CC6001NI Advanced Database System Development

40% Individual Coursework

Student Name: Bijay Bharati

London Met ID: 19030824

College ID: NP01CP4A190041

Assignment Due Date: 23rd March 2022

Assignment Submission Date: 23rd March 2022

Word Count: 2747

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline 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.

Contents

Figures

Introduction

This report contains the development process for creating a web application using ASP .NET with C# and Oracle SQL. Web forms are created to display and modify data in the database. Forms allow users to easily interact with the database without having to type the queries.

Textual Analysis

Berkeley college manages lots of data about its students, teachers, departments, assignments and fees and needs a database that is easy to use and maintain and gets the job done for the college. It is apparent that entities like student, teacher, department, module, fee, etc will exist and maintained it is also clear from the requirement analysis that it is very important to have record for student's attendance and student's fee payment details so the database will be designed by making is a priority to have easy access to those details. The example data provided will help us form a database. Next section of the report deals with normalization and producing a database.

Normalization

Normalization is the process by which we try to reduce anomalies like insertion anomalies and deletion anomalies and redundancy in a relational database. The goal of normalization is to reduce anomalies and redundancies, not eliminate them and in some cases, redundancies may even be kept by organizations according to their needs.

Normalization of fig 1

Normalization Process:

UNF (identify repeating groups)

Teacher (Teacher Name, {Address}, Email, {Module Code, Module Name, Credit Hours})

1NF (remove repeating groups)

Address -1 (Address ID, City, Zip)

Teacher -1 (Teacher ID, Name, Email, Address*)

Module -1 (Module ID, Module Code, Module Name, Credit Hours)

2NF (remove partial dependency)

Partial dependencies only exist where there are composite primary keys

Address -2 (Address ID, City, Zip)

Module -2 (Module ID, Module Code, Module Name, Credit Hours)

Teacher -1 (Teacher ID, Name, Email, Module*, Address *)

Teacher ID → Name, Email,

Teacher ID, Address →

Teacher ID, Module →

Teacher ID gives all other values, so it is in 2NF

Teacher -2 (Teacher ID, Name, Email, Module*, Address *)

3NF (remove transitive dependency)

Address -3 (Address ID, City, Zip) city and zip can give value for one another, but further separation is not needed

Module -3 (Module ID, Module Code, Module Name, Credit Hours) (module code can give credit hrs. but it does not make sense to further break the table module id is used as primary key because module id will be smaller in size and can save space in our database)

Teacher -3 (Teacher ID, Name, Email, Module*, Address *) transitive dependencies do not exist in teacher table

Normalization of fig 2.

The example data provided is like normalization in figure 1 and the steps are same additional entity assignment is introduced.

Student (Student id, student name, address, {module code, module name, assignment, grade, status})

After normalization

Address -3 (Address ID, Street, City)

Student -3 (Student id, name, address*)

Module -3 (Module code, module name)

Assignment can be separated into tables assignment and result but for easier integration with the requirements of Berkeley college, they are kept as shown below. The next section of the report will provide the whole database and assumptions made.

Assignment -3 (Assignment ID, type, grade, status, module*, student*)

Integration and Assumption

Some additional entities and attributes are introduced to make the database more meaningful.

Address (Address ID, City, Zip)

Course (Course ID, Course Name)

Module (Module ID, Course*, Module Code, Module Name, Credit Hours)

Department (Department ID, Department Name, Department Head)

Student (Student ID, Name, DOB, Address*, Course*, Email, Phone, Attendance)

Teacher (Teacher ID, Name, Email, Module*, Department*, Address *, phone)

Assignment (Assignment ID, Type, Grade, Status, Module*, Student*)

Fee (Fee ID, Student*, Amount, Fee Year/ Semester, Due Date, Payment Date, Remarks)

Assumptions:

- Students are enrolled in every module of a course.
- Students can become teachers only after graduation at which point, they are no longer considered students.
- attendance attribute stores attendance as % as overall attendance for every module.
- 1 teacher teaches only 1 module, but many teachers can teach the same module.

Final ERD

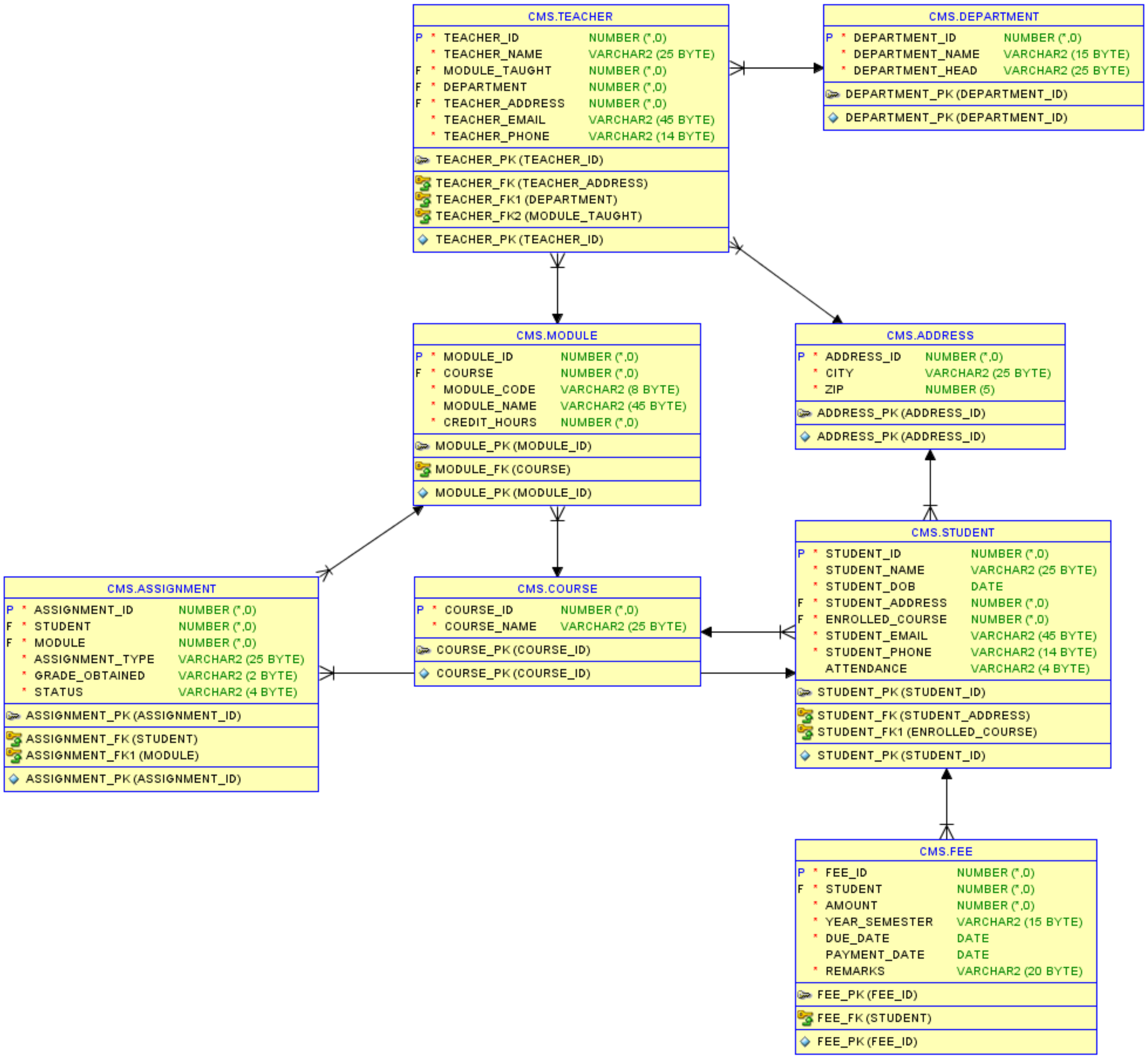

CMS.TEACHER
P * TEACHER_ID NUMBER (*,0)
* TEACHER_NAME VARCHAR2 (25 BYTE)
F * MODULE_TAUGHT NUMBER (*,0)
F * DEPARTMENT NUMBER (*,0)
F * TEACHER_ADDRESS NUMBER (*,0)
* TEACHER_EMAIL VARCHAR2 (45 BYTE)
* TEACHER_PHONE VARCHAR2 (14 BYTE)
TEACHER_PK (TEACHER_ID)
TEACHER_FK (TEACHER_ADDRESS)
TEACHER_FK1 (DEPARTMENT)
TEACHER_FK2 (MODULE_TAUGHT)
TEACHER_PK (TEACHER_ID)
CMS.DEPARTMENT
P * DEPARTMENT_ID NUMBER (*,0)
* DEPARTMENT_NAME VARCHAR2 (15 BYTE)
* DEPARTMENT_HEAD VARCHAR2 (25 BYTE)
DEPARTMENT_PK (DEPARTMENT_ID)
DEPARTMENT_PK (DEPARTMENT_ID)
CMS.MODULE
P * MODULE_ID NUMBER (*,0)
F * COURSE NUMBER (*,0)
* MODULE_CODE VARCHAR2 (8 BYTE)
* MODULE_NAME VARCHAR2 (45 BYTE)
* CREDIT_HOURS NUMBER (*,0)
MODULE_PK (MODULE_ID)
MODULE_FK (COURSE)
MODULE_PK (MODULE_ID)
CMS.ADDRESS
P * ADDRESS_ID NUMBER (*,0)
* CITY VARCHAR2 (25 BYTE)
* ZIP NUMBER (5)
ADDRESS_PK (ADDRESS_ID)
ADDRESS_PK (ADDRESS_ID)
CMS.ASSIGNMENT
P * ASSIGNMENT_ID NUMBER (*,0)
F * STUDENT NUMBER (*,0)
F * MODULE NUMBER (*,0)
* ASSIGNMENT_TYPE VARCHAR2 (25 BYTE)
* GRADE_OBTAINED VARCHAR2 (2 BYTE)
* STATUS VARCHAR2 (4 BYTE)
ASSIGNMENT_PK (ASSIGNMENT_ID)
ASSIGNMENT_FK (STUDENT)
ASSIGNMENT_FK1 (MODULE)
ASSIGNMENT_PK (ASSIGNMENT_ID)
CMS.COURSE
P * COURSE_ID NUMBER (*,0)
* COURSE_NAME VARCHAR2 (25 BYTE)
COURSE_PK (COURSE_ID)
COURSE_PK (COURSE_ID)
CMS.STUDENT
P * STUDENT_ID NUMBER (*,0)
* STUDENT_NAME VARCHAR2 (25 BYTE)
* STUDENT_DOB DATE
F * STUDENT_ADDRESS NUMBER (*,0)
F * ENROLLED_COURSE NUMBER (*,0)
* STUDENT_EMAIL VARCHAR2 (45 BYTE)
* STUDENT_PHONE VARCHAR2 (14 BYTE)
ATTENDANCE VARCHAR2 (4 BYTE)
STUDENT_PK (STUDENT_ID)
STUDENT_FK (STUDENT_ADDRESS)
STUDENT_FK1 (ENROLLED_COURSE)
STUDENT_PK (STUDENT_ID)
CMS.FEE
P * FEE_ID NUMBER (*,0)
F * STUDENT NUMBER (*,0)
* AMOUNT NUMBER (*,0)
* YEAR_SEMESTER VARCHAR2 (15 BYTE)
* DUE_DATE DATE
PAYMENT_DATE DATE
* REMARKS VARCHAR2 (20 BYTE)
FEE_PK (FEE_ID)
FEE_FK (STUDENT)
FEE_PK (FEE_ID)

Figure 1: Final ERD

Data Dictionary

Column Name	Data Type	Size	Constraint	Reference Table	Reference Column	Description	Example Data
ADDRESS_ID	Number	38	Primary Key			To uniquely identify Each address	100
CITY	Varchar	25	Not null			To store the city name	Kathmandu
ZIP	Number	5	Not null			To store the area zip code	44600
ASSIGNMENT_ID	Number	38	Primary Key			Identify assignments	1
STUDENT	Number	38	Foreign Key	STUDENT	STUDENT_ID	Identify student	100
MODULE	Number	38	Foreign Key	MODULE	MODULE_ID	Identify module	1
ASSIGNMENT_TYPE	Varchar	25	Foreign Key			Store assignment type	Coursework
GRADE_OBTAINED	Varchar	2	Not null			Store grade	A+
STATUS	Varchar	4	Not null			Store pass, fail status	Pass
COURSE_ID	Number	38	Primary Key			Identify course	1
COURSE_NAME	Varchar	25	Not null			Store course name	Computing
DEPARTMENT_ID	Number	38	Primary Key			Identify department	1
DEPARTMENT_NAME	Varchar	15	Not null			Store department name	Finance
DEPARTMENT_HEAD	Varchar	25	Not null			Store name of department head	Bobby Brown
FEE_ID	Number	38	Primary Key			Identify dee	1
STUDENT	Number	38	Foreign Key	STUDENT	STUDENT_ID	Identify student	100
AMOUNT	Number	38	Not null			Store fee amount	100000

YEAR_SEMESTER	Varchar	15	Not null			Store year and semester detail	Year 3 sem 2
DUE_DATE	DATE		Not null			Store fee deadline	01-JAN-2001
PAYMENT_DATE	DATE		Nullable			Store fee payment date	
REMARKS	Varchar	20	Not null			Store remarks	Pending
MODULE_ID	Number	38	Primary Key			Identify module	1
COURSE	Number	38	Foreign Key	COURSE	COURSE_ID	Identify course	1
MODULE_CODE	Varchar	8	Not null			Store module code	CC6001NI
MODULE_NAME	Varchar	45	Not null			Store module name	Databases
CREDIT_HOURS	Number	38	Not null			Store credit hours	4
STUDENT_ID	Number	38	Primary Key			Identify student	100001
STUDENT_NAME	Varchar	25	Not null			Store student name	Bob
STUDENT_DOB	Varchar		Not null			Store student DOB	01-JAN-1999
STUDENT_ADDRESS	Number	38	Foreign Key	ADDRESS	ADDRESS_ID	Identify address	2
ENROLLED_COURSE	Number	38	Foreign Key	COURSE	COURSE_ID	Identify course	2
STUDENT_EMAIL	Varchar	45	Not null			Store student email	generic@mail.com
STUDENT_PHONE	Varchar	14	Not null			Store student phone number	+9779898989898
ATTENDANCE	Varchar	4	Not null			Store attendance record	100%
TEACHER_ID	Number	38	Primary Key			Identify teacher	1
TEACHER_NAME	Varchar	25	Not null			Store teacher name	Sam
MODULE_TAUGHT	Number	38	Foreign Key	MODULE	MODULE_ID	Identify module taught by teacher	2

DEPARTMENT	Number	38	Foreign Key	DEPARTMENT	DEPARTMENT_ID	Identify teacher’s department	2
TEACHER_ADDRESS	Number	38	Foreign Key	ADDRESS	ADDRESS_ID	Identify teacher’s address	2
TEACHER_EMAIL	Varchar	45	Not null			Store email details	generic1@mail.com
TEACHER_PHONE	Varchar	14	Not null			Store phone numbet	6849898989

Script

Scripts used to create tables

```
CREATE TABLE address (

    address_id INT NOT NULL,

    city VARCHAR (25) NOT NULL,

    zip NUMBER(5) NOT NULL,

    CONSTRAINT address_pk PRIMARY KEY (address_id)

);


CREATE TABLE course (

    course_id INT NOT NULL,

    course_name VARCHAR (25) NOT NULL,

    CONSTRAINT course_pk PRIMARY KEY (course_id)

);


CREATE TABLE module(

    module_id INT NOT NULL,

    course INT NOT NULL,

    module_code VARCHAR (8) NOT NULL,

    module_name VARCHAR (45) NOT NULL,

    credit_hours INT NOT NULL,

    CONSTRAINT module_pk PRIMARY KEY (module_id),

    CONSTRAINT module_fk FOREIGN KEY(course) REFERENCES course(course_id)

);


CREATE TABLE department(

    department_id INT NOT NULL,

    department_name VARCHAR (15) NOT NULL,
    department_head VARCHAR (25) NOT NULL,
    CONSTRAINT department_pk PRIMARY KEY (department_id)
);

CREATE TABLE student (
    student_id INT NOT NULL,
    student_name VARCHAR (25) NOT NULL,
    student_DOB DATE NOT NULL,
    student_address INT NOT NULL,
    enrolled_course INT NOT NULL,
    student_email VARCHAR (45) NOT NULL,
    student_phone VARCHAR (14) NOT NULL,
    CONSTRAINT student_pk PRIMARY KEY (student_id),
    CONSTRAINT student_fk FOREIGN KEY(student_address) REFERENCES address(address_id),
    CONSTRAINT student_fk1 FOREIGN KEY(enrolled_course) REFERENCES course(course_id)
);

CREATE TABLE teacher(
    teacher_id INT NOT NULL,
    teacher_name VARCHAR (25) NOT NULL,
    module_taught INT NOT NULL,
    department INT NOT NULL,
    teacher_address INT NOT NULL,
    teacher_email VARCHAR (45) NOT NULL,
    teacher_phone VARCHAR (14) NOT NULL,
    CONSTRAINT teacher_pk PRIMARY KEY (teacher_id),
    CONSTRAINT teacher_fk FOREIGN KEY(teacher_address) REFERENCES address(address_id),
    CONSTRAINT teacher_fk1 FOREIGN KEY(department) REFERENCES department(department_id),
    CONSTRAINT teacher_fk2 FOREIGN KEY(module_taught) REFERENCES module(module_id)
);

CREATE TABLE assignment(
    assignment_id INT NOT NULL,
    student INT NOT NULL,
    module INT NOT NULL,
    assignment_type VARCHAR (25) NOT NULL,
    grade_obtained VARCHAR (2) NOT NULL,
    status VARCHAR (4) NOT NULL,
    CONSTRAINT assignment_pk PRIMARY KEY (assignment_id),
    CONSTRAINT assignment_fk FOREIGN KEY(student) REFERENCES student(student_id),
    CONSTRAINT assignment_fk1 FOREIGN KEY(module) REFERENCES module(module_id)
);


CREATE TABLE fee(
    fee_id INT NOT NULL,
    student INT NOT NULL,
    amount INT NOT NULL,
    year_semester VARCHAR (15) NOT NULL,
    due_date DATE NOT NULL,
    payment_date DATE,
    remarks VARCHAR (20) NOT NULL,
    CONSTRAINT fee_pk PRIMARY KEY (fee_id),
    CONSTRAINT fee_fk FOREIGN KEY(student) REFERENCES student(student_id)
);

CREATE SEQUENCE address_sequence
    start with 100
    increment by 1;

CREATE OR REPLACE TRIGGER address_on_insert
    BEFORE INSERT ON address
    FOR EACH ROW
BEGIN
    SELECT address_sequence.nextval
    INTO :new.address_id
    FROM dual;
END;

CREATE SEQUENCE course_sequence
    start with 1
    increment by 1;

CREATE OR REPLACE TRIGGER course_on_insert
    BEFORE INSERT ON course
    FOR EACH ROW
BEGIN
    SELECT course_sequence.nextval
    INTO :new.course_id
    FROM dual;
```

```
END;

CREATE SEQUENCE assignment_sequence
    start with 1
    increment by 1;

CREATE OR REPLACE TRIGGER assignment_on_insert
    BEFORE INSERT ON assignment
    FOR EACH ROW
BEGIN
    SELECT assignment_sequence.nextval
    INTO :new.assignment_id
    FROM dual;
END;

CREATE SEQUENCE fee_sequence
    start with 1
    increment by 1;

CREATE OR REPLACE TRIGGER fee_on_insert
    BEFORE INSERT ON fee
    FOR EACH ROW
BEGIN
    SELECT fee_sequence.nextval
    INTO :new.fee_id
    FROM dual;
END;
```

```
CREATE SEQUENCE department_sequence
   start with 1
   increment by 1;

CREATE OR REPLACE TRIGGER department_on_insert
   BEFORE INSERT ON department
   FOR EACH ROW
BEGIN
   SELECT department_sequence.nextval
   INTO :new.department_id
   FROM dual;
END;

CREATE SEQUENCE module_sequence
   start with 1
   increment by 1;

CREATE OR REPLACE TRIGGER module_on_insert
   BEFORE INSERT ON module
   FOR EACH ROW
BEGIN
   SELECT module_sequence.nextval
   INTO :new.module_id
   FROM dual;
END;

CREATE SEQUENCE student_sequence
   start with 10000
```

```
    increment by 1;

CREATE OR REPLACE TRIGGER student_on_insert
    BEFORE INSERT ON student
    FOR EACH ROW
BEGIN
    SELECT student_sequence.nextval
    INTO :new.student_id
    FROM dual;
END;

CREATE SEQUENCE teacher_sequence
    start with 10000
    increment by 1;

CREATE OR REPLACE TRIGGER teacher_on_insert
    BEFORE INSERT ON teacher
    FOR EACH ROW
BEGIN
    SELECT teacher_sequence.nextval
    INTO :new.teacher_id
    FROM dual;
END;
```

Scripts used to modify tables

ALTER TABLE student ADD attendance VARCHAR (4) ;

UPDATE student SET attendance = '100%' WHERE student_id = 10005;

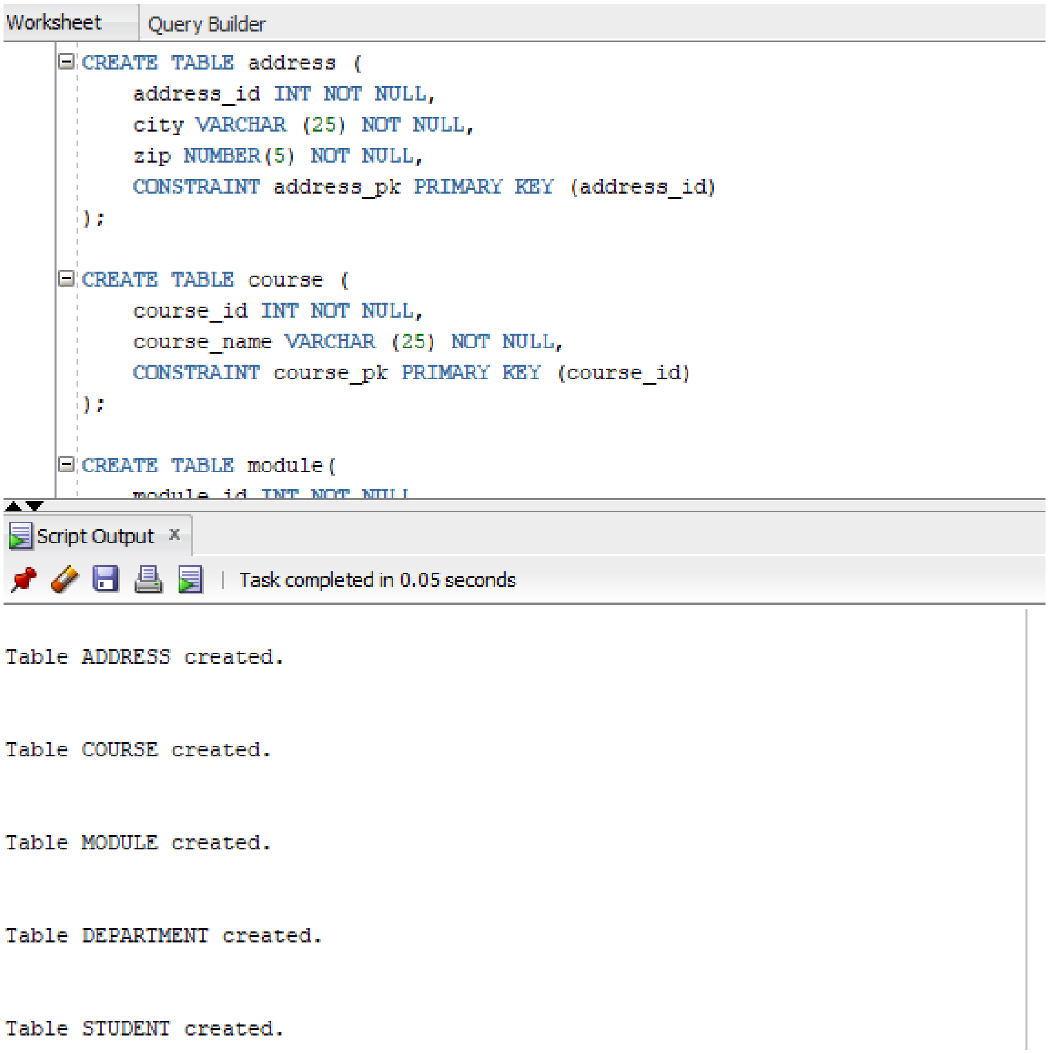
UPDATE student SET attendance = '80%' WHERE student_id = 10006;

UPDATE student SET attendance = '70%' WHERE student_id = 10007;

UPDATE student SET attendance = '60%' WHERE student_id = 10008;

UPDATE student SET attendance = '50%' WHERE student_id = 10009;

UPDATE student SET attendance = '40%' WHERE student_id = 10020;



```
CREATE TABLE address (
    address_id INT NOT NULL,
    city VARCHAR (25) NOT NULL,
    zip NUMBER(5) NOT NULL,
    CONSTRAINT address_pk PRIMARY KEY (address_id)
);

CREATE TABLE course (
    course_id INT NOT NULL,
    course_name VARCHAR (25) NOT NULL,
    CONSTRAINT course_pk PRIMARY KEY (course_id)
);

CREATE TABLE module(
    module_id INT NOT NULL
```

Task completed in 0.05 seconds

```
Table ADDRESS created.


Table COURSE created.


Table MODULE created.


Table DEPARTMENT created.


Table STUDENT created.
```

Figure 2: Table Creation

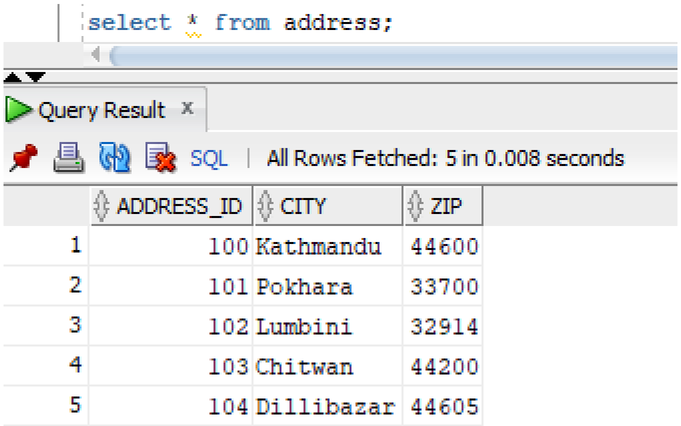
Insert Statement

```
INSERT ALL
   INTO address (city, zip) VALUES ('Kathmandu', 44600)
   INTO address (city, zip) VALUES ('Pokhara', 33700)
   INTO address (city, zip) VALUES ('Lumbini', 32914)
   INTO address (city, zip) VALUES ('Chitwan', 44200)
   INTO address (city, zip) VALUES ('Dillibazar', 44605)
SELECT * FROM dual;


INSERT ALL
   INTO course (course_name) VALUES ('Computing')
   INTO course (course_name) VALUES ('Networking')
   INTO course (course_name) VALUES ('Multimedia')
   INTO course (course_name) VALUES ('Marketing')
   INTO course (course_name) VALUES ('Artificial Intelligence')
SELECT * FROM dual;


INSERT ALL
   INTO module (course,module_code,module_name,credit_hours) VALUES (1,'CS0134NI','Databases',4)
   INTO module (course,module_code,module_name,credit_hours) VALUES (1,'CS0134NI','Application Development',3)
   INTO module (course,module_code,module_name,credit_hours) VALUES (2,'CS0154NA','Ethical Hacking',4)
   INTO module (course,module_code,module_name,credit_hours) VALUES (2,'CS1134NA','Networks',4)
   INTO module (course,module_code,module_name,credit_hours) VALUES (3,'CS5134MI','3D Modelling',2)
   INTO module (course,module_code,module_name,credit_hours) VALUES (3,'CS6134MI','Game Design',2)
   INTO module (course,module_code,module_name,credit_hours) VALUES
(4,'CS7134BB','Accounting',3)

   INTO module (course,module_code,module_name,credit_hours) VALUES
(4,'CS8199BB','Business',6)

   INTO module (course,module_code,module_name,credit_hours) VALUES
(5,'CS9114NE','Algorithms',5)

   INTO module (course,module_code,module_name,credit_hours) VALUES
(5,'CS0104NE','Mathematics',4)

SELECT * FROM dual;


INSERT ALL

   INTO department (department_name,department_head) VALUES ('Digital Design','John Doe')

   INTO department (department_name,department_head) VALUES ('Computing','Bob Vance')

   INTO department (department_name,department_head) VALUES ('Maths','Michael Scott')

   INTO department (department_name,department_head) VALUES ('Business','Paul Newman')

   INTO department (department_name,department_head) VALUES ('Finance','Ben Simmons')

SELECT * FROM dual;


INSERT ALL

   INTO student
(student_name,student_DOB,student_address,enrolled_course,student_email,student_phone)
VALUES ('Scott Lang','01-JAN-1999',100,5,'sl1@mail.com','8998948984')

   INTO student
(student_name,student_DOB,student_address,enrolled_course,student_email,student_phone)
VALUES ('Sam Lee','03-FEB-1997',102,4,'slee@mail.com','1238763903')

   INTO student
(student_name,student_DOB,student_address,enrolled_course,student_email,student_phone)
VALUES ('Xiao Lang','01-JAN-2001',103,3,'xcn@mail.com','7865656676')

   INTO student
(student_name,student_DOB,student_address,enrolled_course,student_email,student_phone)
VALUES ('Hank Pym','11-Mar-2000',104,2,'hpy@mail.com','8000888989')
```

```
   INTO student
(student_name,student_DOB,student_address,enrolled_course,student_email,student_phone)
VALUES ('Happy H','21-AUG-1998',101,5,'hh@mail.com','8787878784')

SELECT * FROM dual;


INSERT ALL

   INTO teacher
(teacher_name,module_taught,department,teacher_address,teacher_email,teacher_phone)
VALUES ('Simon Fox',1,2,104,'sf@mail.com','8447447447')

   INTO teacher
(teacher_name,module_taught,department,teacher_address,teacher_email,teacher_phone)
VALUES ('Sam Wilson',8,5,103,'swll@mail.com','8777747447')

   INTO teacher
(teacher_name,module_taught,department,teacher_address,teacher_email,teacher_phone)
VALUES ('Barry Don',5,1,102,'bdn@mail.com','1117447447')

   INTO teacher
(teacher_name,module_taught,department,teacher_address,teacher_email,teacher_phone)
VALUES ('Pen Smith',6,1,101,'psm@mail.com','9908947447')

   INTO teacher
(teacher_name,module_taught,department,teacher_address,teacher_email,teacher_phone)
VALUES ('Will Man',2,2,100,'wma@mail.com','1278364098')

SELECT * FROM dual;




INSERT ALL

   INTO assignment (student,module,assignment_type,grade_obtained,status) VALUES
(10005,10,'Individual Coursework','A','PASS')

   INTO assignment (student,module,assignment_type,grade_obtained,status) VALUES
(10005,9,'Written Exam','A','PASS')

   INTO assignment (student,module,assignment_type,grade_obtained,status) VALUES
(10006,8,'Group Coursework','A+','PASS')

   INTO assignment (student,module,assignment_type,grade_obtained,status) VALUES
(10006,7,'Presentation','A','PASS')
```

```
   INTO assignment (student,module,assignment_type,grade_obtained,status) VALUES
(10007,6,'MCQ Exam','A+','PASS')

   INTO assignment (student,module,assignment_type,grade_obtained,status) VALUES
(10007,5,'Individual Coursework','A','PASS')

   INTO assignment (student,module,assignment_type,grade_obtained,status) VALUES
(10008,4,'Written Examination','A','PASS')

   INTO assignment (student,module,assignment_type,grade_obtained,status) VALUES
(10008,3,'Viva','B','PASS')

   INTO assignment (student,module,assignment_type,grade_obtained,status) VALUES
(10009,2,'Individual Coursework','B+','PASS')

   INTO assignment (student,module,assignment_type,grade_obtained,status) VALUES
(10009,1,'Individual Coursework','F','Fail')

SELECT * FROM dual;


INSERT ALL

   INTO fee (student,amount,year_semester,due_date,payment_date,remarks) VALUES
(10005,100000,'Year 3 sem 2','01-JAN-2021',",'Not paid')

   INTO fee (student,amount,year_semester,due_date,payment_date,remarks) VALUES
(10004,110000,'Year 3 sem 2','01-FEB-2021','01-FEB-2021','Paid')

   INTO fee (student,amount,year_semester,due_date,payment_date,remarks) VALUES
(10003,120000,'Year 3 sem 2','01-APR-2021',",'Pending')

   INTO fee (student,amount,year_semester,due_date,payment_date,remarks) VALUES
(10002,130000,'Year 3 sem 2','01-MAR-2021','11-MAR-2021','Paid')

   INTO fee (student,amount,year_semester,due_date,payment_date,remarks) VALUES
(10001,140000,'Year 3 sem 2','01-MAR-2021','07-MAR-2021','Paid')

SELECT * FROM dual;
```

Select Statement



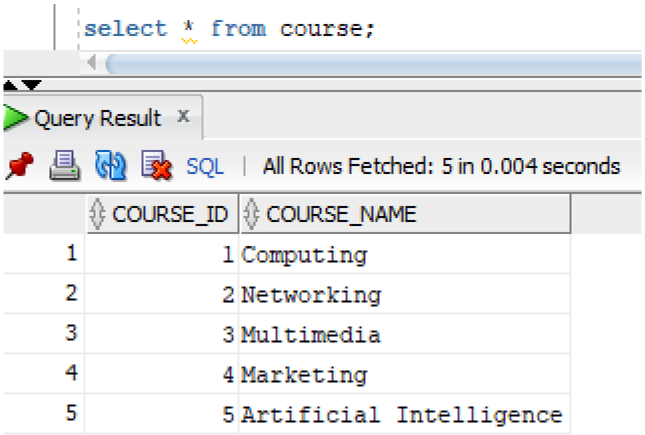
```
select * from address;
```

Query Result

SQL | All Rows Fetched: 5 in 0.008 seconds

	ADDRESS_ID	CITY	ZIP
1	100	Kathmandu	44600
2	101	Pokhara	33700
3	102	Lumbini	32914
4	103	Chitwan	44200
5	104	Dillibazar	44605

Figure 3: Select statement on address



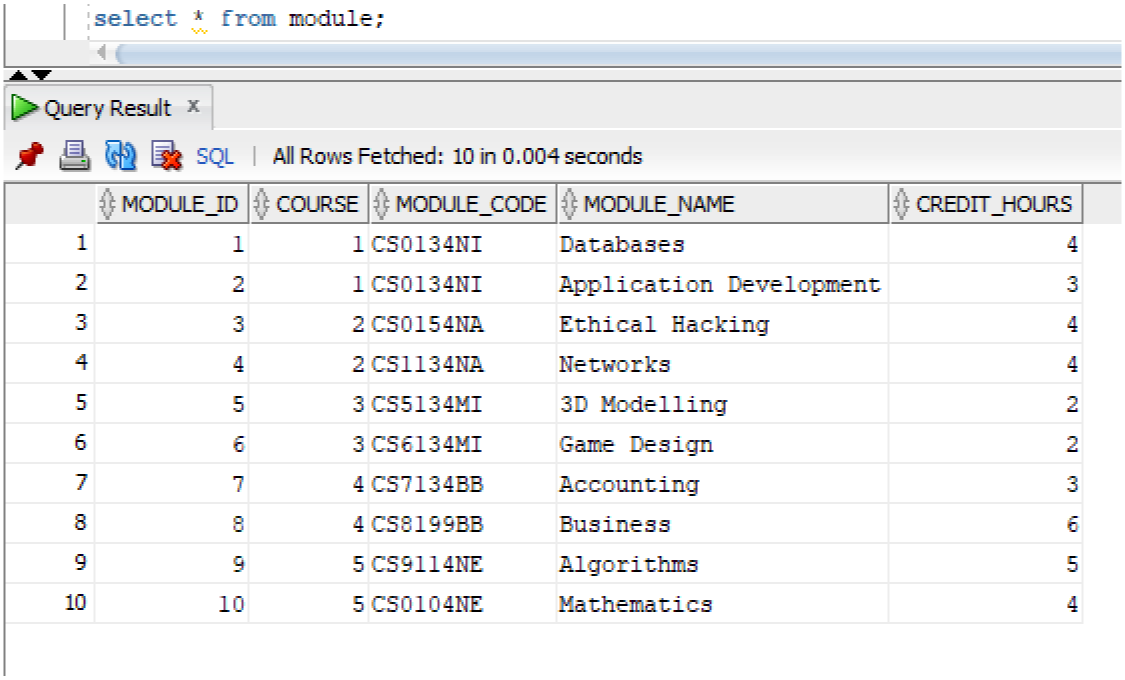
```
select * from course;
```

Query Result

SQL | All Rows Fetched: 5 in 0.004 seconds

	COURSE_ID	COURSE_NAME
1	1	Computing
2	2	Networking
3	3	Multimedia
4	4	Marketing
5	5	Artificial Intelligence

Figure 4: Select statement on course

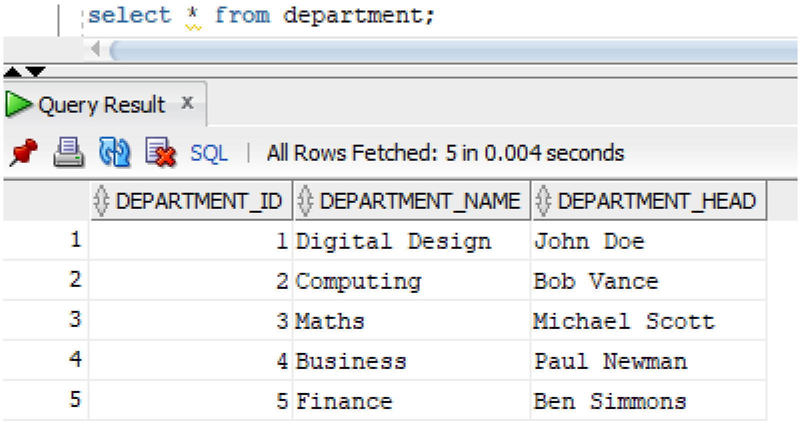


```
select * from module;
```

All Rows Fetched: 10 in 0.004 seconds

	MODULE_ID	COURSE	MODULE_CODE	MODULE_NAME	CREDIT_HOURS
1	1	1	CS0134NI	Databases	4
2	2	1	CS0134NI	Application Development	3
3	3	2	CS0154NA	Ethical Hacking	4
4	4	2	CS1134NA	Networks	4
5	5	3	CS5134MI	3D Modelling	2
6	6	3	CS6134MI	Game Design	2
7	7	4	CS7134BB	Accounting	3
8	8	4	CS8199BB	Business	6
9	9	5	CS9114NE	Algorithms	5
10	10	5	CS0104NE	Mathematics	4

Figure 5: Select statement on module

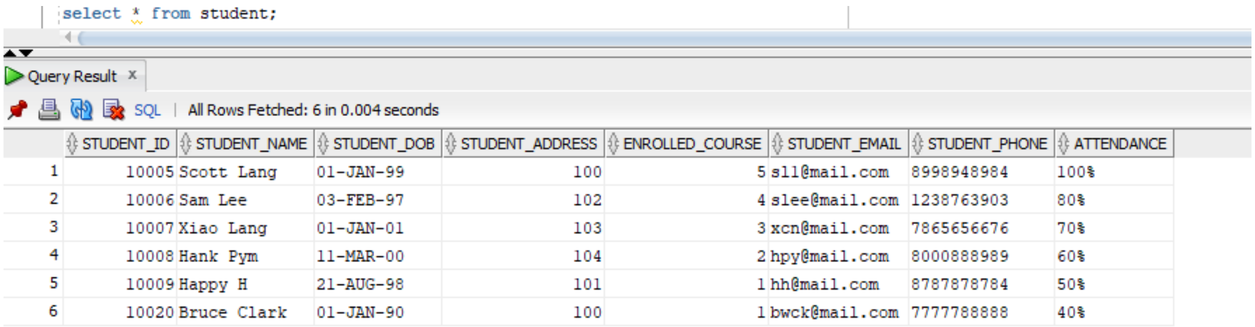


```
select * from department;
```

All Rows Fetched: 5 in 0.004 seconds

	DEPARTMENT_ID	DEPARTMENT_NAME	DEPARTMENT_HEAD
1	1	Digital Design	John Doe
2	2	Computing	Bob Vance
3	3	Maths	Michael Scott
4	4	Business	Paul Newman
5	5	Finance	Ben Simmons

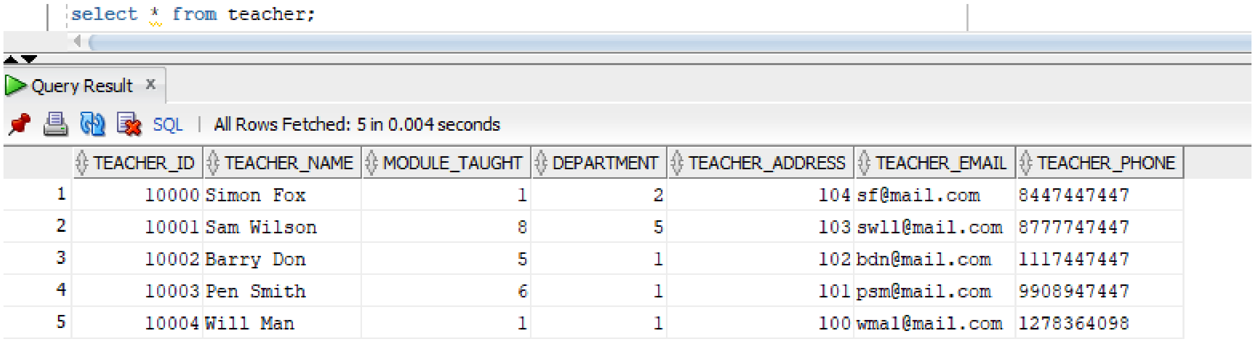
Figure 6: Select statement on department



```
select * from student;
```

	STUDENT_ID	STUDENT_NAME	STUDENT_DOB	STUDENT_ADDRESS	ENROLLED_COURSE	STUDENT_EMAIL	STUDENT_PHONE	ATTENDANCE
1	10005	Scott Lang	01-JAN-99	100	5	sl1@mail.com	8998948984	100%
2	10006	Sam Lee	03-FEB-97	102	4	slee@mail.com	1238763903	80%
3	10007	Xiao Lang	01-JAN-01	103	3	xcn@mail.com	7865656676	70%
4	10008	Hank Pym	11-MAR-00	104	2	hpy@mail.com	8000888989	60%
5	10009	Happy H	21-AUG-98	101	1	hh@mail.com	8787878784	50%
6	10020	Bruce Clark	01-JAN-90	100	1	bwck@mail.com	7777788888	40%

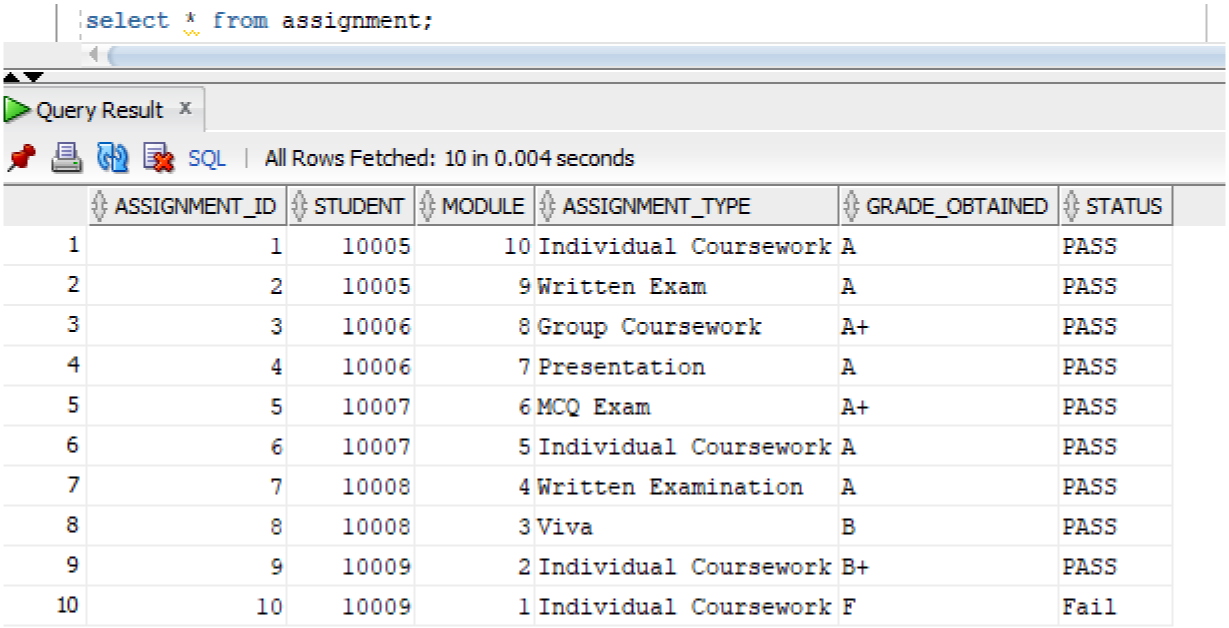
Figure 7: Select statement on student



```
select * from teacher;
```

	TEACHER_ID	TEACHER_NAME	MODULE_TAUGHT	DEPARTMENT	TEACHER_ADDRESS	TEACHER_EMAIL	TEACHER_PHONE
1	10000	Simon Fox	1	2	104	sf@mail.com	8447447447
2	10001	Sam Wilson	8	5	103	swl1@mail.com	8777747447
3	10002	Barry Don	5	1	102	bdn@mail.com	1117447447
4	10003	Pen Smith	6	1	101	psm@mail.com	9908947447
5	10004	Will Man	1	1	100	wmal@mail.com	1278364098

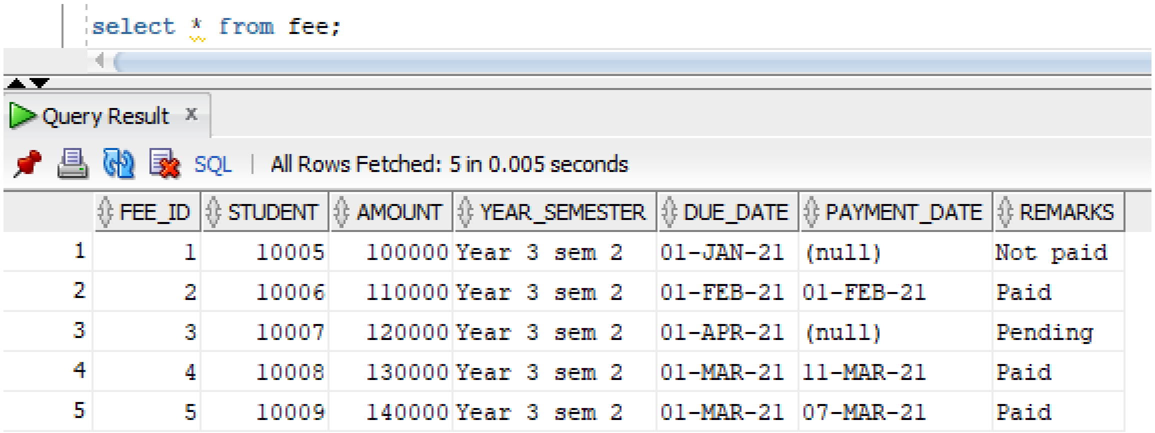
Figure 8: Select statement on teacher



```
select * from assignment;
```

	ASSIGNMENT_ID	STUDENT	MODULE	ASSIGNMENT_TYPE	GRADE_OBTAINED	STATUS
1	1	10005	10	Individual Coursework	A	PASS
2	2	10005	9	Written Exam	A	PASS
3	3	10006	8	Group Coursework	A+	PASS
4	4	10006	7	Presentation	A	PASS
5	5	10007	6	MCQ Exam	A+	PASS
6	6	10007	5	Individual Coursework	A	PASS
7	7	10008	4	Written Examination	A	PASS
8	8	10008	3	Viva	B	PASS
9	9	10009	2	Individual Coursework	B+	PASS
10	10	10009	1	Individual Coursework	F	Fail

Figure 9: Select statement on assignment



```
select * from fee;
```

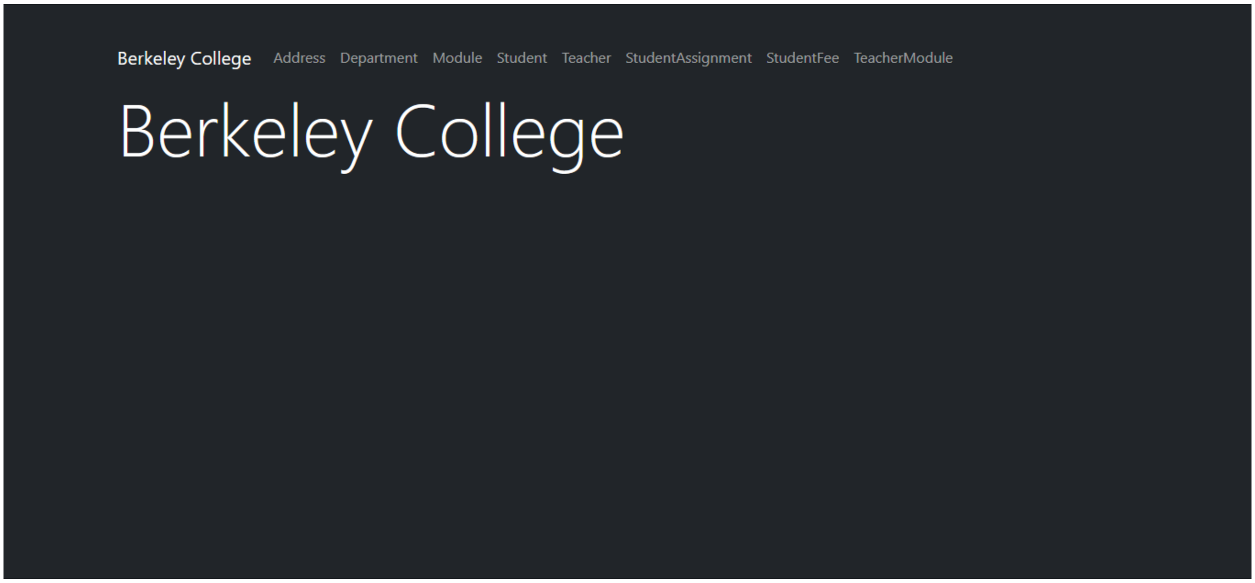
All Rows Fetched: 5 in 0.005 seconds

	FEE_ID	STUDENT	AMOUNT	YEAR_SEMESTER	DUE_DATE	PAYMENT_DATE	REMARKS
1	1	10005	100000	Year 3 sem 2	01-JAN-21	(null)	Not paid
2	2	10006	110000	Year 3 sem 2	01-FEB-21	01-FEB-21	Paid
3	3	10007	120000	Year 3 sem 2	01-APR-21	(null)	Pending
4	4	10008	130000	Year 3 sem 2	01-MAR-21	11-MAR-21	Paid
5	5	10009	140000	Year 3 sem 2	01-MAR-21	07-MAR-21	Paid

Figure 10: Select statement on fee

Forms

Dashboard or Home Page



Berkeley College

Figure 11: Home Page

Complex Form and Queries

SQL Queries

Query for studentfee.aspx form search function:

```
@"SELECT student_name AS ""Student Name"" , student_id AS ""Student ID"", amount AS ""Fee
Amount"", year_semester AS ""Year/Semester"", due_date AS ""Due Date"", payment_date AS
""Payment Date"", remarks AS ""Remarks"" FROM student inner join fee on
student_id=student WHERE student_id=" + studentID + "";
```

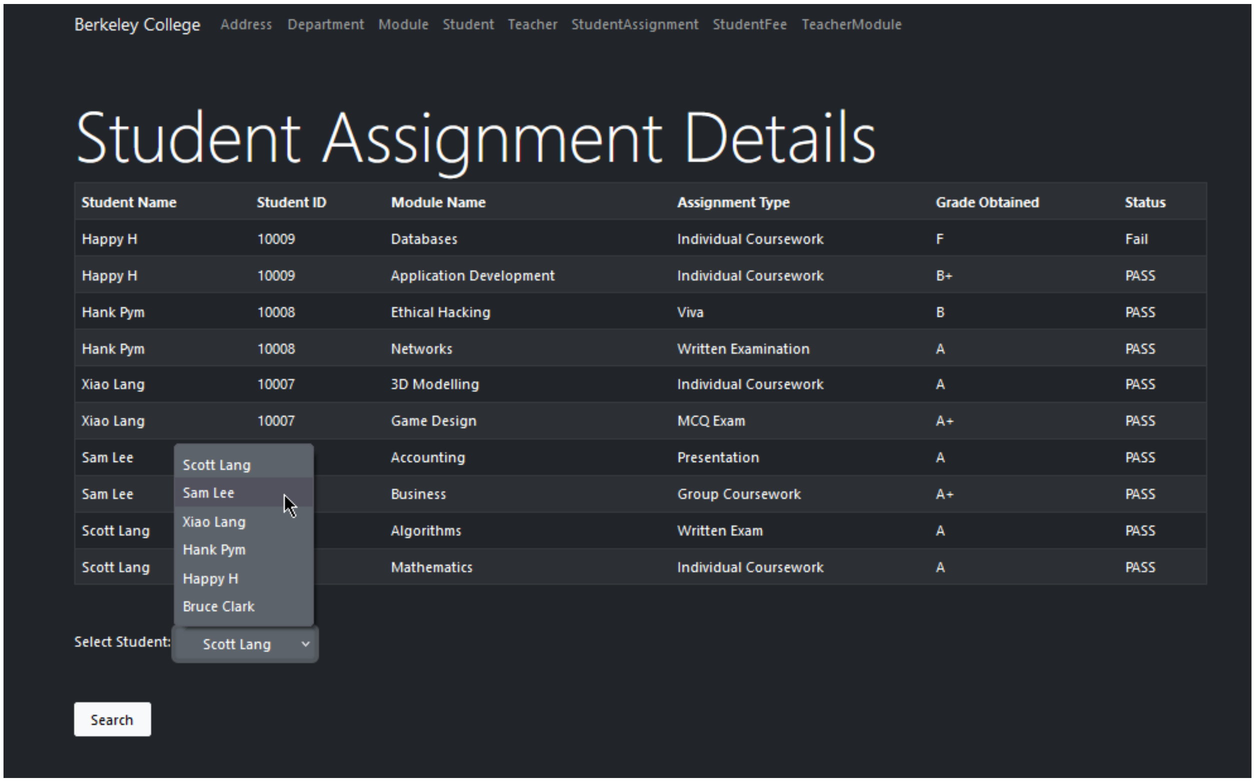
Query for studentassignment.aspx form search function:

```
@"SELECT student_name AS ""Student Name"" , student_id AS ""Student ID"", module_name AS
""Module Name"", assignment_type AS ""Assignment Type"", grade_obtained AS ""Grade
Obtained"", status AS ""Status""  FROM student  JOIN assignment ON
student.student_id=assignment.student JOIN module ON module.module_id=assignment.module
AND assignment.student=student.student_id WHERE student_id=" + studentID + "";
```

Query for teachermodule.aspx form search function:

```
@"SELECT teacher_name AS ""Teacher Name"" , module_name AS ""Module Taught"", module_code
AS ""Module Code"", course AS ""Course ID"", credit_hours AS ""Module Credit Hours"" FROM
module inner join teacher on module_id = module_taught WHERE teacher_id="+teacherid+"";
```

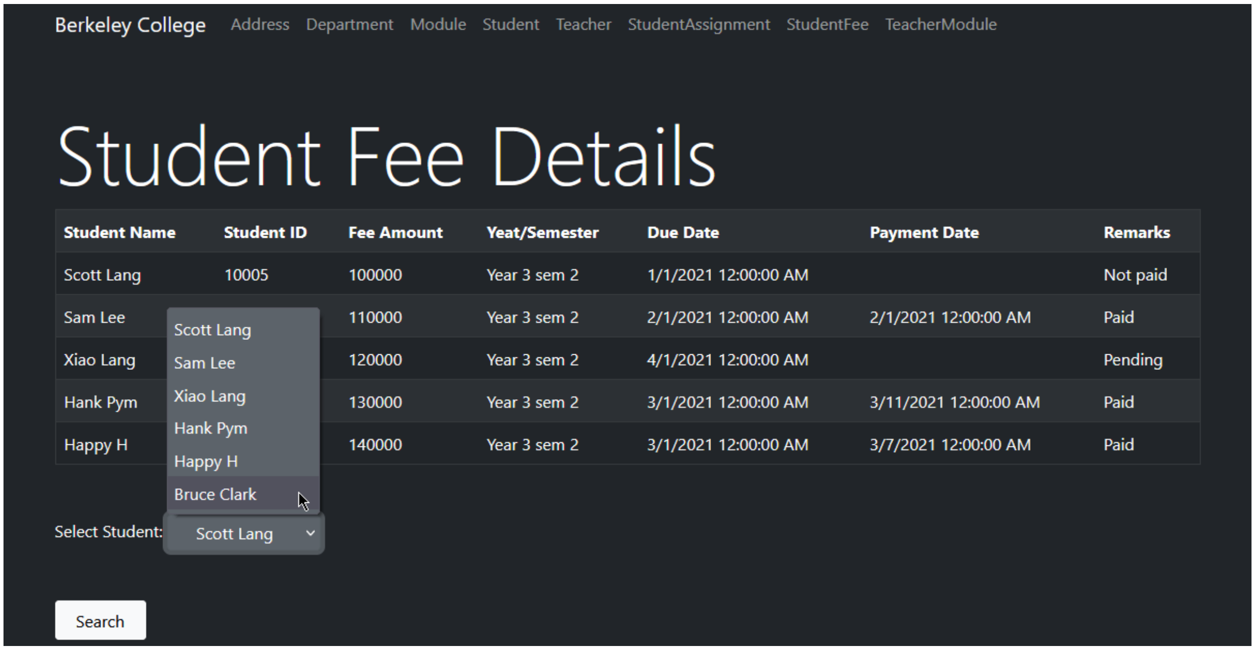
Complex Forms



Student Assignment Details

Student Name	Student ID	Module Name	Assignment Type	Grade Obtained	Status
Happy H	10009	Databases	Individual Coursework	F	Fail
Happy H	10009	Application Development	Individual Coursework	B+	PASS
Hank Pym	10008	Ethical Hacking	Viva	B	PASS
Hank Pym	10008	Networks	Written Examination	A	PASS
Xiao Lang	10007	3D Modelling	Individual Coursework	A	PASS
Xiao Lang	10007	Game Design	MCQ Exam	A+	PASS
Sam Lee		Accounting	Presentation	A	PASS
Sam Lee		Business	Group Coursework	A+	PASS
Scott Lang		Algorithms	Written Exam	A	PASS
Scott Lang		Mathematics	Individual Coursework	A	PASS

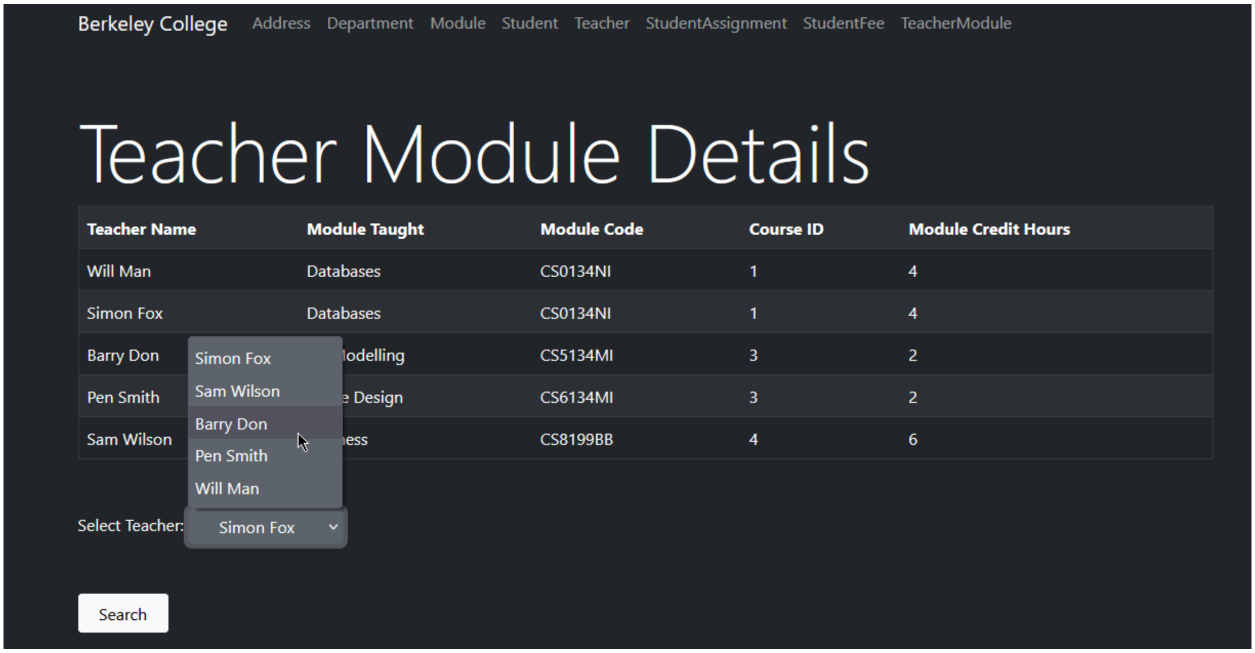
Figure 12: student-assignment complex form



Student Fee Details

Student Name	Student ID	Fee Amount	Yeat/Semester	Due Date	Payment Date	Remarks
Scott Lang	10005	100000	Year 3 sem 2	1/1/2021 12:00:00 AM		Not paid
Sam Lee		110000	Year 3 sem 2	2/1/2021 12:00:00 AM	2/1/2021 12:00:00 AM	Paid
Xiao Lang		120000	Year 3 sem 2	4/1/2021 12:00:00 AM		Pending
Hank Pym		130000	Year 3 sem 2	3/1/2021 12:00:00 AM	3/11/2021 12:00:00 AM	Paid
Happy H		140000	Year 3 sem 2	3/1/2021 12:00:00 AM	3/7/2021 12:00:00 AM	Paid

Figure 13: student-fee complex form



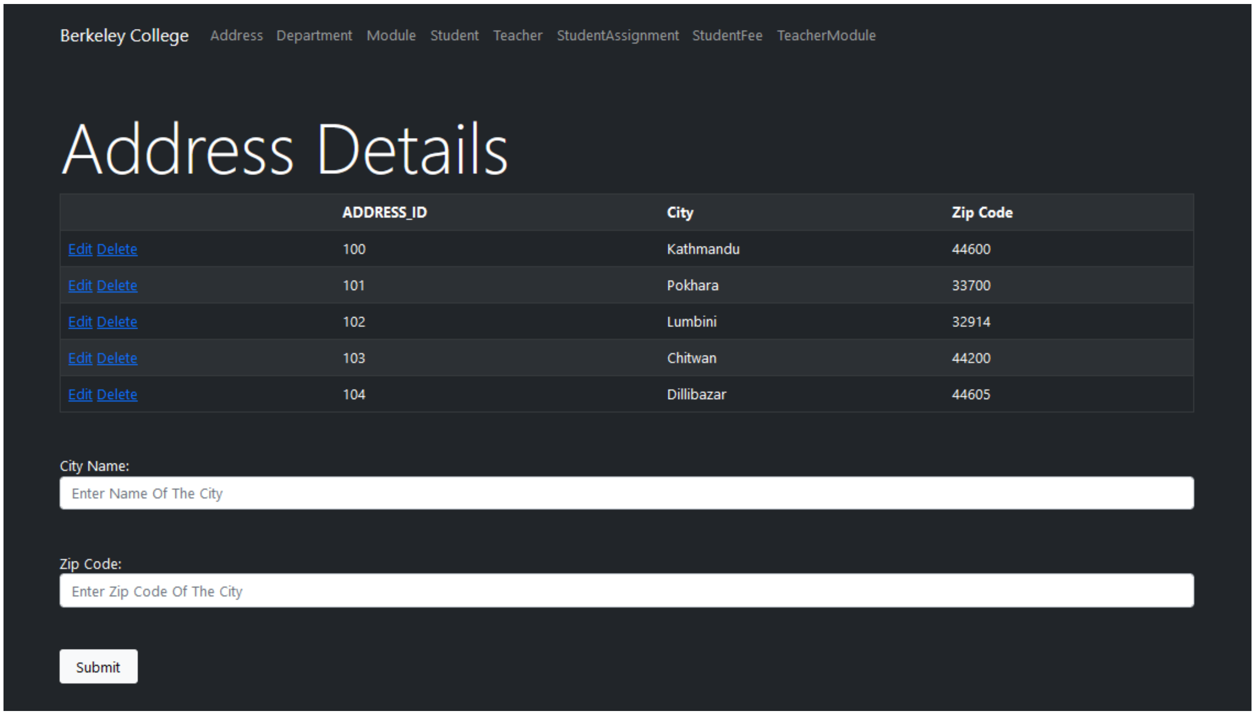
Teacher Module Details

Teacher Name	Module Taught	Module Code	Course ID	Module Credit Hours
Will Man	Databases	CS0134NI	1	4
Simon Fox	Databases	CS0134NI	1	4
Barry Don	lodelling	CS5134MI	3	2
Pen Smith	e Design	CS6134MI	3	2
Sam Wilson	ess	CS8199BB	4	6

Select Teacher: Simon Fox

Figure 14: teacher-module complex form

Simple Form



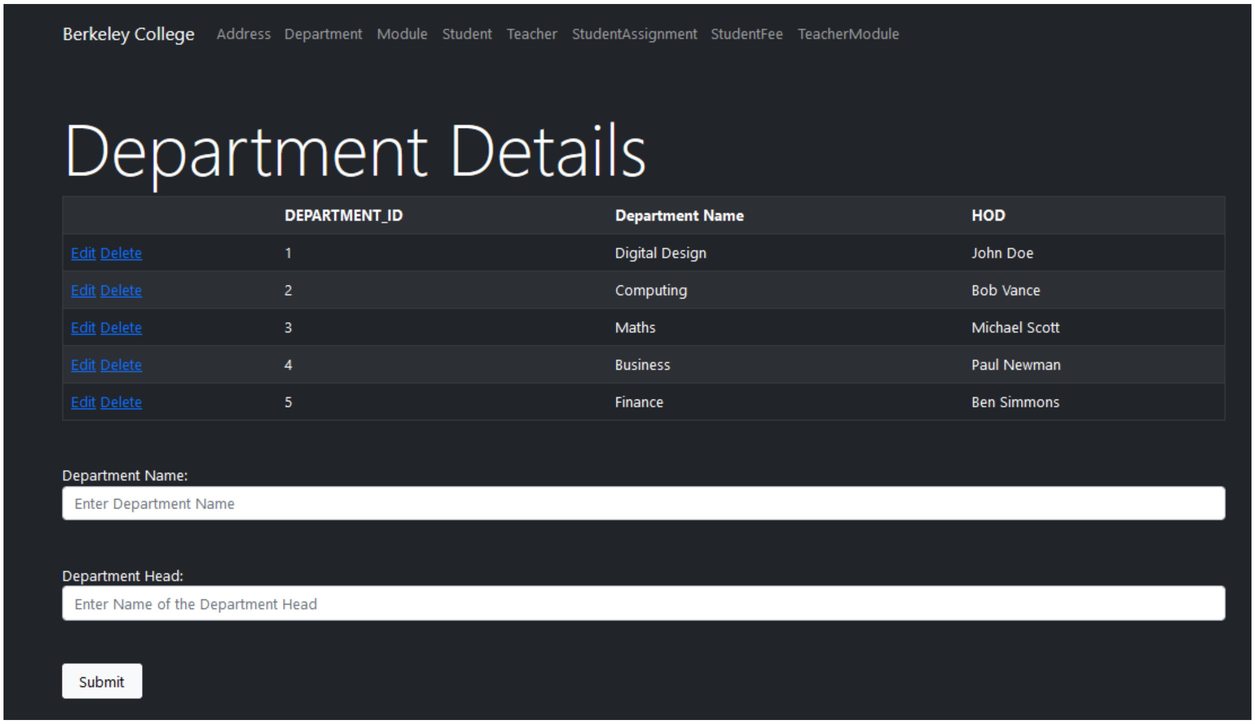
Address Details

	ADDRESS_ID	City	Zip Code
Edit Delete	100	Kathmandu	44600
Edit Delete	101	Pokhara	33700
Edit Delete	102	Lumbini	32914
Edit Delete	103	Chitwan	44200
Edit Delete	104	Dillibazar	44605

City Name: Enter Name Of The City

Zip Code: Enter Zip Code Of The City

Figure 15: address form



Department Details

	DEPARTMENT_ID	Department Name	HOD
Edit Delete	1	Digital Design	John Doe
Edit Delete	2	Computing	Bob Vance
Edit Delete	3	Maths	Michael Scott
Edit Delete	4	Business	Paul Newman
Edit Delete	5	Finance	Ben Simmons

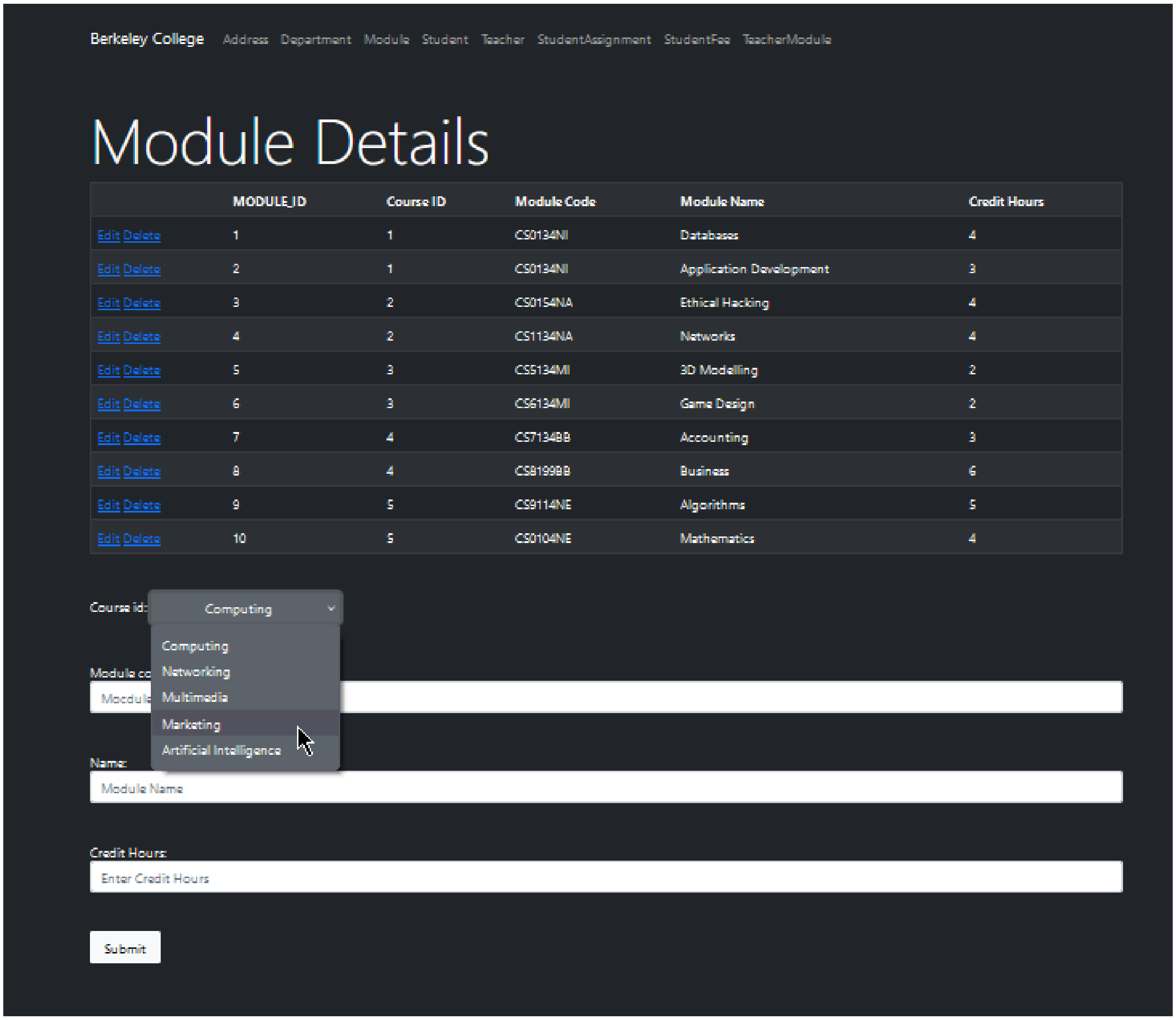
Department Name:

Enter Department Name

Department Head:

Enter Name of the Department Head

Figure 16: Department Form



Berkeley College Address Department Module Student Teacher StudentAssignment StudentFee TeacherModule

Module Details

	MODULE_ID	Course ID	Module Code	Module Name	Credit Hours
Edit Delete	1	1	CS0134NI	Databases	4
Edit Delete	2	1	CS0134NI	Application Development	3
Edit Delete	3	2	CS0154NA	Ethical Hacking	4
Edit Delete	4	2	CS1134NA	Networks	4
Edit Delete	5	3	CS5134MI	3D Modelling	2
Edit Delete	6	3	CS6134MI	Game Design	2
Edit Delete	7	4	CS7134BB	Accounting	3
Edit Delete	8	4	CS8199BB	Business	6
Edit Delete	9	5	CS9114NE	Algorithms	5
Edit Delete	10	5	CS0104NE	Mathematics	4

Course id: Computing

Module co

Module

Name:

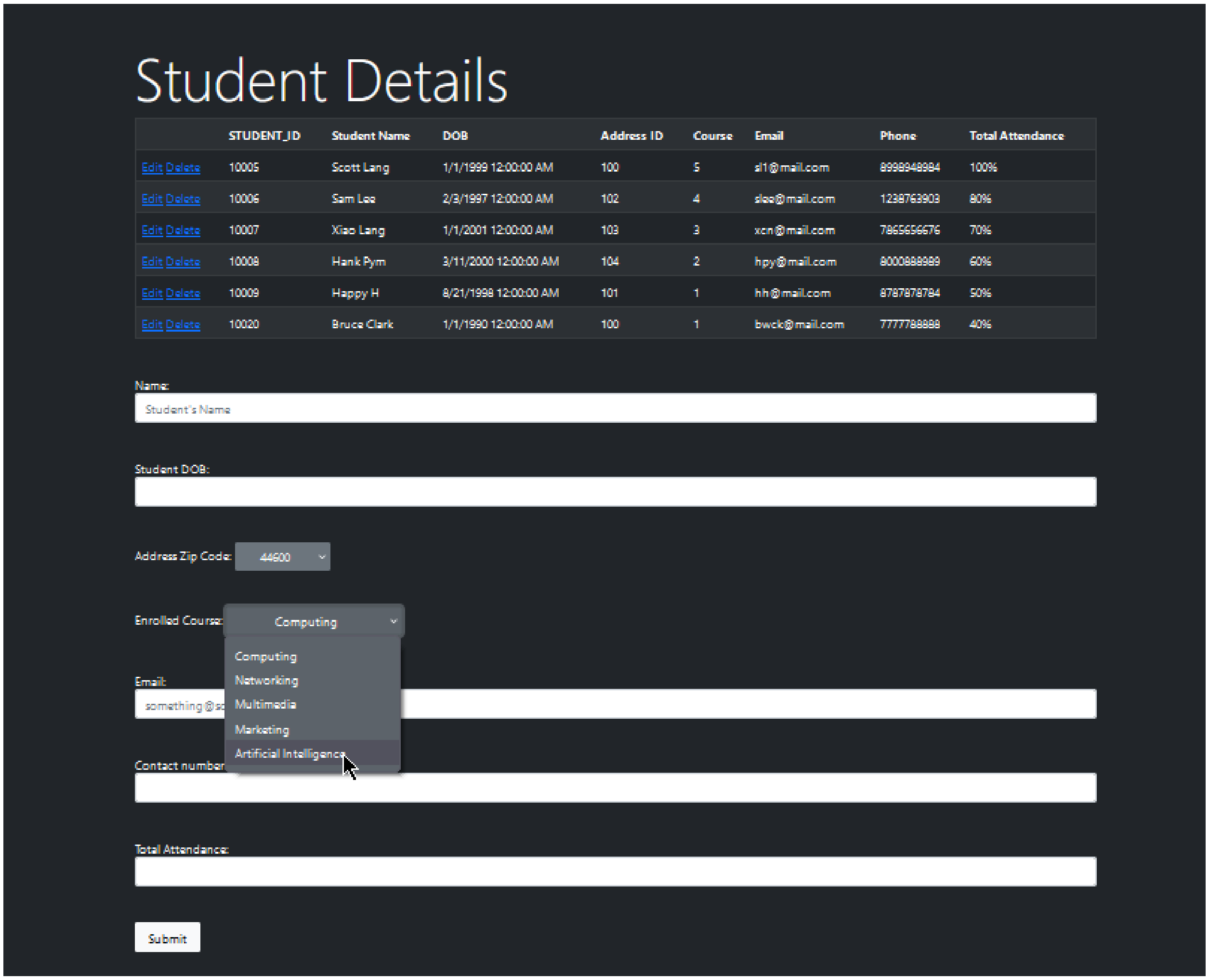
Module Name

Credit Hours:

Enter Credit Hours

Submit

Figure 17: module form



Student Details

	STUDENT_ID	Student Name	DOB	Address ID	Course	Email	Phone	Total Attendance
Edit Delete	10005	Scott Lang	1/1/1999 12:00:00 AM	100	5	sl1@mail.com	8998948984	100%
Edit Delete	10006	Sam Lee	2/3/1997 12:00:00 AM	102	4	slee@mail.com	1238763903	80%
Edit Delete	10007	Xiao Lang	1/1/2001 12:00:00 AM	103	3	xcn@mail.com	7865656676	70%
Edit Delete	10008	Hank Pym	3/11/2000 12:00:00 AM	104	2	hpy@mail.com	8000888989	60%
Edit Delete	10009	Happy H	8/21/1998 12:00:00 AM	101	1	hh@mail.com	8787878784	50%
Edit Delete	10020	Bruce Clark	1/1/1990 12:00:00 AM	100	1	bwck@mail.com	7777788888	40%

Name: Student's Name

Student DOB:

Address Zip Code: 44600

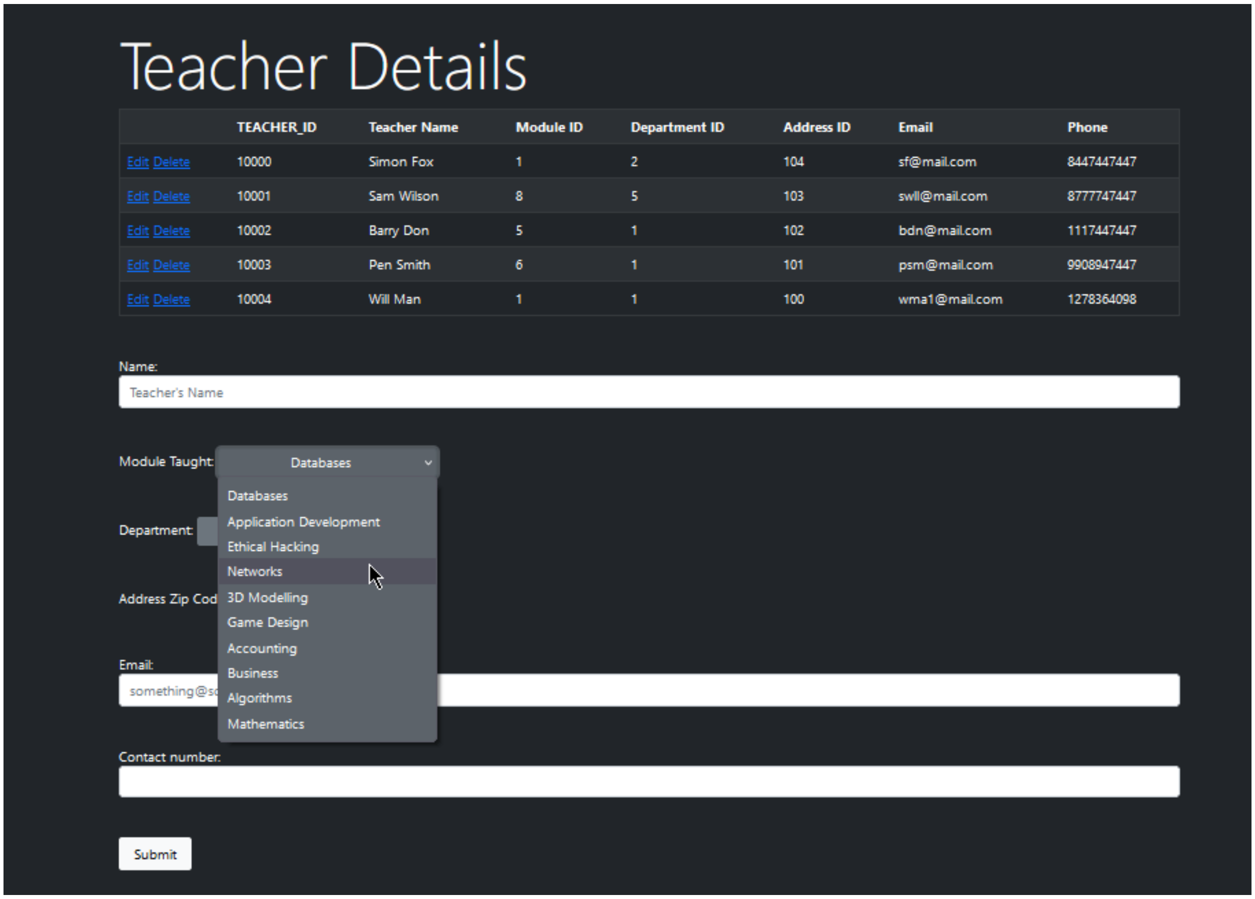
Enrolled Course: Computing

Email: something@so

Contact number

Total Attendance:

Figure 18: student form



Teacher Details

	TEACHER_ID	Teacher Name	Module ID	Department ID	Address ID	Email	Phone
Edit Delete	10000	Simon Fox	1	2	104	sf@mail.com	8447447447
Edit Delete	10001	Sam Wilson	8	5	103	swll@mail.com	8777747447
Edit Delete	10002	Barry Don	5	1	102	bdn@mail.com	1117447447
Edit Delete	10003	Pen Smith	6	1	101	psm@mail.com	9908947447
Edit Delete	10004	Will Man	1	1	100	wma1@mail.com	1278364098

Name: Teacher's Name

Module Taught: Databases

Department:

Address Zip Cod

Email: something@s

Contact number:

Figure 19: teacher form

User Manual

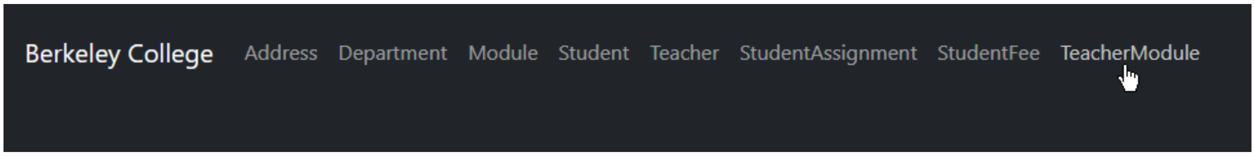
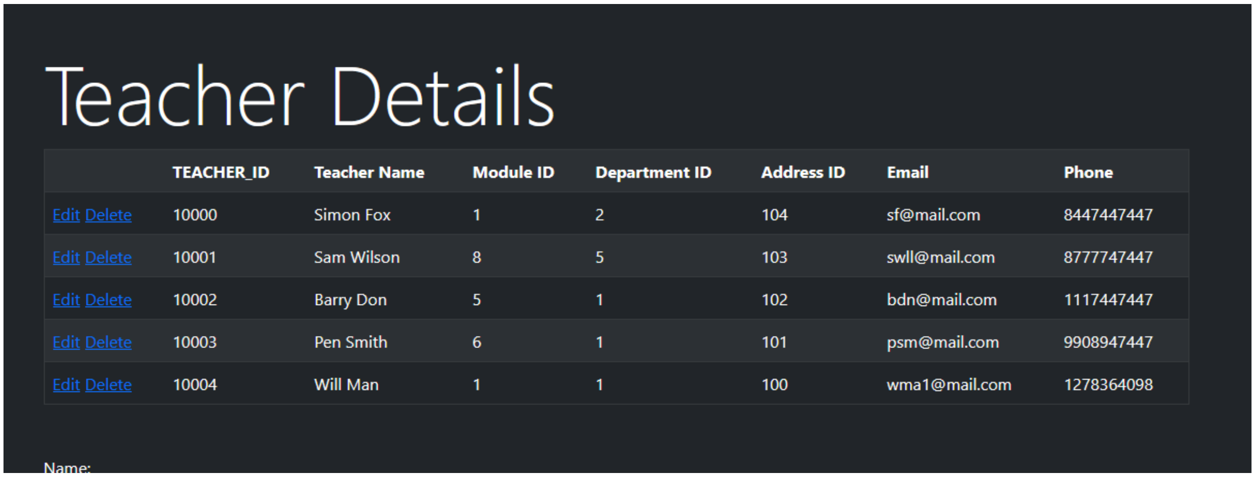


Figure 20: navigation

Each page contains navigation bar as shown above. Each link will take the user to its respective form.



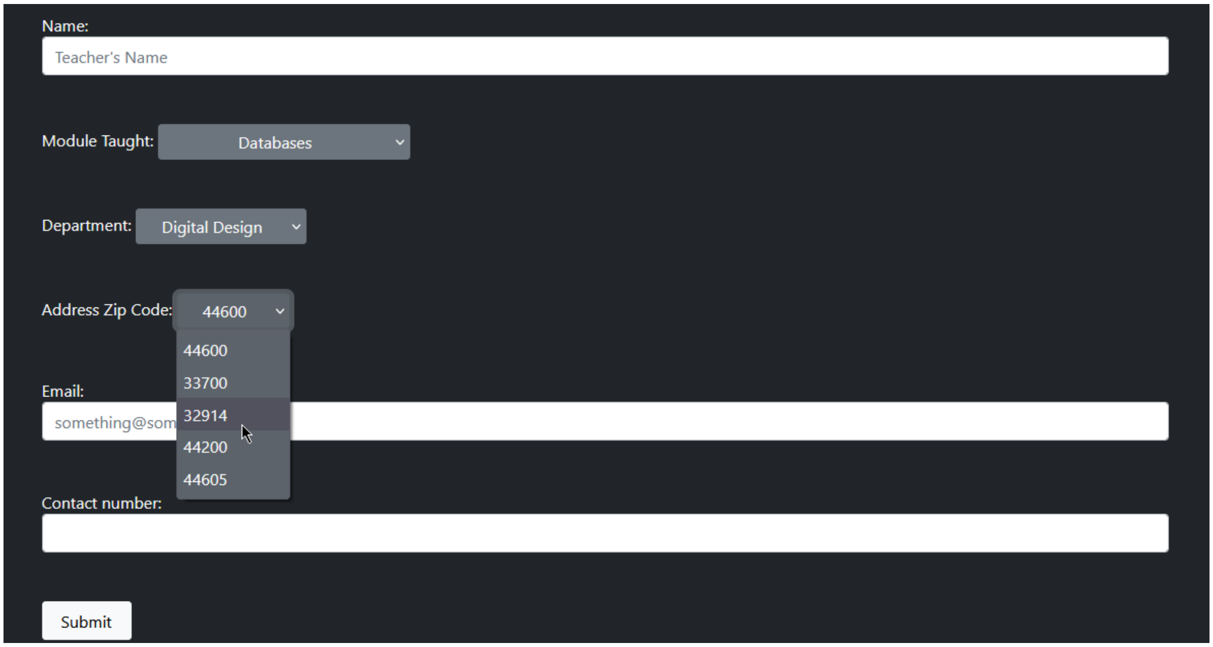
Teacher Details

	TEACHER_ID	Teacher Name	Module ID	Department ID	Address ID	Email	Phone
Edit Delete	10000	Simon Fox	1	2	104	sf@mail.com	8447447447
Edit Delete	10001	Sam Wilson	8	5	103	swll@mail.com	8777747447
Edit Delete	10002	Barry Don	5	1	102	bdn@mail.com	1117447447
Edit Delete	10003	Pen Smith	6	1	101	psm@mail.com	9908947447
Edit Delete	10004	Will Man	1	1	100	wma1@mail.com	1278364098

Name:

Figure 21: grid view

Each page (simple form) will contain a grid view where data is displayed



Name: Teacher's Name

Module Taught: Databases

Department: Digital Design

Address Zip Code: 44600

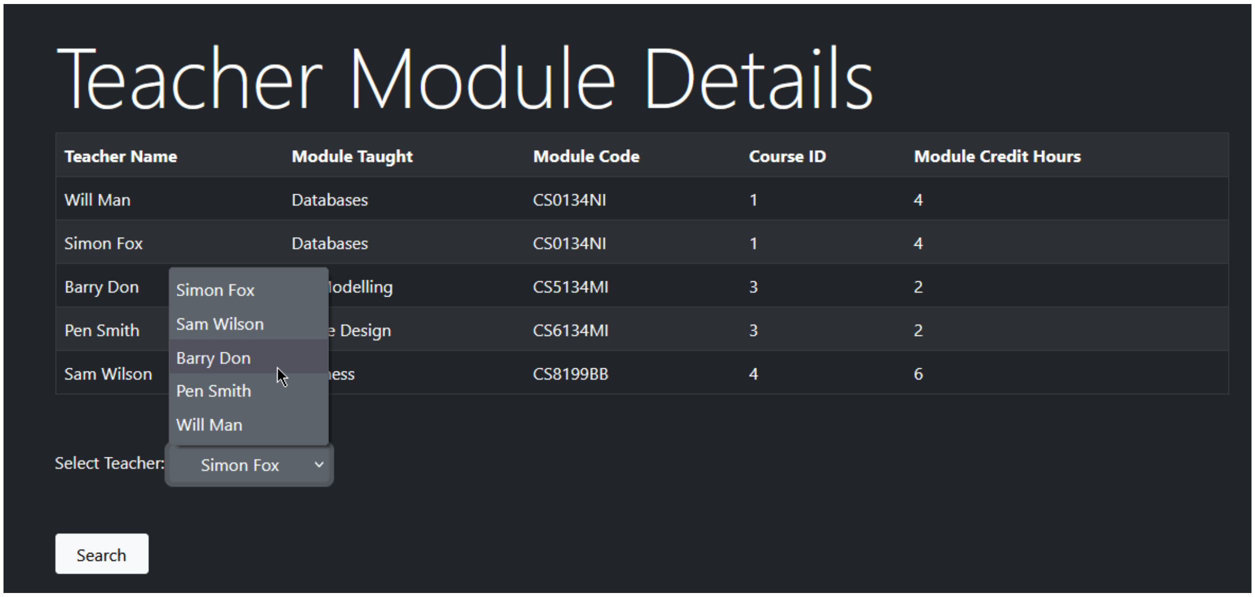
Email: something@som

Contact number:

Figure 22: input fields

Forms that support full CRUD operations will contain input fields which the users can use to insert data to the database or modify data. Submit button needs to be pressed to write data to database. See testing section.

Refer to data dictionary to get idea about what types of values to enter.



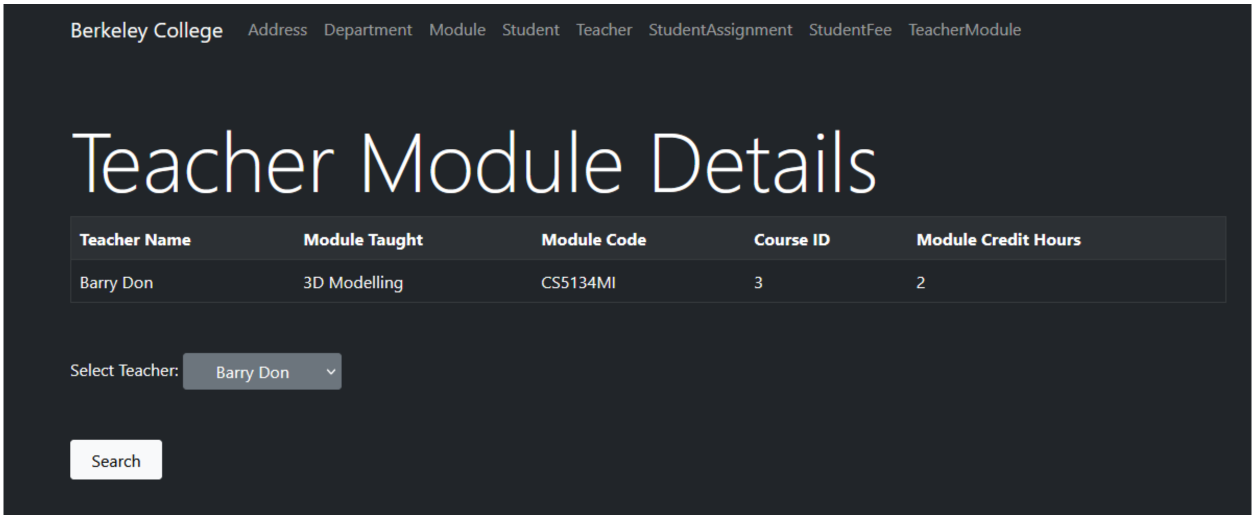
Teacher Module Details

Teacher Name	Module Taught	Module Code	Course ID	Module Credit Hours
Will Man	Databases	CS0134NI	1	4
Simon Fox	Databases	CS0134NI	1	4
Barry Don	lodelling	CS5134MI	3	2
Pen Smith	e Design	CS6134MI	3	2
Sam Wilson	ess	CS8199BB	4	6

Select Teacher: Simon Fox

Figure 23: search

Complex forms will contain a drop-down list and a search button to search for details about a specific person. It is demonstrated below.



Teacher Module Details

Teacher Name	Module Taught	Module Code	Course ID	Module Credit Hours
Barry Don	3D Modelling	CS5134MI	3	2

Select Teacher: Barry Don

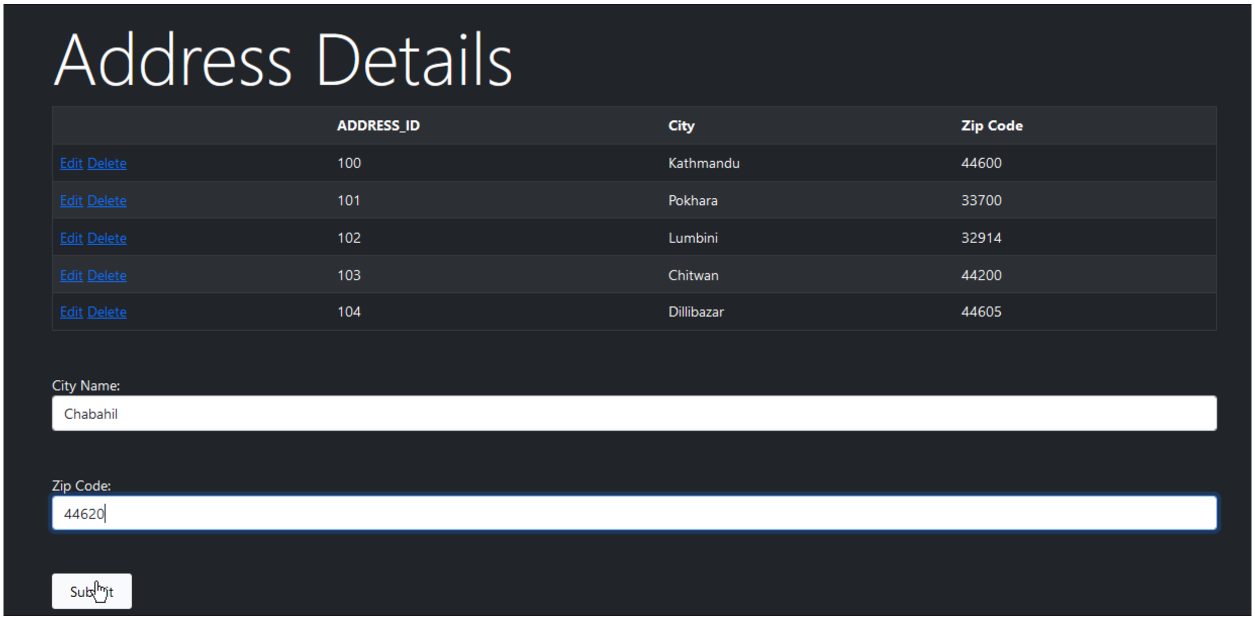
Figure 24: searching

Only details about Barry Don are displayed after pressing search button.

For further assistance or any confusion contact: np01cp4a190041@islingtoncollege.edu.np

Testing

Inserting new value



Address Details

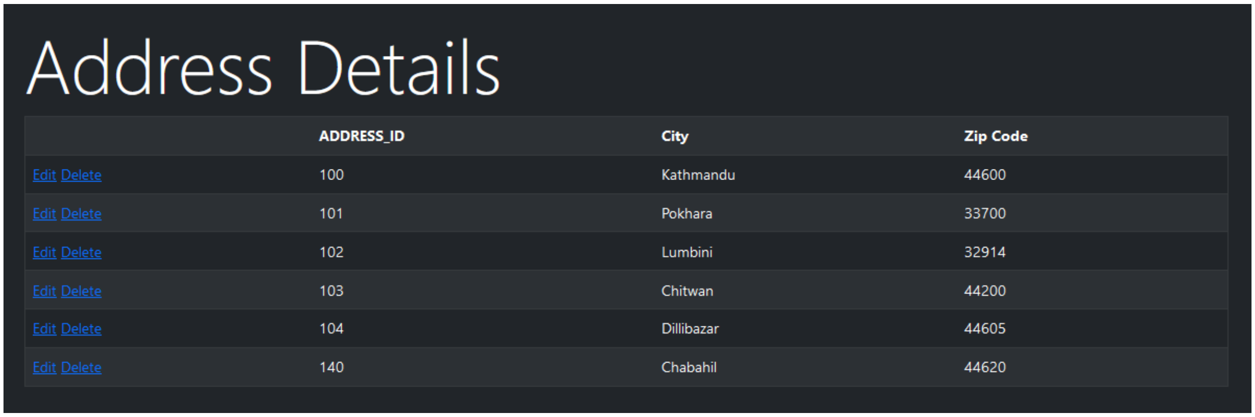
	ADDRESS_ID	City	Zip Code
Edit Delete	100	Kathmandu	44600
Edit Delete	101	Pokhara	33700
Edit Delete	102	Lumbini	32914
Edit Delete	103	Chitwan	44200
Edit Delete	104	Dillibazar	44605

City Name: Chabahil

Zip Code: 44620

Figure 25: inserting new address

Values can be inserted by adding values in input field and pressing submit.

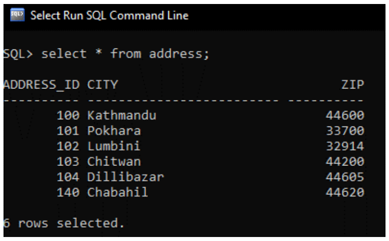


Address Details

	ADDRESS_ID	City	Zip Code
Edit Delete	100	Kathmandu	44600
Edit Delete	101	Pokhara	33700
Edit Delete	102	Lumbini	32914
Edit Delete	103	Chitwan	44200
Edit Delete	104	Dillibazar	44605
Edit Delete	140	Chabahil	44620

Figure 26: new value is added

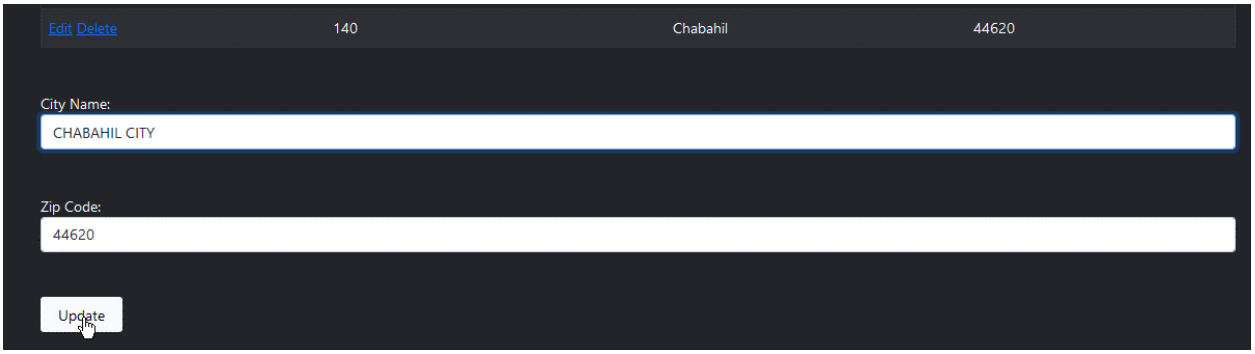
New value is added to database.



```
SQL> select * from address;

ADDRESS_ID CITY                              ZIP
---------- ------------------------- ----------
       100 Kathmandu                       44600
       101 Pokhara                         33700
       102 Lumbini                         32914
       103 Chitwan                         44200
       104 Dillibazar                      44605
       140 Chabahil                        44620

6 rows selected.
```

Figure 27:value added to database

Updating value



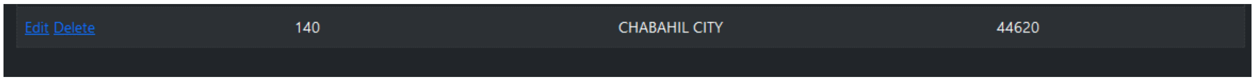
140 Chabahil 44620

City Name: CHABAHIL CITY

Zip Code: 44620

Figure 28: before update

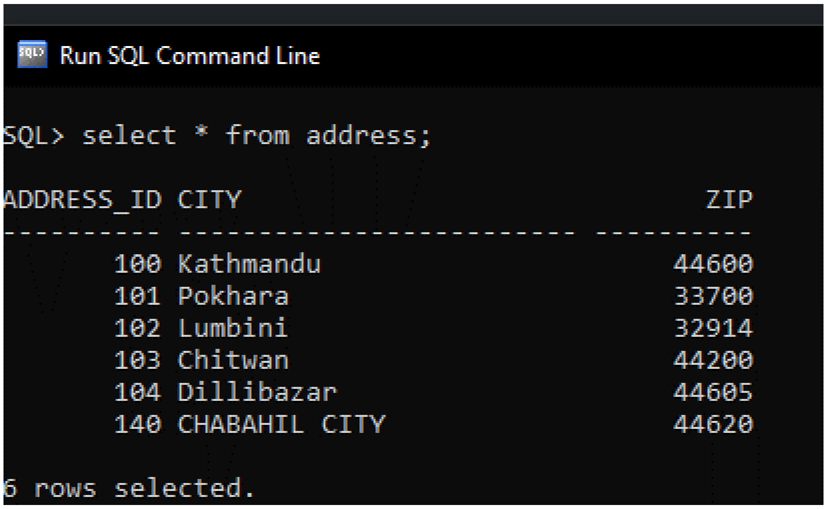
A row is selected, and changes are made.



140 CHABAHIL CITY 44620

Figure 29: After update

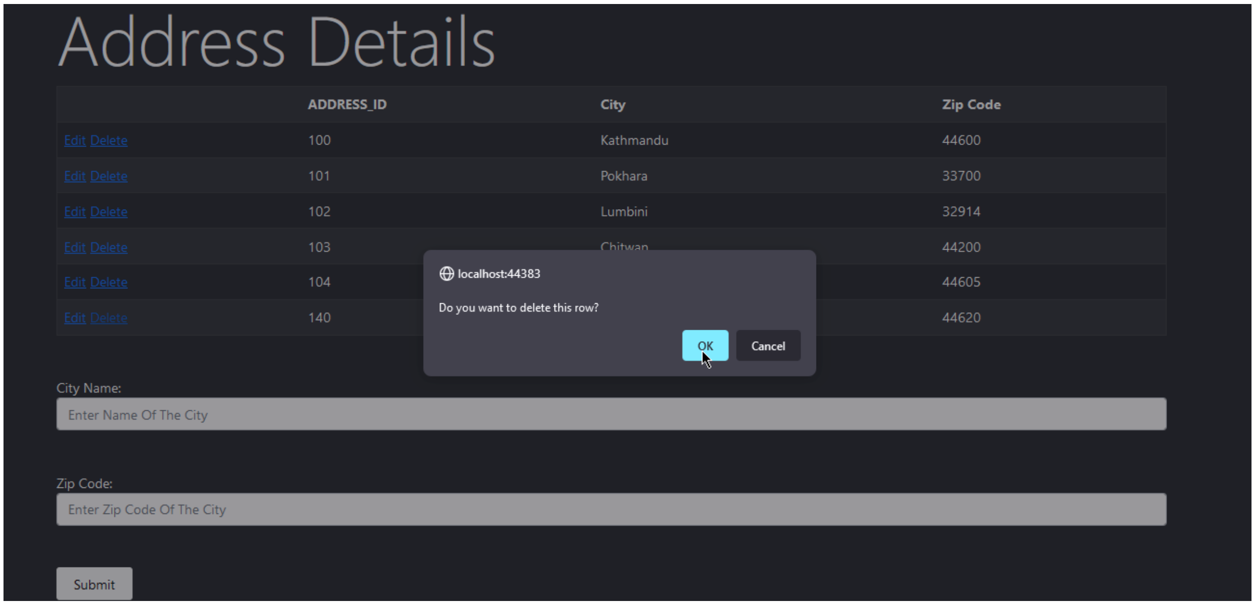
Row is updated and database is also updated



```
SQL> select * from address;

ADDRESS_ID CITY                            ZIP
---------- ------------------------- ----------
       100 Kathmandu                      44600
       101 Pokhara                        33700
       102 Lumbini                        32914
       103 Chitwan                        44200
       104 Dillibazar                     44605
       140 CHABAHIL CITY                  44620

6 rows selected.
```

Figure 30: value updated in database

Deleting value



Address Details

	ADDRESS_ID	City	Zip Code
Edit Delete	100	Kathmandu	44600
Edit Delete	101	Pokhara	33700
Edit Delete	102	Lumbini	32914
Edit Delete	103	Chitwan	44200
Edit Delete	104		44605
Edit Delete	140		44620

localhost:44383

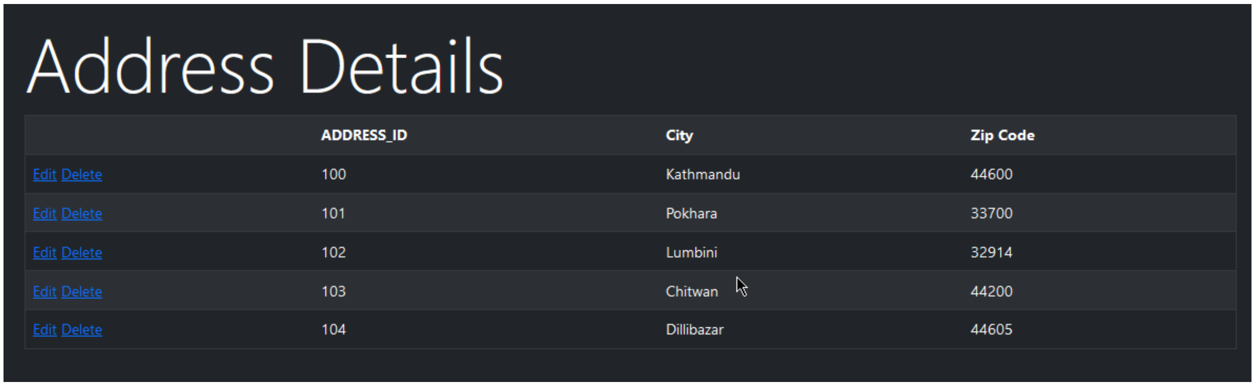
Do you want to delete this row?

City Name: Enter Name Of The City

Zip Code: Enter Zip Code Of The City

Figure 31: deleting data

A row is selected to be deleted

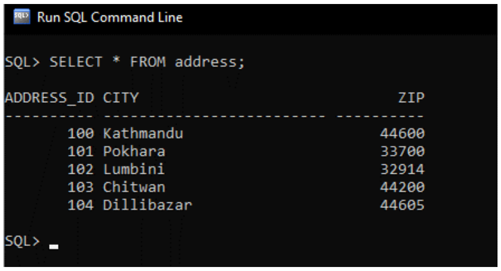


Address Details

	ADDRESS_ID	City	Zip Code
Edit Delete	100	Kathmandu	44600
Edit Delete	101	Pokhara	33700
Edit Delete	102	Lumbini	32914
Edit Delete	103	Chitwan	44200
Edit Delete	104	Dillibazar	44605

Figure 32: deleted

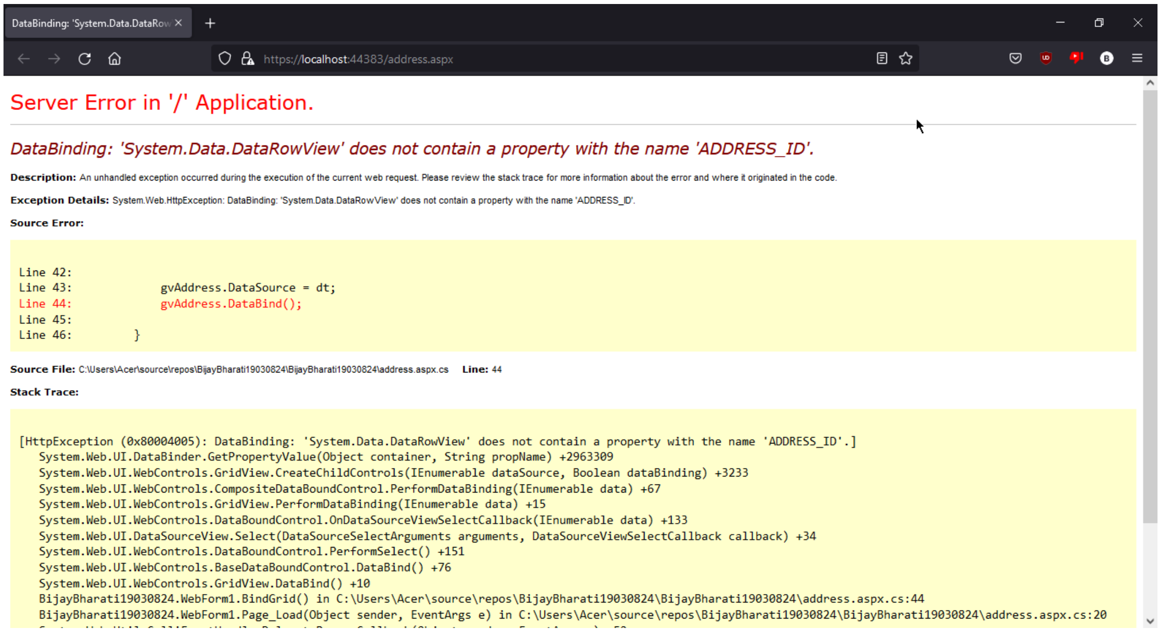
Value is deleted from database and grid view



```
SQL> SELECT * FROM address;

ADDRESS_ID CITY                           ZIP
---------- ------------------------ ---------
       100 Kathmandu                    44600
       101 Pokhara                      33700
       102 Lumbini                      32914
       103 Chitwan                      44200
       104 Dillibazar                   44605

SQL> _
```

Figure 33: value deleted from database

Incorrect query


DataBinding: 'System.Data.DataRow
https://localhost:44383/address.aspx

Server Error in '/' Application.

DataBinding: 'System.Data.DataRowView' does not contain a property with the name 'ADDRESS_ID'.

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.Web.HttpException: DataBinding: 'System.Data.DataRowView' does not contain a property with the name 'ADDRESS_ID'.

Source Error:

```
Line 42:
Line 43:             gvAddress.DataSource = dt;
Line 44:             gvAddress.DataBind();
Line 45:
Line 46:         }
```

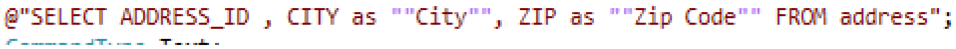
Source File: C:\Users\Acer\source\repos\BijayBharati19030824\BijayBharati19030824\address.aspx.cs **Line:** 44

Stack Trace:

```
[HttpException (0x80004005): DataBinding: 'System.Data.DataRowView' does not contain a property with the name 'ADDRESS_ID'.]
   System.Web.UI.DataBinder.GetPropertyValue(Object container, String propName) +2963309
   System.Web.UI.WebControls.GridView.CreateChildControls(IEnumerable dataSource, Boolean dataBinding) +3233
   System.Web.UI.WebControls.CompositeDataBoundControl.PerformDataBinding(IEnumerable data) +67
   System.Web.UI.WebControls.GridView.PerformDataBinding(IEnumerable data) +15
   System.Web.UI.WebControls.DataBoundControl.OnDataSourceViewSelectCallback(IEnumerable data) +133
   System.Web.UI.DataSourceView.Select(DataSourceSelectArguments arguments, DataSourceViewSelectCallback callback) +34
   System.Web.UI.WebControls.DataBoundControl.PerformSelect() +151
   System.Web.UI.WebControls.BaseDataBoundControl.DataBind() +76
   System.Web.UI.WebControls.GridView.DataBind() +10
   BijayBharati19030824.WebForm1.BindGrid() in C:\Users\Acer\source\repos\BijayBharati19030824\BijayBharati19030824\address.aspx.cs:44
   BijayBharati19030824.WebForm1.Page_Load(Object sender, EventArgs e) in C:\Users\Acer\source\repos\BijayBharati19030824\BijayBharati19030824\address.aspx.cs:20
```

Figure 34: query error

The error was caused because of incorrect formatted query, the error was solved by correcting the query.

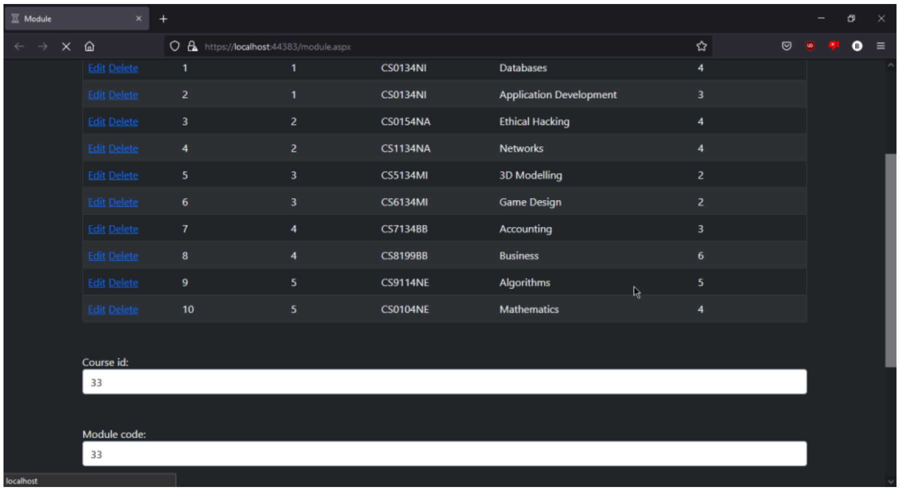


```
@"SELECT ADDRESS_ID , CITY as ""City"", ZIP as ""Zip Code"" FROM address";
```

Figure 35: correction

The query was corrected after which the application was running without any problems.

Integrity violation

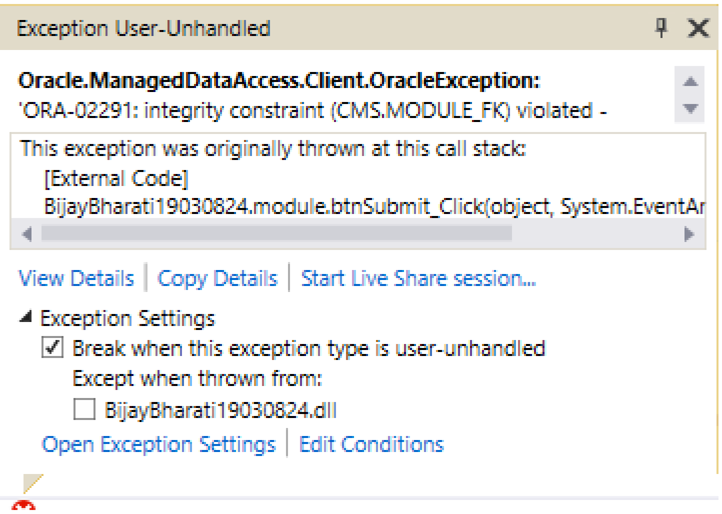


Edit Delete	1	1	CS0134NI	Databases	4
Edit Delete	2	1	CS0134NI	Application Development	3
Edit Delete	3	2	CS0154NA	Ethical Hacking	4
Edit Delete	4	2	CS1134NA	Networks	4
Edit Delete	5	3	CS5134MI	3D Modelling	2
Edit Delete	6	3	CS6134MI	Game Design	2
Edit Delete	7	4	CS7134BB	Accounting	3
Edit Delete	8	4	CS8199BB	Business	6
Edit Delete	9	5	CS9114NE	Algorithms	5
Edit Delete	10	5	CS0104NE	Mathematics	4

Course id: 33

Module code: 33

Figure 36: inserting invalid value for course



Exception User-Unhandled

Oracle.ManagedDataAccess.Client.OracleException: 'ORA-02291: integrity constraint (CMS.MODULE_FK) violated -

```
This exception was originally thrown at this call stack:
    [External Code]
    BijayBharati19030824.module.btnSubmit_Click(object, System.EventAr
```

Exception Settings

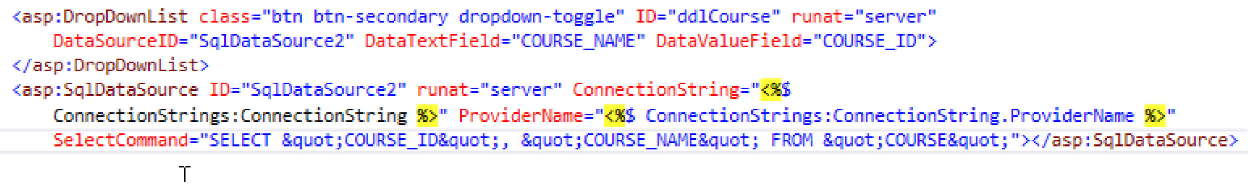
☑ Break when this exception type is user-unhandled

Except when thrown from:

□ BijayBharati19030824.dll

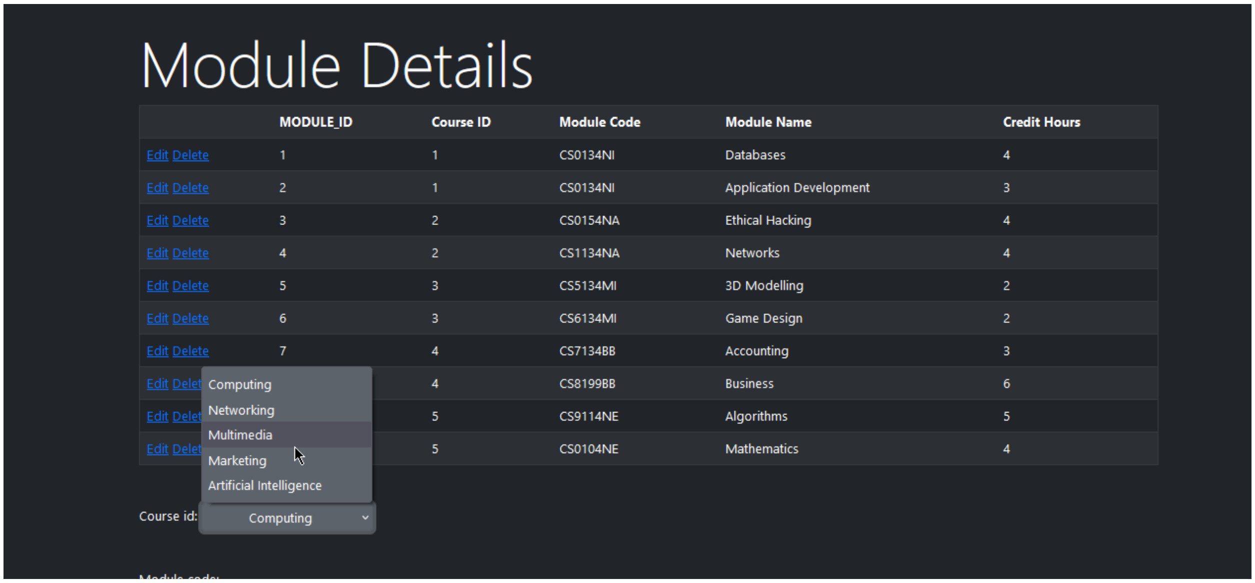
Figure 37: integrity violation

The following error was caused when using an invalid value for foreign key. The error was corrected by implementing dropdown list for all fields that are foreign keys so that only proper values can be entered by the users.



```
<asp:DropDownList class="btn btn-secondary dropdown-toggle" ID="ddlCourse" runat="server"
    DataSourceID="SqlDataSource2" DataTextField="COURSE_NAME" DataValueField="COURSE_ID">
</asp:DropDownList>
<asp:SqlDataSource ID="SqlDataSource2" runat="server" ConnectionString="<%$
    ConnectionStrings:ConnectionString %>" ProviderName="<%$ ConnectionStrings:ConnectionString.ProviderName %>"
    SelectCommand="SELECT &quot;COURSE_ID&quot;, &quot;COURSE_NAME&quot; FROM &quot;COURSE&quot;"></asp:SqlDataSource>
```

Figure 38: implementing drop down list



Module Details

	MODULE_ID	Course ID	Module Code	Module Name	Credit Hours
Edit Delete	1	1	CS0134NI	Databases	4
Edit Delete	2	1	CS0134NI	Application Development	3
Edit Delete	3	2	CS0154NA	Ethical Hacking	4
Edit Delete	4	2	CS1134NA	Networks	4
Edit Delete	5	3	CS5134MI	3D Modelling	2
Edit Delete	6	3	CS6134MI	Game Design	2
Edit Delete	7	4	CS7134BB	Accounting	3
Edit Delet		4	CS8199BB	Business	6
Edit Delet		5	CS9114NE	Algorithms	5
Edit Delet		5	CS0104NE	Mathematics	4

Course id: Computing

Module code:

Figure 39: drop down list

The drop-down list is implemented after which there is no possibility of encountering the error.

Further Discussion

This coursework has helped me develop my understanding on databases and improved my web development and .NET skills. This coursework has also helped me be familiar with tools like SQL developer, Microsoft Visual Studio and Oracle Data Modeller. Through the coursework, I learnt how to connect database to our application and perform CRUD operations. This was also a great opportunity to refresh my knowledge on SQL queries.

The coursework was very insightful, and research done to complete the coursework and problems encountered helped me learn and improve what I know about Oracle and ASP .NET.