Kingdom of Saudi Arabia Royal Commission at Yanbu Colleges & Institutes Division Yanbu University College Computer Science & Engineering Dept Information & Computer Technology De		المـمــــــــلكـــــة العربية الســــعودية بـينبع الـــــهـــــــــيـــئة الملــــكـــــية قـــطــاع الـكـــــــــلـيات والـمـعــاهـد كـــــية ينبع الجــــــــــامــــــــــعيةقسم علوم وهندسة الحاسب الألي قسم تقنية المعلومات والحاسب الآلي

PROJECT _____

ACADEMIC YEAR 1441/1442 H (2020/2021 G), SEMESTER II (202)

FUNDAMENTALS OF DATABASE SYSTEMS CS 311

DATE:	Sunday, January 31, 2021	**START TIME:**	**Week 03**
		FINISH TIME:	**Week 15**

STUDENT NAME:	Aya Alharbi _		
STUDENT ID:	3 8 2 0 1 2 3	**SECTION:**	1

FOR INSTRUCTOR USE ONLY					GENERAL INSTRUCTIONS
Q. No.	**CLOs**	**PLO**	**MAX MARK**	**MARKS OBTAINED**	
Phase I	**1.02 1.03 2.01 2.02**	**CE A,CS A CE B,CS B,C CE B,C,E,I CS B,C**	**20**		
Phase II	**2.02 3.01**	**CE B,C,E CS C,F**	**10**		
Phase III	**3.01**	**CE B,C,I CS C**	**10**		
Phase IV	**4.01**	**CE K CS D,F**	**10**		
Phase V	**3.02**	**CS D**	**05**		
TOTAL MARKS			**55**		

MARKED BY:	**Signature:**
CHECKED BY:	**Signature:**

Student management system

Prepared by: Dr.Kajal Nusratullah

Description

The Student management system is to automate the functionalities of a college or university. The employee of the Admission and Registration Department will be able to store, view or update data and information about students and staff easily.

The system can be used to store student information like student's name, national id, student ID, phone number, email, and specialty (department). The employee will be able to use this system to register a new student. Employees can also check course details such as course name, course code, for which department, and prerequisites. The employee will be able to view sections for each course and instructor id, lecturer id, and student id. Moreover, the Employee shall be able to view the day, time, and room for each section for a section(lecture). The Employee will be also able to add a new employee to the system and can check the employee details like name, employee id, phone number, email, and department.

The student management system will store and manage all the data and information for the students, courses, and faculty. It will provide the easiest and efficient way to maintain the functionality related to the staff and the students of the college.

The employee will be able to retrieve any information related to any department such as department name, department code, the head of the department.

Business Rules:

- Instructor email should be end with (___@rcyci.edu.sa)
- Student email should be end with (___@stu.rcyci.edu.sa)
- Department code should be one of ("CSE", "MS", "ID", "AL")),

· The courses should be in one of these department (Computer Science and Engineering Department, Management Science Department, Interior Design Department, Applied Linguistics Department)

Entities and Attributes:

Student						
Student_id	Name	National_id	Phone_no	Email	Dep_code	Dep_name

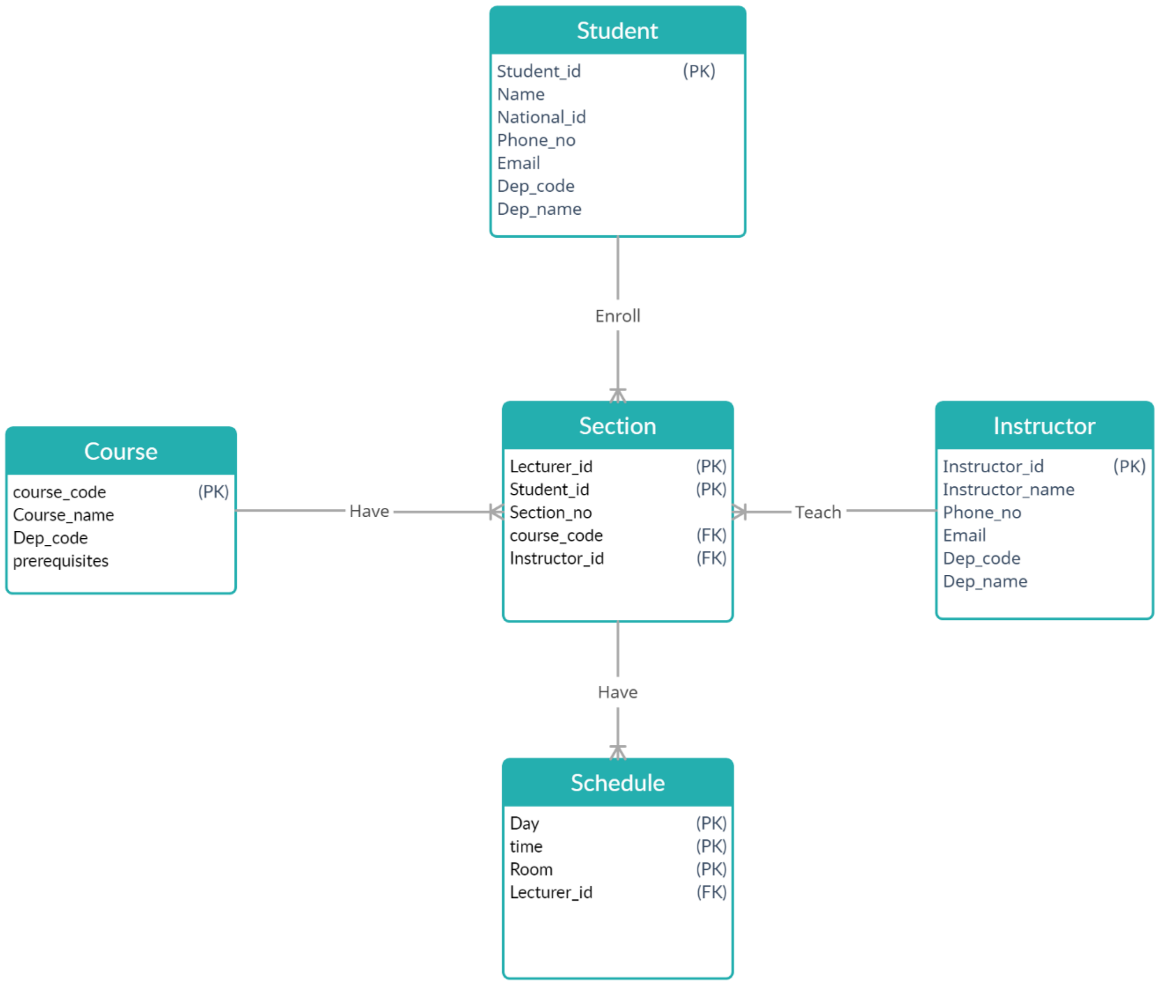
Course			
course_code	Course_name	Dep_code	prerequisites

Section				
Lecture_id	**Student_id**	course_code	Instructor_id	Section_no

Schedule			
Day	**time**	**Room**	Lecture_id

Instructor					
Instructor_id	Instructor_name	Phone_no	Email	Dep_code	Dep_name

Initial ERD:



Database Normalization

Database normalization is a process used to organize a database into tables and columns. The main idea with this is that a table should be about a specific topic and only supporting topics included.

Reasons for Database Normalization

Prepared by: Dr.Kajal Nusratullah

- Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate the undesirable characteristics like Insertion, Update and Deletion Anomalies.
- Normalization divides the larger table into the smaller table and links them using relationship.
- The normal form is used to reduce redundancy from the database table.

Conversion to First Normal Form:

- Step 1: Eliminate the Repeating Groups
- Step 2: Identify the Primary Key
- Step 3: Identify All Dependencies

Conversion to Second Normal Form:

- Step 1: Make New Tables to Eliminate Partial Dependencies
- Step 2: Reassign Corresponding Dependent Attributes

Conversion to Third Normal Form:

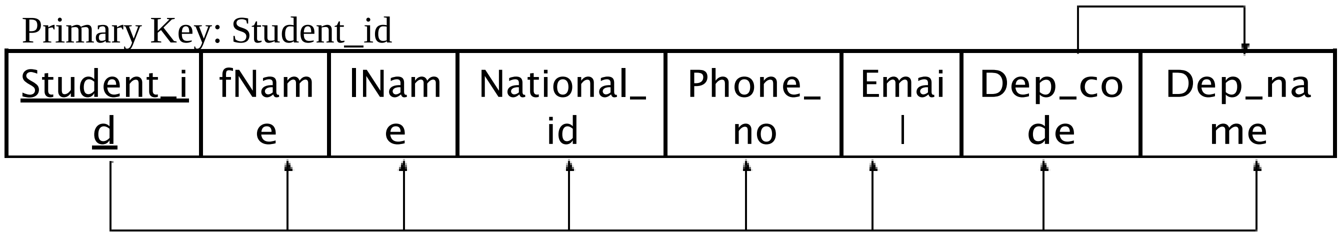
- Step 1: Make New Tables to Eliminate Transitive Dependencies
- Step 2: Reassign Corresponding Dependent Attributes

Student Table

Student_id	fName	lName	National_id	Phone_no	Email	Dep_code	Dep_name

1NF

Primary Key: Student_id



Student_id	fName	lName	National_id	Phone_no	Email	Dep_code	Dep_name

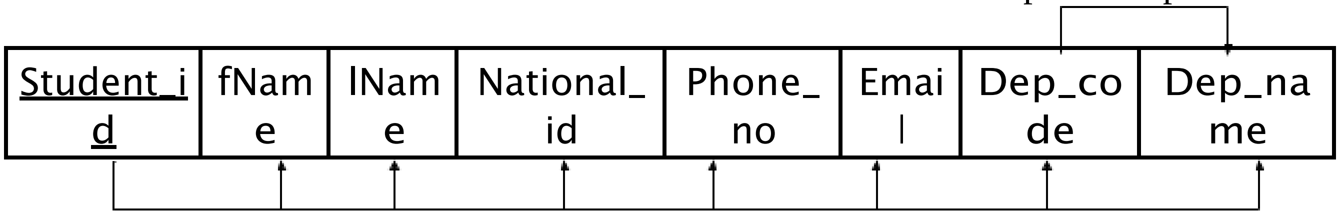
(Student_id => fName, lName, National_id, Phone_no, Email, Dep_code, Dep_name)

TRANSITIVE DEPENDCIES:

(Dep_code => Dep_name)

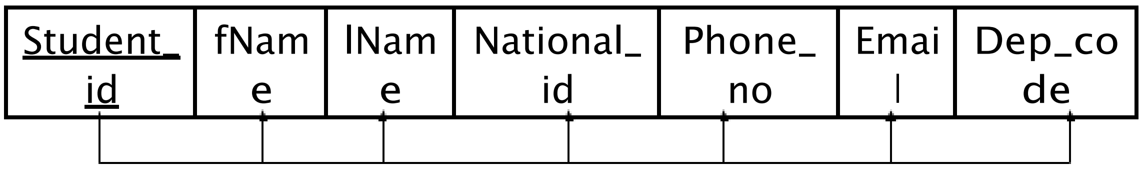
Prepared by: Dr.Kajal Nusratullah

2NF: It is in second normal form since it is in 1NF and includes no partial dependencies

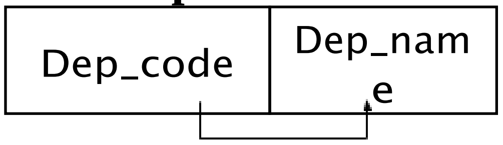


3NF

Student Table



Department Table



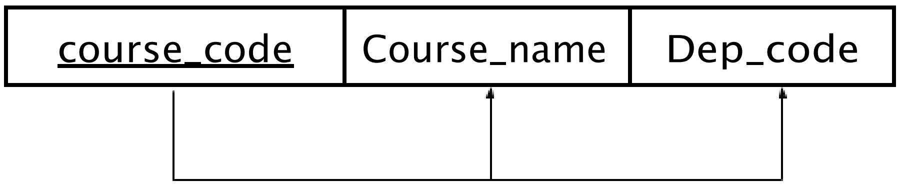
Course Table

<u>course_code</u>	Course_name	Dep_code	prerequisites

1NF

This table is holding repeating groups of data (if a course has two courses as prerequisites, we will have course_code, Course_name and Dep_code repeated in two records). Therefore, Identity tables and fields that will hold this data without the repeating groups which means taking the primary key and the prerequisites column in a separate table.

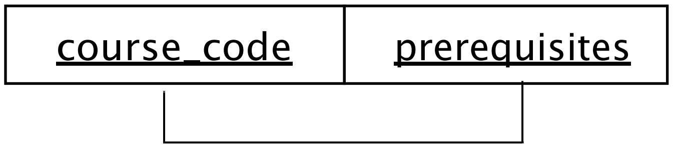
Primary Key: course_code



(course_code => Course_name, Dep_code)

Prerequisite Table

Primary Key: course_code and prerequisites



course_code	prerequisites

(course_code, prerequisites)

2NF: It is in second normal form since it is in 1NF and includes no partial dependencies

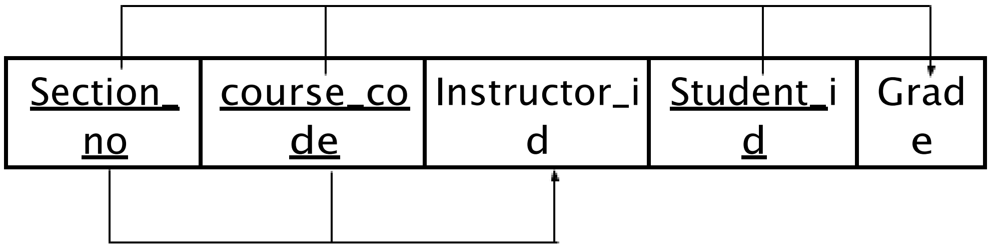
3NF: It is in third normal form since it is in 2NF and it contains no transitive dependencies

Section Table

Primary Key: Section_no, course_code, Student_id

Section_no	course_code	Instructor_id	Student_id	Grade

1NF



Section_no	course_code	Instructor_id	Student_id	Grade

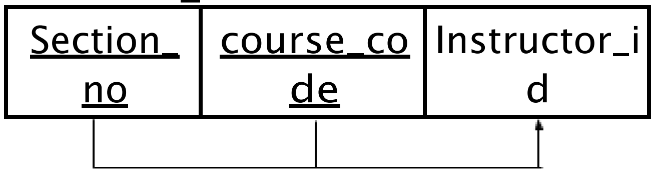
Partial Dependencies:

(Section_no, course_code => Instructor_id)

(Section_no, course_code, Student_id => Grade)

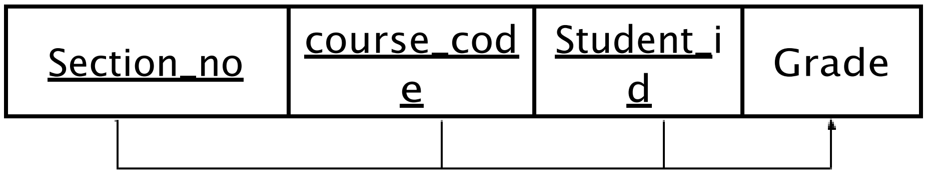
2NF

Instructor_Section Table



Section_no	course_code	Instructor_id

Grade Table



Section_no	course_code	Student_id	Grade

3NF: It is in third normal form since:

– It is in 2NF

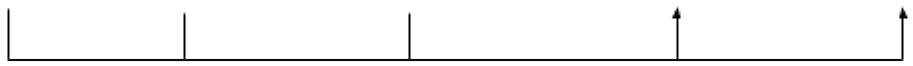
– It contains no transitive dependencies

Schedule Table

Day	time	Room	Section_no	course_code

1NF
Day, time and room should be unique and not null for each (section and course).
Therefore, Primary Key: Day, time, room

Day	time	Room	Section_no	course_code



2NF: It is in second normal form since it is in 1NF and includes no partial dependencies
3NF: It is in third normal form since it is in 2NF and it contains no transitive dependencies.

Schedule Table

Day	time	Room	Section_no	course_code

1NF
Day, time and room should be unique and not null for each (section and course).
Therefore, Primary Key: Day, time, room

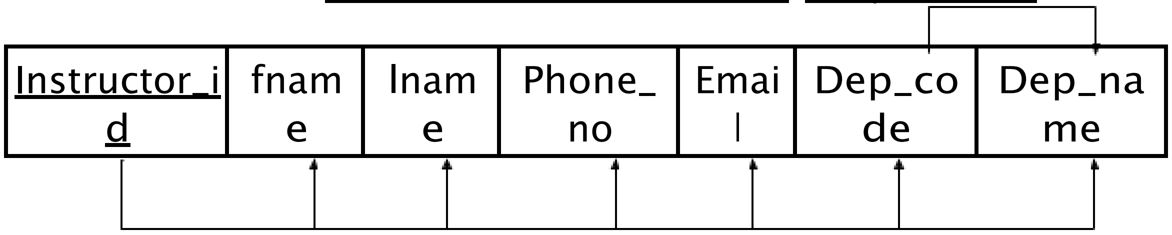
Day	time	Room	Section_no	course_code

2NF: It is in second normal form since it is in 1NF and includes no partial dependencies
3NF: It is in third normal form since it is in 2NF and it contains no transitive dependencies.

Instructor Table

Instructor_id	fname	lname	Phone_no	Email	Dep_code	Dep_name

1NF
Primary Key: Instructor_id



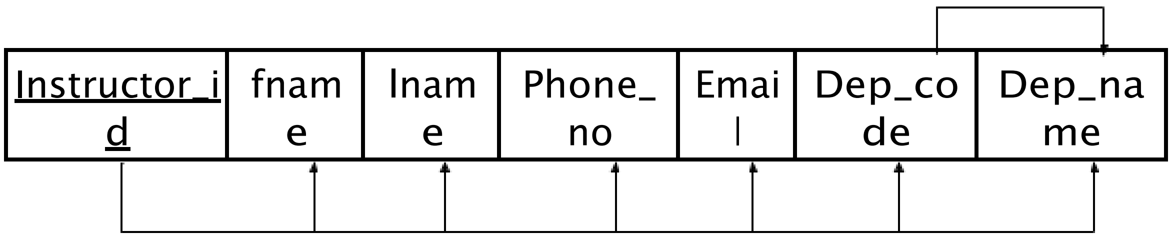
<u>Instructor_id</u>	fname	lname	Phone_no	Email	Dep_code	Dep_name

(<u>Instructor_id</u> => Instructor_name, Phone_no, Email, Dep_code, Dep_name)

TRANSITIVE DEPENDCIES:

(Dep_code => Dep_name)

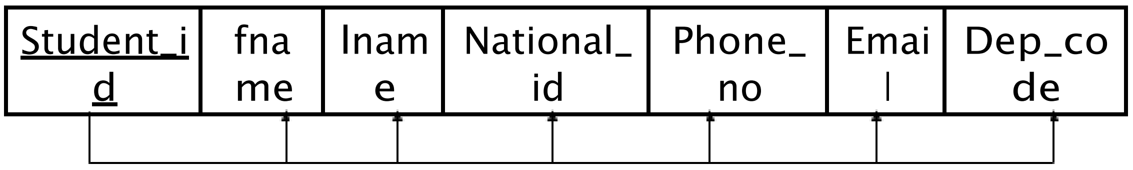
2NF: It is in second normal form since it is in 1NF and includes no partial dependencies.



<u>Instructor_id</u>	fname	lname	Phone_no	Email	Dep_code	Dep_name

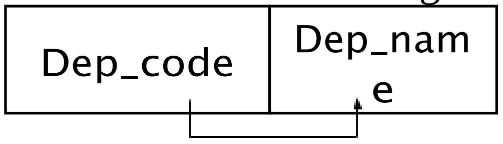
3NF

Instructor Table



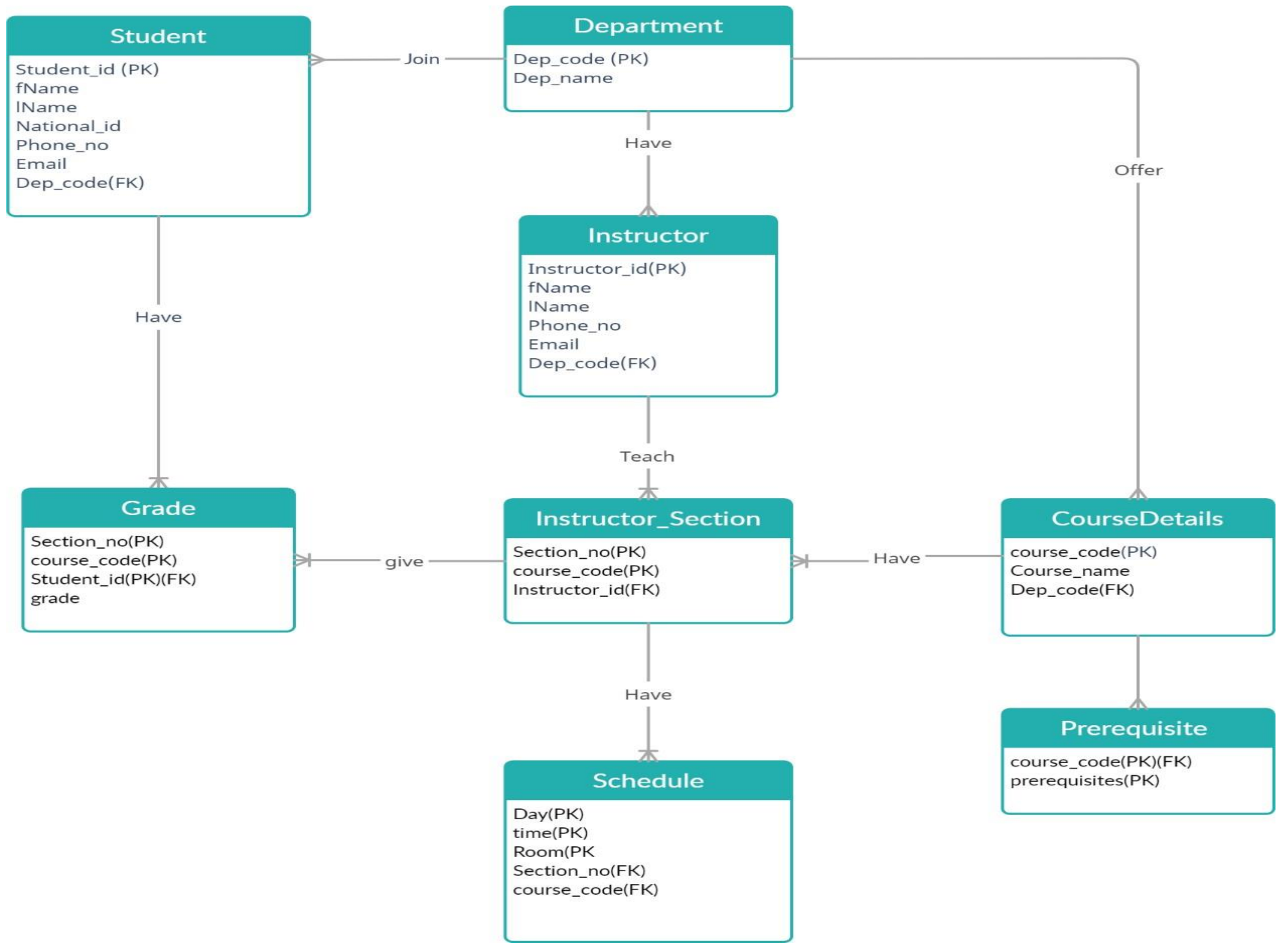
<u>Student_id</u>	fname	lname	National_id	Phone_no	Email	Dep_code

Department Table: it is the same table created while normalization table student since it is having the same attributes.

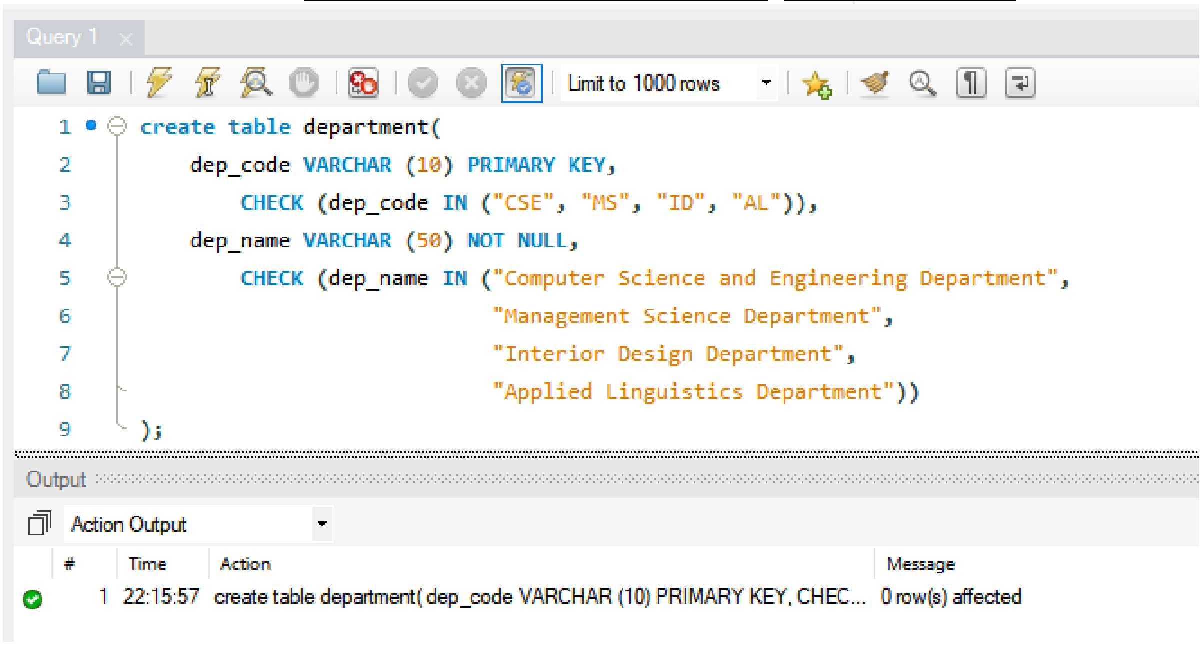


Dep_code	Dep_name

Final ERD



Department Table

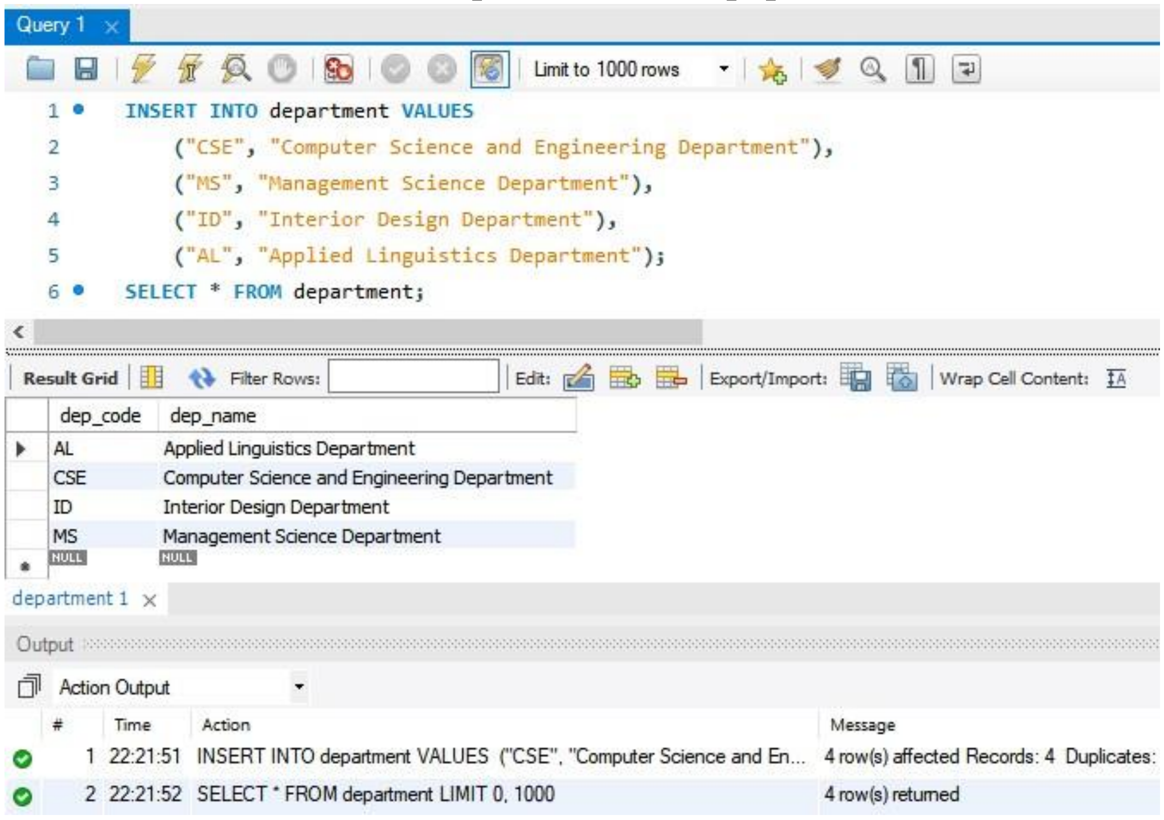


```
create table department(
    dep_code VARCHAR (10) PRIMARY KEY,
        CHECK (dep_code IN ("CSE", "MS", "ID", "AL")),
    dep_name VARCHAR (50) NOT NULL,
        CHECK (dep_name IN ("Computer Science and Engineering Department",
                            "Management Science Department",
                            "Interior Design Department",
                            "Applied Linguistics Department"))
);
```

```
#   Time      Action                                                          Message
1   22:15:57  create table department( dep_code VARCHAR (10) PRIMARY KEY, CHEC...  0 row(s) affected
```

```
CREATE TABLE department(
        dep_code VARCHAR (10) PRIMARY KEY,
                CHECK (dep_code IN ("CSE", "MS", "ID", "AL")),
        dep_name VARCHAR (50) NOT NULL,
                CHECK (dep_name IN ("Computer Science and Engineering Department",
                                    "Management Science Department",
                                    "Interior Design Department",
                                    "Applied Linguistics Department"))
);
```

Department Table population



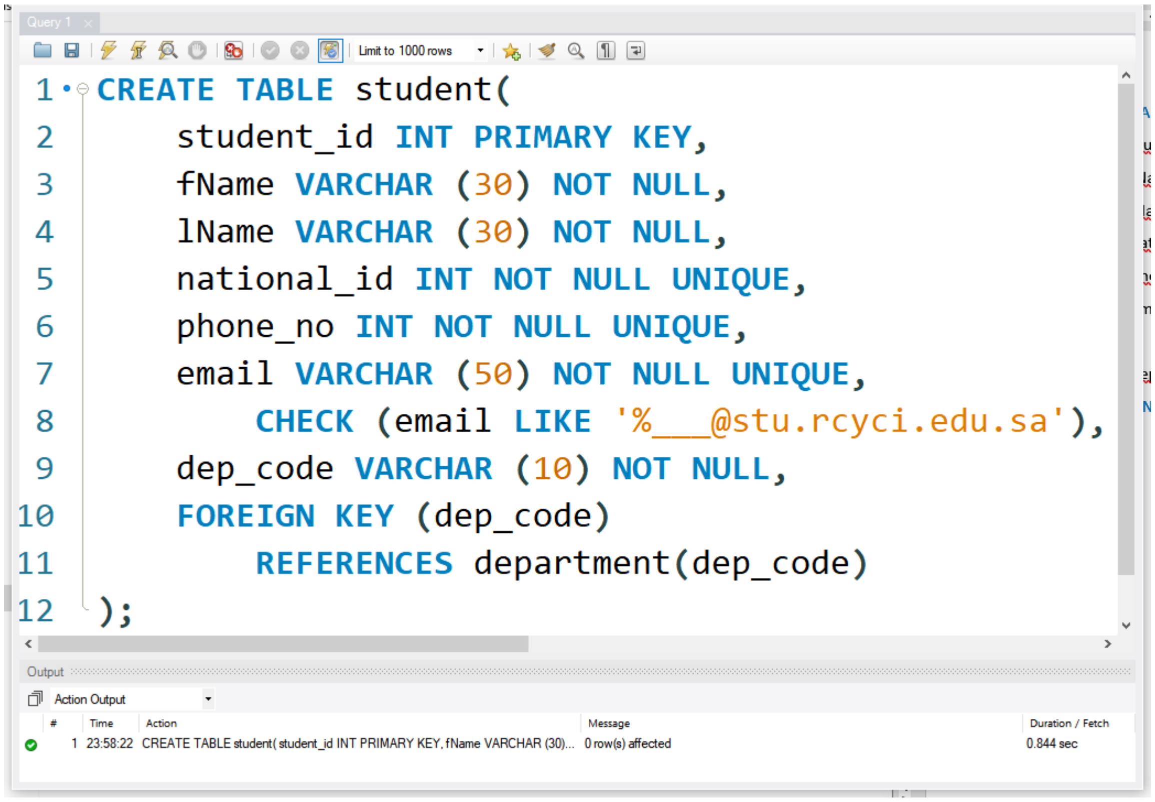
```
INSERT INTO department VALUES
    ("CSE", "Computer Science and Engineering Department"),
    ("MS", "Management Science Department"),
    ("ID", "Interior Design Department"),
    ("AL", "Applied Linguistics Department");
SELECT * FROM department;
```

dep_code	dep_name
AL	Applied Linguistics Department
CSE	Computer Science and Engineering Department
ID	Interior Design Department
MS	Management Science Department
NULL	NULL

#	Time	Action	Message
1	22:21:51	INSERT INTO department VALUES ("CSE", "Computer Science and En...	4 row(s) affected Records: 4 Duplicates:
2	22:21:52	SELECT * FROM department LIMIT 0, 1000	4 row(s) returned

```
INSERT INTO department VALUES
        ("CSE", "Computer Science and Engineering Department"),
        ("MS", "Management Science Department"),
        ("ID", "Interior Design Department"),
        ("AL", "Applied Linguistics Department");
SELECT * FROM department;
```

Student Table

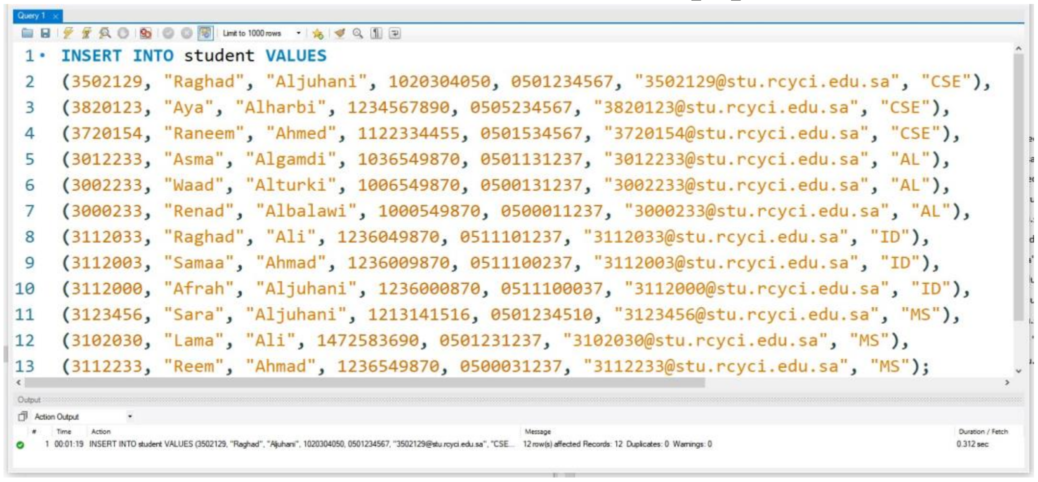


```
CREATE TABLE student(
      student_id INT PRIMARY KEY,
      fName VARCHAR (30) NOT NULL,
      lName VARCHAR (30) NOT NULL,
      national_id INT NOT NULL UNIQUE,
      phone_no INT NOT NULL UNIQUE,
      email VARCHAR (50) NOT NULL UNIQUE,
          CHECK (email LIKE '%___@stu.rcyci.edu.sa'),
      dep_code VARCHAR (10) NOT NULL,
      FOREIGN KEY (dep_code)
          REFERENCES department(dep_code)
);
```

```
Output
Action Output
#   Time      Action                                                          Message            Duration / Fetch
1   23:58:22  CREATE TABLE student( student_id INT PRIMARY KEY, fName VARCHAR (30)...  0 row(s) affected  0.844 sec
```

```
CREATE TABLE student(
        student_id INT PRIMARY KEY,
        fName VARCHAR (30) NOT NULL,
        lName VARCHAR (30) NOT NULL,
        national_id INT NOT NULL UNIQUE,
        phone_no INT NOT NULL UNIQUE,
        email VARCHAR (50) NOT NULL UNIQUE,
                CHECK (email LIKE '%___@stu.rcyci.edu.sa'),
        dep_code VARCHAR (10) NOT NULL,
   FOREIGN KEY (dep_code)
                REFERENCES department(dep_code)
);
```

Student Table population



```
INSERT INTO student VALUES
(3502129, "Raghad", "Aljuhani", 1020304050, 0501234567, "3502129@stu.rcyci.edu.sa", "CSE"),
(3820123, "Aya", "Alharbi", 1234567890, 0505234567, "3820123@stu.rcyci.edu.sa", "CSE"),
(3720154, "Raneem", "Ahmed", 1122334455, 0501534567, "3720154@stu.rcyci.edu.sa", "CSE"),
(3012233, "Asma", "Algamdi", 1036549870, 0501131237, "3012233@stu.rcyci.edu.sa", "AL"),
(3002233, "Waad", "Alturki", 1006549870, 0500131237, "3002233@stu.rcyci.edu.sa", "AL"),
(3000233, "Renad", "Albalawi", 1000549870, 0500011237, "3000233@stu.rcyci.edu.sa", "AL"),
(3112033, "Raghad", "Ali", 1236049870, 0511101237, "3112033@stu.rcyci.edu.sa", "ID"),
(3112003, "Samaa", "Ahmad", 1236009870, 0511100237, "3112003@stu.rcyci.edu.sa", "ID"),
(3112000, "Afrah", "Aljuhani", 1236000870, 0511100037, "3112000@stu.rcyci.edu.sa", "ID"),
(3123456, "Sara", "Aljuhani", 1213141516, 0501234510, "3123456@stu.rcyci.edu.sa", "MS"),
(3102030, "Lama", "Ali", 1472583690, 0501231237, "3102030@stu.rcyci.edu.sa", "MS"),
(3112233, "Reem", "Ahmad", 1236549870, 0500031237, "3112233@stu.rcyci.edu.sa", "MS");
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	00:01:19	INSERT INTO student VALUES (3502129, "Raghad", "Aljuhani", 1020304050, 0501234567, "3502129@stu.rcyci.edu.sa", "CSE...	12 row(s) affected Records: 12 Duplicates: 0 Warnings: 0	0.312 sec

INSERT INTO student VALUES

(3502129, "Raghad", "Aljuhani", 1020304050, 0501234567, "3502129@stu.rcyci.edu.sa", "CSE"),

(3820123, "Aya", "Alharbi", 1234567890, 0505234567, "3820123@stu.rcyci.edu.sa", "CSE"),

(3720154, "Raneem", "Ahmed", 1122334455, 0501534567, "3720154@stu.rcyci.edu.sa", "CSE"),

(3012233, "Asma", "Algamdi", 1036549870, 0501131237, "3012233@stu.rcyci.edu.sa", "AL"),

(3002233, "Waad", "Alturki", 1006549870, 0500131237, "3002233@stu.rcyci.edu.sa", "AL"),

(3000233, "Renad", "Albalawi", 1000549870, 0500011237, "3000233@stu.rcyci.edu.sa", "AL"),

(3112033, "Raghad", "Ali", 1236049870, 0511101237, "3112033@stu.rcyci.edu.sa", "ID"),

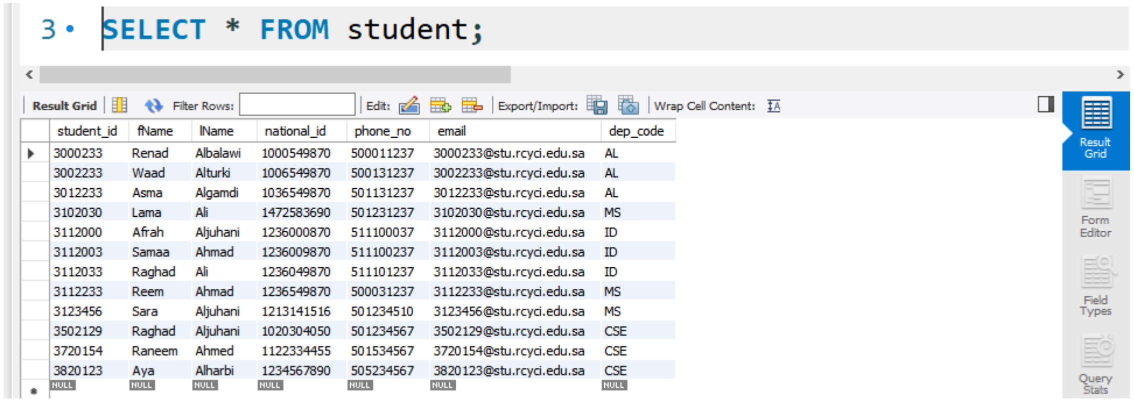
(3112003, "Samaa", "Ahmad", 1236009870, 0511100237, "3112003@stu.rcyci.edu.sa", "ID"),

(3112000, "Afrah", "Aljuhani", 1236000870, 0511100037, "3112000@stu.rcyci.edu.sa", "ID"),

(3123456, "Sara", "Aljuhani", 1213141516, 0501234510, "3123456@stu.rcyci.edu.sa", "MS"),

(3102030, "Lama", "Ali", 1472583690, 0501231237, "3102030@stu.rcyci.edu.sa", "MS"),

(3112233, "Reem", "Ahmad", 1236549870, 0500031237, "3112233@stu.rcyci.edu.sa", "MS");

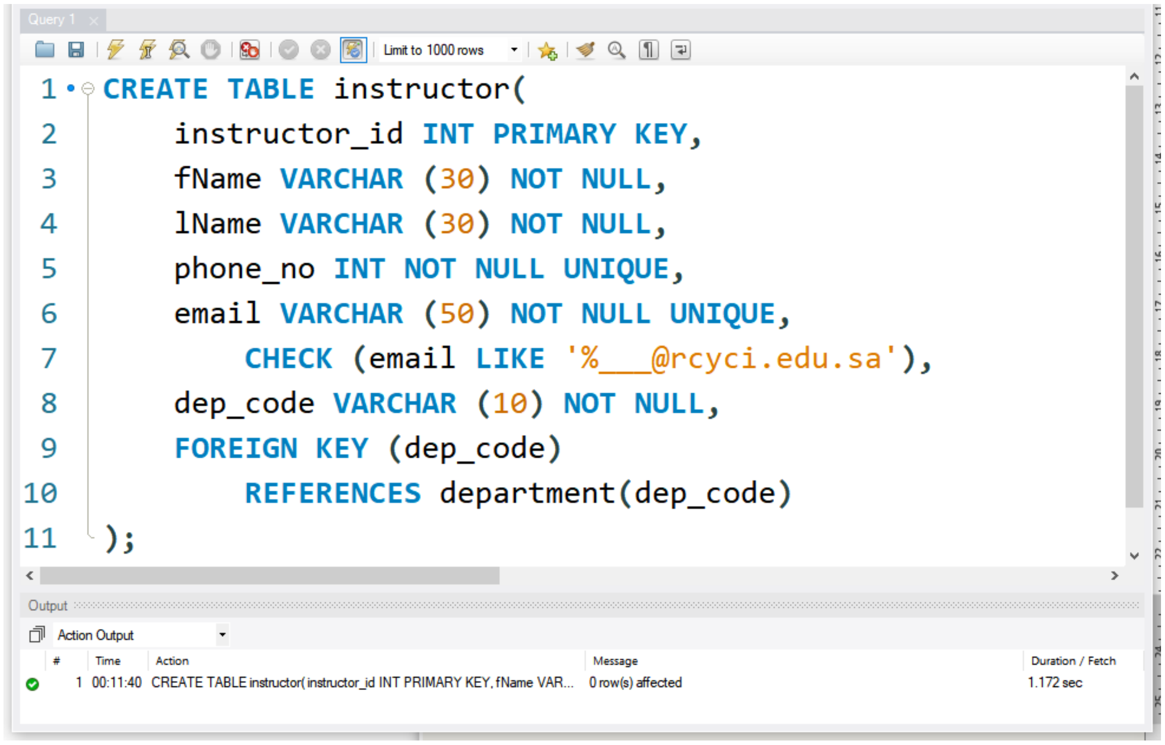


```
SELECT * FROM student;
```

student_id	fName	lName	national_id	phone_no	email	dep_code
3000233	Renad	Albalawi	1000549870	500011237	3000233@stu.rcyci.edu.sa	AL
3002233	Waad	Alturki	1006549870	500131237	3002233@stu.rcyci.edu.sa	AL
3012233	Asma	Algamdi	1036549870	501131237	3012233@stu.rcyci.edu.sa	AL
3102030	Lama	Ali	1472583690	501231237	3102030@stu.rcyci.edu.sa	MS
3112000	Afrah	Aljuhani	1236000870	511100037	3112000@stu.rcyci.edu.sa	ID
3112003	Samaa	Ahmad	1236009870	511100237	3112003@stu.rcyci.edu.sa	ID
3112033	Raghad	Ali	1236049870	511101237	3112033@stu.rcyci.edu.sa	ID
3112233	Reem	Ahmad	1236549870	500031237	3112233@stu.rcyci.edu.sa	MS
3123456	Sara	Aljuhani	1213141516	501234510	3123456@stu.rcyci.edu.sa	MS
3502129	Raghad	Aljuhani	1020304050	501234567	3502129@stu.rcyci.edu.sa	CSE
3720154	Raneem	Ahmed	1122334455	501534567	3720154@stu.rcyci.edu.sa	CSE
3820123	Aya	Alharbi	1234567890	505234567	3820123@stu.rcyci.edu.sa	CSE
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Prepared by: Dr.Kajal Nusratullah

Instructor Table

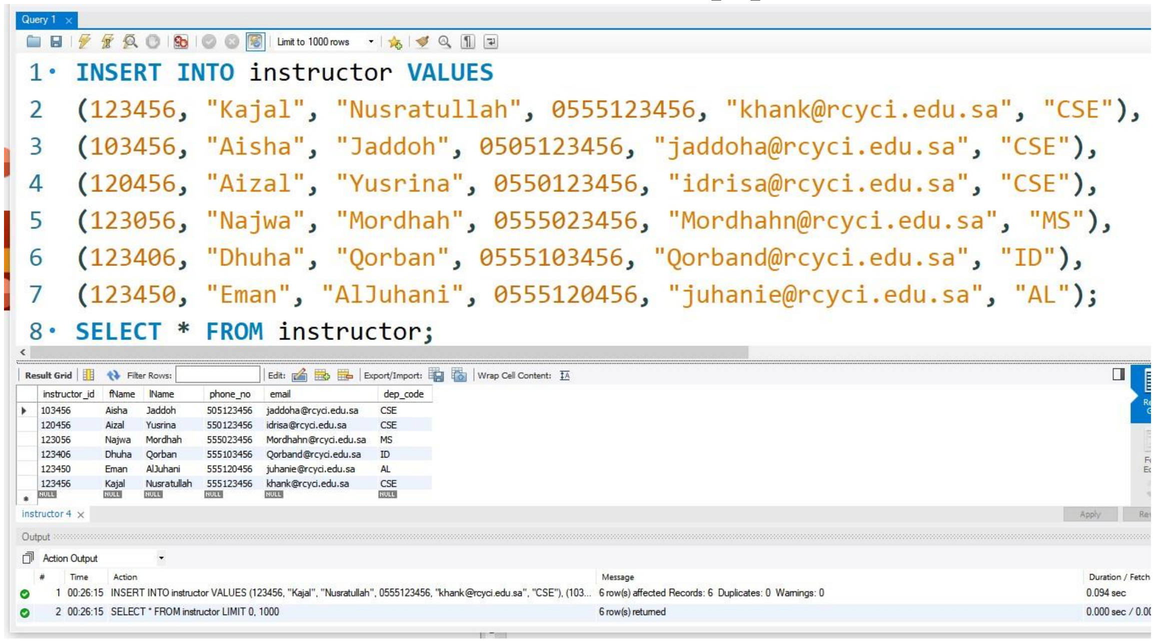


```
CREATE TABLE instructor(
      instructor_id INT PRIMARY KEY,
      fName VARCHAR (30) NOT NULL,
      lName VARCHAR (30) NOT NULL,
      phone_no INT NOT NULL UNIQUE,
      email VARCHAR (50) NOT NULL UNIQUE,
          CHECK (email LIKE '%___@rcyci.edu.sa'),
      dep_code VARCHAR (10) NOT NULL,
      FOREIGN KEY (dep_code)
          REFERENCES department(dep_code)
);
```

```
#   Time      Action                                                          Message              Duration / Fetch
1   00:11:40  CREATE TABLE instructor( instructor_id INT PRIMARY KEY, fName VAR...  0 row(s) affected    1.172 sec
```

```
CREATE TABLE instructor(
        instructor_id INT PRIMARY KEY,
        fName VARCHAR (30) NOT NULL,
        lName VARCHAR (30) NOT NULL,
        phone_no INT NOT NULL UNIQUE,
        email VARCHAR (50) NOT NULL UNIQUE,
                CHECK (email LIKE '%___@rcyci.edu.sa'),
        dep_code VARCHAR (10) NOT NULL,
        FOREIGN KEY (dep_code)
                REFERENCES department(dep_code)
);
```

Instructor Table population



```
INSERT INTO instructor VALUES
(123456, "Kajal", "Nusratullah", 0555123456, "khank@rcyci.edu.sa", "CSE"),
(103456, "Aisha", "Jaddoh", 0505123456, "jaddoha@rcyci.edu.sa", "CSE"),
(120456, "Aizal", "Yusrina", 0550123456, "idrisa@rcyci.edu.sa", "CSE"),
(123056, "Najwa", "Mordhah", 0555023456, "Mordhahn@rcyci.edu.sa", "MS"),
(123406, "Dhuha", "Qorban", 0555103456, "Qorband@rcyci.edu.sa", "ID"),
(123450, "Eman", "AlJuhani", 0555120456, "juhanie@rcyci.edu.sa", "AL");
SELECT * FROM instructor;
```

instructor_id	fName	lName	phone_no	email	dep_code
103456	Aisha	Jaddoh	505123456	jaddoha@rcyci.edu.sa	CSE
120456	Aizal	Yusrina	550123456	idrisa@rcyci.edu.sa	CSE
123056	Najwa	Mordhah	555023456	Mordhahn@rcyci.edu.sa	MS
123406	Dhuha	Qorban	555103456	Qorband@rcyci.edu.sa	ID
123450	Eman	AlJuhani	555120456	juhanie@rcyci.edu.sa	AL
123456	Kajal	Nusratullah	555123456	khank@rcyci.edu.sa	CSE
NULL	NULL	NULL	NULL	NULL	NULL

```
#  Time      Action                                                                                          Message                                                       Duration / Fetch
1  00:26:15  INSERT INTO instructor VALUES (123456, "Kajal", "Nusratullah", 0555123456, "khank@rcyci.edu.sa", "CSE"), (103...  6 row(s) affected Records: 6  Duplicates: 0  Warnings: 0  0.094 sec
2  00:26:15  SELECT * FROM instructor LIMIT 0, 1000                                                          6 row(s) returned                                             0.000 sec / 0.00
```

INSERT INTO instructor VALUES

(123456, "Kajal", "Nusratullah", 0555123456, "khank@rcyci.edu.sa", "CSE"),

(103456, "Aisha", "Jaddoh", 0505123456, "jaddoha@rcyci.edu.sa", "CSE"),

(120456, "Aizal", "Yusrina", 0550123456, "idrisa@rcyci.edu.sa", "CSE"),

(123056, "Najwa", "Mordhah", 0555023456, "Mordhahn@rcyci.edu.sa", "MS"),

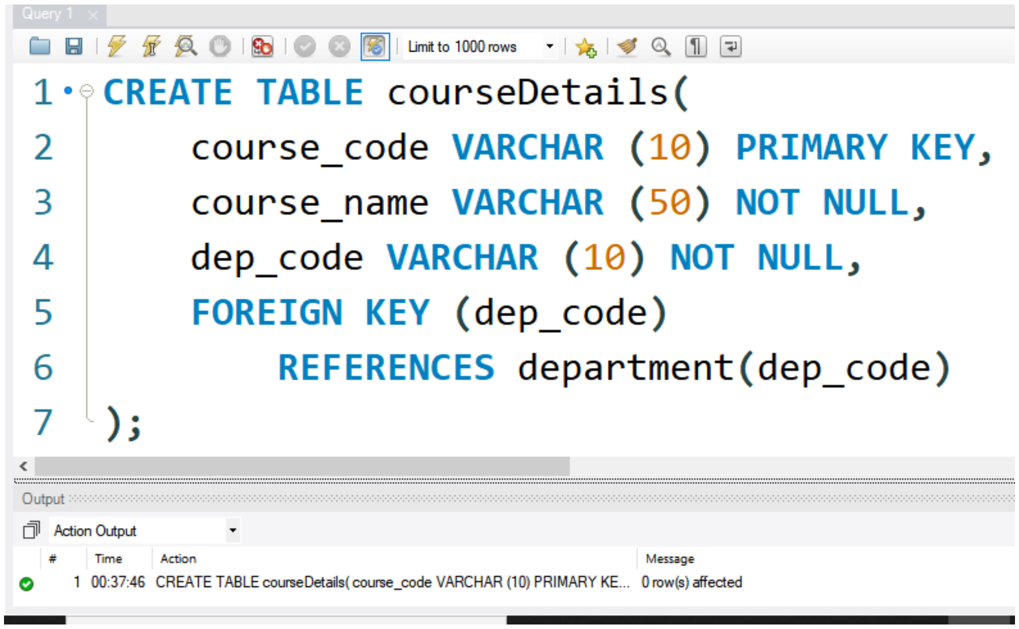
(123406, "Dhuha", "Qorban", 0555103456, "Qorband@rcyci.edu.sa", "ID"),

(123450, "Eman", "AlJuhani", 0555120456, "juhanie@rcyci.edu.sa", "AL");

SELECT * FROM instructor;

---

courseDetails Table

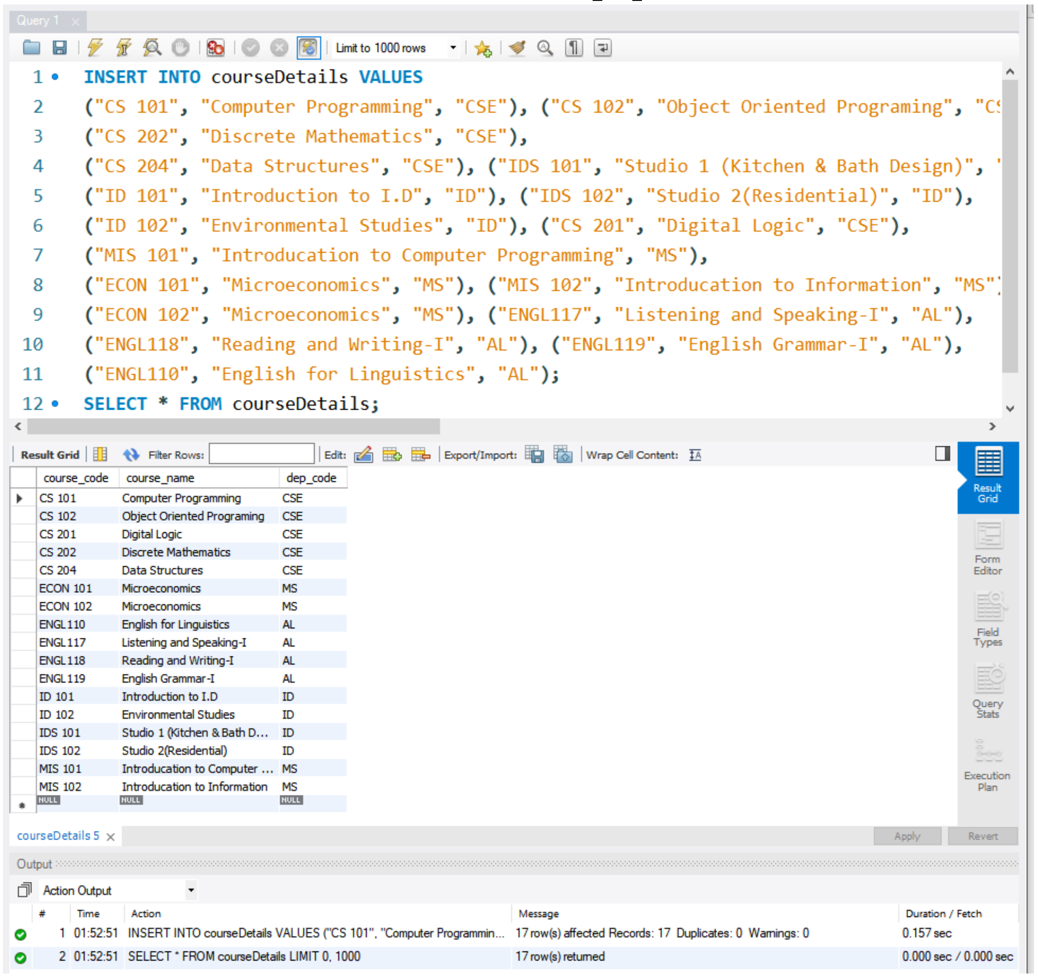


```
CREATE TABLE courseDetails(
        course_code VARCHAR (10) PRIMARY KEY,
        course_name VARCHAR (50) NOT NULL,
        dep_code VARCHAR (10) NOT NULL,
        FOREIGN KEY (dep_code)
            REFERENCES department(dep_code)
);
```

```
#   Time      Action                                                          Message
1   00:37:46  CREATE TABLE courseDetails( course_code VARCHAR (10) PRIMARY KE...  0 row(s) affected
```

```
CREATE TABLE courseDetails(
        course_code VARCHAR (10) PRIMARY KEY,
        course_name VARCHAR (50) NOT NULL,
        dep_code VARCHAR (10) NOT NULL,
        FOREIGN KEY (dep_code)
                REFERENCES department(dep_code)
);
```

courseDetails Table population



```
INSERT INTO courseDetails VALUES
("CS 101", "Computer Programming", "CSE"), ("CS 102", "Object Oriented Programing", "CS
("CS 202", "Discrete Mathematics", "CSE"),
("CS 204", "Data Structures", "CSE"), ("IDS 101", "Studio 1 (Kitchen & Bath Design)", '
("ID 101", "Introduction to I.D", "ID"), ("IDS 102", "Studio 2(Residential)", "ID"),
("ID 102", "Environmental Studies", "ID"), ("CS 201", "Digital Logic", "CSE"),
("MIS 101", "Introducation to Computer Programming", "MS"),
("ECON 101", "Microeconomics", "MS"), ("MIS 102", "Introducation to Information", "MS")
("ECON 102", "Microeconomics", "MS"), ("ENGL117", "Listening and Speaking-I", "AL"),
("ENGL118", "Reading and Writing-I", "AL"), ("ENGL119", "English Grammar-I", "AL"),
("ENGL110", "English for Linguistics", "AL");
SELECT * FROM courseDetails;
```

course_code	course_name	dep_code
CS 101	Computer Programming	CSE
CS 102	Object Oriented Programing	CSE
CS 201	Digital Logic	CSE
CS 202	Discrete Mathematics	CSE
CS 204	Data Structures	CSE
ECON 101	Microeconomics	MS
ECON 102	Microeconomics	MS
ENGL110	English for Linguistics	AL
ENGL117	Listening and Speaking-I	AL
ENGL118	Reading and Writing-I	AL
ENGL119	English Grammar-I	AL
ID 101	Introduction to I.D	ID
ID 102	Environmental Studies	ID
IDS 101	Studio 1 (Kitchen & Bath D...	ID
IDS 102	Studio 2(Residential)	ID
MIS 101	Introducation to Computer ...	MS
MIS 102	Introducation to Information	MS
NULL	NULL	NULL

courseDetails 5

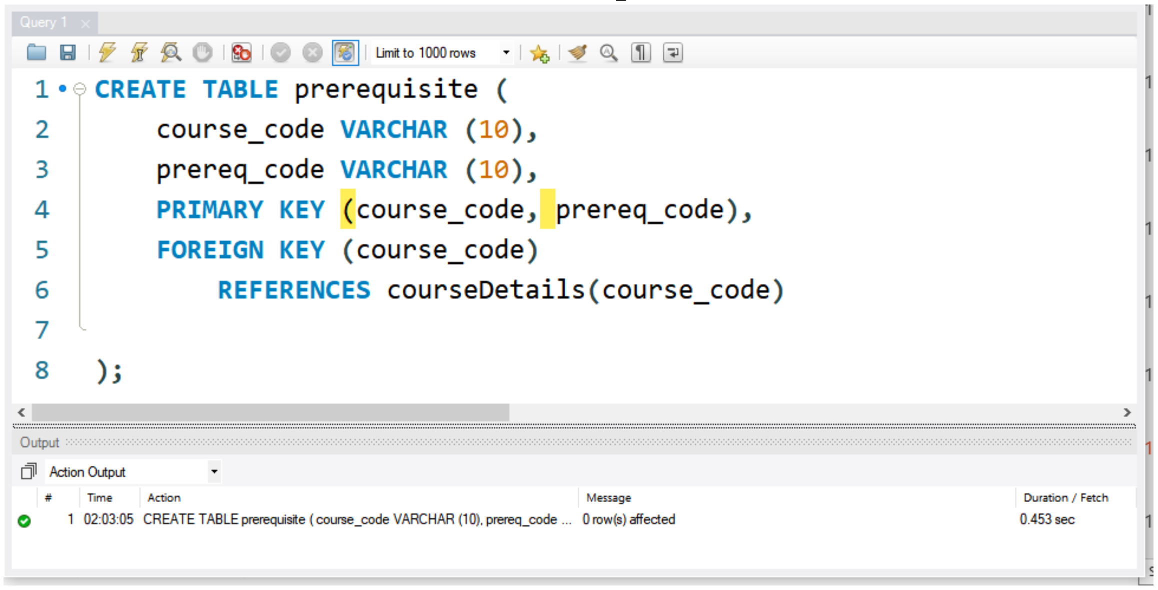
Output

Action Output

```
#  Time      Action                                                         Message                                                Duration / Fetch
1  01:52:51  INSERT INTO courseDetails VALUES ("CS 101", "Computer Programmin...  17 row(s) affected Records: 17  Duplicates: 0  Warnings: 0  0.157 sec
2  01:52:51  SELECT * FROM courseDetails LIMIT 0, 1000                      17 row(s) returned                                     0.000 sec / 0.000 sec
```

```
INSERT INTO courseDetails VALUES
("CS 101", "Computer Programming", "CSE"), ("CS 102", "Object Oriented Programing", "CSE"),
("CS 202", "Discrete Mathematics", "CSE"),
("CS 204", "Data Structures", "CSE"), ("IDS 101", "Studio 1 (Kitchen & Bath Design)", "ID"),
("ID 101", "Introduction to I.D", "ID"), ("IDS 102", "Studio 2(Residential)", "ID"),
("ID 102", "Environmental Studies", "ID"), ("CS 201", "Digital Logic", "CSE"),
("MIS 101", "Introducation to Computer Programming", "MS"),
("ECON 101", "Microeconomics", "MS"), ("MIS 102", "Introducation to Information", "MS"),
("ECON 102", "Microeconomics", "MS"), ("ENGL117", "Listening and Speaking-I", "AL"),
("ENGL118", "Reading and Writing-I", "AL"), ("ENGL119", "English Grammar-I", "AL"),
("ENGL110", "English for Linguistics", "AL");
SELECT * FROM courseDetails;
```

Prerequisite Table



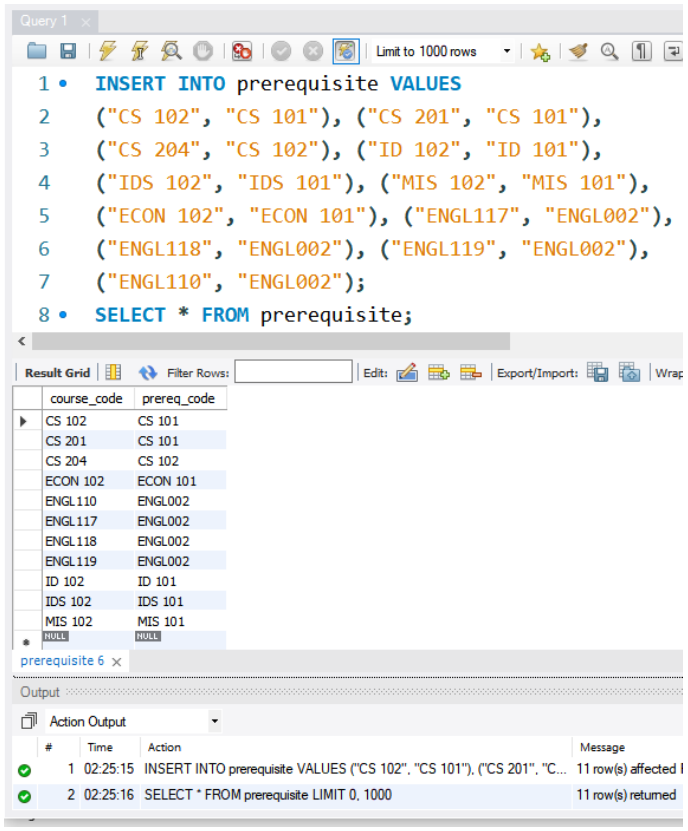
```
CREATE TABLE prerequisite (
    course_code VARCHAR (10),
    prereq_code VARCHAR (10),
    PRIMARY KEY (course_code, prereq_code),
    FOREIGN KEY (course_code)
        REFERENCES courseDetails(course_code)

);
```

```
#   Time      Action                                                                    Message              Duration / Fetch
1   02:03:05  CREATE TABLE prerequisite ( course_code VARCHAR (10), prereq_code ...    0 row(s) affected    0.453 sec
```

```
CREATE TABLE prerequisite (
        course_code VARCHAR (10),
        prereq_code VARCHAR (10),
        PRIMARY KEY (course_code, prereq_code),
        FOREIGN KEY (course_code)
                REFERENCES courseDetails(course_code)
);
```

Prerequisite Table population



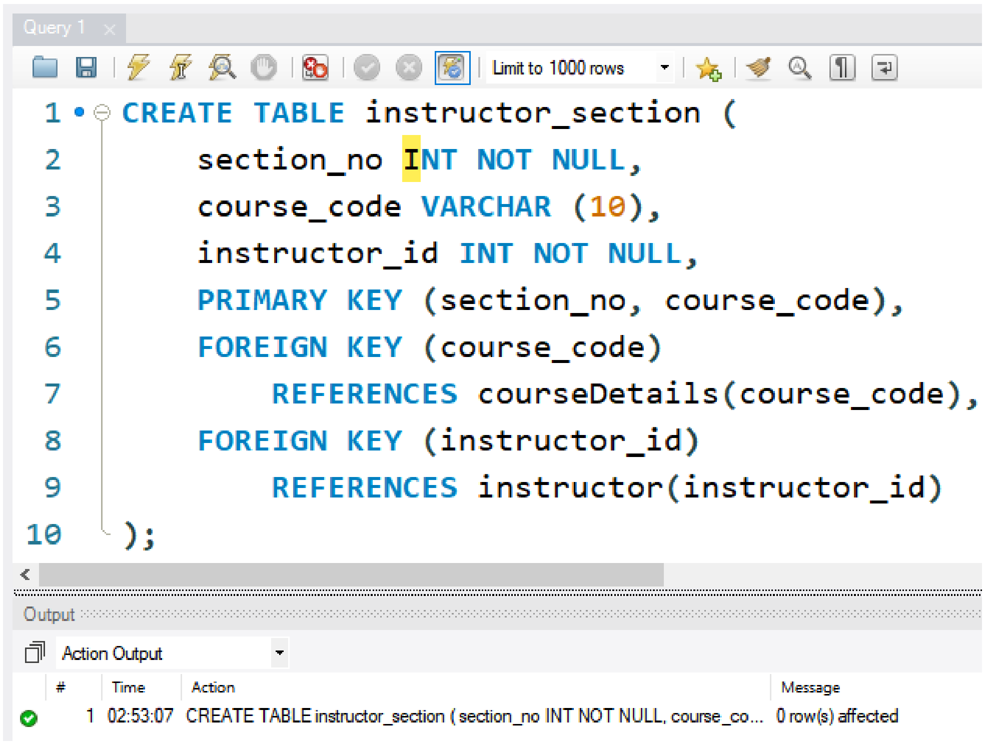
```
INSERT INTO prerequisite VALUES
("CS 102", "CS 101"), ("CS 201", "CS 101"),
("CS 204", "CS 102"), ("ID 102", "ID 101"),
("IDS 102", "IDS 101"), ("MIS 102", "MIS 101"),
("ECON 102", "ECON 101"), ("ENGL117", "ENGL002"),
("ENGL118", "ENGL002"), ("ENGL119", "ENGL002"),
("ENGL110", "ENGL002");
SELECT * FROM prerequisite;
```

course_code	prereq_code
CS 102	CS 101
CS 201	CS 101
CS 204	CS 102
ECON 102	ECON 101
ENGL110	ENGL002
ENGL117	ENGL002
ENGL118	ENGL002
ENGL119	ENGL002
ID 102	ID 101
IDS 102	IDS 101
MIS 102	MIS 101
NULL	NULL

```
#  Time      Action                                                         Message
1  02:25:15  INSERT INTO prerequisite VALUES ("CS 102", "CS 101"), ("CS 201", "C...   11 row(s) affected
2  02:25:16  SELECT * FROM prerequisite LIMIT 0, 1000                          11 row(s) returned
```

INSERT INTO prerequisite VALUES
("CS 102", "CS 101"), ("CS 201", "CS 101"),
("CS 204", "CS 102"), ("ID 102", "ID 101"),
("IDS 102", "IDS 101"), ("MIS 102", "MIS 101"),
("ECON 102", "ECON 101"), ("ENGL117", "ENGL002"),
("ENGL118", "ENGL002"), ("ENGL119", "ENGL002"),
("ENGL110", "ENGL002");
SELECT * FROM prerequisite;

Instructor_Section Table

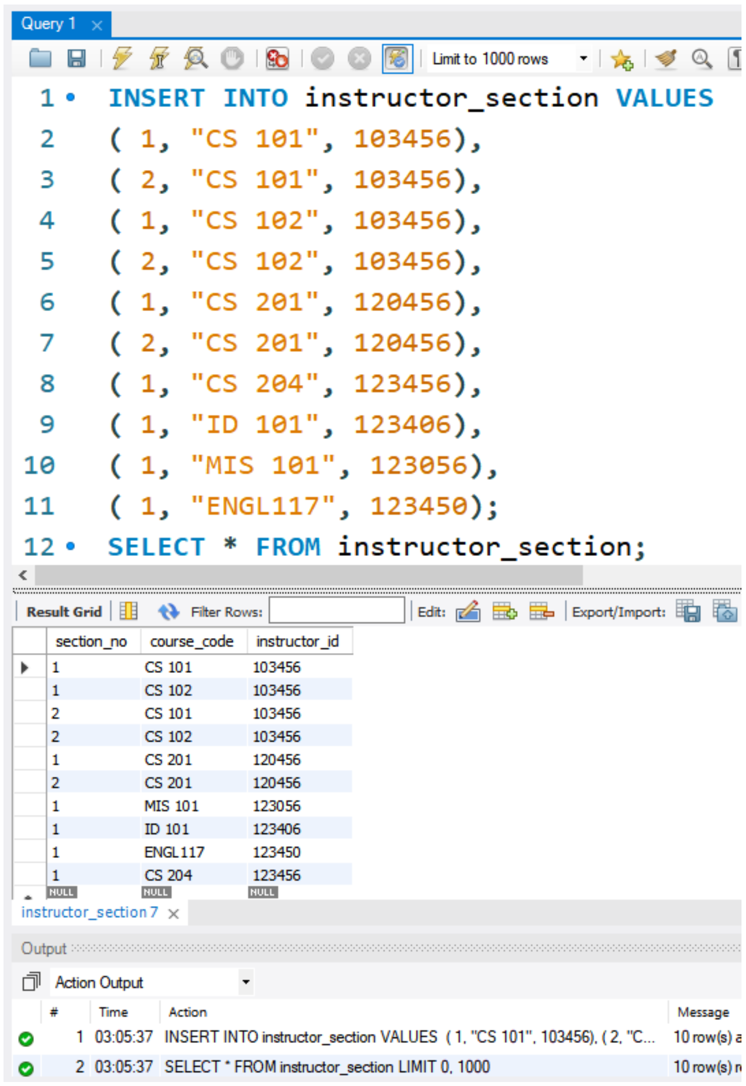


```
CREATE TABLE instructor_section (
    section_no INT NOT NULL,
    course_code VARCHAR (10),
    instructor_id INT NOT NULL,
    PRIMARY KEY (section_no, course_code),
    FOREIGN KEY (course_code)
        REFERENCES courseDetails(course_code),
    FOREIGN KEY (instructor_id)
        REFERENCES instructor(instructor_id)
);
```

```
#   Time      Action                                                            Message
1   02:53:07  CREATE TABLE instructor_section ( section_no INT NOT NULL, course_co...  0 row(s) affected
```

```
CREATE TABLE instructor_section (
        section_no INT NOT NULL,
        course_code VARCHAR (10),
        instructor_id INT NOT NULL,
        PRIMARY KEY (section_no, course_code),
        FOREIGN KEY (course_code)
                REFERENCES courseDetails(course_code),
        FOREIGN KEY (instructor_id)
                REFERENCES instructor(instructor_id)
);
```

Instructor_Section Table population



```
INSERT INTO instructor_section VALUES
( 1, "CS 101", 103456),
( 2, "CS 101", 103456),
( 1, "CS 102", 103456),
( 2, "CS 102", 103456),
( 1, "CS 201", 120456),
( 2, "CS 201", 120456),
( 1, "CS 204", 123456),
( 1, "ID 101", 123406),
( 1, "MIS 101", 123056),
( 1, "ENGL117", 123450);
SELECT * FROM instructor_section;
```

section_no	course_code	instructor_id
1	CS 101	103456
1	CS 102	103456
2	CS 101	103456
2	CS 102	103456
1	CS 201	120456
2	CS 201	120456
1	MIS 101	123056
1	ID 101	123406
1	ENGL117	123450
1	CS 204	123456
NULL	NULL	NULL

```
#  Time      Action                                                              Message
1  03:05:37  INSERT INTO instructor_section VALUES ( 1, "CS 101", 103456), ( 2, "C...  10 row(s) a
2  03:05:37  SELECT * FROM instructor_section LIMIT 0, 1000                          10 row(s) r
```

INSERT INTO instructor_section VALUES

(1, "CS 101", 103456), (2, "CS 101", 103456),

(1, "CS 102", 103456), (2, "CS 102", 103456),

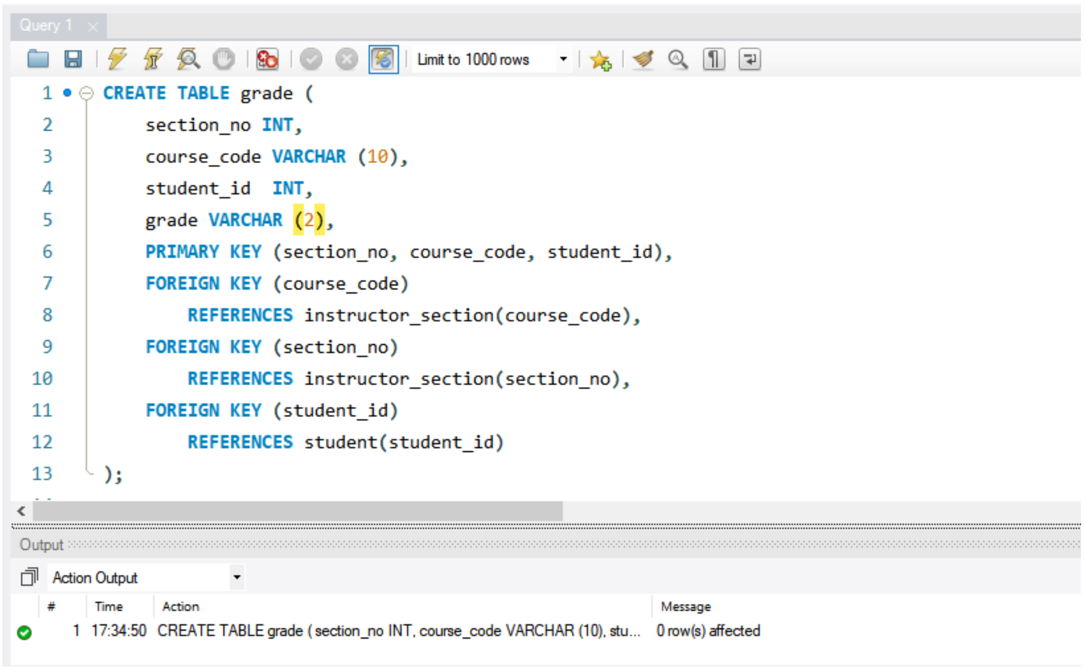
(1, "CS 201", 120456), (2, "CS 201", 120456),

(1, "CS 204", 123456), (1, "ID 101", 123406),

(1, "MIS 101", 123056), (1, "ENGL117", 123450);

SELECT * FROM instructor_section;

Grade Table



Query 1

Limit to 1000 rows

```
CREATE TABLE grade (
    section_no INT,
    course_code VARCHAR (10),
    student_id  INT,
    grade VARCHAR (2),
    PRIMARY KEY (section_no, course_code, student_id),
    FOREIGN KEY (course_code)
        REFERENCES instructor_section(course_code),
    FOREIGN KEY (section_no)
        REFERENCES instructor_section(section_no),
    FOREIGN KEY (student_id)
        REFERENCES student(student_id)
);
```

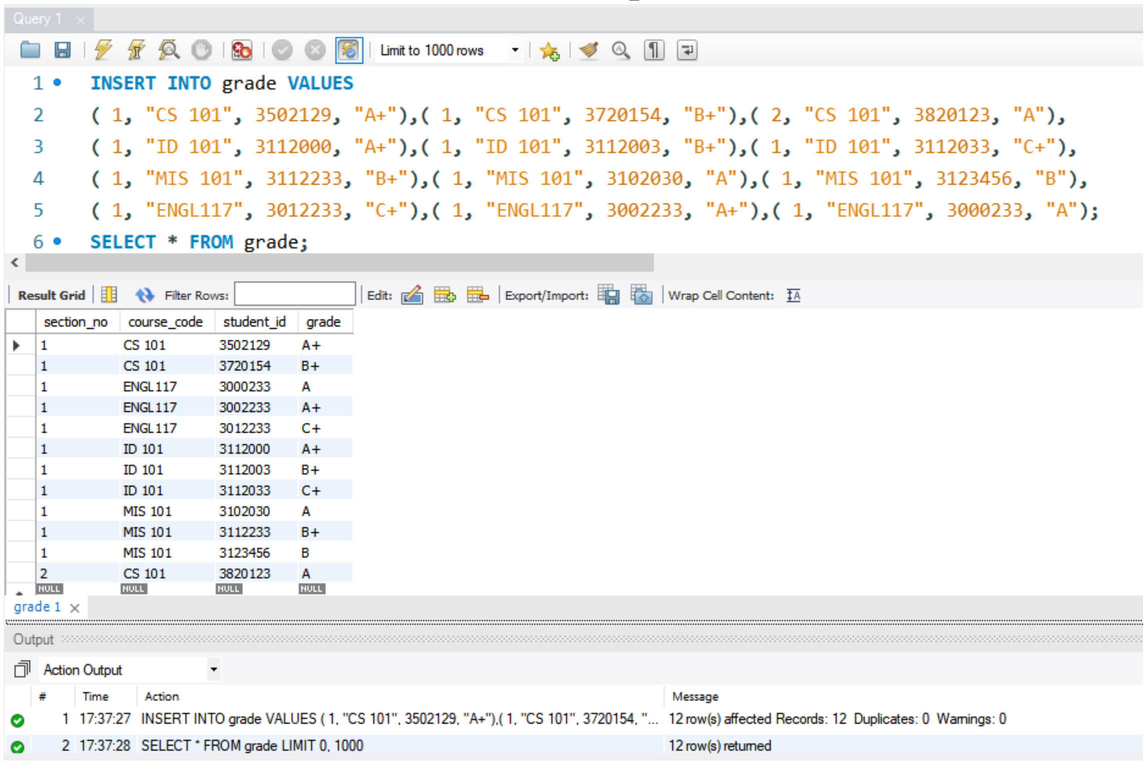
Output

Action Output

```
#   Time       Action                                                            Message
1   17:34:50   CREATE TABLE grade ( section_no INT, course_code VARCHAR (10), stu...   0 row(s) affected
```

```
CREATE TABLE grade (
        section_no INT,
        course_code VARCHAR (10),
        student_id INT,
        grade VARCHAR (2),
        PRIMARY KEY (section_no, course_code, student_id),
        FOREIGN KEY (course_code)
                REFERENCES instructor_section(course_code),
        FOREIGN KEY (section_no)
                REFERENCES instructor_section(section_no),
        FOREIGN KEY (student_id)
                REFERENCES student(student_id)
);
```

Grade Table Population



```
INSERT INTO grade VALUES
( 1, "CS 101", 3502129, "A+"),( 1, "CS 101", 3720154, "B+"),( 2, "CS 101", 3820123, "A"),
( 1, "ID 101", 3112000, "A+"),( 1, "ID 101", 3112003, "B+"),( 1, "ID 101", 3112033, "C+"),
( 1, "MIS 101", 3112233, "B+"),( 1, "MIS 101", 3102030, "A"),( 1, "MIS 101", 3123456, "B"),
( 1, "ENGL117", 3012233, "C+"),( 1, "ENGL117", 3002233, "A+"),( 1, "ENGL117", 3000233, "A");
SELECT * FROM grade;
```

section_no	course_code	student_id	grade
1	CS 101	3502129	A+
1	CS 101	3720154	B+
1	ENGL117	3000233	A
1	ENGL117	3002233	A+
1	ENGL117	3012233	C+
1	ID 101	3112000	A+
1	ID 101	3112003	B+
1	ID 101	3112033	C+
1	MIS 101	3102030	A
1	MIS 101	3112233	B+
1	MIS 101	3123456	B
2	CS 101	3820123	A
NULL	NULL	NULL	NULL

```
#  Time      Action                                                                          Message
1  17:37:27  INSERT INTO grade VALUES ( 1, "CS 101", 3502129, "A+"),( 1, "CS 101", 3720154, "...  12 row(s) affected Records: 12  Duplicates: 0  Warnings: 0
2  17:37:28  SELECT * FROM grade LIMIT 0, 1000                                               12 row(s) returned
```

INSERT INTO grade VALUES

(1, "CS 101", 3502129 , "A+"),(1, "CS 101", 3720154 , "B+"),(2, "CS 101", 3820123 , "A"),

(1, "ID 101", 3112000 , "A+"),(1, "ID 101", 3112003 , "B+"),(1, "ID 101", 3112033 , "C+"),

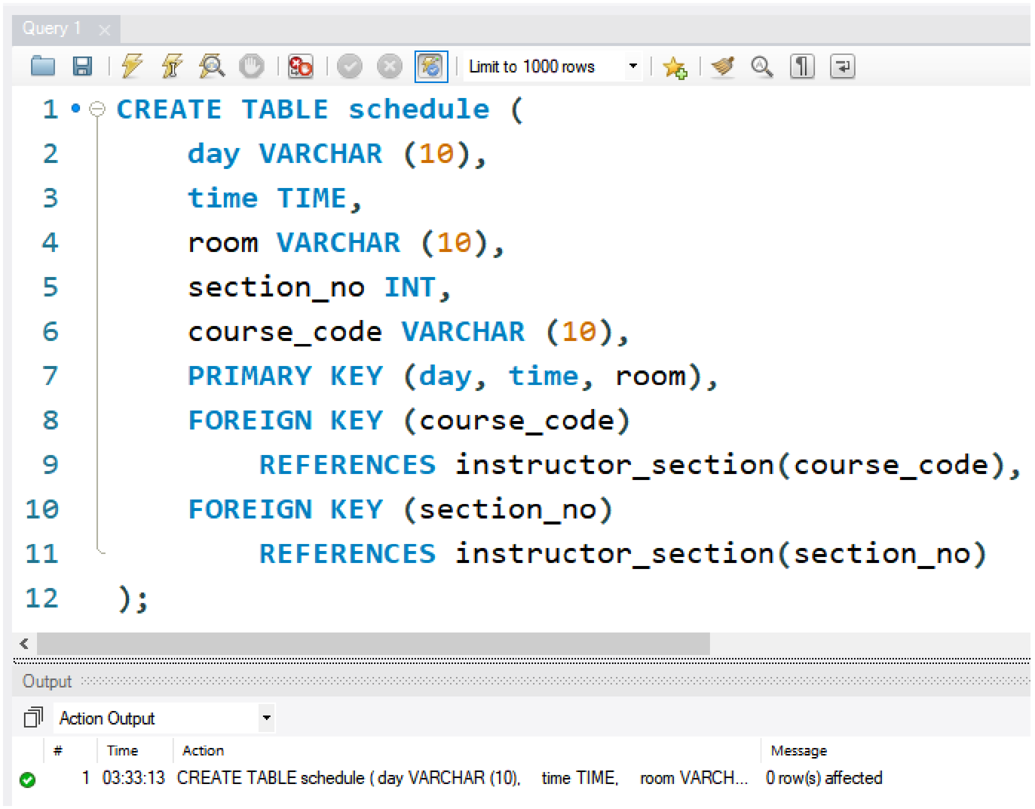
(1, "MIS 101", 3112233 , "B+"),(1, "MIS 101", 3102030 , "A"),(1, "MIS 101", 3123456 , "B"),

(1, "ENGL117", 3012233 , "C+"),(1, "ENGL117", 3002233 , "A+"),(1, "ENGL117", 3000233 , "A");

SELECT * FROM grade;

---

schedule Table

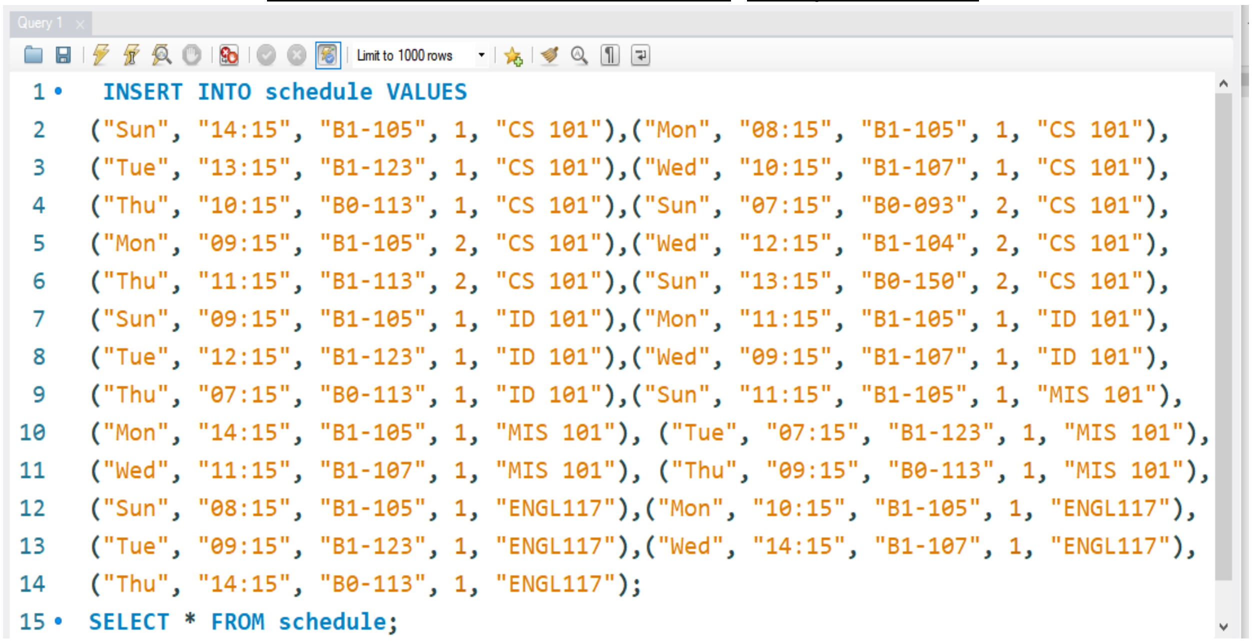


```
CREATE TABLE schedule (
	day VARCHAR (10),
	time TIME,
	room VARCHAR (10),
	section_no INT,
	course_code VARCHAR (10),
	PRIMARY KEY (day, time, room),
	FOREIGN KEY (course_code)
		REFERENCES instructor_section(course_code),
	FOREIGN KEY (section_no)
		REFERENCES instructor_section(section_no)
);
```

```
#  Time      Action                                                      Message
1  03:33:13  CREATE TABLE schedule ( day VARCHAR (10),  time TIME,  room VARCH...  0 row(s) affected
```

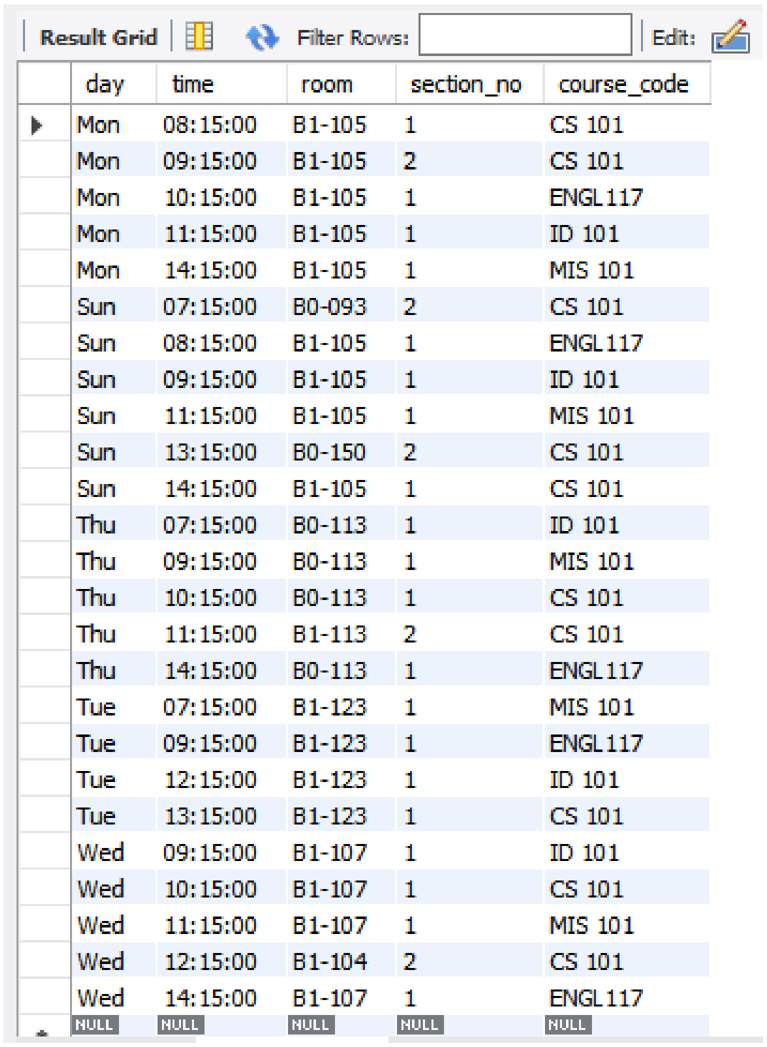
```
CREATE TABLE schedule (
	day VARCHAR (10),
	time TIME,
	room VARCHAR (10),
	section_no INT,
	course_code VARCHAR (10),
	PRIMARY KEY (day, time, room),
	FOREIGN KEY (course_code)
		REFERENCES instructor_section(course_code),
	FOREIGN KEY (section_no)
		REFERENCES instructor_section(section_no)
);
```

schedule Table Population



Limit to 1000 rows

```
INSERT INTO schedule VALUES
("Sun", "14:15", "B1-105", 1, "CS 101"),("Mon", "08:15", "B1-105", 1, "CS 101"),
("Tue", "13:15", "B1-123", 1, "CS 101"),("Wed", "10:15", "B1-107", 1, "CS 101"),
("Thu", "10:15", "B0-113", 1, "CS 101"),("Sun", "07:15", "B0-093", 2, "CS 101"),
("Mon", "09:15", "B1-105", 2, "CS 101"),("Wed", "12:15", "B1-104", 2, "CS 101"),
("Thu", "11:15", "B1-113", 2, "CS 101"),("Sun", "13:15", "B0-150", 2, "CS 101"),
("Sun", "09:15", "B1-105", 1, "ID 101"),("Mon", "11:15", "B1-105", 1, "ID 101"),
("Tue", "12:15", "B1-123", 1, "ID 101"),("Wed", "09:15", "B1-107", 1, "ID 101"),
("Thu", "07:15", "B0-113", 1, "ID 101"),("Sun", "11:15", "B1-105", 1, "MIS 101"),
("Mon", "14:15", "B1-105", 1, "MIS 101"), ("Tue", "07:15", "B1-123", 1, "MIS 101"),
("Wed", "11:15", "B1-107", 1, "MIS 101"), ("Thu", "09:15", "B0-113", 1, "MIS 101"),
("Sun", "08:15", "B1-105", 1, "ENGL117"),("Mon", "10:15", "B1-105", 1, "ENGL117"),
("Tue", "09:15", "B1-123", 1, "ENGL117"),("Wed", "14:15", "B1-107", 1, "ENGL117"),
("Thu", "14:15", "B0-113", 1, "ENGL117");
SELECT * FROM schedule;
```

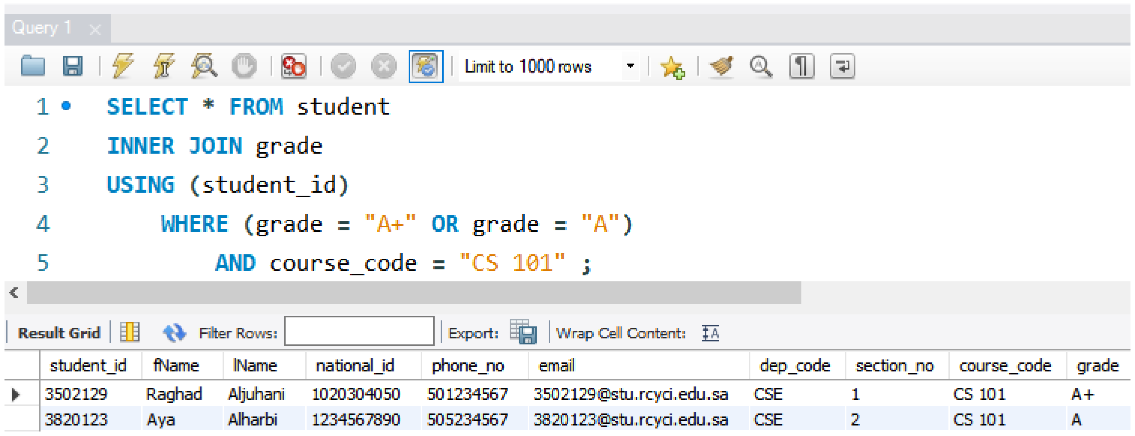


Result Grid | Filter Rows: | Edit:

day	time	room	section_no	course_code
Mon	08:15:00	B1-105	1	CS 101
Mon	09:15:00	B1-105	2	CS 101
Mon	10:15:00	B1-105	1	ENGL117
Mon	11:15:00	B1-105	1	ID 101
Mon	14:15:00	B1-105	1	MIS 101
Sun	07:15:00	B0-093	2	CS 101
Sun	08:15:00	B1-105	1	ENGL117
Sun	09:15:00	B1-105	1	ID 101
Sun	11:15:00	B1-105	1	MIS 101
Sun	13:15:00	B0-150	2	CS 101
Sun	14:15:00	B1-105	1	CS 101
Thu	07:15:00	B0-113	1	ID 101
Thu	09:15:00	B0-113	1	MIS 101
Thu	10:15:00	B0-113	1	CS 101
Thu	11:15:00	B1-113	2	CS 101
Thu	14:15:00	B0-113	1	ENGL117
Tue	07:15:00	B1-123	1	MIS 101
Tue	09:15:00	B1-123	1	ENGL117
Tue	12:15:00	B1-123	1	ID 101
Tue	13:15:00	B1-123	1	CS 101
Wed	09:15:00	B1-107	1	ID 101
Wed	10:15:00	B1-107	1	CS 101
Wed	11:15:00	B1-107	1	MIS 101
Wed	12:15:00	B1-104	2	CS 101
Wed	14:15:00	B1-107	1	ENGL117
NULL	NULL	NULL	NULL	NULL

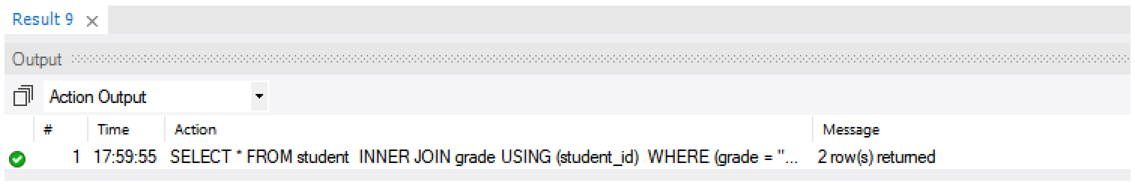
```
INSERT INTO schedule VALUES
("Sun", "14:15", "B1-105", 1, "CS 101"),("Mon", "08:15", "B1-105", 1, "CS 101"),
("Tue", "13:15", "B1-123", 1, "CS 101"),("Wed", "10:15", "B1-107", 1, "CS 101"),
("Thu", "10:15", "B0-113", 1, "CS 101"),("Sun", "07:15", "B0-093", 2, "CS 101"),
("Mon", "09:15", "B1-105", 2, "CS 101"),("Wed", "12:15", "B1-104", 2, "CS 101"),
("Thu", "11:15", "B1-113", 2, "CS 101"),("Sun", "13:15", "B0-150", 2, "CS 101"),
("Sun", "09:15", "B1-105", 1, "ID 101"),("Mon", "11:15", "B1-105", 1, "ID 101"),
("Tue", "12:15", "B1-123", 1, "ID 101"),("Wed", "09:15", "B1-107", 1, "ID 101"),
("Thu", "07:15", "B0-113", 1, "ID 101"),("Sun", "11:15", "B1-105", 1, "MIS 101"),
("Mon", "14:15", "B1-105", 1, "MIS 101"), ("Tue", "07:15", "B1-123", 1, "MIS 101"),
("Wed", "11:15", "B1-107", 1, "MIS 101"), ("Thu", "09:15", "B0-113", 1, "MIS 101"),
("Sun", "08:15", "B1-105", 1, "ENGL117"),("Mon", "10:15", "B1-105", 1, "ENGL117"),
("Tue", "09:15", "B1-123", 1, "ENGL117"),("Wed", "14:15", "B1-107", 1, "ENGL117"),
("Thu", "14:15", "B0-113", 1, "ENGL117");
SELECT * FROM schedule;
```

Retrieve the student's information and grade information for each student who got A+ or A in the CS101 course.



```
SELECT * FROM student
INNER JOIN grade
USING (student_id)
    WHERE (grade = "A+" OR grade = "A")
        AND course_code = "CS 101" ;
```

student_id	fName	lName	national_id	phone_no	email	dep_code	section_no	course_code	grade
3502129	Raghad	Aljuhani	1020304050	501234567	3502129@stu.rcyci.edu.sa	CSE	1	CS 101	A+
3820123	Aya	Alharbi	1234567890	505234567	3820123@stu.rcyci.edu.sa	CSE	2	CS 101	A



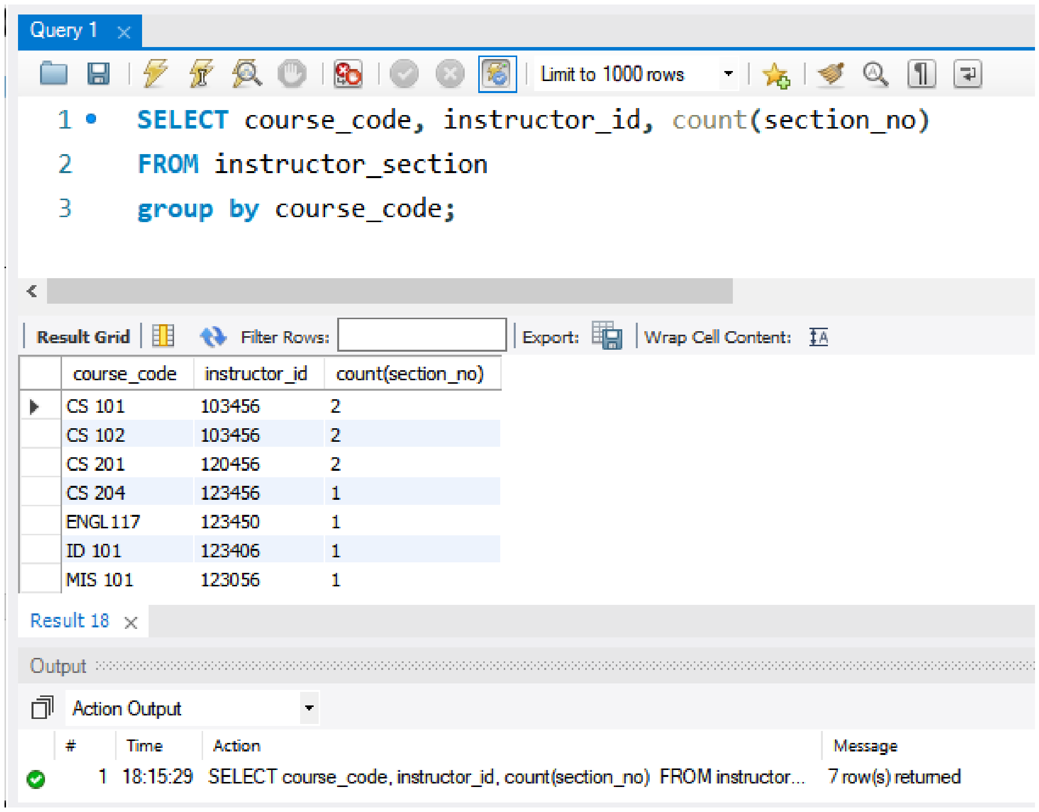
```
#   Time      Action                                                                Message
1   17:59:55  SELECT * FROM student  INNER JOIN grade USING (student_id)  WHERE (grade = "...   2 row(s) returned
```

By using multiple conditions (OR operators) and JOIN between student and grade, we can retrieve the records

```
SELECT * FROM student
INNER JOIN grade
```

```
USING (student_id)
    WHERE (grade = "A+" OR grade = "A")
        AND course_code = "CS 101" ;
```

Retrieve the total section in each course.



```
SELECT course_code, instructor_id, count(section_no)
FROM instructor_section
group by course_code;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

course_code	instructor_id	count(section_no)
CS 101	103456	2
CS 102	103456	2
CS 201	120456	2
CS 204	123456	1
ENGL117	123450	1
ID 101	123406	1
MIS 101	123056	1

Result 18

Output

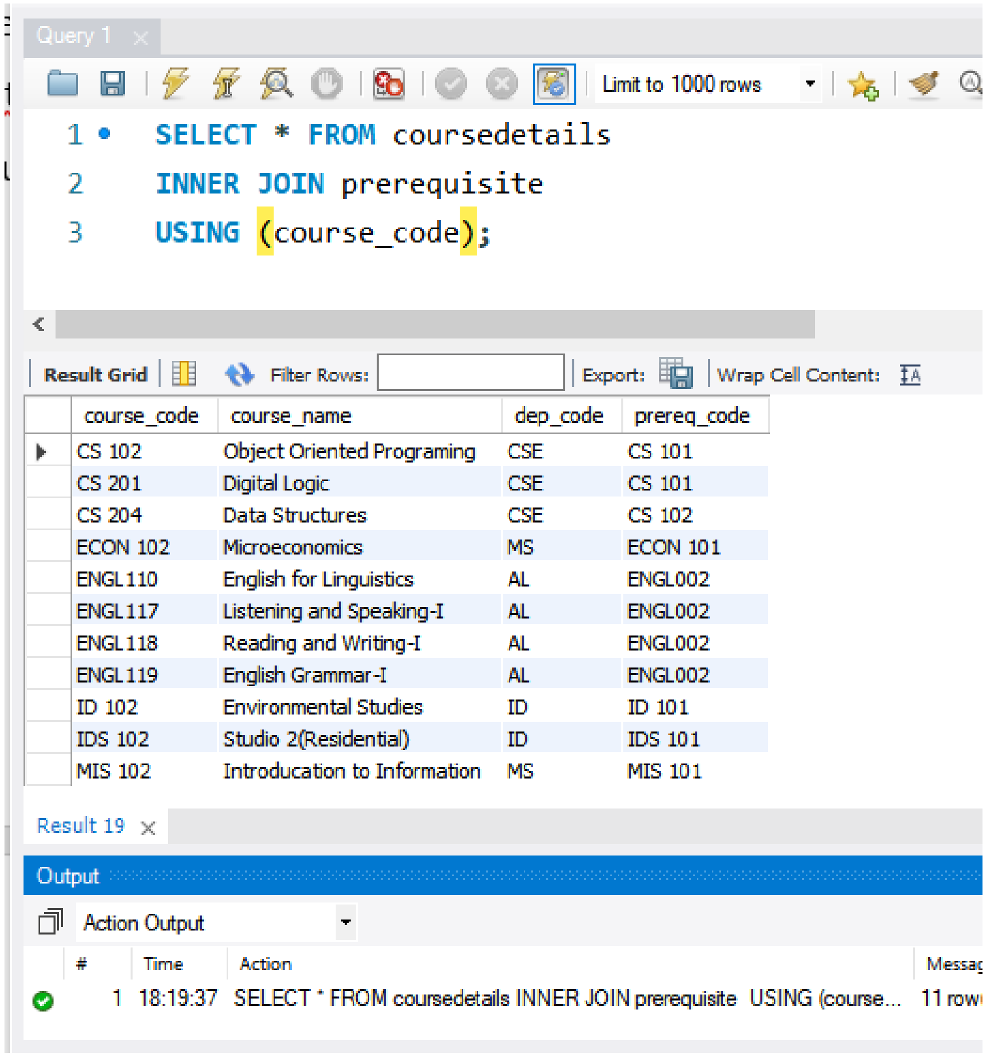
Action Output

```
#  Time      Action                                                            Message
1  18:15:29  SELECT course_code, instructor_id, count(section_no) FROM instructor...  7 row(s) returned
```

By using aggregate functions COUNT() to count number of section for one course, we can retrieve the records.

```
SELECT course_code, instructor_id, count(section_no)
FROM instructor_section
GROUP BY course_code;
```

Retrieve the complete course details with the prerequisite for each course



```
SELECT * FROM coursedetails
INNER JOIN prerequisite
USING (course_code);
```

course_code	course_name	dep_code	prereq_code
CS 102	Object Oriented Programing	CSE	CS 101
CS 201	Digital Logic	CSE	CS 101
CS 204	Data Structures	CSE	CS 102
ECON 102	Microeconomics	MS	ECON 101
ENGL110	English for Linguistics	AL	ENGL002
ENGL117	Listening and Speaking-I	AL	ENGL002
ENGL118	Reading and Writing-I	AL	ENGL002
ENGL119	English Grammar-I	AL	ENGL002
ID 102	Environmental Studies	ID	ID 101
IDS 102	Studio 2(Residential)	ID	IDS 101
MIS 102	Introducation to Information	MS	MIS 101

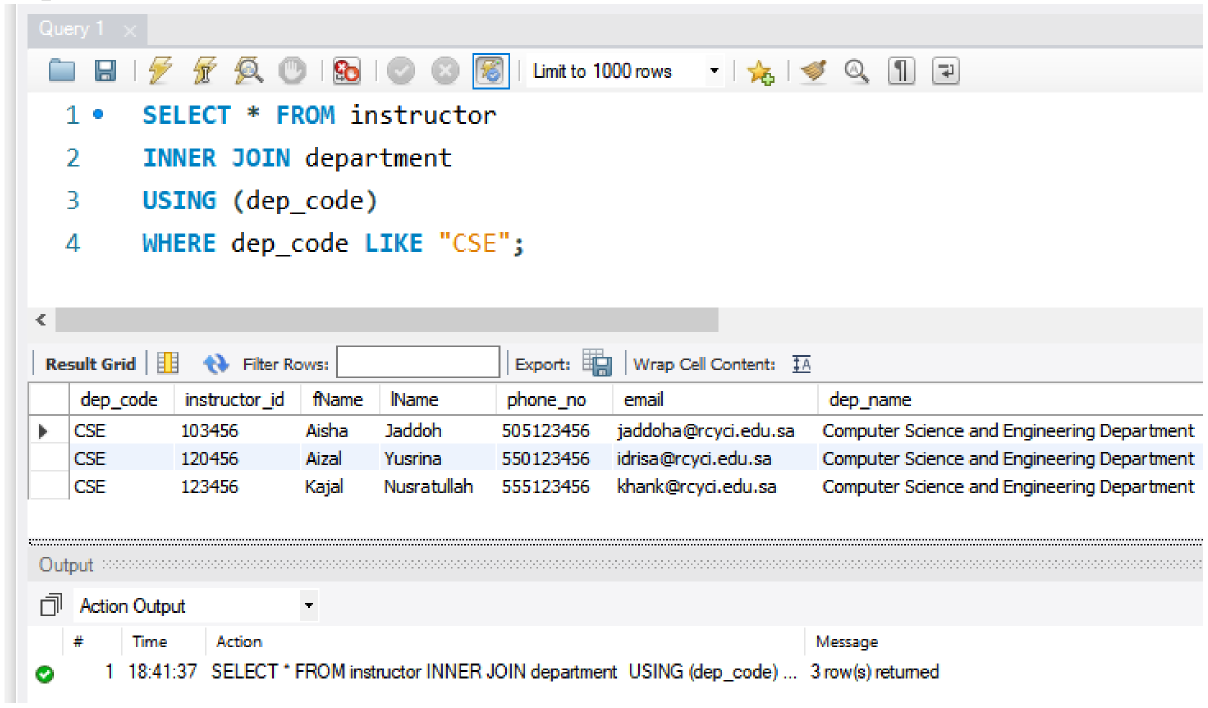
```
#   Time      Action                                                          Messag
1   18:19:37  SELECT * FROM coursedetails INNER JOIN prerequisite  USING (course...   11 row
```

By using INNER JOIN between coursedetails and prerequisite tables, we can retrieve the records.

SELECT * FROM coursedetails
INNER JOIN prerequisite
USING (course_code);

Prepared by: Dr.Kajal Nusratullah

Retrieve the complete Instructor information in CSE department with department name



```
SELECT * FROM instructor
INNER JOIN department
USING (dep_code)
WHERE dep_code LIKE "CSE";
```

	dep_code	instructor_id	fName	lName	phone_no	email	dep_name
▶	CSE	103456	Aisha	Jaddoh	505123456	jaddoha@rcyci.edu.sa	Computer Science and Engineering Department
	CSE	120456	Aizal	Yusrina	550123456	idrisa@rcyci.edu.sa	Computer Science and Engineering Department
	CSE	123456	Kajal	Nusratullah	555123456	khank@rcyci.edu.sa	Computer Science and Engineering Department

```
#   Time      Action                                                     Message
1   18:41:37  SELECT * FROM instructor INNER JOIN department USING (dep_code) ...   3 row(s) returned
```

By using INNER JOIN between instructor and department tables with condition dep_code = "CSE";

```
SELECT * FROM instructor
INNER JOIN department
USING (dep_code)
WHERE dep_code LIKE "CSE";
```

Prepared by: Dr.Kajal Nusratullah