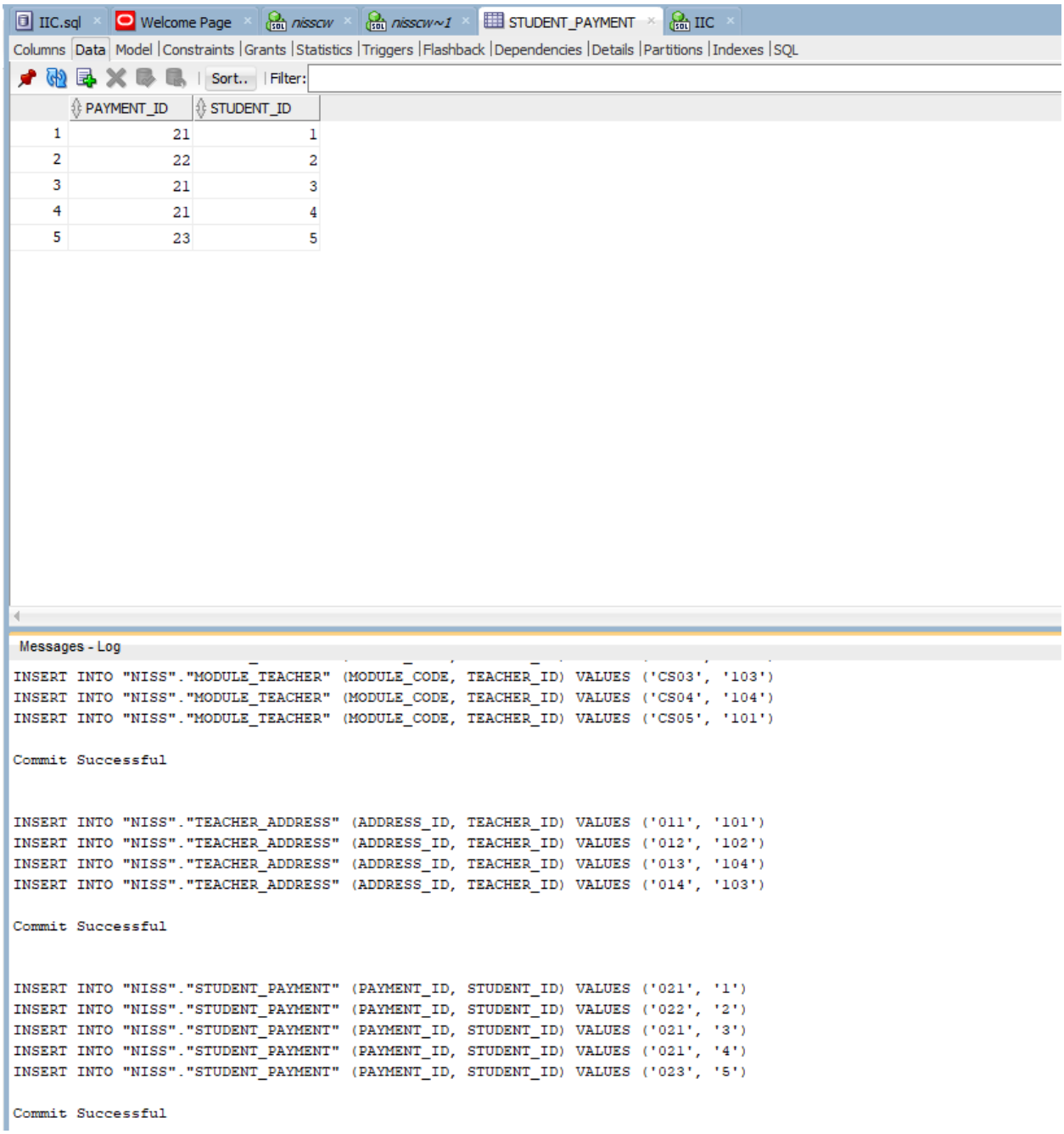


Figure 30: Inserting Values in Student_Payment Table

6.30. Showing the values of student table

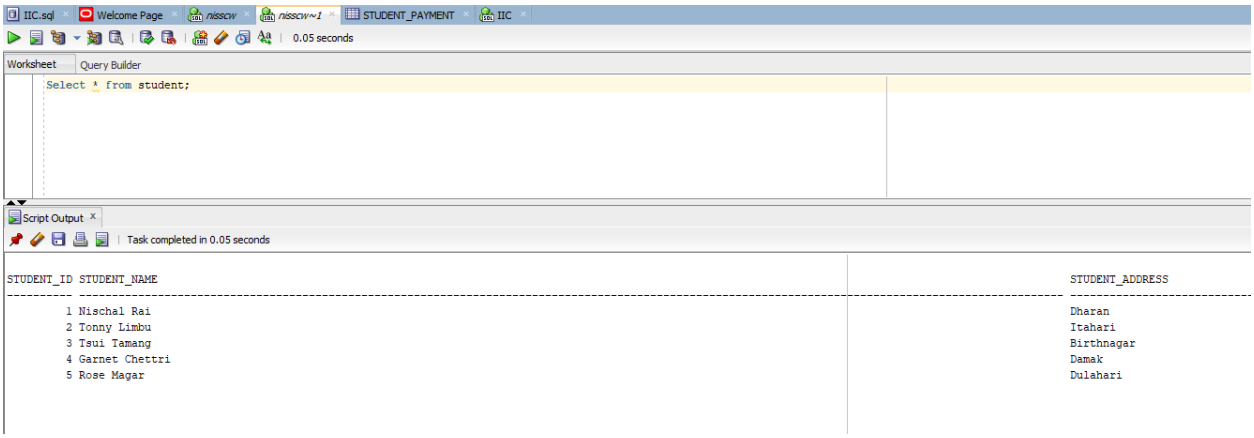


Figure 31: Showing the Values of Student_Address Table

6.31. Showing the values of grade table

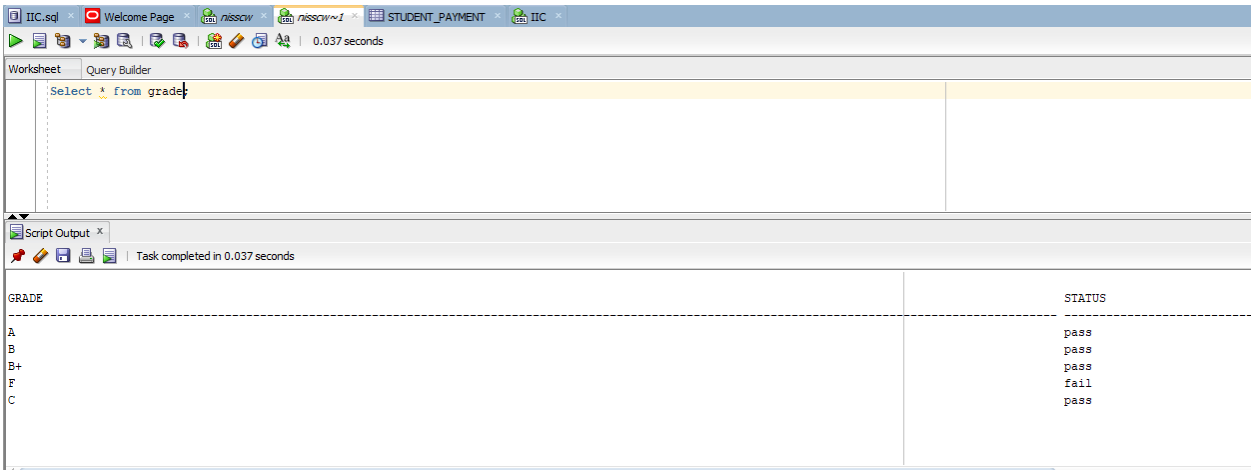


Figure 32: Showing the Values of Grade Table

6.32. Showing the values of assignment table

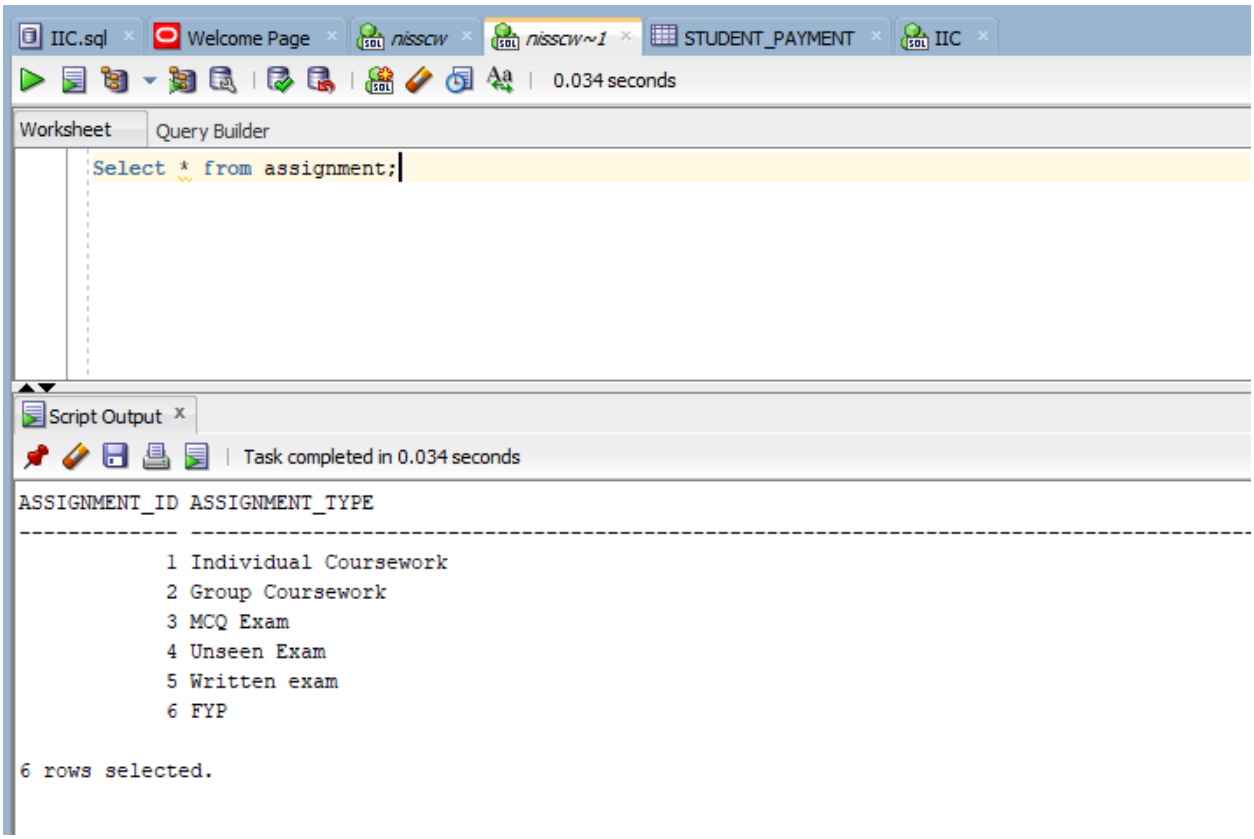


Figure 33: Showing the Values of Assignment Table

6.33. Showing the values of department table

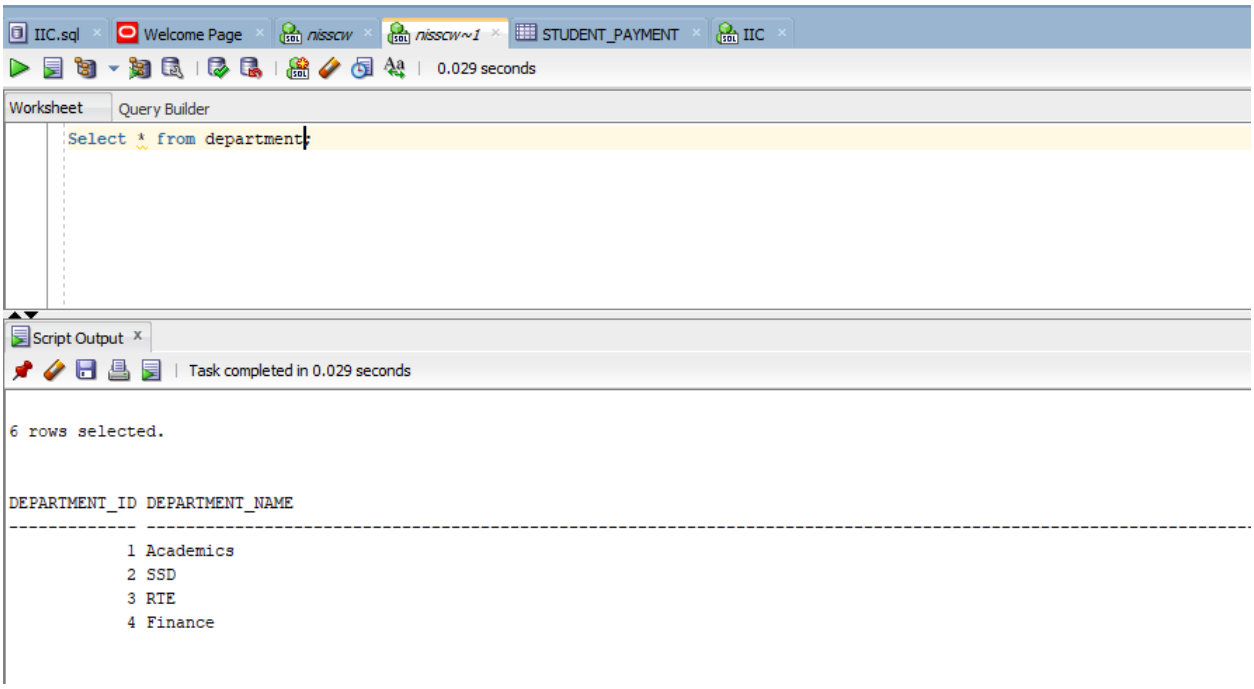


Figure 34: Showing the Values of Department Table

6.34. Showing the values of address table

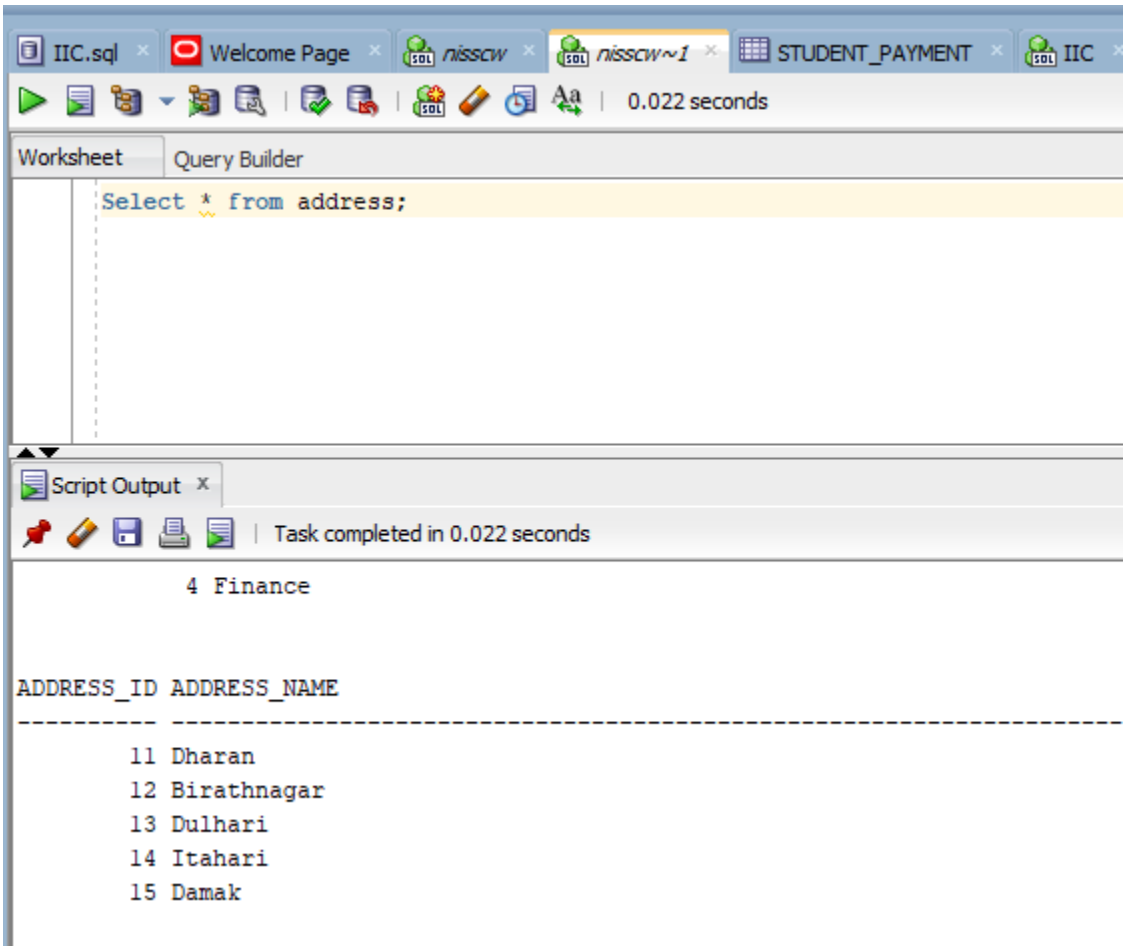


Figure 35: Showing the Values of Address Table

6.35. Showing the values of payment table

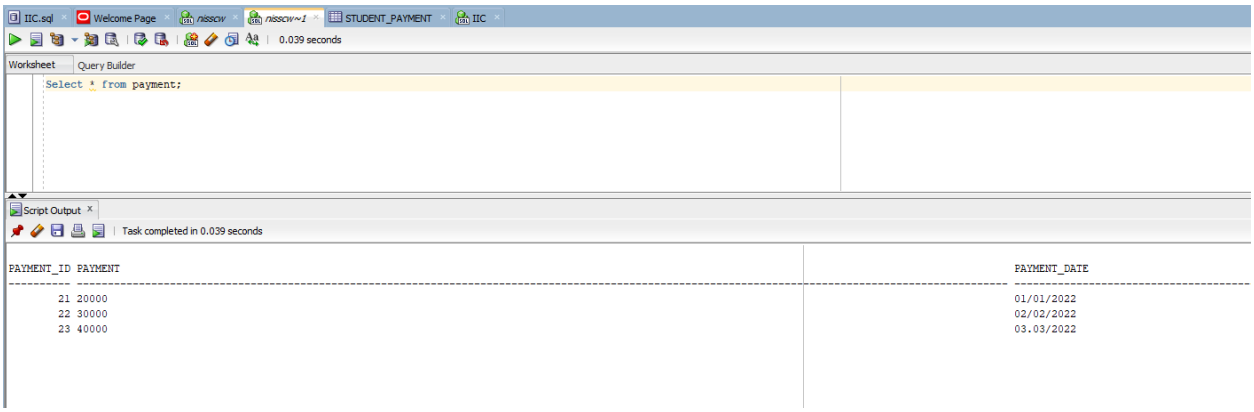


Figure 36: Showing the Values of Payment Table

6.36. Showing the values of module table

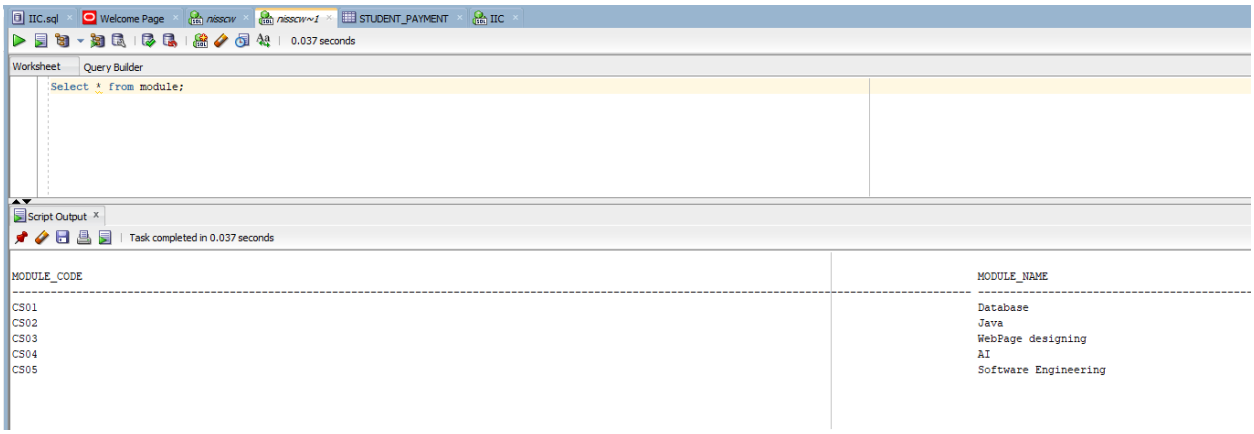


Figure 37: Showing the Values of Modul Table

6.37. Showing the values of student_details table

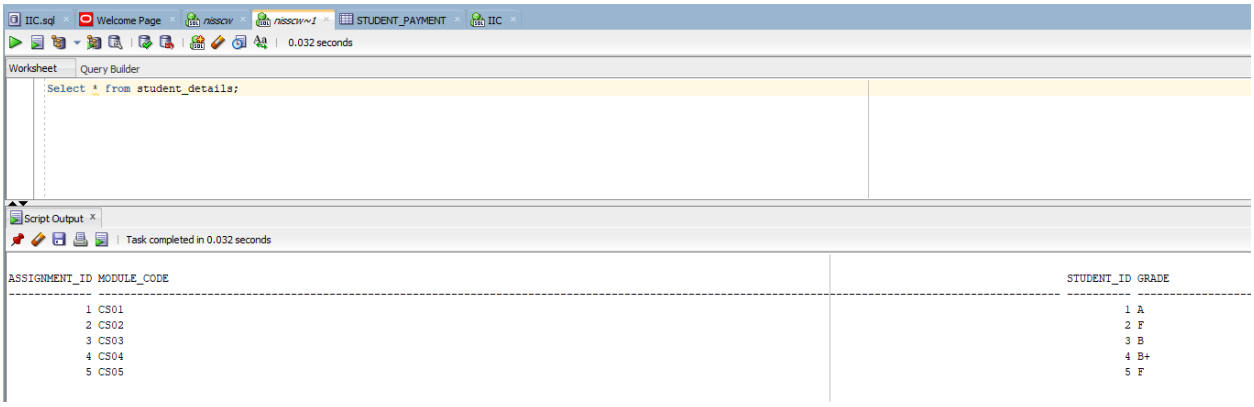


Figure 38: Showing the Values of Student Table

6.38. Showing the values of module_student table

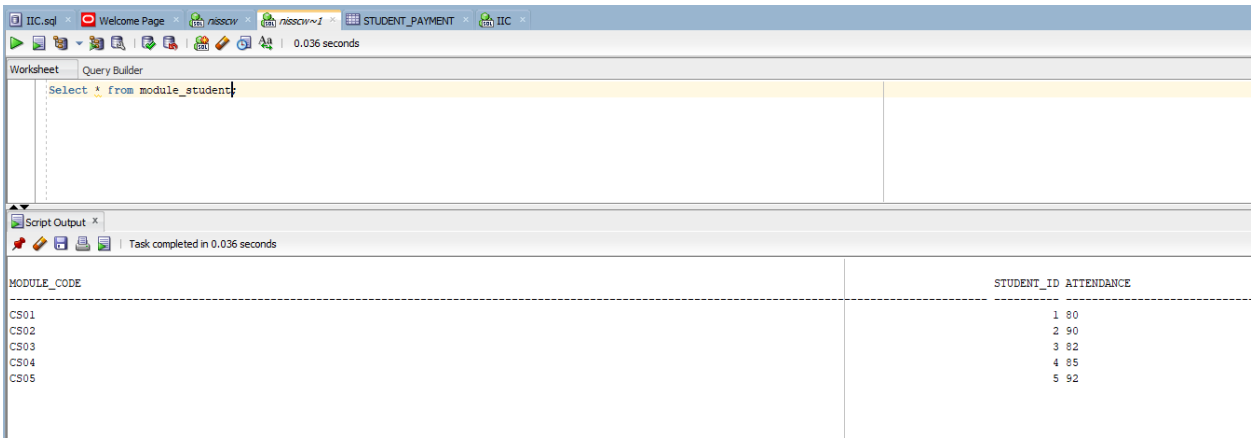


Figure 39: Showing the Values of Module_Student Table

6.39. Showing the values of teacher table

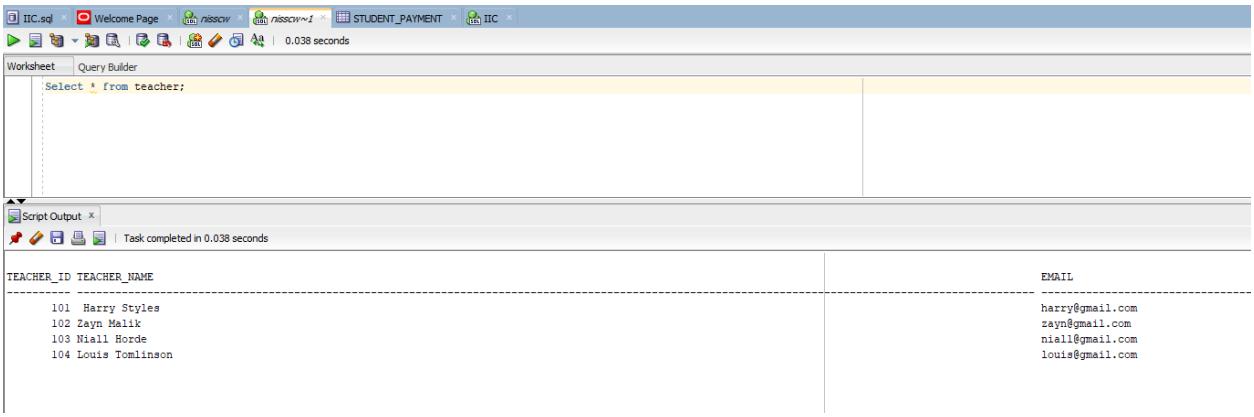


Figure 40: Showing the Values of Teacher Table

6.40. Showing the values of module_teacher table

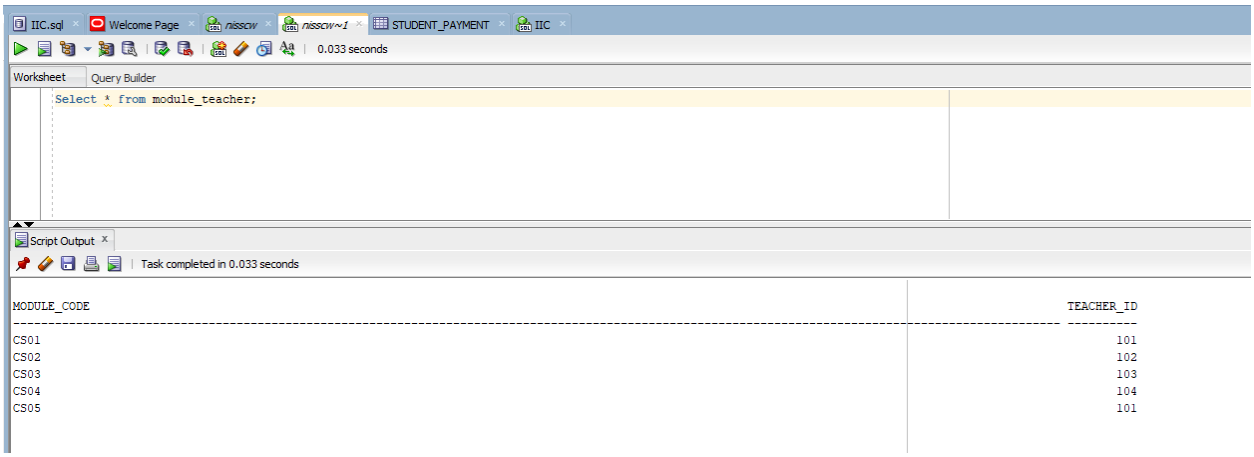


Figure 41: Showing the Values of Module_Teacher

6.41. Showing the values of teacher_address table

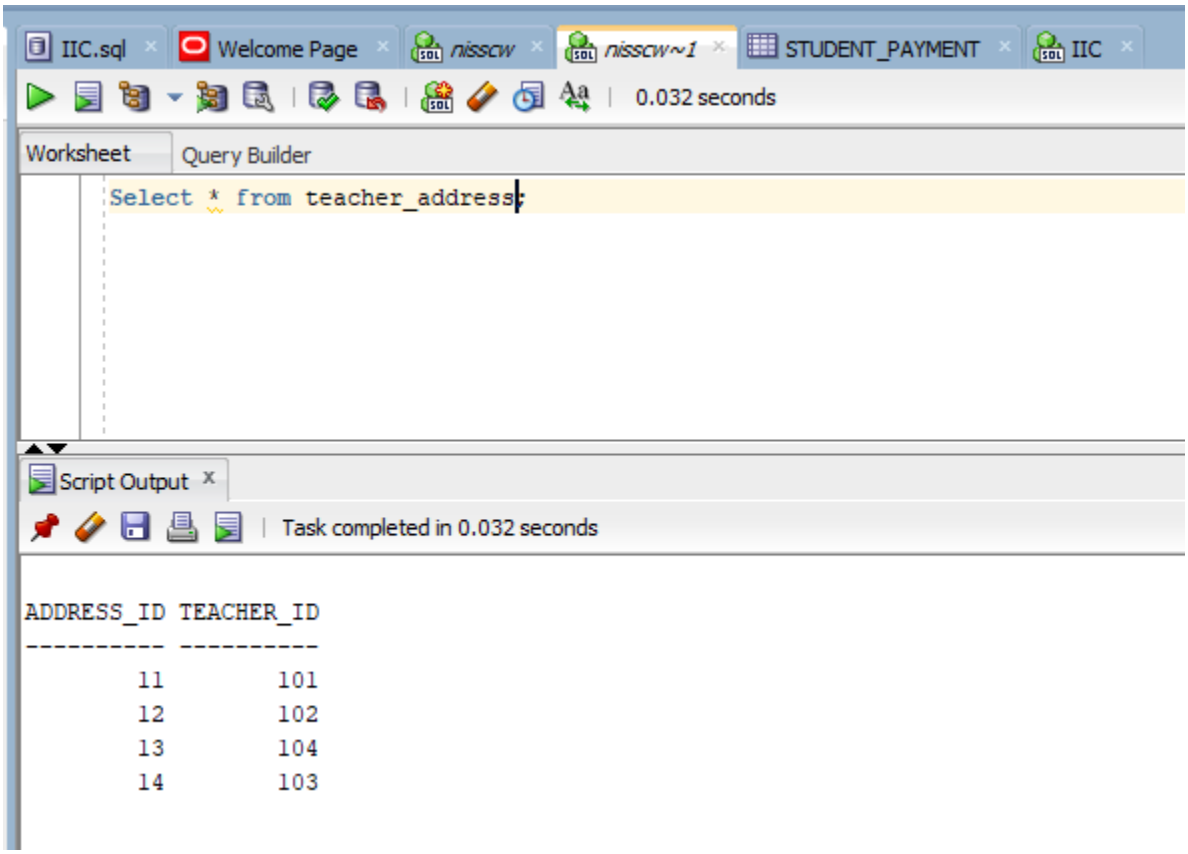


Figure 42: Showing the Values of Teacher Address

6.42. Showing the values of student_payment table

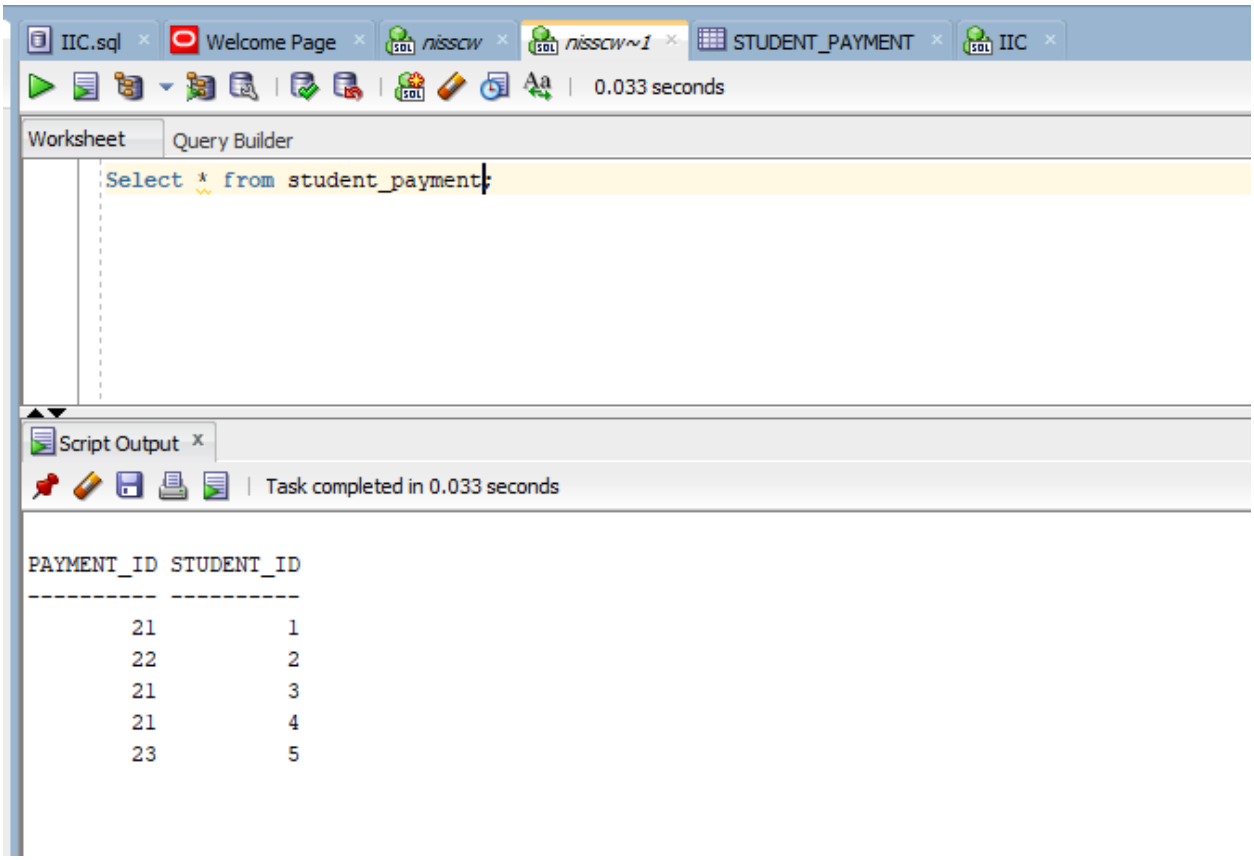


Figure 43: Showing the Values of Student_Payment

7. Implementation of Web-based Database Application

7.1. Connecting database

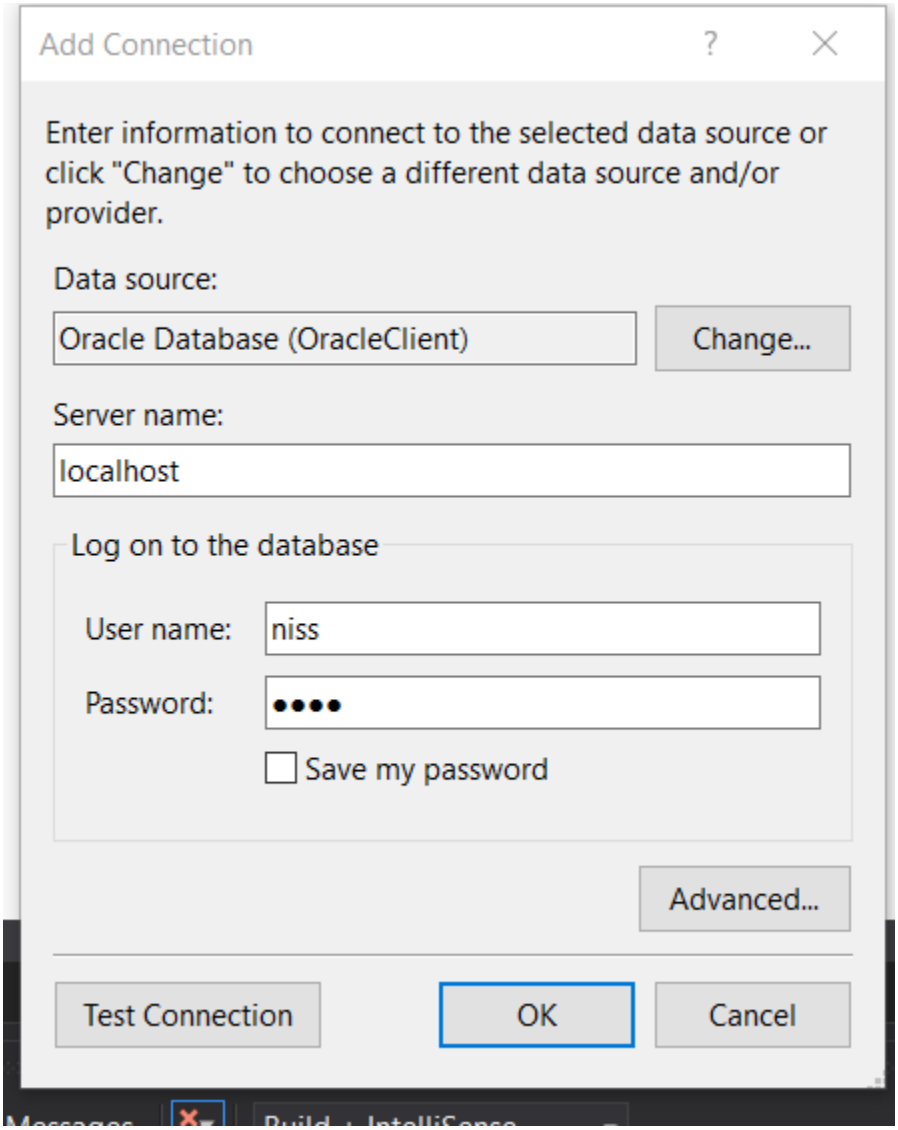


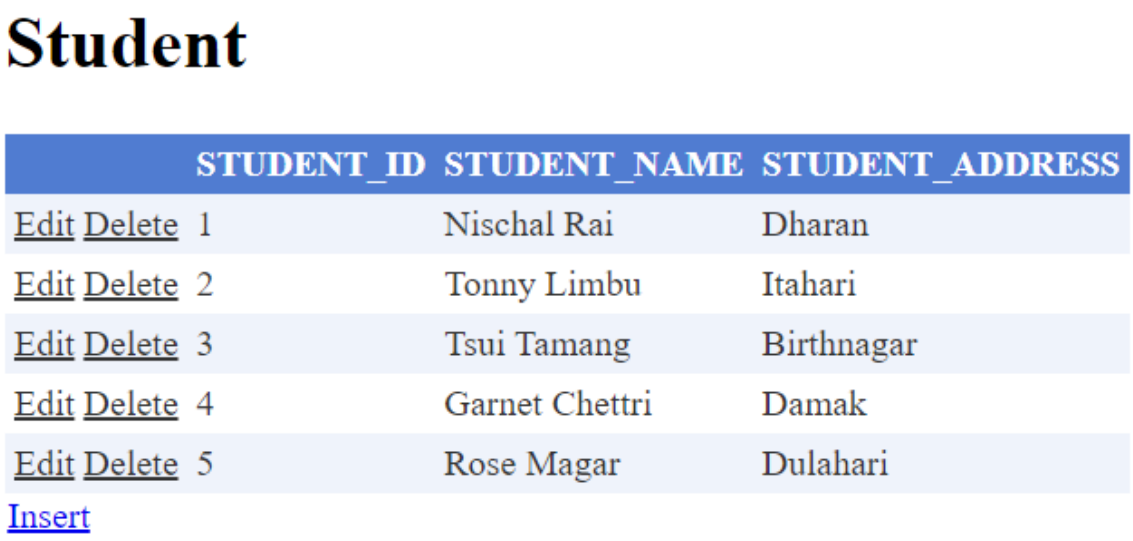
Figure 44: Connecting Database

7.2. Basic form

The basic forms consist of 5 web pages that shows Student Details, Department Details, Teacher Details, Address Details and Module Details. All these forms can perform CRUD operations. So, users of this web application can easily add, update, and delete the table details.

7.2.1. Student Details

This basic form consists of details of student table, and we can perform CRUD operation in this basic form. Users of this web application can easily add, update, and delete the table details.



Student

	STUDENT_ID	STUDENT_NAME	STUDENT_ADDRESS
Edit Delete	1	Nischal Rai	Dharan
Edit Delete	2	Tonny Limbu	Itahari
Edit Delete	3	Tsui Tamang	Birthnagar
Edit Delete	4	Garnet Chettri	Damak
Edit Delete	5	Rose Magar	Dulahari

Insert

Figure 45: Basic Form of Student Table

7.2.2. Department Details

This basic form consists of details of department table, and we can perform CRUD operation in this basic form. Users of this web application can easily add, update, and delete the table details.

Department

DEPARTMENT_ID	DEPARTMENT_NAME
1	Academics
2	SSD
3	RTE
4	Finance

Insert

Figure 46: Basic Form of Department Table

7.2.3. Teacher Details

This basic form consists of details of teacher table, and we can perform CRUD operation in this basic form. Users of this web application can easily add, update, and delete the table details.

Teacher

TEACHER_ID	TEACHER_NAME	EMAIL	DEPARTMENT_ID
101	Harry Styles	harry@gmail.com	1
102	Zayn Malik	zayn@gmail.com	2
103	Niall Horde	niall@gmail.com	3
104	Louis Tomlinson	louis@gmail.com	4

Insert

Figure 47: Basic Form of Teacher Table

7.2.4. Address Details

This basic form consists of details of address table, and we can perform CRUD operation in this basic form. Users of this web application can easily add, update, and delete the table details.

Address

ADDRESS_ID	ADDRESS_NAME
11	Dharan
12	Birathnagar
13	Dulhari
14	Itahari
15	Damak

Insert

Figure 48: Basic Form of Address Table

7.2.5. Module Details

This basic form consists of details of module table, and we can perform CRUD operation in this basic form. Users of this web application can easily add, update, and delete the table details.

Module

MODULE_CODE	MODULE_NAME	CREDIT_HOUR
CS01	Database	60
CS02	Java	65
CS03	WebPage designing	40
CS04	AI	60
CS05	Software Engineering	90

Insert

Figure 49: Basic Form of Module Table

7.3. Complex Form

The complex webforms consist of joined tables with multiple values. The complex forms were developed to display Teacher – Module Mapping details, Student Fee Payment details and Student-Assignment Details.

7.3.1. Teacher – Module Mapping form

This complex form gives the detail about teacher name, email of the teacher, module code, module name and credit hours based of the teacher id provided to it.

Teacher - Module Mapping Form

101

TEACHER_NAME	EMAIL	MODULE_CODE	MODULE_NAME	CREDIT_HOUR
Harry Styles	harry@gmail.com	CS01	Database	60
Harry Styles	harry@gmail.com	CS05	Software Engineering	90

Figure 50: Complex Form of Teacher - Module Mapping

Query:

SQL statement:

```
select teacher_name, email, module_code, module_name, credit_hour from module_teacher join module
using(module_code) join teacher using(teacher_id) where teacher_id =:teacher_id
```

Query Builder...

Figure 51: Query for Teacher - Module Mapping

7.3.2. Student Fee Payment Form

This complex form gives the detail about student name, student address, payment and payment date based on the student id provided to it.

Student Fee Payment Form

1

STUDENT_NAME	STUDENT_ADDRESS	PAYMENT	PAYMENT_DATE
Nischal Rai	Dharan	20000	01/01/2022

Figure 52: Complex Form of Student Fee Payment Form

<u>Query:</u>

SQL statement:

```
select student_name, student_address, payment, payment_date from student_payment join student using
(student_id) join payment using(payment_id) where student_id =:student_id
```

Query Builder...

Figure 53: Query for Student Fee Payment

7.3.3. Student – Assignment Form

This complex form gives the detail about student name, module name, assignment type, grade and status based on the student id provided to it.

Student-Assignment Form

1

STUDENT_NAME	MODULE_NAME	ASSIGNMENT_TYPE	GRADE	STATUS
Nischal Rai	Database	Individual Coursework	A	pass

Figure 54: Complex Form of Student – Assignment Form

Query:

SQL statement:

```
select student_name, module_name, assignment_type, grade, status from student_details join student using
(student_id) join module using(module_code) join assignment using(assignment_id) join grade using
(grade) where student_id =:student_id
```

Query Builder...

Figure 55:Query for Student-Assignment

8. Testing

8.1. Inserting into Student Table

Test Case	1
Action	The new button was clicked then the new student's information was entered and entered was pressed.
Expect Result	The table will display the newly added record of the student.
Actual Result	The table displayed the newly added record of the student.
Conclusion	Successful

Table 15: Test Case 1

Before:

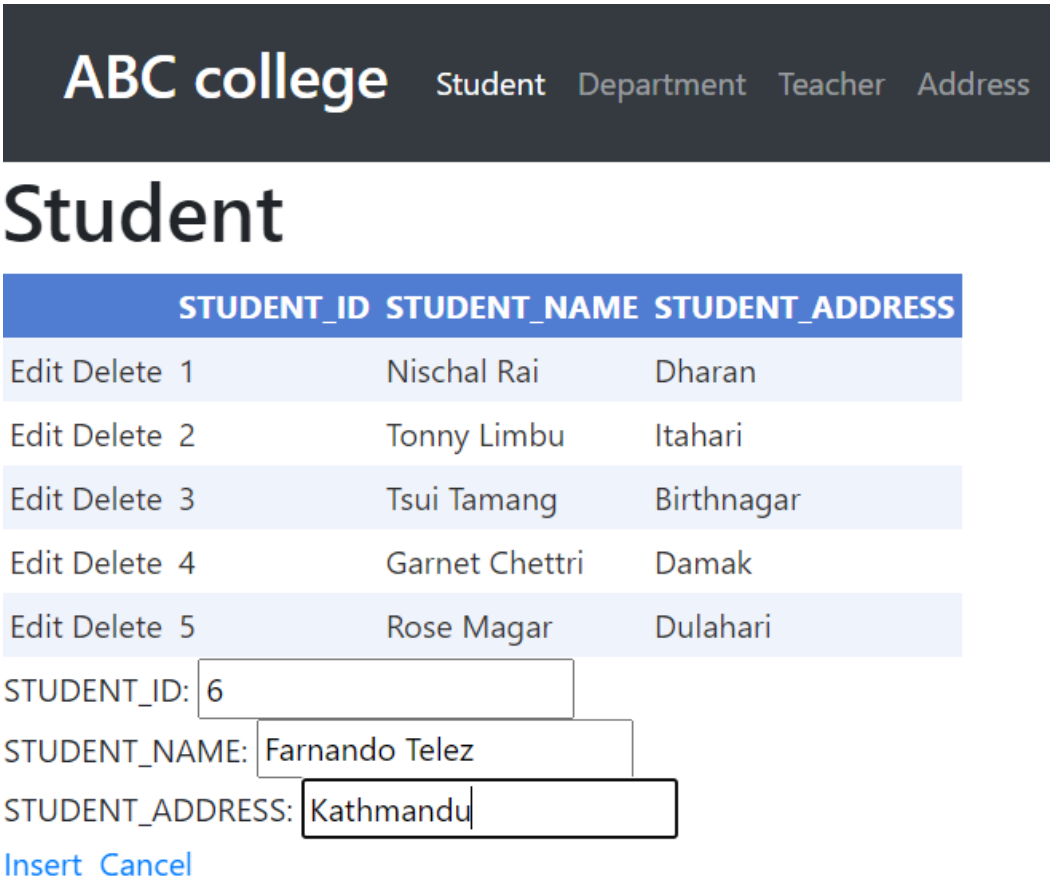


Figure 56: Before Inserting New Student

After:

ABC college Student Department Teacher Addr

Student

	STUDENT_ID	STUDENT_NAME	STUDENT_ADDRESS
Edit Delete	1	Nischal Rai	Dharan
Edit Delete	2	Tonny Limbu	Itahari
Edit Delete	3	Tsui Tamang	Birthnagar
Edit Delete	4	Garnet Chettri	Damak
Edit Delete	5	Rose Magar	Dulahari
Edit Delete	6	Farnando Telez	Kathmandu

Insert

Figure 57: After Inserting New Student

8.2. Editing in Student Table

Test Case	2
Action	The edit button was clicked and a value in the grid view was edited.
Expect Result	The edited value would be displayed in the table.
Actual Result	The edited value was displayed in the table.
Conclusion	Successful

Table 16: Test Case 2

Before:

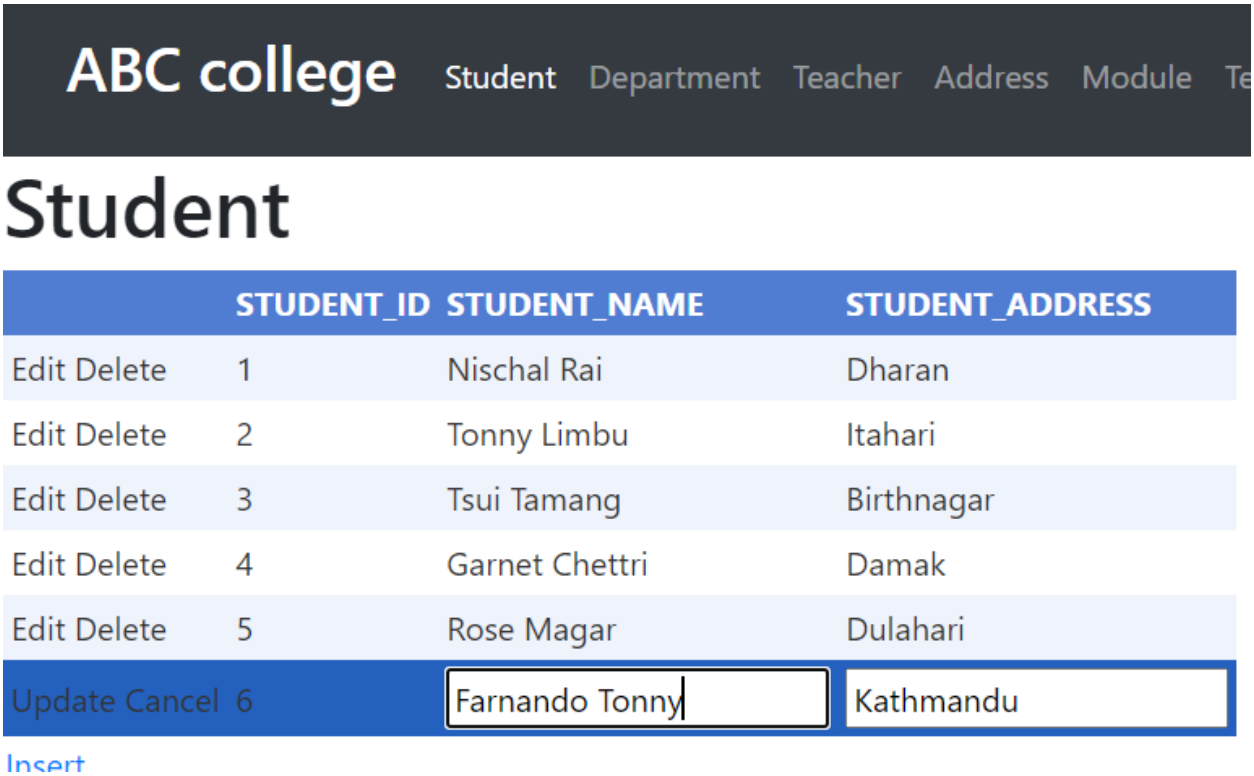


Figure 58: Before Editing value to Student

After:

ABC college Student Department Teacher Addr

Student

	STUDENT_ID	STUDENT_NAME	STUDENT_ADDRESS
Edit Delete	1	Nischal Rai	Dharan
Edit Delete	2	Tonny Limbu	Itahari
Edit Delete	3	Tsui Tamang	Birthnagar
Edit Delete	4	Garnet Chettri	Damak
Edit Delete	5	Rose Magar	Dulahari
Edit Delete	6	Farnando Tonny	Kathmandu

Insert

Figure 59: After Editing Values to Student

8.3. Deleting from Student Table

Test Case	3
Action	The delete button was clicked for a row in the grid view.
Expect Result	The value would be deleted from the table.
Actual Result	The value was be deleted from the table.
Conclusion	Successful

Table 17: Test Case 3

Before:

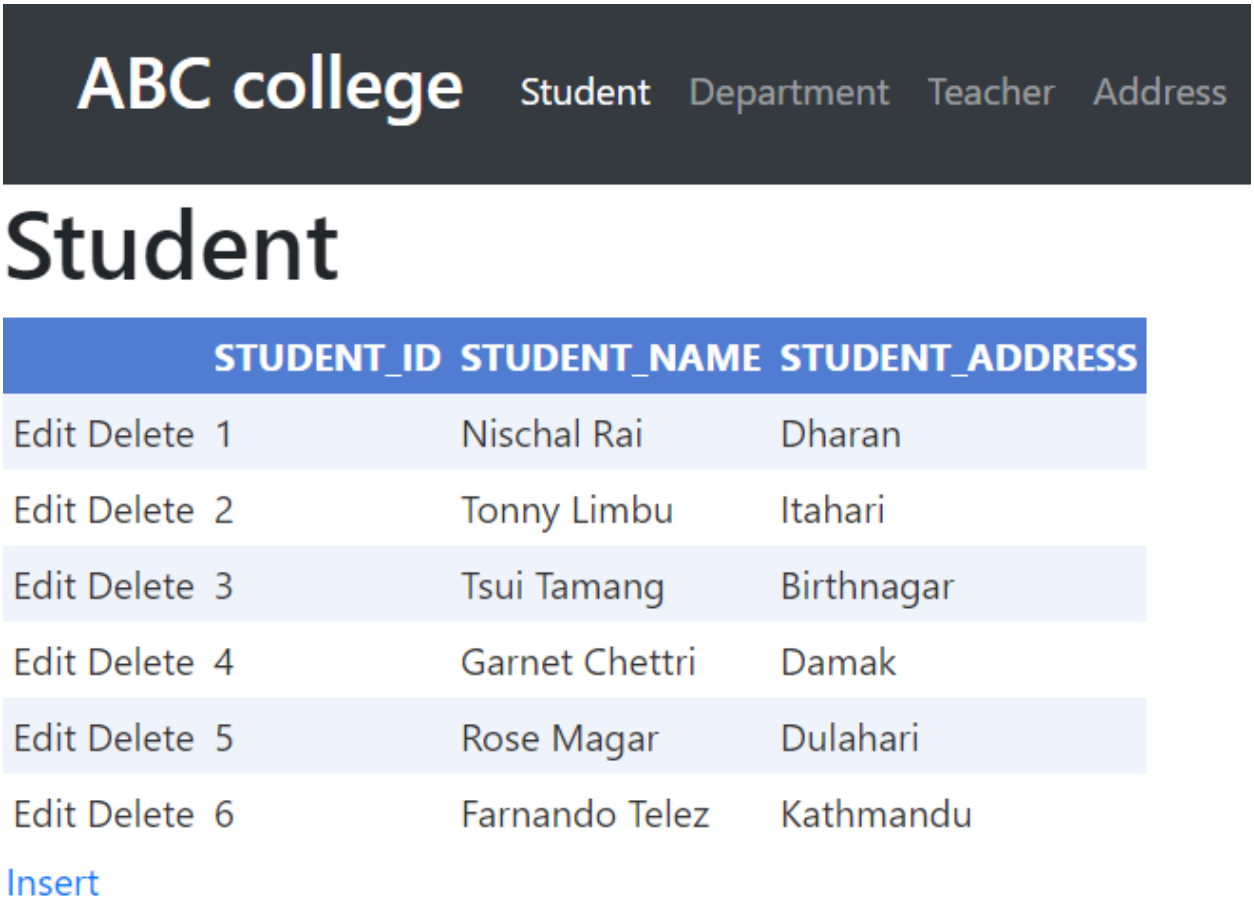


Figure 60: Before Deleting from Student

After:

ABC college Student Department Teacher Addres

Student

	STUDENT_ID	STUDENT_NAME	STUDENT_ADDRESS
Edit Delete	1	Nischal Rai	Dharan
Edit Delete	2	Tonny Limbu	Itahari
Edit Delete	3	Tsui Tamang	Birthnagar
Edit Delete	4	Garnet Chettri	Damak
Edit Delete	5	Rose Magar	Dulahari

Insert

Figure 61: After Deleting from Student

8.4. Inserting into Department Table

Test Case	4
Action	The new button was clicked then the new Department's information was entered and entered was pressed.
Expect Result	The table will display the newly added record of the Department.
Actual Result	The table displayed the newly added record of the Department.
Conclusion	Successful

Table 18: Test 4

Before:

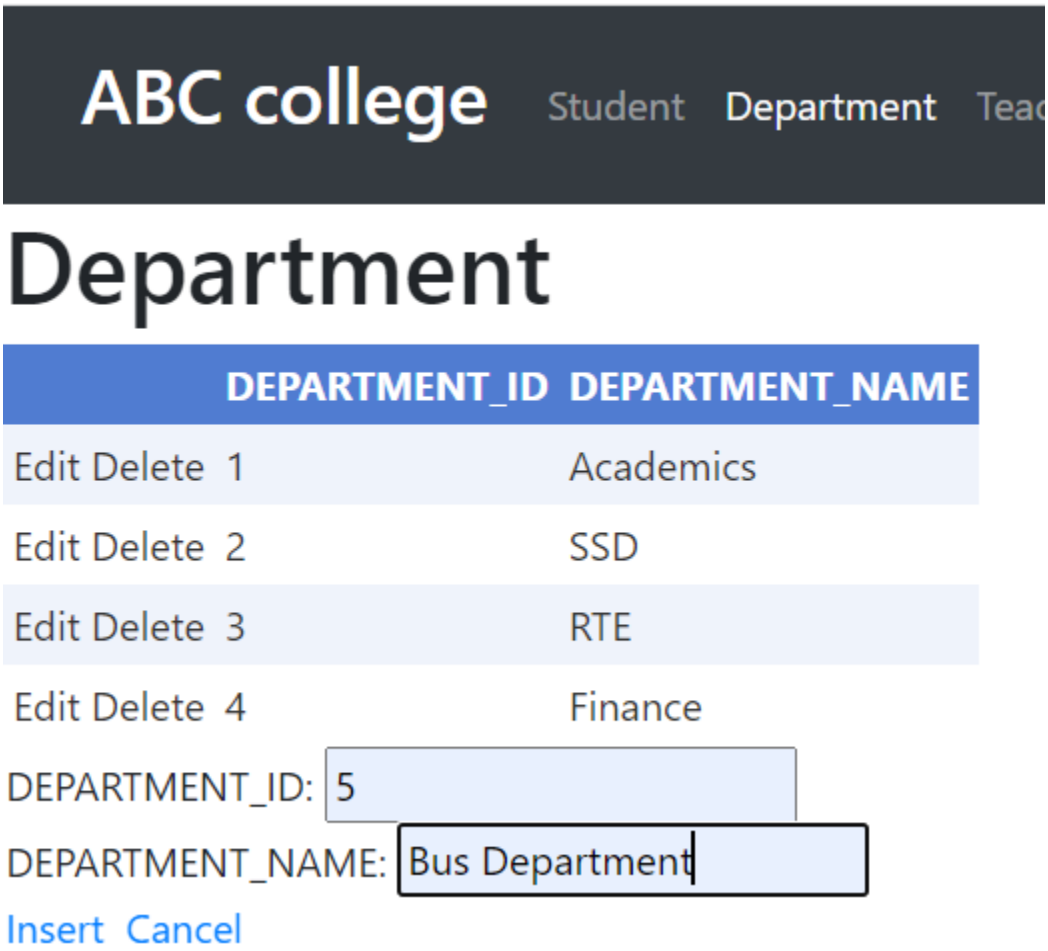


Figure 62: Before Inerting new Department

After:

ABC college Student Department Tea

Department

	DEPARTMENT_ID	DEPARTMENT_NAME
Edit Delete	1	Academics
Edit Delete	2	SSD
Edit Delete	3	RTE
Edit Delete	4	Finance
Edit Delete	5	Bus Department

Insert

Figure 63: After Inserting new Department

8.5. Editing value from Department

Test Case	5
Action	The edit button was clicked and a value in the grid view was edited.
Expect Result	The edited value would be displayed in the table.
Actual Result	The edited value was displayed in the table.
Conclusion	Successful

Table 19: Test Case 5

Before:

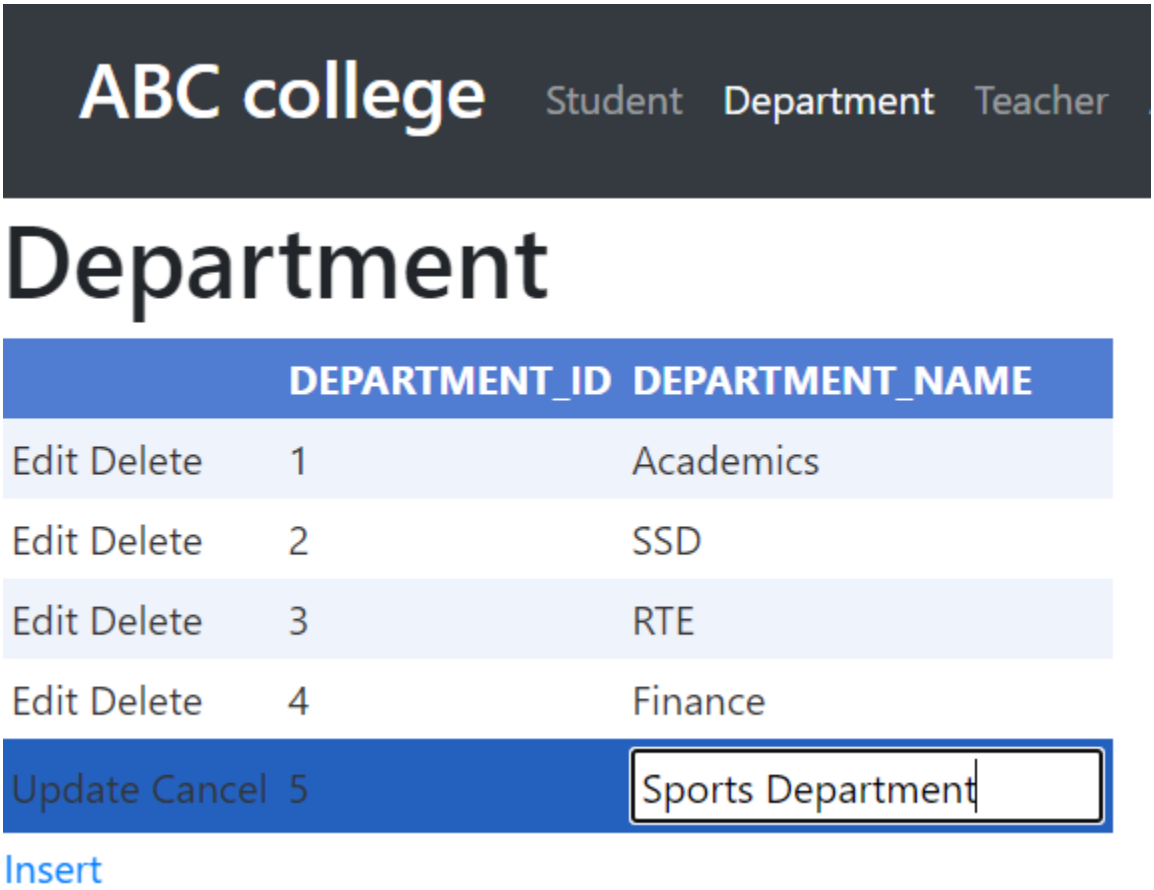


Figure 64: Before Editing the value of department

After:

ABC college Student Department

Department

	DEPARTMENT_ID	DEPARTMENT_NAME
Edit Delete	1	Academics
Edit Delete	2	SSD
Edit Delete	3	RTE
Edit Delete	4	Finance
Edit Delete	5	Sports Department

Insert

Figure 65: After Editing the Value of Department

8.6. Deleting Value form Department

Test Case	3
Action	The delete button was clicked for a row in the grid view.
Expect Result	The value would be deleted from the table.
Actual Result	The value was be deleted from the table.
Conclusion	Successful

Table 20: Test Case 6

Before:

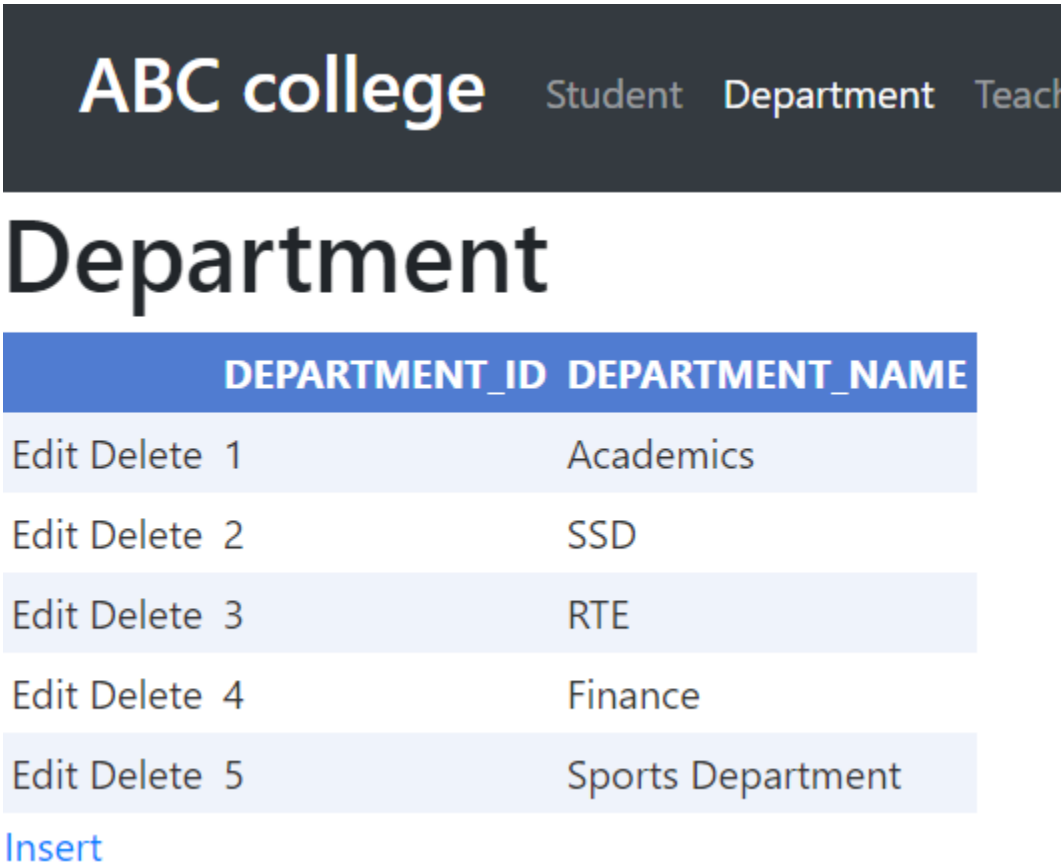


Figure 66: Before Deleting the Value from Department

After:

ABC college Student Department Teac

Department

	DEPARTMENT_ID	DEPARTMENT_NAME
Edit Delete	1	Academics
Edit Delete	2	SSD
Edit Delete	3	RTE
Edit Delete	4	Finance

Insert

Figure 67: After Deleting the Value from the Department

8.7. Inserting into Teacher

Test Case	7
Action	The new button was clicked then the new Teacher's information was entered and entered was pressed.
Expect Result	The table will display the newly added record of the teacher.
Actual Result	The table displayed the newly added record of the teacher.
Conclusion	Successful

Table 21: Test Case 7

Before:

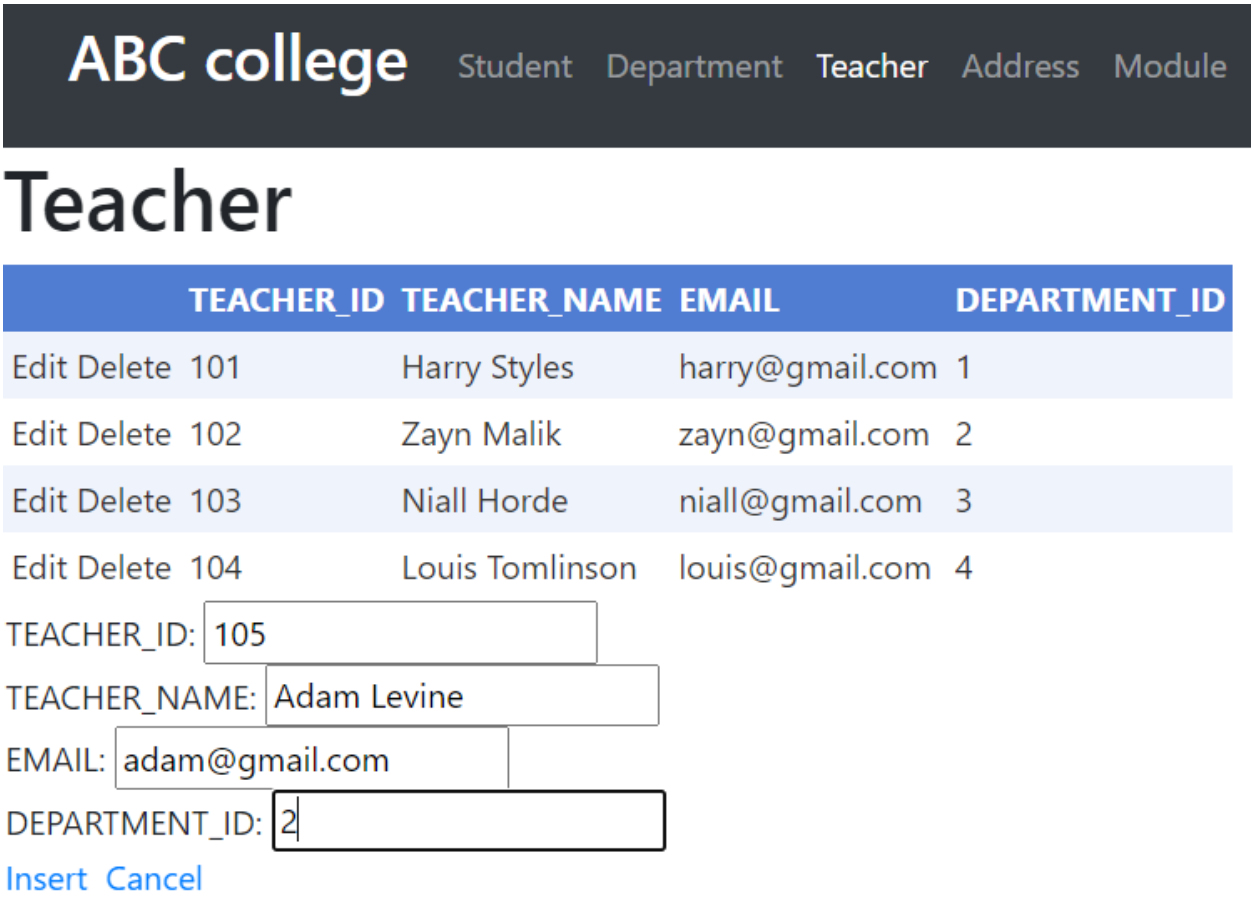


Figure 68: Before Inserting new Teacher

After:

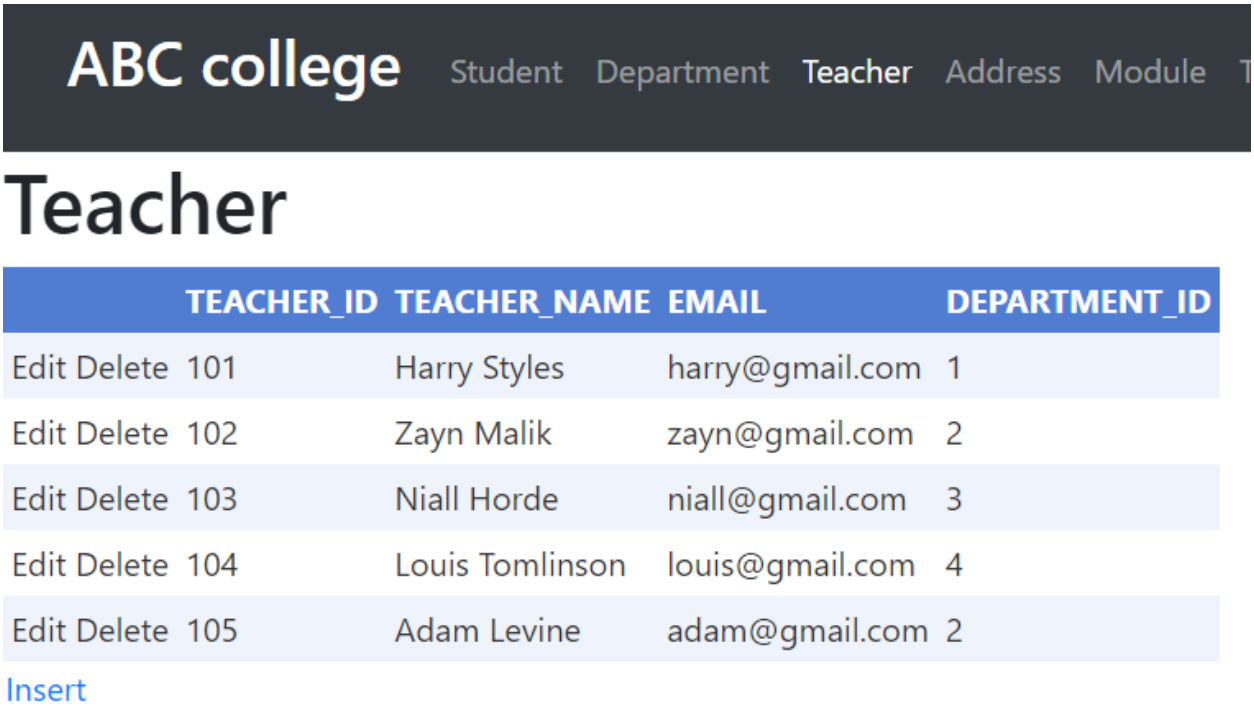


Figure 69: After Inserting new Teacher

8.8. Editing Values of Teacher

Test Case	8
Action	The edit button was clicked and a value in the grid view was edited.
Expect Result	The edited value would be displayed in the table.
Actual Result	The edited value was displayed in the table.
Conclusion	Successful

Table 22: Test Case 8

Before:

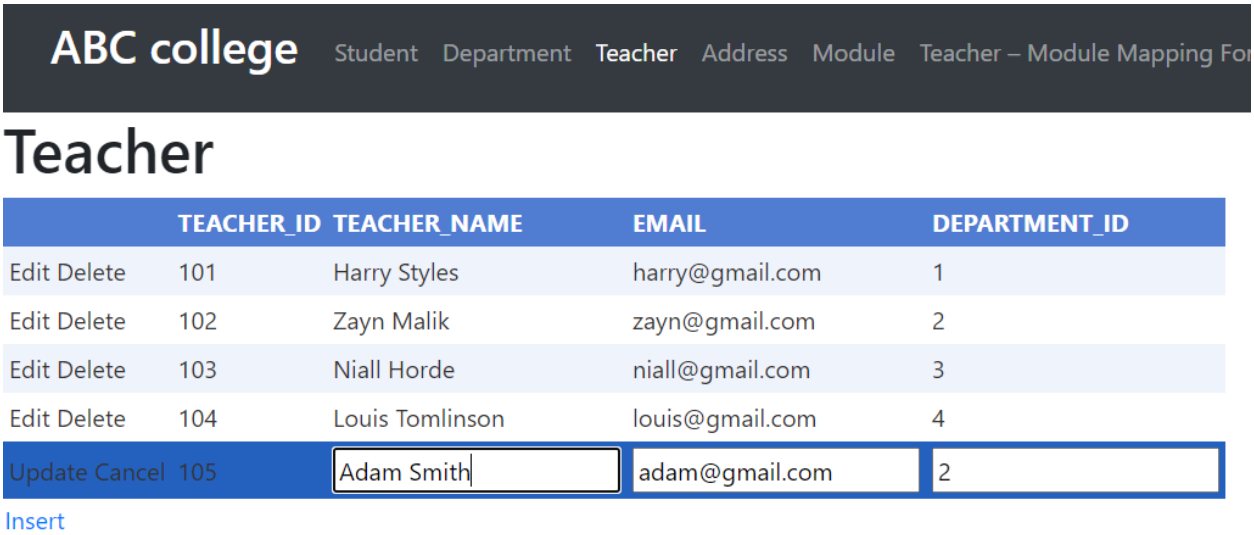


Figure 70: Before Editing in Teacher

After:

ABC college Student Department Teacher Address Module

Teacher

	TEACHER_ID	TEACHER_NAME	EMAIL	DEPARTMENT_ID
Edit Delete	101	Harry Styles	harry@gmail.com	1
Edit Delete	102	Zayn Malik	zayn@gmail.com	2
Edit Delete	103	Niall Horde	niall@gmail.com	3
Edit Delete	104	Louis Tomlinson	louis@gmail.com	4
Edit Delete	105	Adam Smith	adam@gmail.com	2

Insert

Figure 71: After Editing in Teacher

8.9. Deleting Values from Teacher

Test Case	9
Action	The delete button was clicked for a row in the grid view.
Expect Result	The value would be deleted from the table.
Actual Result	The value was be deleted from the table.
Conclusion	Successful

Table 23: Test Case 9

Before:

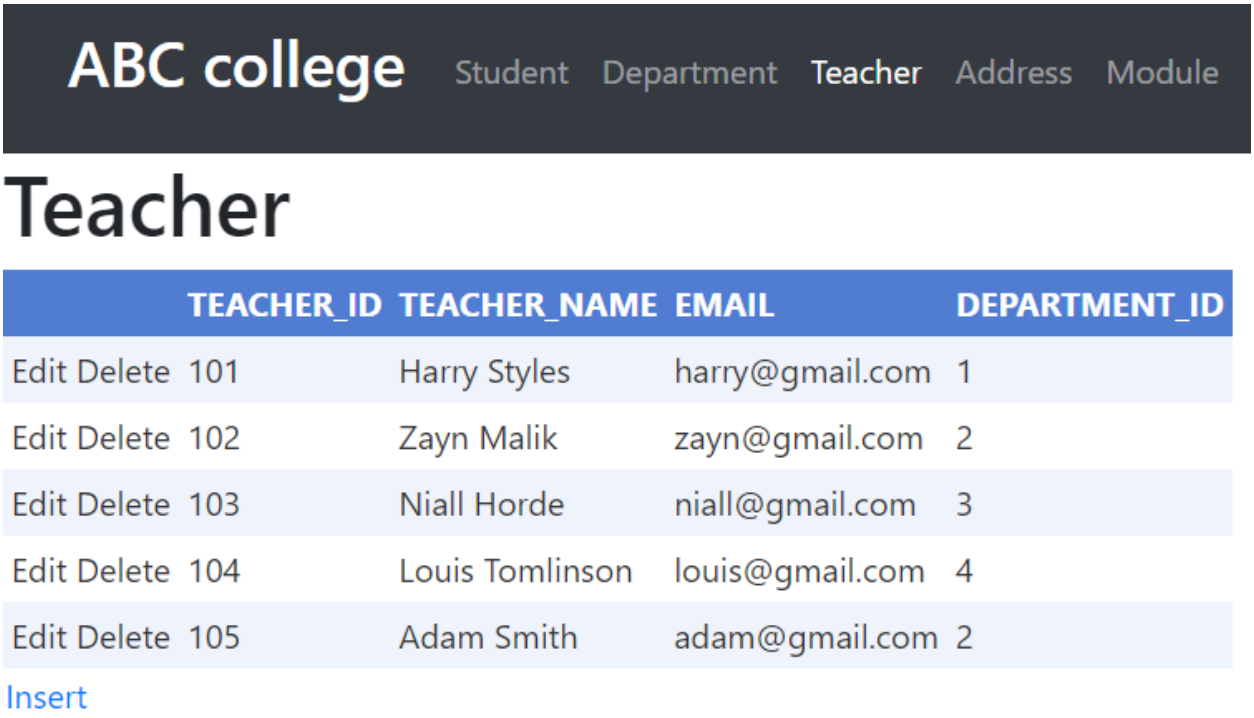


Figure 72: Before Deleting Value of Teacher

After:

ABC college Student Department Teacher Address Module

Teacher

	TEACHER_ID	TEACHER_NAME	EMAIL	DEPARTMENT_ID
Edit Delete	101	Harry Styles	harry@gmail.com	1
Edit Delete	102	Zayn Malik	zayn@gmail.com	2
Edit Delete	103	Niall Horde	niall@gmail.com	3
Edit Delete	104	Louis Tomlinson	louis@gmail.com	4

Insert

Figure 73: After Deleting the Value in Teacher

8.10. Inserting Values to Address

Test Case	10
Action	The new button was clicked then the new Address's information was entered and entered was pressed.
Expect Result	The table will display the newly added record of the address.
Actual Result	The table displayed the newly added record of the address.
Conclusion	Successful

Table 24: Test Case 10

Before:

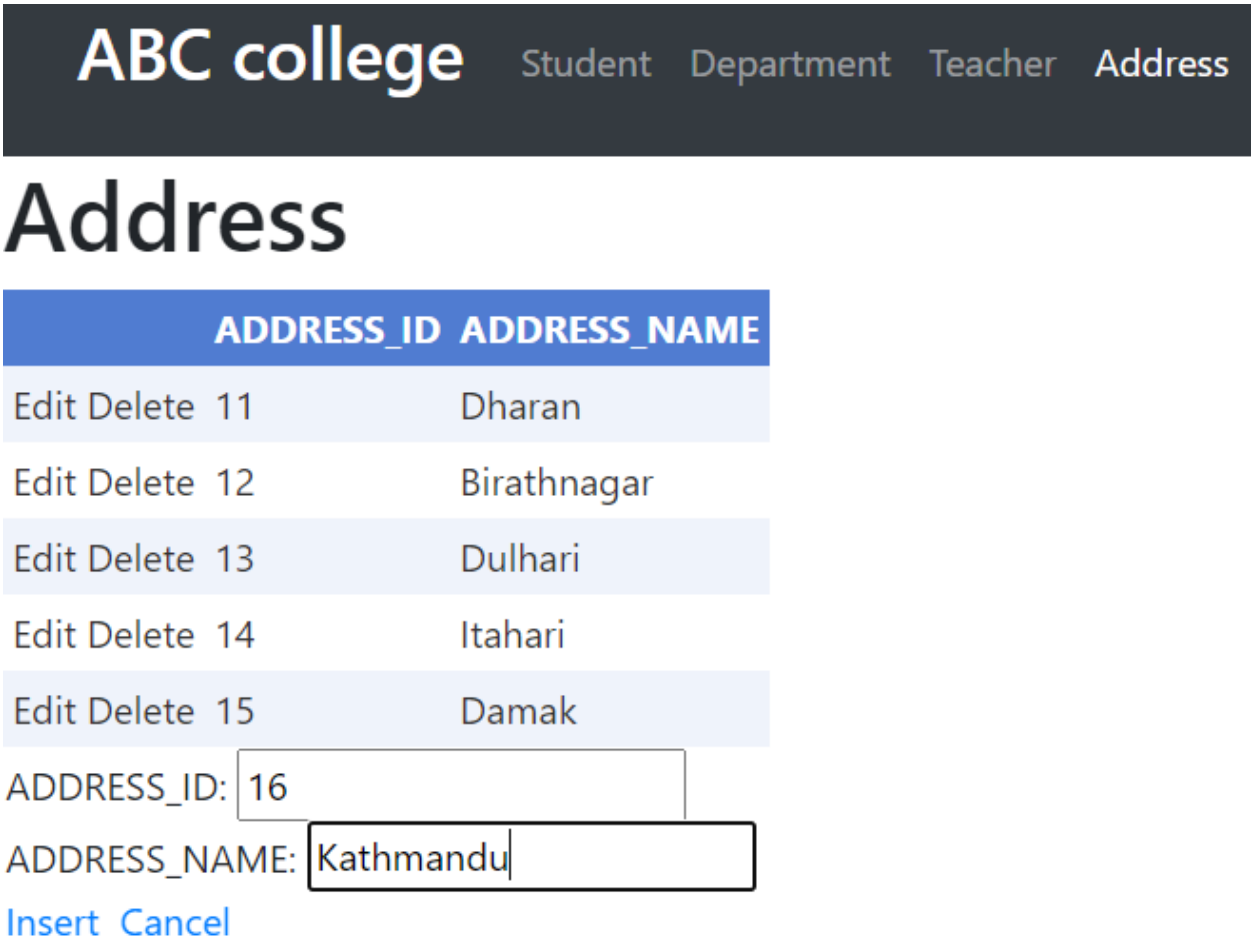


Figure 74: Before Inserting new Address

After:

ABC college Student Department Teacher Address

Address

	ADDRESS_ID	ADDRESS_NAME
Edit Delete	11	Dharan
Edit Delete	12	Birathnagar
Edit Delete	13	Dulhari
Edit Delete	14	Itahari
Edit Delete	15	Damak
Edit Delete	16	Kathmandu

Insert

Figure 75: After Inserting New Address

8.11. Editing Values of Address

Test Case	11
Action	The edit button was clicked and a value in the grid view was edited.
Expect Result	The edited value would be displayed in the table.
Actual Result	The edited value was displayed in the table.
Conclusion	Successful

Table 25: Test Case 11

Before:

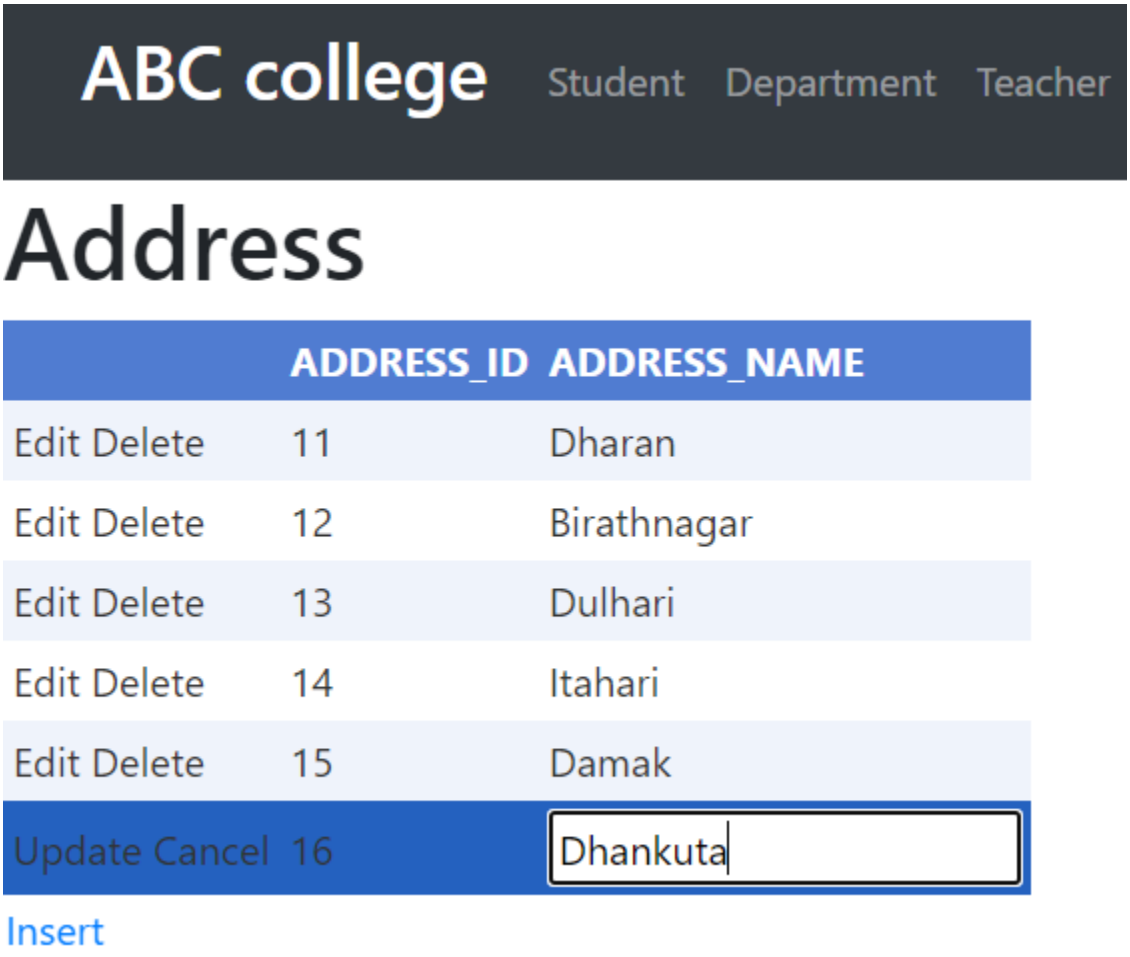


Figure 76: Before Editing the Value of Address

After:

ABC college Student Department Teacher Address

Address

	ADDRESS_ID	ADDRESS_NAME
Edit Delete	11	Dharan
Edit Delete	12	Birathnagar
Edit Delete	13	Dulhari
Edit Delete	14	Itahari
Edit Delete	15	Damak
Edit Delete	16	Dhankuta

Insert

Figure 77: After Editing the Address

8.12. Deleting Values from Address

Test Case	12
Action	The delete button was clicked for a row in the grid view.
Expect Result	The value would be deleted from the table.
Actual Result	The value was be deleted from the table.
Conclusion	Successful

Table 26: Test Case 12

Before:

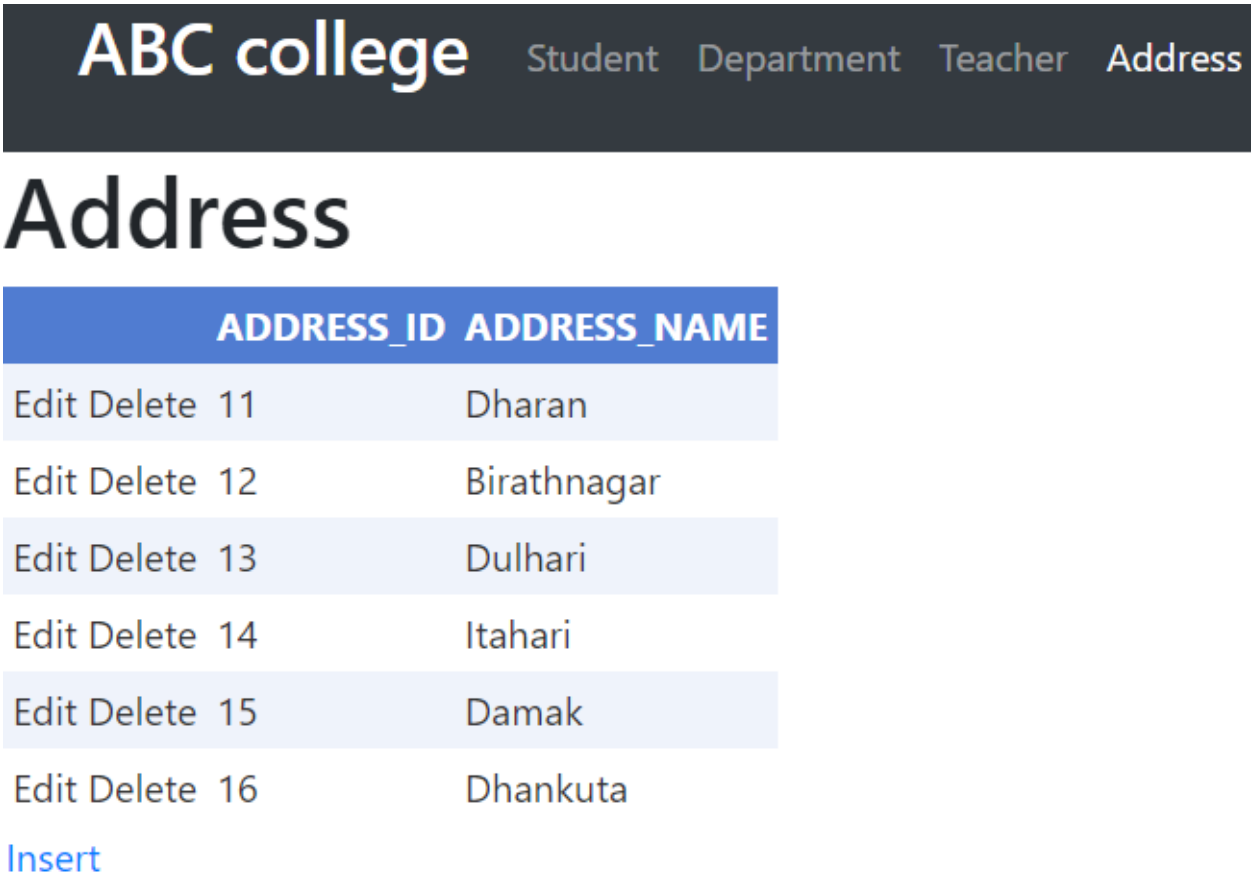


Figure 78: Before Deleting the Value from Address

After:

ABC college Student Department Teacher Address

Address

	ADDRESS_ID	ADDRESS_NAME
Edit Delete	11	Dharan
Edit Delete	12	Birathnagar
Edit Delete	13	Dulhari
Edit Delete	14	Itahari
Edit Delete	15	Damak

Insert

Figure 79: After Deleting the Value from Address

8.13. Inserting Values to Module

Test Case	13
Action	The new button was clicked then the new Module's information was entered and entered was pressed.
Expect Result	The table will display the newly added record of the module.
Actual Result	The table displayed the newly added record of the module.
Conclusion	Successful

Table 27: Test Case 13

Before:

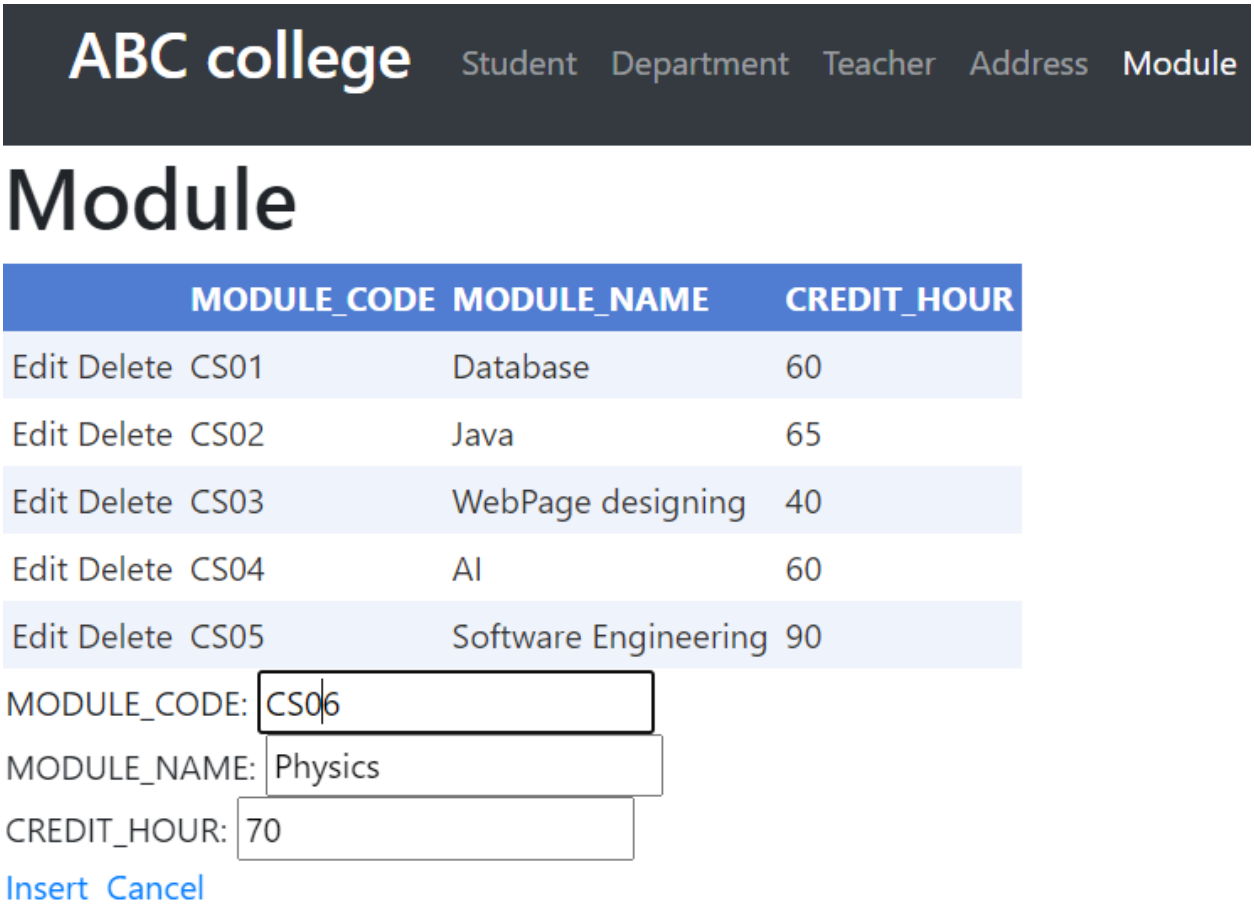


Figure 80: Before Inserting new Module

After:

ABC college Student Department Teacher Address Module

Module

	MODULE_CODE	MODULE_NAME	CREDIT_HOUR
Edit Delete	CS01	Database	60
Edit Delete	CS02	Java	65
Edit Delete	CS03	WebPage designing	40
Edit Delete	CS04	AI	60
Edit Delete	CS05	Software Engineering	90
Edit Delete	CS06	Physics	70

Insert

Figure 81: After Inserting new Module

8.14. Editing Values of Module

Test Case	14
Action	The edit button was clicked and a value in the grid view was edited.
Expect Result	The edited value would be displayed in the table.
Actual Result	The edited value was displayed in the table.
Conclusion	Successful

Table 28: Test Case 14

Before:

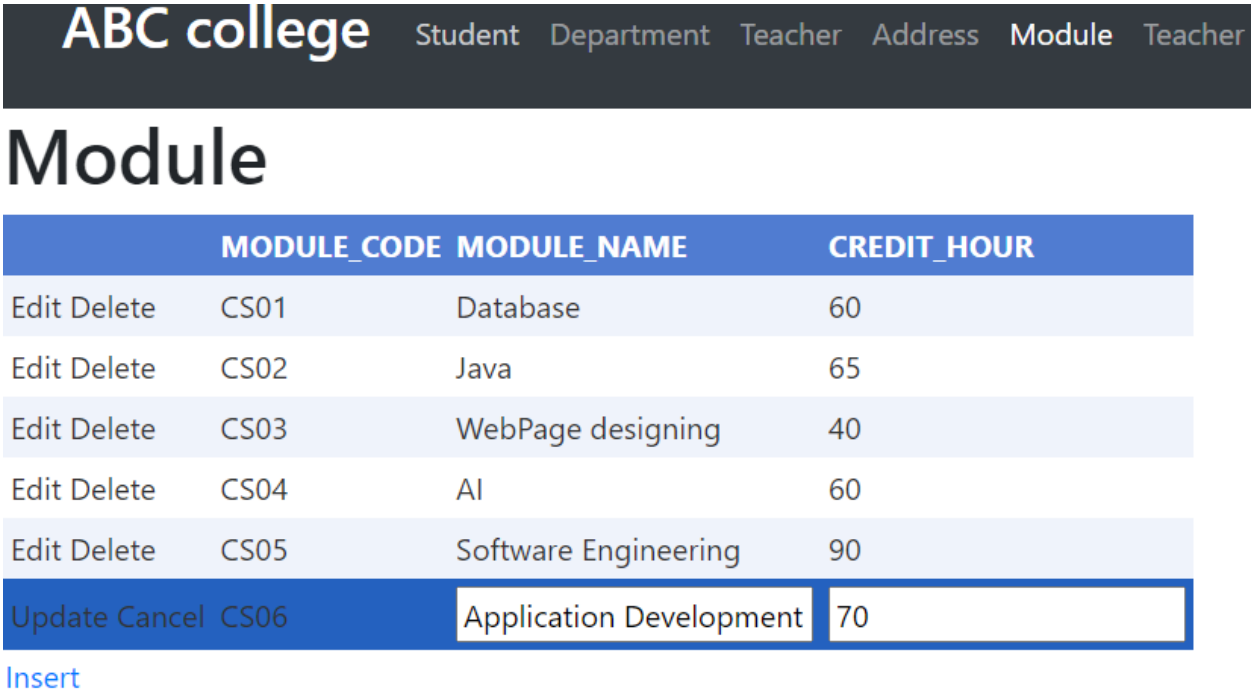


Figure 82: Before Editing values of Module

After:

ABC college Student Department Teacher Address Module

Module

	MODULE_CODE	MODULE_NAME	CREDIT_HOUR
Edit Delete	CS01	Database	60
Edit Delete	CS02	Java	65
Edit Delete	CS03	WebPage designing	40
Edit Delete	CS04	AI	60
Edit Delete	CS05	Software Engineering	90
Edit Delete	CS06	Application Development	70

Insert

Figure 83: After Editing the Value in Module

8.15. Deleting Values from Module

Test Case	15
Action	The delete button was clicked for a row in the grid view.
Expect Result	The value would be deleted from the table.
Actual Result	The value was be deleted from the table.
Conclusion	Successful

Table 29: Test Case 15

Before:

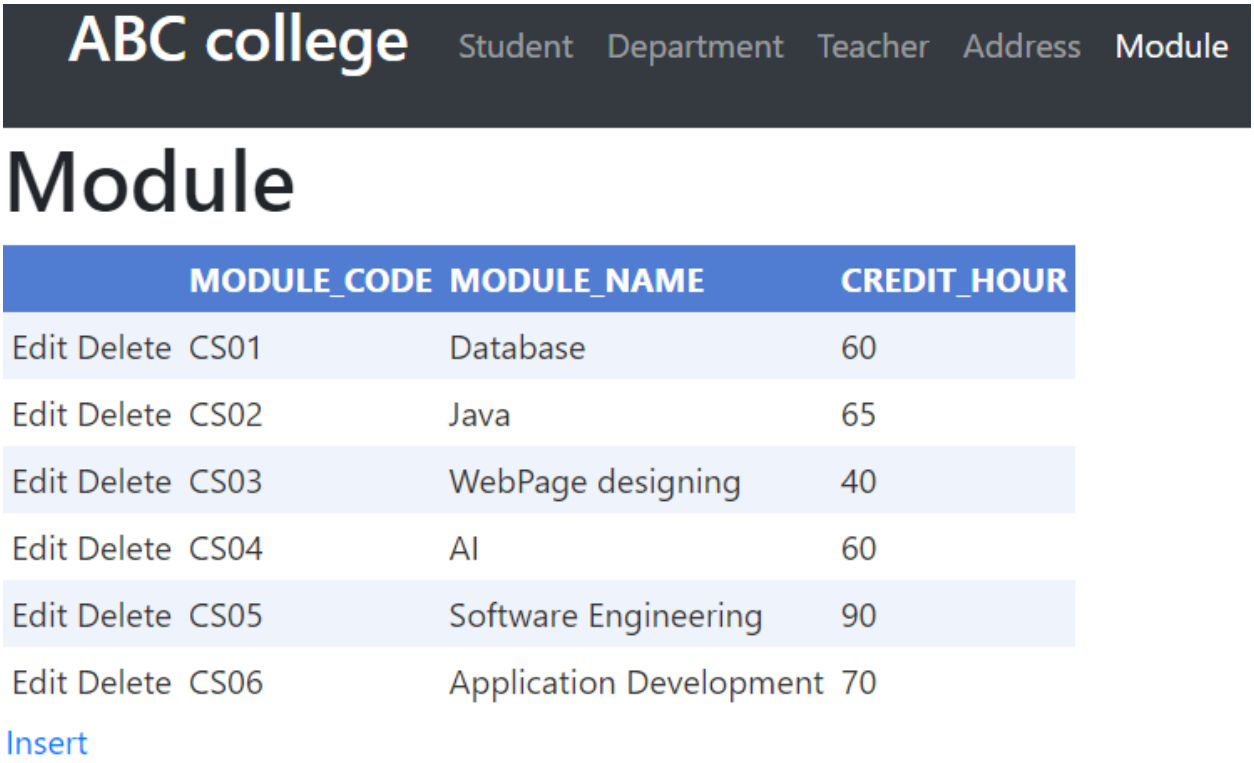


Figure 84: Before Deleting the Value from Module

After:

ABC college Student Department Teacher Address Module

Module

	MODULE_CODE	MODULE_NAME	CREDIT_HOUR
Edit Delete	CS01	Database	60
Edit Delete	CS02	Java	65
Edit Delete	CS03	WebPage designing	40
Edit Delete	CS04	AI	60
Edit Delete	CS05	Software Engineering	90

Insert

Figure 85: After Deleting the Value from Module

8.16. Teacher – Module Mapping Form Test

Test Case	16
Action	The drop-down value is changed.
Expect Result	The detail of new value would be displayed.
Actual Result	The detail of new value was be displayed.
Conclusion	Successful

Table 30: Test Case 16

Before:

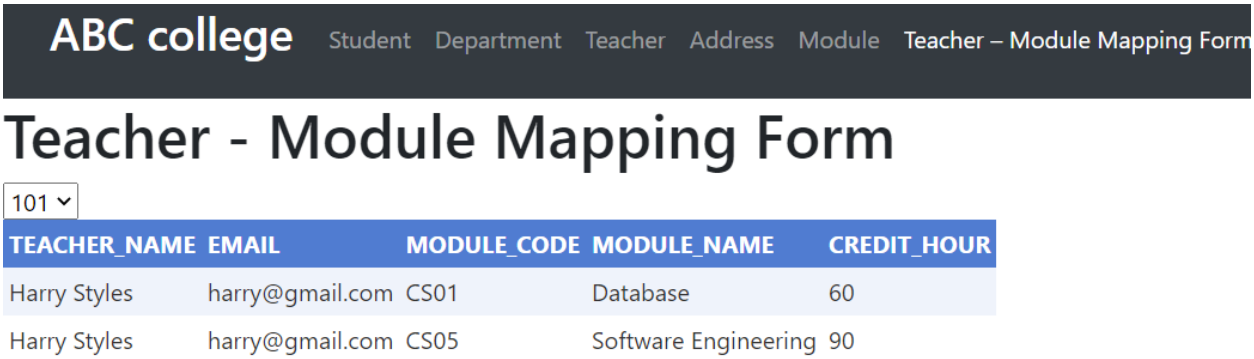


Figure 86: Before the drop-down value is changed in Teacher-Module Mapping

After:

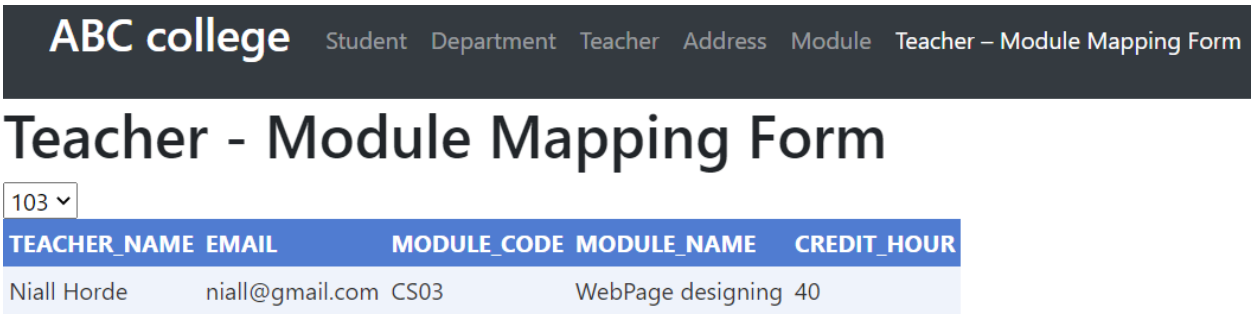


Figure 87: After the drop-down value is changed in Teacher-Module Mapping

8.17. Student Fee Payment Form Test

Test Case	17
Action	The drop-down value is changed.
Expect Result	The detail of new value would be displayed.
Actual Result	The detail of new value was be displayed.
Conclusion	Successful

Table 31: Test Case 17

Before:

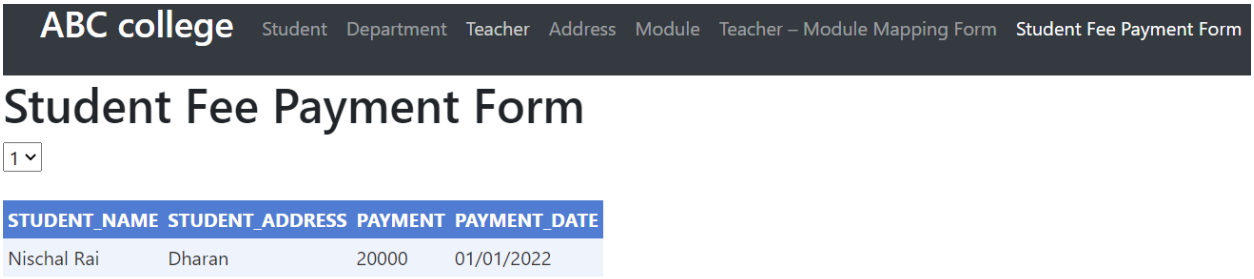


Table 32: Before the drop-down value is changed in Student Fee Payment Form

After:

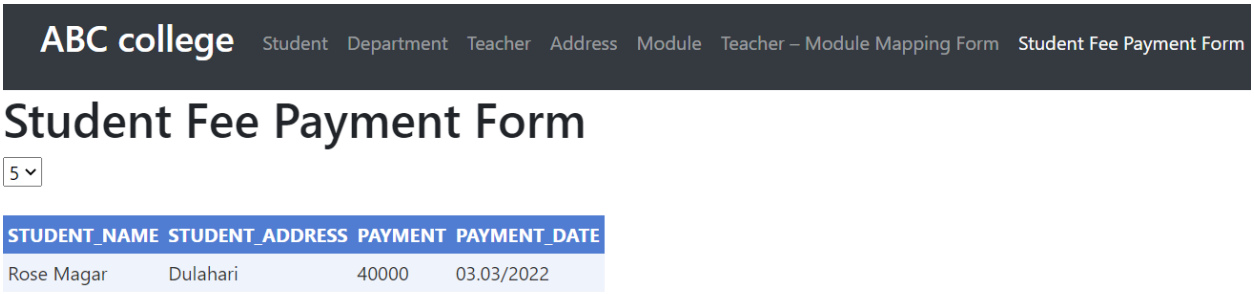


Table 33: After the drop-down value is changed in Student Fee Payment Form

8.18. Student – Assignment Form Test

Test Case	18
Action	The drop-down value is changed.
Expect Result	The detail of new value would be displayed.
Actual Result	The detail of new value was be displayed.
Conclusion	Successful

Table 34: Test Case 18

Before:

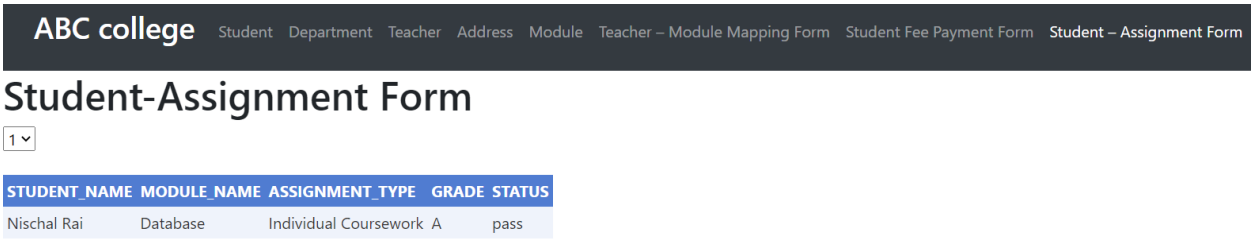


Table 35: Before the drop-down value is changed in Student – Assignment Form

After:

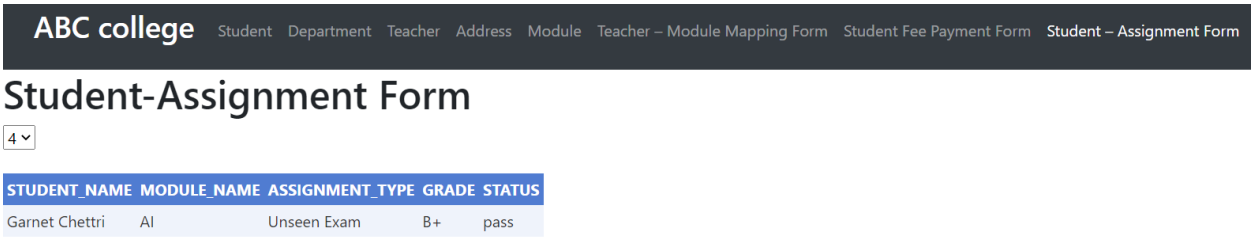


Table 36: After the drop-down value is changed in Student – Assignment Form

8.19. Error handling 1 (Using False foreign Key for Department_ID in teacher)

Test Case	19
Action	Inserting the Mistake Department_ID.
Expect Result	Table to take the value
Actual Result	An error message was appeared
Conclusion	Failed

Table 37: Test Case 19

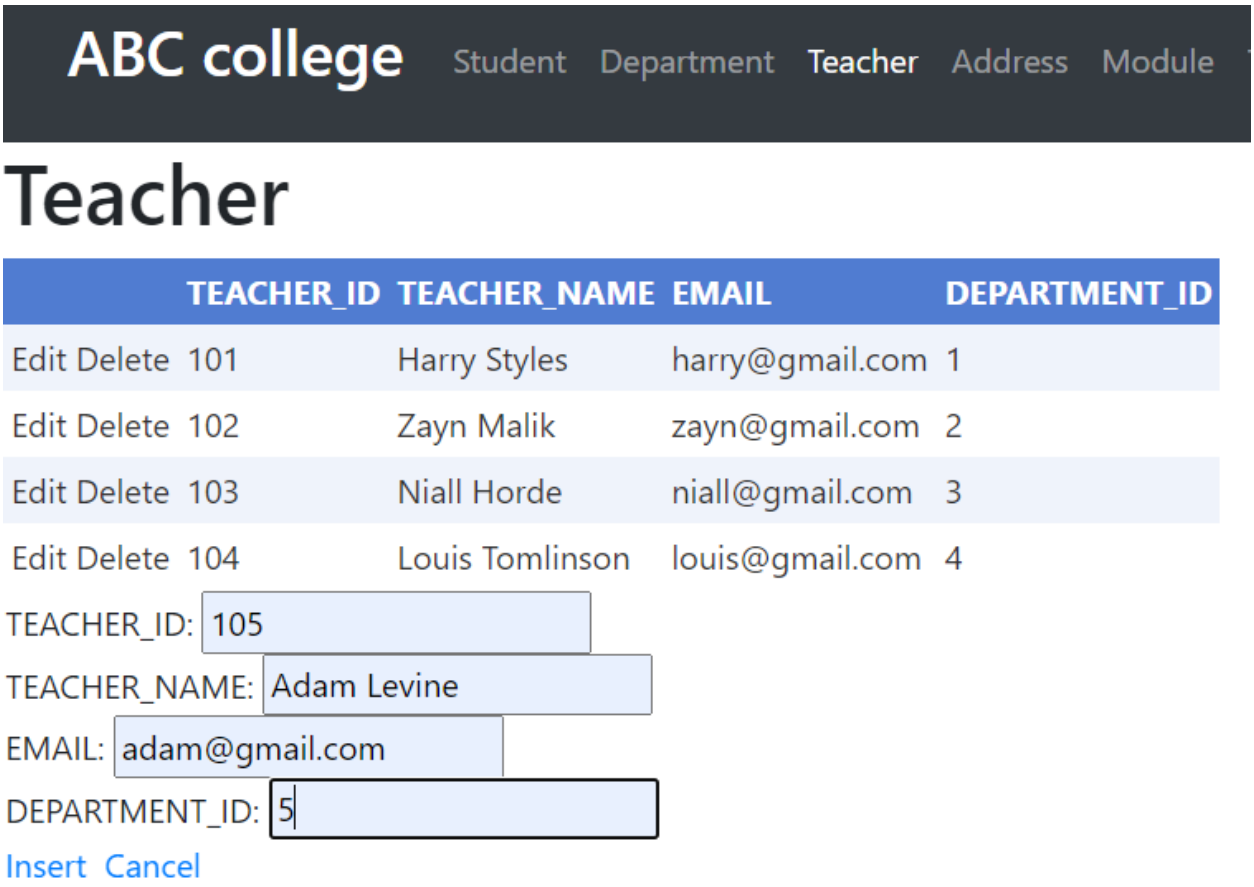


Figure 88: Part 1 of error

Server Error in '/' Application.

ORA-02291: integrity constraint (NISS.TEACHER_DEPARTMENT_FK) violated - parent key not found

Description: An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it originated in the code.

Exception Details: System.Data.OracleClient.OracleException: ORA-02291: integrity constraint (NISS.TEACHER_DEPARTMENT_FK) violated - parent key not found

Source Error:

An unhandled exception was generated during the execution of the current web request. Information regarding the origin and location of the exception can be identified using the exception stack trace below.

Stack Trace:

```
[OracleException (0x80131938): ORA-02291: integrity constraint (NISS.TEACHER_DEPARTMENT_FK) violated - parent key not found
]
   System.Data.OracleClient.OracleConnection.CheckError(OciErrorHandle errorHandle, Int32 rc) +337331
   System.Data.OracleClient.OracleCommand.Execute(OciStatementHandle statementHandle, CommandBehavior behavior, Boolean needRowid, OciRowidDescriptor& rowidDescriptor, ArrayList& resul
   System.Data.OracleClient.OracleCommand.ExecuteNonQueryInternal(Boolean needRowid, OciRowidDescriptor& rowidDescriptor) +547
   System.Data.OracleClient.OracleCommand.ExecuteNonQuery() +115
   System.Web.UI.WebControls.SqlDataSourceView.ExecuteDbCommand(DbCommand command, DataSourceOperation operation) +400
   System.Web.UI.WebControls.SqlDataSourceView.ExecuteInsert(IDictionary values) +419
   System.Web.UI.DataSourceView.Insert(IDictionary values, DataSourceViewOperationCallback callback) +100
   System.Web.UI.WebControls.FormView.HandleInsert(String commandArg, Boolean causesValidation) +379
   System.Web.UI.WebControls.FormView.HandleEvent(EventArgs e, Boolean causesValidation, String validationGroup) +647
   System.Web.UI.WebControls.FormView.OnBubbleEvent(Object source, EventArgs e) +92
   System.Web.UI.Control.RaiseBubbleEvent(Object source, EventArgs args) +37
   System.Web.UI.WebControls.FormViewRow.OnBubbleEvent(Object source, EventArgs e) +88
   System.Web.UI.Control.RaiseBubbleEvent(Object source, EventArgs args) +37
   System.Web.UI.WebControls.LinkButton.OnCommand(CommandEventArgs e) +127
   System.Web.UI.WebControls.LinkButton.RaisePostBackEvent(String eventArgument) +168
   System.Web.UI.WebControls.LinkButton.System.Web.UI.IPostBackEventHandler.RaisePostBackEvent(String eventArgument) +12
   System.Web.UI.Page.RaisePostBackEvent(IPostBackEventHandler sourceControl, String eventArgument) +15
   System.Web.UI.Page.RaisePostBackEvent(NameValueCollection postData) +9858668
   System.Web.UI.Page.ProcessRequestMain(Boolean includeStagesBeforeAsyncPoint, Boolean includeStagesAfterAsyncPoint) +1696
```

Figure 89: Part 2 of error (an error message)

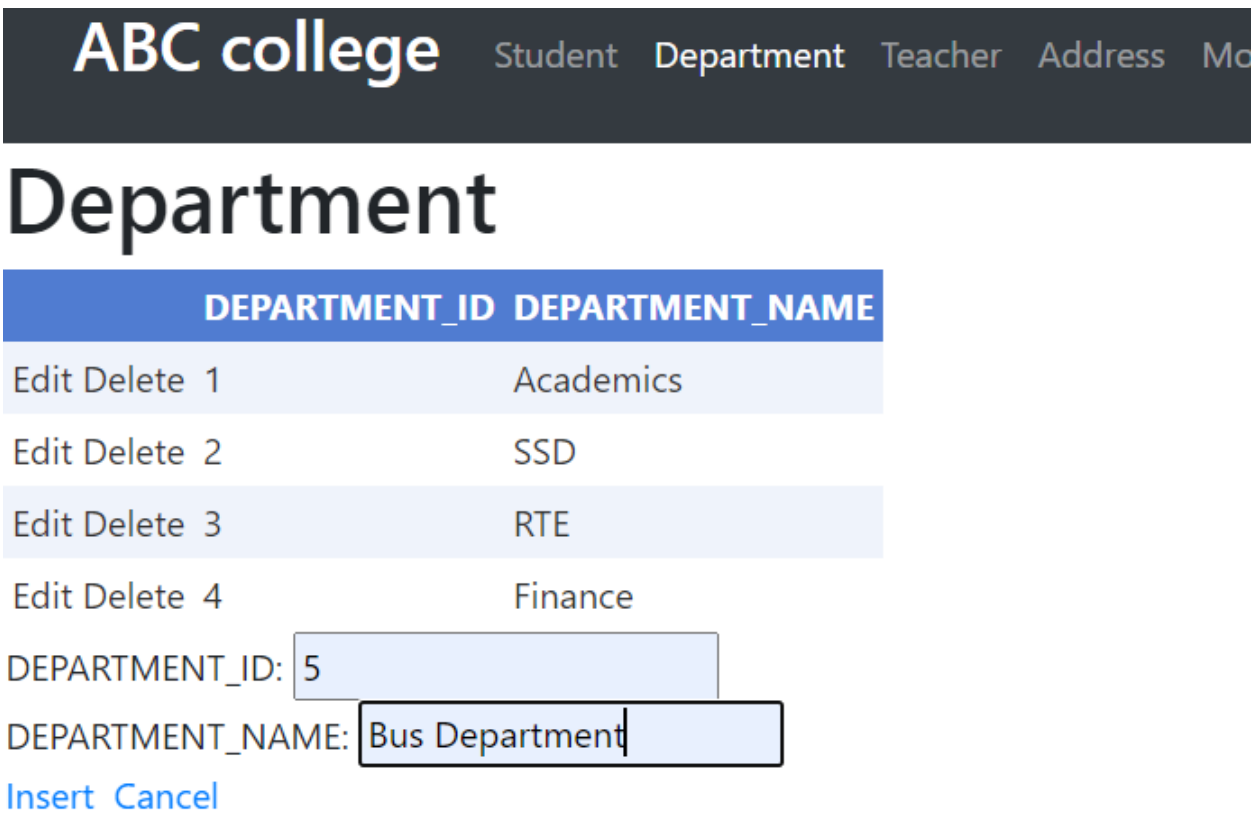


Figure 90: Fixing error Part 1

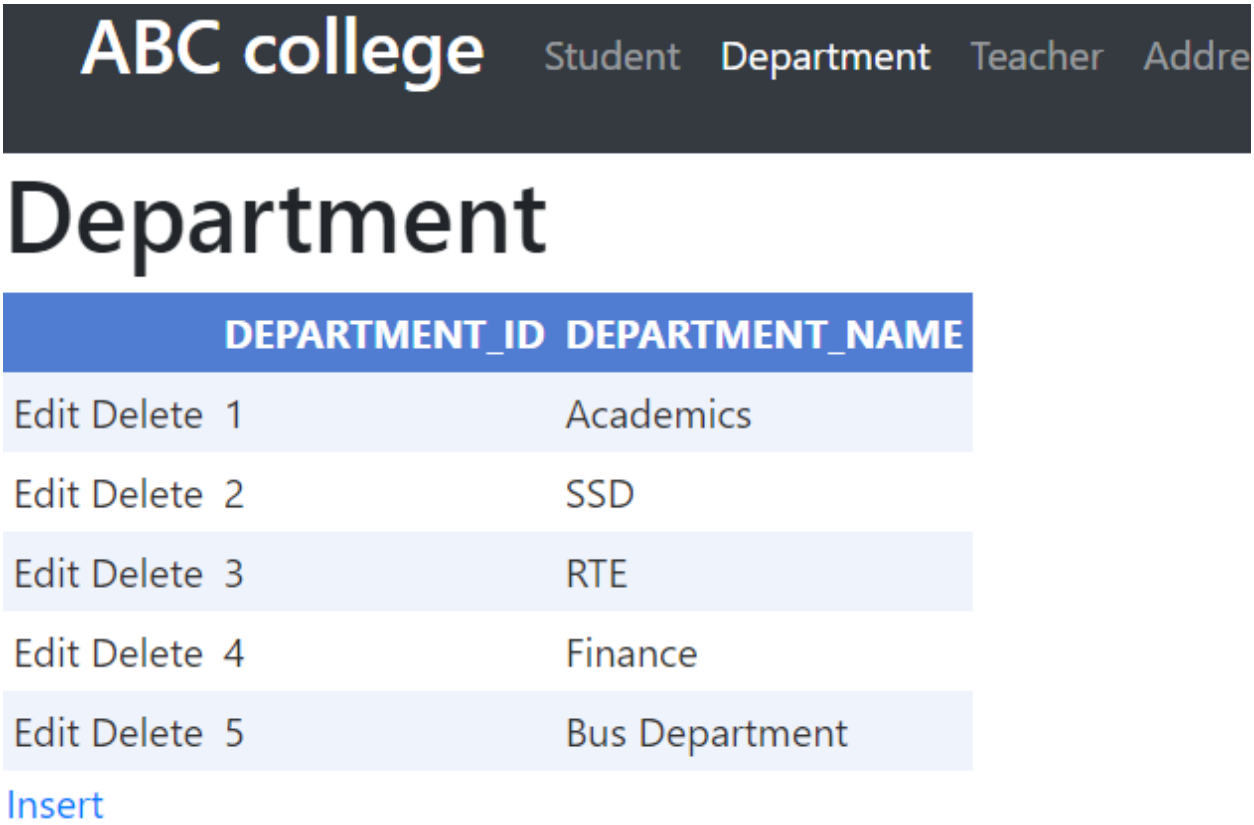


Figure 91: Fixing error Part 2

Test Case	19
Action	Inserting the Mistake Department_ID.
Expect Result	Table to take the value
Actual Result	Table took the value
Conclusion	Successful

Table 38: Test 19 after fixing the error

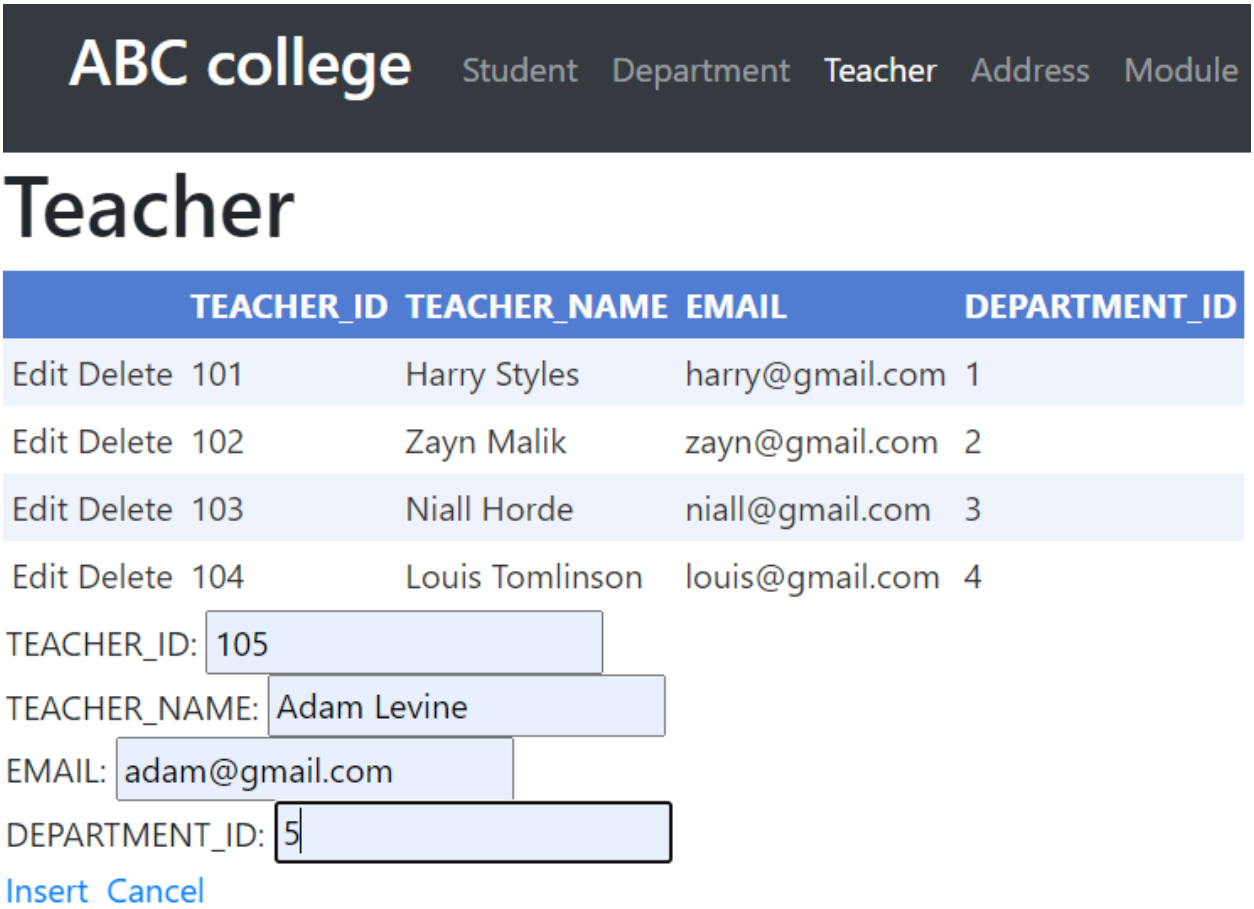


Figure 92: Checking the error

ABC college Student Department Teacher Address Module Te

Teacher

	TEACHER_ID	TEACHER_NAME	EMAIL	DEPARTMENT_ID
Edit Delete	101	Harry Styles	harry@gmail.com	1
Edit Delete	102	Zayn Malik	zayn@gmail.com	2
Edit Delete	103	Niall Horde	niall@gmail.com	3
Edit Delete	104	Louis Tomlinson	louis@gmail.com	4
Edit Delete	105	Adam Levine	adam@gmail.com	5

Insert

Table 39: Fixed

9. User Manual

The ABC College website consist of Navigation bar and a college logo.

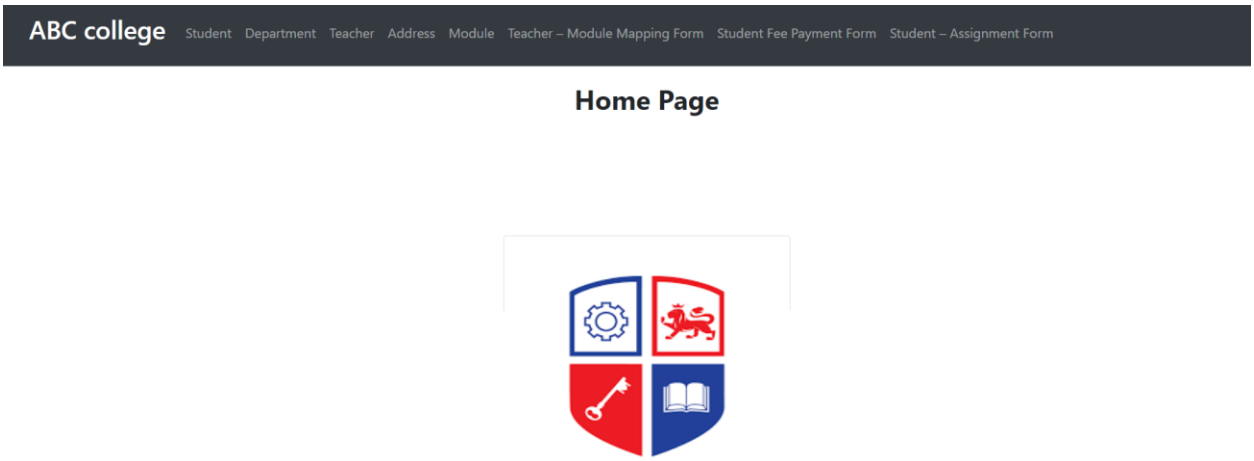


Table 40: Home Page

Pressing the ABC College will take us to the home.

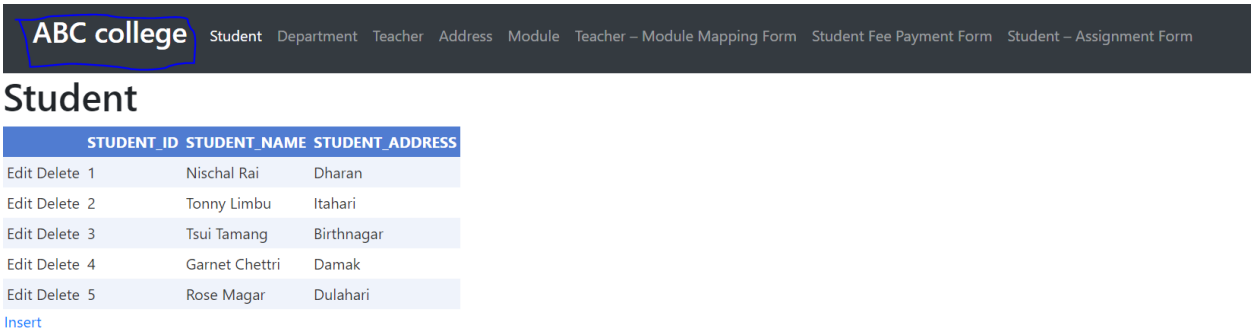


Table 41: Student Form

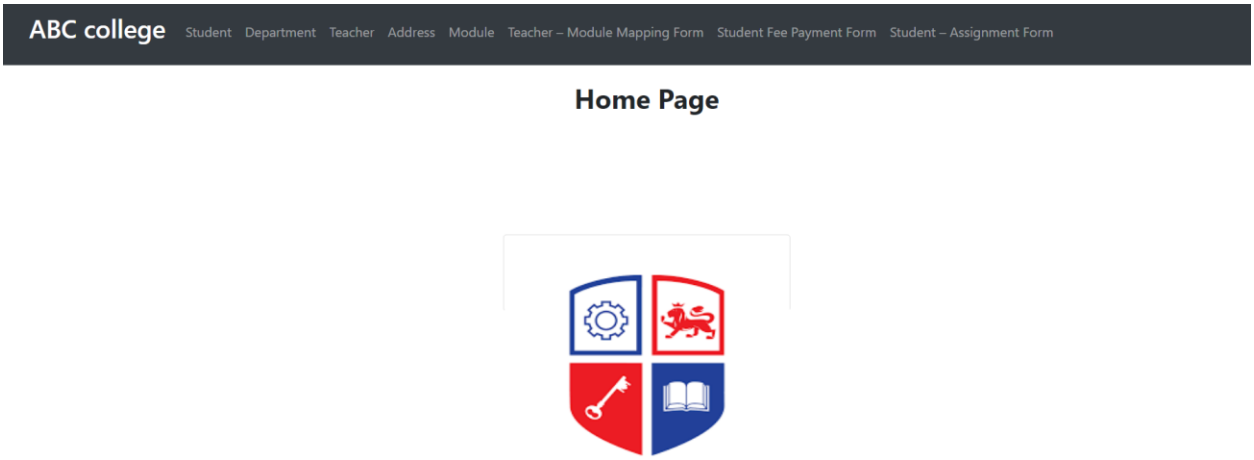


Table 42: Back to Home Page

Pressing other navigation will take us to their respective places and the active navigation will indicate the active form or page.

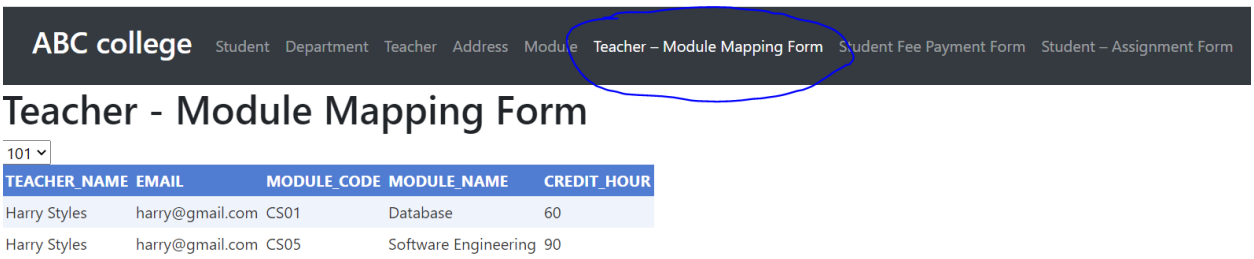


Table 43: Teacher - Module Mapping Form is active

10.Conclusion

The database is being created for an online college platform called ABC college. Since its inception, the student and teacher information has been physically recorded and registered. Because the old culture resulted in data duplication, the company required a web application that could perform CRUD operations with database integration. This would assist the organization in maintaining records in a systematic manner. The coursework provided an opportunity to investigate the situation of a hospitality industry and expand a database that meets the needs through the development of the database. The respective coursework has enabled the growth of learning skills and has proven to be an excellent method for learning database with proper implementation on a real-time web application. In addition, the implementation of the ASP.NET framework and architecture were taught throughout the course.

The normalization and data insertion processes were both time-consuming and difficult to complete. When database integration was implemented in web forms, this issue frequently arose. Various problems arose during the project's completion. Nonetheless, with proper research, self-learning, and helpful guidance from module teachers, all the problems were resolved. The coursework was successfully completed on time and according to the guidelines, and it turned out to be a great learning experience.

The following are some of the useful outcomes and techniques learned during the coursework completion process:

- ✓ Creating a database with appropriate tables and attributes based on the case study provided.
- ✓ Normalization of the examples provided.
- ✓ Using Data Modeler to create ERD and DDL scripts.
- ✓ Creating web applications with the ASP.NET framework.
- ✓ Integrating a database into an ASP.NET web application.

11.References

Hotka, 2006. *Oracle Sql Developer Handbook.* 1st ed. New Dehli: Tata McGraw-Hill Publishing Company Limited.

Powell, G., 2006. *Beginning Database Design.* 1st ed. Crosspoint Boulevard: Wiley Publishing, Inc..