LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title
CC5051NA Advanced Database Systems Development

Assessment Weightage &
Type 50% Individual Coursework

Year and Semester
2019-20 Autumn

Student Name: Shubhangi Piya

London Met ID: 19031687

College ID: np01cp4a190196

Assignment Due Date: 20th Dec, 2020

Assignment Submission Date: 20th Dec,2020

Title (Where Required):

Word Count (Where Required): 4357

TABLE OF CONTENTS

TABLE OF FIGURES

1. Introduction

1.1 Introduction of college

Established in April of 1860 the Brown College is one of the oldest private research university in New York. A place where brightest minds from across the globe come to explore, indulge in their curiosity and make the world a better place. The college offers over 150 degree programs and enroll some 41,000 students. The brown college has played a prominent role in many scientific advances from the Germany project to discovery of chemical elements and breakthroughs in computer science.

Majority of the enrollment is done under undergraduate programs but we also offer a comprehensive doctoral program. The four-year undergraduate programs offers its most popular majors in computer science, electrical engineering, business, finance and economics. We also offer interdisciplinary graduate programs in Agriculture, Chemistry, Computer Science, English, Epidemiology, Geography, Mathematics, Mechanical Engineering, Molecular Biology, Genetics and Physics.

The current faculty of Brown includes ten Nobel Prize winners, four Pulitzer Prize winners and five Wolf Prize winners. Brown's 5 libraries together contain 11 thousand volumes and cover over 5 h.a. of land, forming one of the largest library complexes in NY. The campus is also home to several museums including the California Museum of Paleontology, the Beck Art Museum and the Oscar Hall of Science.

The lecture halls and tutor rooms in Brown are fully furnished and air-conditioned with hygienic environment. They are well equipped with best multimedia tools for instructional needs. We have a Wi-Fi enabled campus which is easily accessible around any corner of the college. The college has designed a special app for the students and instructors that offers a beautiful classroom management platform. The Brown College has students coming from across the globe and this diversity only makes the campus more beautiful. What happens here at Brown College is not magic, it just feels that way.

1.2 Current business activities and operations

I. The college has six buildings out of which four are fully dedicated to the undergraduate students. The blocks are named Michigan, Arizona, Massachusetts, Nevada, Missouri and Wisconsin.

II. Brown has a total of 6 lecture halls that can accommodate around 150 students and 20 tutor rooms designed to fit a strength of 20 to 30 students. The classrooms are fully furnished, air conditioned and equipped with presentation materials.

III. The college has five libraries where four serve as library systems reference, administrative center, while the main collections reside in the Oscid undergraduate library. It has over 111 thousand printed volumes and manuscripts, maps, reference materials, encyclopedias.

IV. Students have their own large dining hall that can accommodate some 200 students with hygienic air. Besides this the college has 4 canteens to offer and a cozy café at the first floor.

V. The campus is also home to the New York botanical garden, one of the most diverse plant collection in the USA which is famous for its rare and endangered species.

VI. Tis oldest research university owns various re-search laboratories and research forest.

VII. Oski bears are Brown's athletic team. With its long history in athletics the college h.a won national titles in football, men's polo, basketball, gymnastics, water polo and rugby.

VIII. Brown holds 15 labs backed by the latest facilities and technical equipment for a superior learning experience.

IX. Not to forget the private swimming pools and theatre room for entertainment and sports purpose. Brown has two interior pools and a exterior. The theatre rooms have excellent sound system and adjustable sofas so you can sit back and enjoy any show.

X. We offer around 150 courses both for the graduated and under graduates so they do not have to limit their options to certain course. The classes are conducted six times a week.

XI. The Brown College conducts classes in a LTW pattern which stands for lecture tutorial workshop. The professors take lectures in the lecture hall where two to three classes are gathered. The tutorial is more interactive as there are Q/a sessions with lesser students. The workshop is where your theories and brought to practical agenda.

XII. The instructors teach and supervise students by directing laboratory workshop sessions, seminars, demonstrations, discussion groups and assigning individual or group projects, case study and field coursework.

1.3 Business Rules

The rules are description of operational procedures that will be used in the college's system. It describes the type of data to be stored in the database and how the college intends to use it.

I. The college database should record address of each person of which exactly one is designated as the mailing address.
II. Each address consists of country, province, city, street, house number and phone numbers.
III. Many instructors can be associated in a course, but an instructor can be associated only in one course.
IV. For each course, there is a course leader, and an instructor can be a leader of only one course.
V. Each instructor can teach any one or many modules at a time.
VI. A student can enroll for only one course.
VII. Each module is taught in any given particular class, but in each class a number of modules are taught.
VIII. The college has to keep yearly record of number of students enrolled in a course.
IX. Each person should provide their contact number, date of birth and house number.
X. The students must list their marks obtained in high school mark sheet and the student id from SEE examinations.

1.4 Identification of Entities and Attributes

Entity in database can be defined as object which has independent and self-contained existence. For example in a college database it ca be students, teachers, classes and courses offered. Al these entities have their attributes that act as their properties and give them their identity.

Entities	Attributes
Student	Student_Id(Pk), Name, Course_Type, Phone_No, Specification Id(Fk)
Instructor	Instructor_Id(Pk), Name, Salary, Phone Num, House No(Fk)
Address	House No(Pk), Province, Country, City, Street, Fax no
Course	Course_Type(Pk), Fees, Total Stds, Instructor Id(Fk)
Specification	Specification Id(Pk), Specification Name, Course Type(Fk)
Module	Module Id, Module Title, Students Enrolled, Specification Id(Fk), Instructor_Id(fk)
Infrastructure	Access Id(Pk), Email Address(fk), Library,Theatre, afé, Lab

1.5.1 Initial ER Diagram

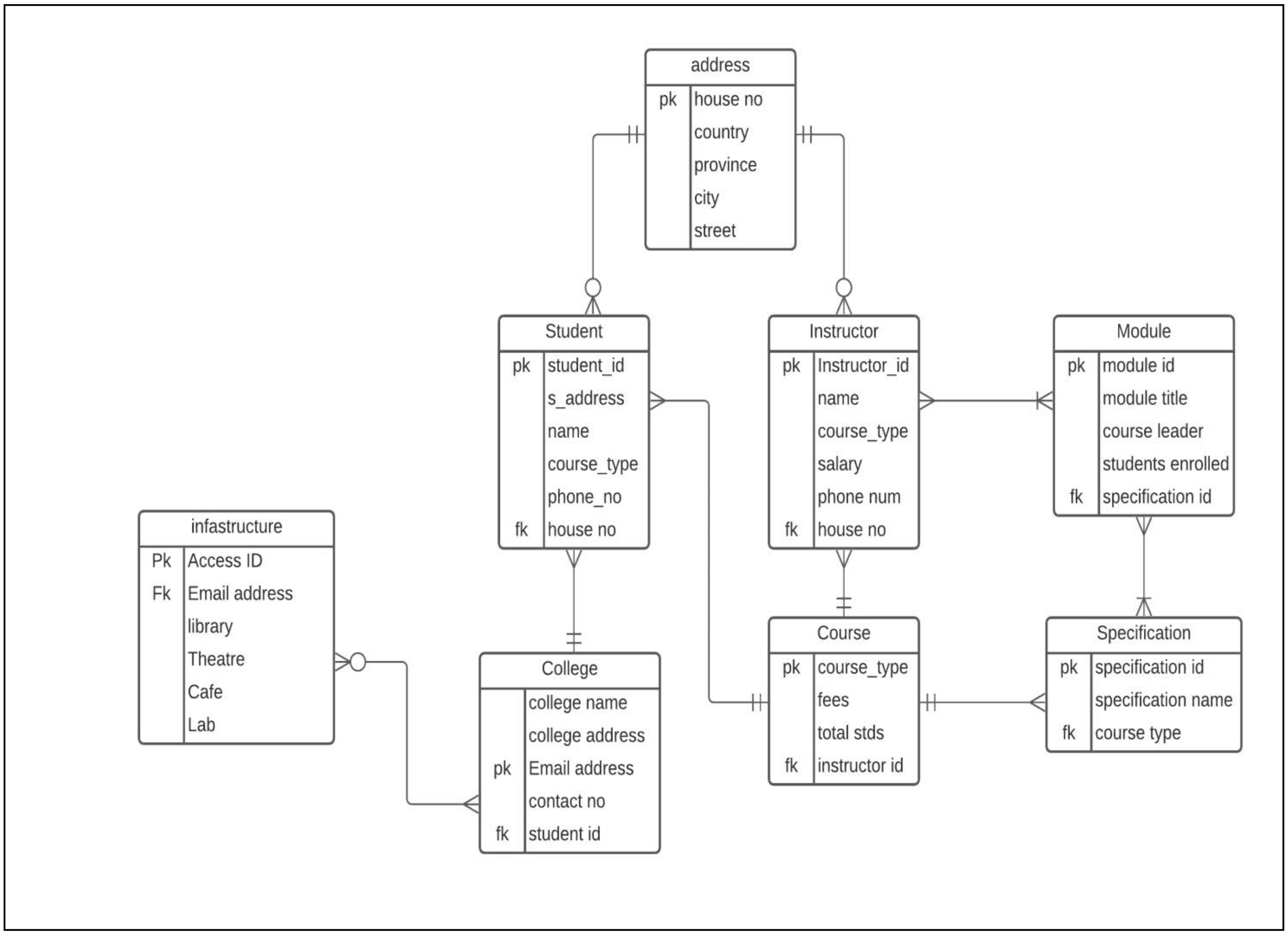


Figure 1: Initial ER Diagram

Anomalies are undesirable in any database. The existence of update, delete and insert anomalies causes data inconsistency, data redundancy and loss of unwanted data. We can see that the ER diagram represented above also has many anomalies. Therefore, in order to get rid of the anomalies normalization needs to be implemented.

1.5.2 Assumptions

Before normalizing the entities and attributes to 3NF (Third Normal Form), some of the assumptions are made which are listed below:

- Every address has a house number, country, province, city and street.
- Students and instructors can have only one mailing address.
- Module Id has been assigned for module and specification id for specifications.
- The infrastructure facilities are both for students and instructor.
- A specification can have many modules
- A module can belong to only one course.
- A module can have only one course leader
- An instructor can teach only one course at a time
- An instructor can be associated with many modules
- Student as well as the instructor must provide their contact number.

2. Normalization

Normalization is a process of organizing the data in database to avoid data redundancy, insertion anomaly, update anomaly & deletion anomaly. (SINGH, 2015)

2.1 UNF (Un Normalized Form)

Un-normalised form is a preparatory stage of the normalisation process allowing us to create a structured frame, representative of a piece of organisational data such as a form or document (e.g invoice, report, purchase order etc.). This is our initial Normalisation ‘relation’ that contains both real data, taken from the form or document, and modelled data, based upon and extended from the original from or document. (rdbms.opengrass, n.d.)

Applying UNF

To write the data in un-normalized form some steps must be followed. All the attributes from the ERD diagram are to be listed down with unique identifier. Then the repeating groups are kept within {}.

Showing repeating groups:

Specification (specification id, specification name, course type

{module id, module title, students enrolled{instructor id, name, phone num, salary{fees, total stds {students id, std_name, phone no, country id, province, city, street, fax no, access id, library, theatre, café, lab}

}

}

)

2.2 1NF (first normal form)

A database comes into 1Nf when the attributes of a table does not hold multiple values. It should hold only atomic values. (SINGH, 2015)

Specification (specification id, specification name, course type)

Specification_module-1(Specification id*, <u>module id</u>, module title, students enrolled, course leader)

Module_instructor-1(module id*, specification id*, <u>instructor id,</u> name, course type, phone num, salary, house no)

Instructor_course-1(specification id*, instructor id*, <u>course type</u>, fees, total st student id, name, phone no, country, province, city, street, fax no, access id, library, theatre, café, lab)

2.3 2NF (second normal form)

A table is said to be in 2NF if both the following conditions hold:

- Table is in 1NF (First normal form)
- No non-prime attribute is dependent on the proper subset of any candidate key of table.

Any subset of attributes of a table that can uniquely identify all the tuples of that table is known as a Super key. All the candidate keys are super keys but candidate keys do not have redundant attributes. This means that from a super key when we remove all the attributes that are unnecessary for its uniqueness, only then it becomes a candidate key. (krishnaswamy, 2019). An attribute that is not part of any candidate key is known as non-prime attribute. (SINGH, 2015)

Applying 2NF

The functional dependencies in 1NF are identified and each determinant primary key of new relation is made. Then all the attributes that depend on given determinant in relation is placed with that determinant as non-key attributes.

Checking partial dependency

Specification_Module-1(Specification Id*, Module Id, Module Title, Students Enrolled, Course Leader)

Specification Id → X

Module Id → Module Title, Students Enrolled, Course Leader

Specification Id, Module Id → X

Module_Instructor-1(Module Id*, Specification Id*, Instructor Id, Name, Course Type, Phone Num, Salary, House No)

Specification Id → X

Module Id → X

Instructor Id → Name, Course Type, Phone Num, Salary

Module Id, Instructor Id → X

Specification Id, Module Id → X

Specification Id, Module Id, Instructor Id → X

Specification_Instructor_Course-1 (Instructor Id*, Specification Id*, Course Type, Fees, Total Students, Student Id, student_Name, Phone No, Country, Province, house no, City, Street, Fax No, Access Id, Library, Theatre, Café, Lab)

Instructor Id → X

Specification Id → X

Course Type → Fees, Total Students

Module Id, Course Type → X

Module Id → Specification Id → X

Specification Id, Course Type → X

Specification Id, Module Id, Course Type → Student Id, Name, Phone No, Country, Province, City, Street, Fax No, Access Id, Library, Theatre, Café, Lab

2 NF

Specification -2 (Specification Id, Specification Name, Course Type*)

Module -2 (Module Id, Module Title, Students Enrolled, Course Leader, Specification Id*, Instructor Id*)

Module_Specification-2 (Specification Id*, Module Id*)

Instructor -2 (Instructor Id, Name, Course Type*, Salary, Phone Num, House No)

Module_Instructor -2 (Module Id*, Instructor Id*)

Specification_Instructor -2(Specification Id*, Instructor Id*)

Specification_Module_Instructor -2(Specification Id*, Module Id, Instructor Id*)

Course-2 (Course Type*, Fees, Total Students, Instructor Id*)

Specification_Module_Course-2(Specification Id*, Module Id*, Course Type*, Student Id, student_Name, Phone No, Country Id, Province, house no, City, Street, Fax No, Access Id, Library, Theatre, Café, Lab)

Specification_Course -2 (Specification Id *, Course Type*)

Module_Course -2 (Module Id*, Course Type*)

2.4 3NF (third normal form)

In order for the database to be in 3NF , the first rule is to have 2 NF and second is to not have any transitive functional dependencies. A transitive dependency in a database is an indirect relationship between values in the same table that causes a functional dependency. (guru99, 2005)

Applying 3NF

All the relations of 2NF are checked for transitive dependency and for each determinant in transitive dependency, a new relation is created.

Checking transitive dependency

Specification_Module_Course-2(Specification Id*, Module Id*, Course Type*, Student Id, Name, Phone No, Country Id, Province, City, Street, Fax No, Access Id, Library, Theatre, Café, Lab)

Specification Id, Module Id, Course Type → Student Id → Name

Specification Id, Module Id, Course Type → Student Id → Phone Number

Specification Id, Module Id, Course Type → Access Id → Library

Specification Id, Module Id, Course Type → Access Id → Theatre

Specification Id, Module Id, Course Type → Access Id → Café

Specification Id, Module Id, Course Type → Country Id →city

Specification Id, Module Id, Course Type → Country Id → Street

Specification Id, Module Id, Course Type → Country Id → Province

Instructor, Course Type → Salary

3 NF

Specification -3 (<u>Specification Id</u>, Specification Name, Course Type*)

Module -3 (<u>Module Id</u>, Module Title, Students Enrolled, Course Leader, Specification Id*, Instructor Id*)

Module_Specification -3 (Module Id*, Specification Id*)

Instructor -3 (<u>Instructor Id</u>, Name, Phone Number)

Instructor_Course Type -3 (Instructor Id*, Course Type*)

Course Type -3 (Salary, Course Type*)

Module_Instructor-3 (Module Id*, Instructor Id*)

Specification_Course-3 (Specification Id*, Course Type*)

Specification_Module_Course-3 (Specification Id*, Module Id*, Course Type*, Student Id*, Access Id*, Country Id*)

Course-3 (Course Type, Fees, Total Students, Instructor Id*)

Student -3 (Student Id, student_Name, Phone No, Specification Id*, Course Type*)

Infrastructure-3 (Access Id, Library, Café, Theatre)

Address-3 (Country Id, City, Street, Fax No, Province,house no)

Student_Address-3 (Student Id*, Country Id*)

Instructor_Address-3(Instructor Id*, Country Id*)

Student_Instructor_Address-3 (Student Id*, Instructor Id*, Country Id*)

Final Table

Specification -3 (Specification Id, Specification Name, Course Type*)

Module -3 (Module Id, Module Title, Students Enrolled, Course Leader, Specification Id*, Instructor Id*)

Specification_Module_Course (Specification Id*, Module Id*, Course Type*, Student Id*, Access Id*, Country Id*)

Instructor -3 (Instructor Id, Name, Phone Number)

Course Type -3 (Salary, Course Type*)

Course-3 (Course Type, Fees, Total Students, Instructor Id*)

Student -3 (Student Id, Name, Phone No, Specification Id*, Course Type*)

Infrastructure-3 (Access Id, Library, Café, Theatre)

Address-3 (Country Id, City, Street, Fax No, Province,house no)

Student_Address-3 (Student Id*, Country Id*)

Instructor_Address-3(Instructor Id*, Country Id*)

Note: In the above final table section some relations are removed because of its futility in the database. Module_specification-3, Specification_course-3 are removed as the table specification_module_course-3 gives all its details. As all the relations are shown by the specification_module_course-3 table it is unnecessary to repeat the relations. The aim of the normalization is to minimize the data redundancy but by including the relations twice the data redundancy increases for which the unnecessary relations are eliminated. Student_Instructor_Address-3 is also removed because of its inconveniency. Altogether, the four relations i.e. Patient_Staff-3, Staff_Appointment-3, Patient_Appointment-3 and Student_Instructor_Address-3 are removed in the final table.

3. Final ER Diagram after normalization

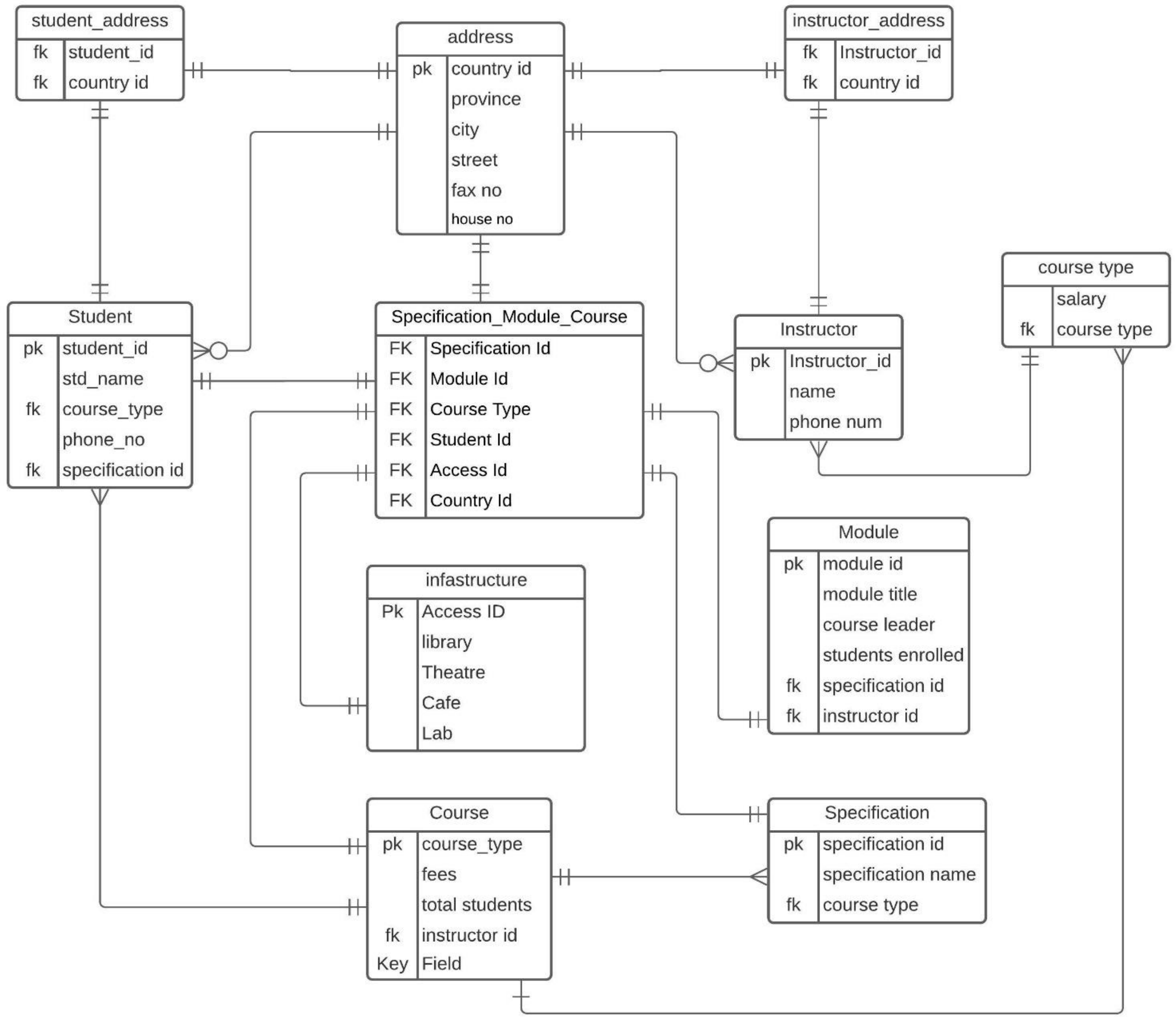


Figure 2: Final ER Diagram after normalization

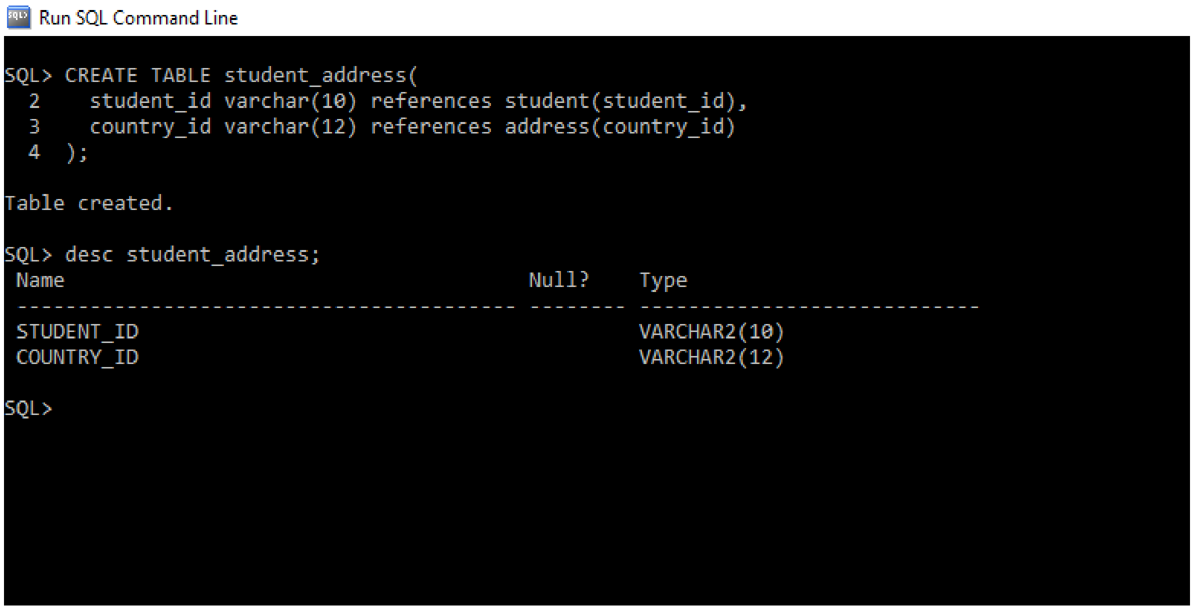
During the conceptual data modeling phase, data requirements are expressed through an ERD. The conceptual data modeling phase in general is independent of a DBMS. Performing normalization during ERD development can improve the conceptual model, and speed its implementation. One of the ways an ERD is enhanced during the logical design phase is through the process of normalization. It is the process of removing redundancy in a table. It usually involves dividing an entity table into two or more tables and defining relationships between the tables.

The ERD above has received many extensions and variations. This transformation involves the addition of an entity type and a one-to-many relationship. This transformation can be useful to record a finer level of detail about an entity. The ER diagram transforms a weak entity type into a strong entity type. This transformation is most useful for associative entity types. (Kaula, 2007).

The final Erd now has a better handle on database security with much more flexible database design. With reduction of redundant data and data inconsistency this Erd provides for a greater overall database organization. (Stephens, 2003)

4. Database Implementation

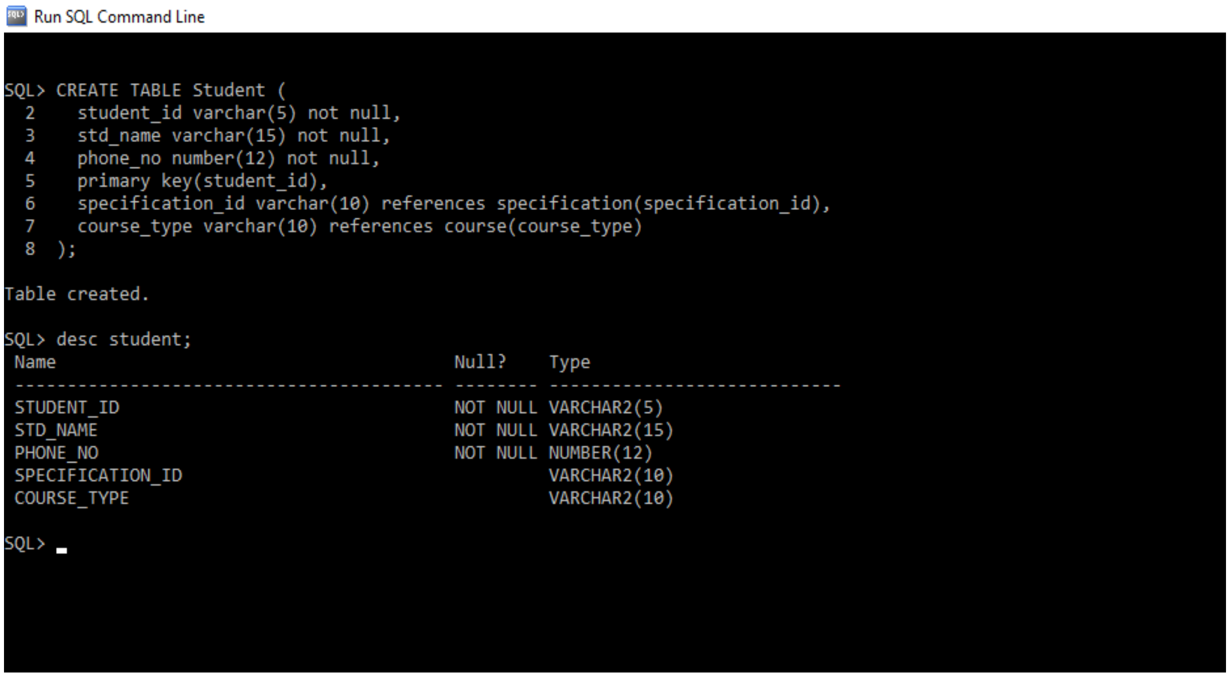
4.1 Table Generation (DDL Scripts)



Run SQL Command Line

```
SQL> CREATE TABLE student_address(
       student_id varchar(10) references student(student_id),
       country_id varchar(12) references address(country_id)
     );

Table created.

SQL> desc student_address;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 STUDENT_ID                                         VARCHAR2(10)
 COUNTRY_ID                                         VARCHAR2(12)

SQL>
```

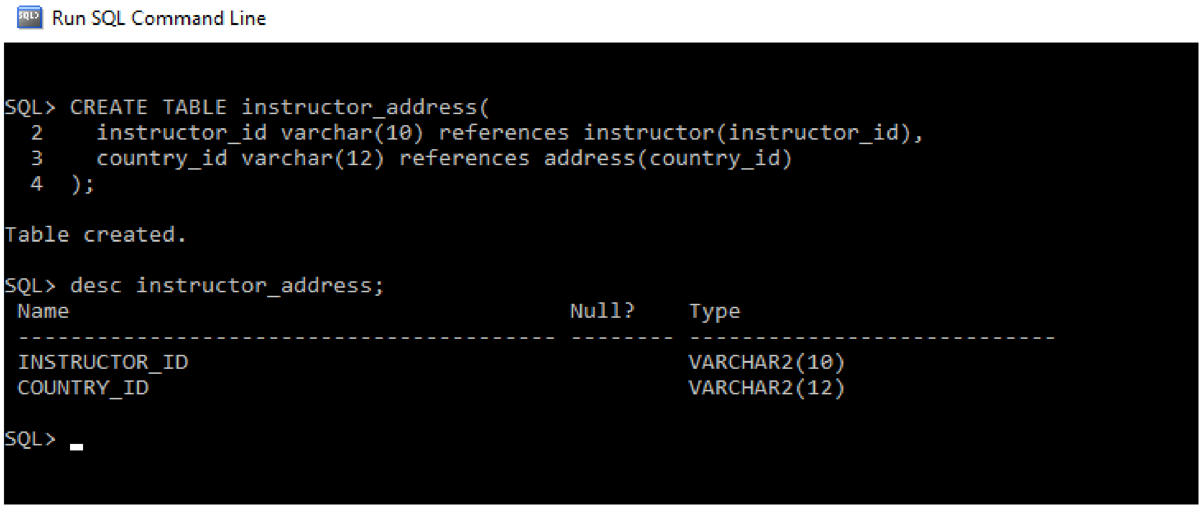
Figure 3: create table student_address



Run SQL Command Line

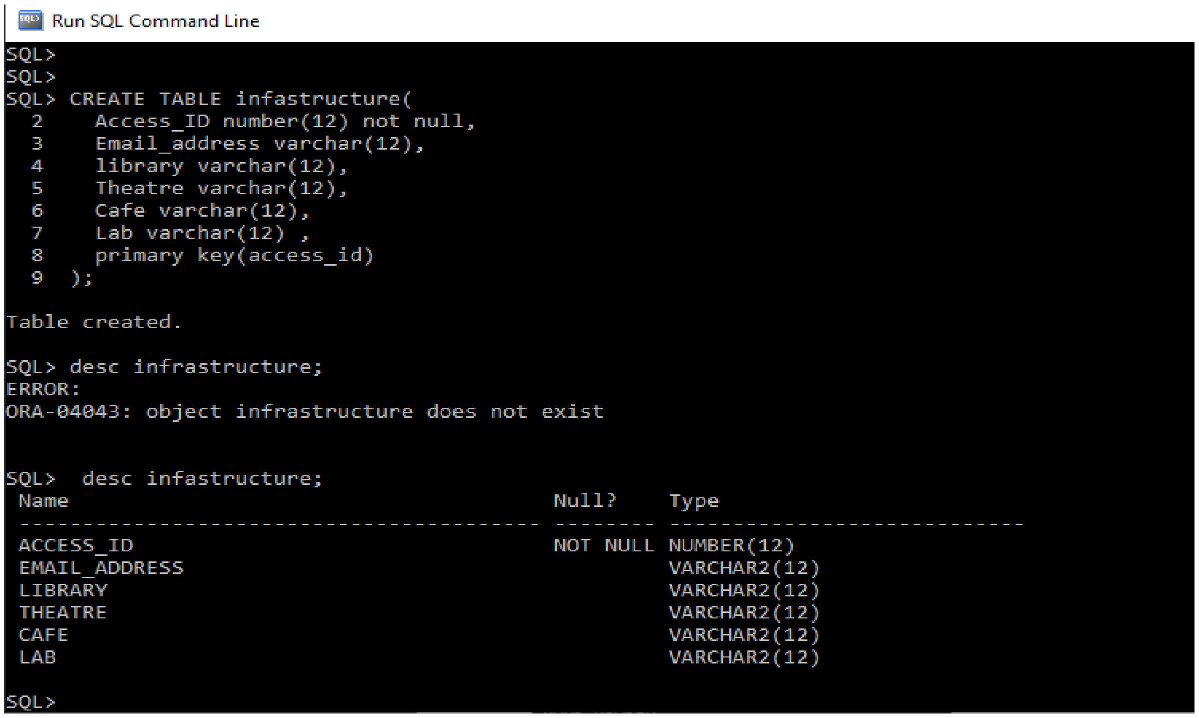
```
SQL> CREATE TABLE Student (
       student_id varchar(5) not null,
       std_name varchar(15) not null,
       phone_no number(12) not null,
       primary key(student_id),
       specification_id varchar(10) references specification(specification_id),
       course_type varchar(10) references course(course_type)
     );

Table created.

SQL> desc student;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 STUDENT_ID                                NOT NULL VARCHAR2(5)
 STD_NAME                                  NOT NULL VARCHAR2(15)
 PHONE_NO                                  NOT NULL NUMBER(12)
 SPECIFICATION_ID                                   VARCHAR2(10)
 COURSE_TYPE                                        VARCHAR2(10)

SQL>
```

Figure 4: create table student



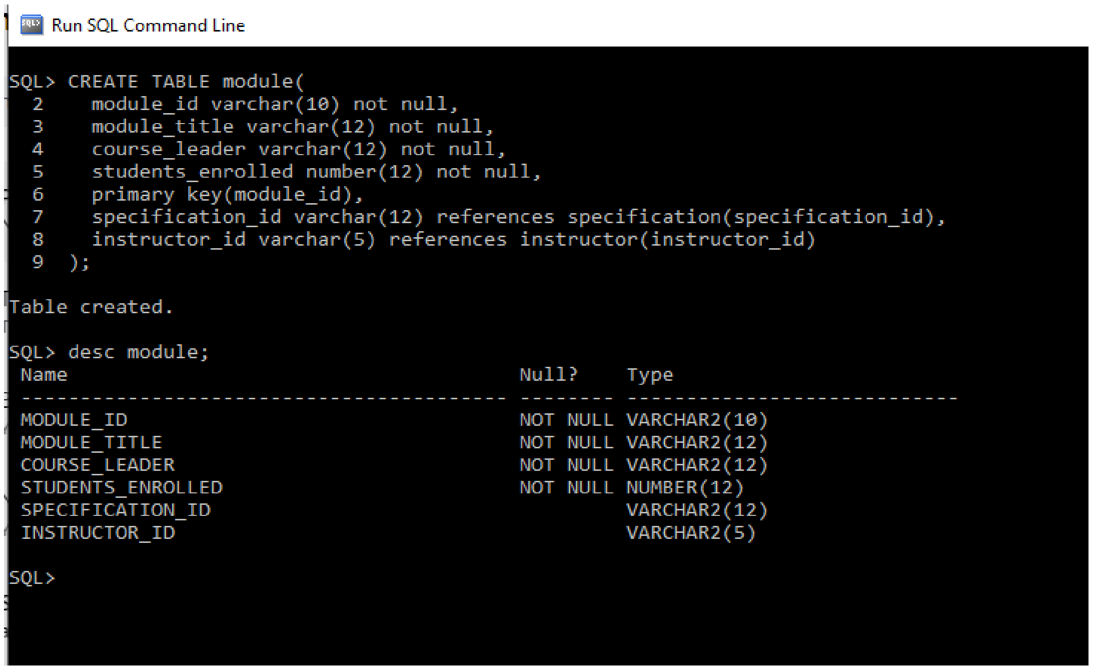
```
SQL> CREATE TABLE instructor_address(
  2    instructor_id varchar(10) references instructor(instructor_id),
  3    country_id varchar(12) references address(country_id)
  4  );

Table created.

SQL> desc instructor_address;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 INSTRUCTOR_ID                                      VARCHAR2(10)
 COUNTRY_ID                                         VARCHAR2(12)

SQL>
```

Figure 5: create table instructor_address



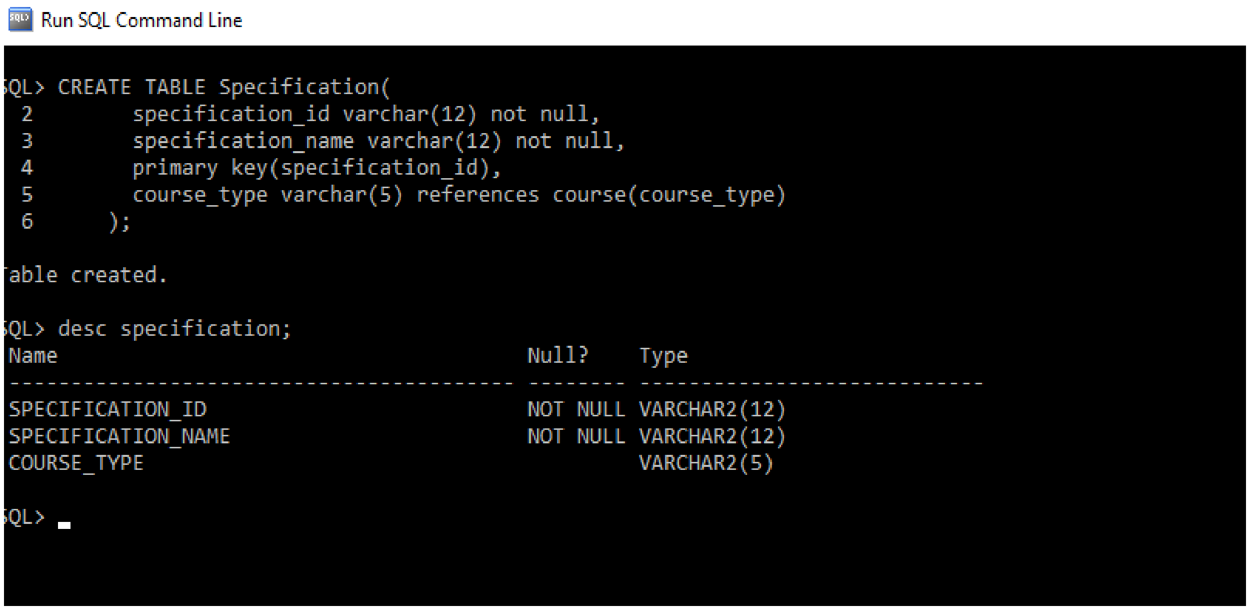
```
SQL>
SQL>
SQL> CREATE TABLE infastructure(
  2    Access_ID number(12) not null,
  3    Email_address varchar(12),
  4    library varchar(12),
  5    Theatre varchar(12),
  6    Cafe varchar(12),
  7    Lab varchar(12) ,
  8    primary key(access_id)
  9  );

Table created.

SQL> desc infrastructure;
ERROR:
ORA-04043: object infrastructure does not exist


SQL>  desc infastructure;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ACCESS_ID                                 NOT NULL NUMBER(12)
 EMAIL_ADDRESS                                      VARCHAR2(12)
 LIBRARY                                            VARCHAR2(12)
 THEATRE                                            VARCHAR2(12)
 CAFE                                               VARCHAR2(12)
 LAB                                                VARCHAR2(12)

SQL>
```

Figure 6: create table infastructure



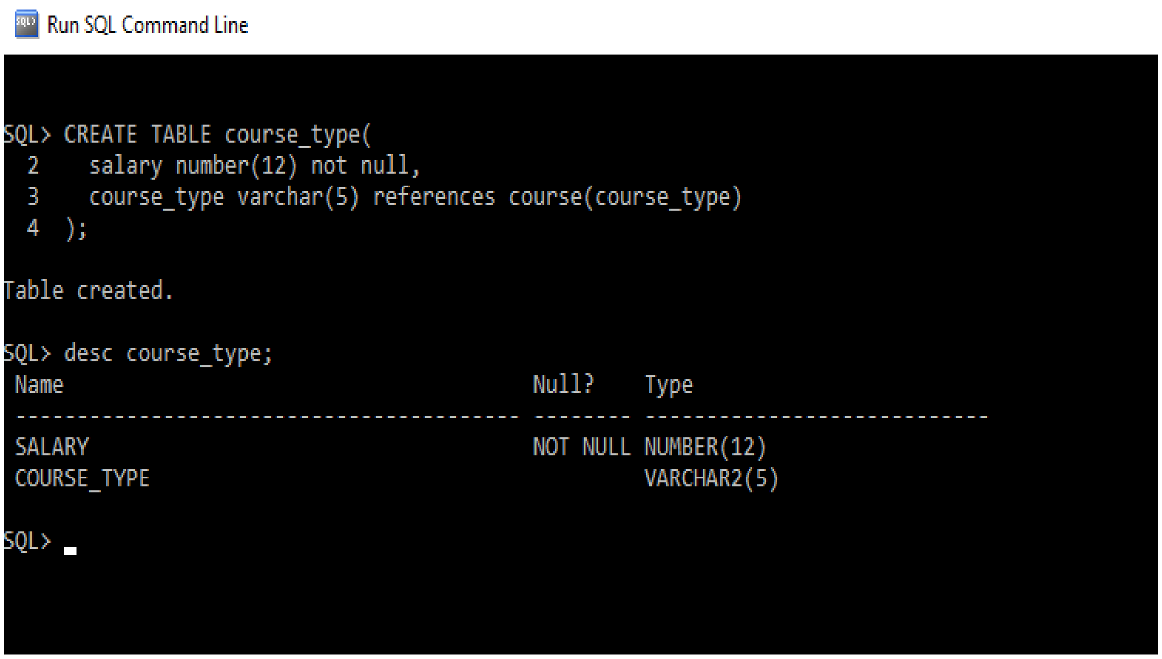
```
SQL> CREATE TABLE module(
       module_id varchar(10) not null,
       module_title varchar(12) not null,
       course_leader varchar(12) not null,
       students_enrolled number(12) not null,
       primary key(module_id),
       specification_id varchar(12) references specification(specification_id),
       instructor_id varchar(5) references instructor(instructor_id)
     );

Table created.

SQL> desc module;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 MODULE_ID                                 NOT NULL VARCHAR2(10)
 MODULE_TITLE                              NOT NULL VARCHAR2(12)
 COURSE_LEADER                             NOT NULL VARCHAR2(12)
 STUDENTS_ENROLLED                         NOT NULL NUMBER(12)
 SPECIFICATION_ID                                   VARCHAR2(12)
 INSTRUCTOR_ID                                      VARCHAR2(5)

SQL>
```

Figure 7: create table infastructure



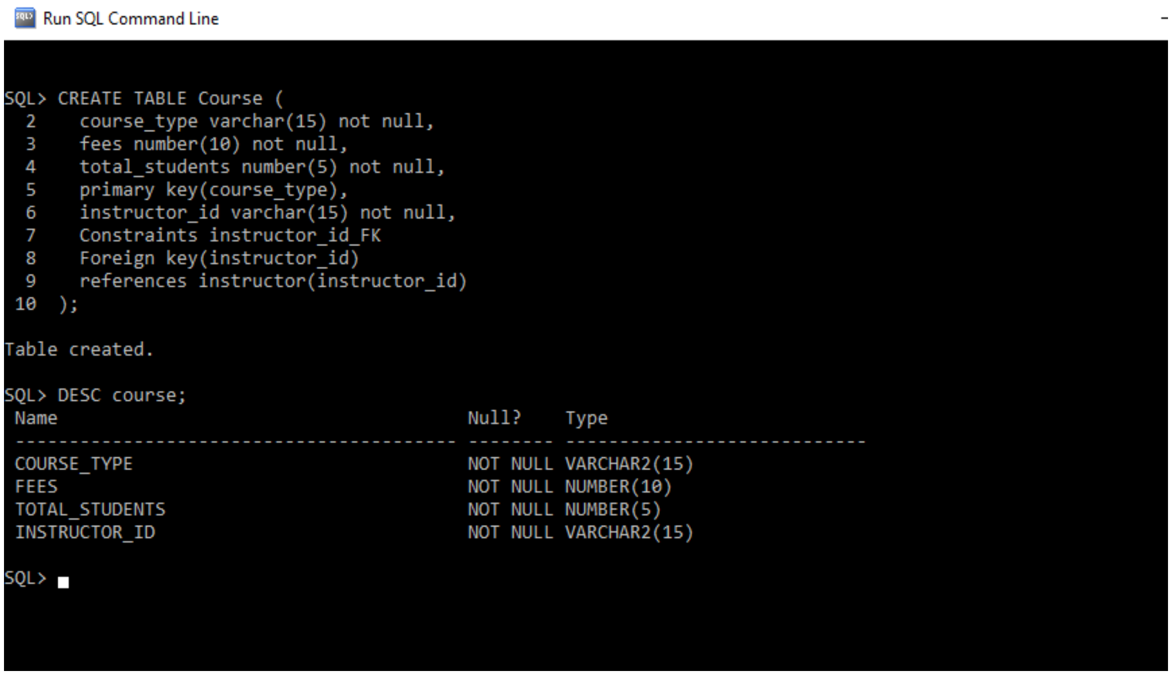
```
QL> CREATE TABLE Specification(
          specification_id varchar(12) not null,
          specification_name varchar(12) not null,
          primary key(specification_id),
          course_type varchar(5) references course(course_type)
        );

able created.

QL> desc specification;
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
SPECIFICATION_ID                          NOT NULL VARCHAR2(12)
SPECIFICATION_NAME                        NOT NULL VARCHAR2(12)
COURSE_TYPE                                        VARCHAR2(5)

QL>
```

Figure 8: create table specification



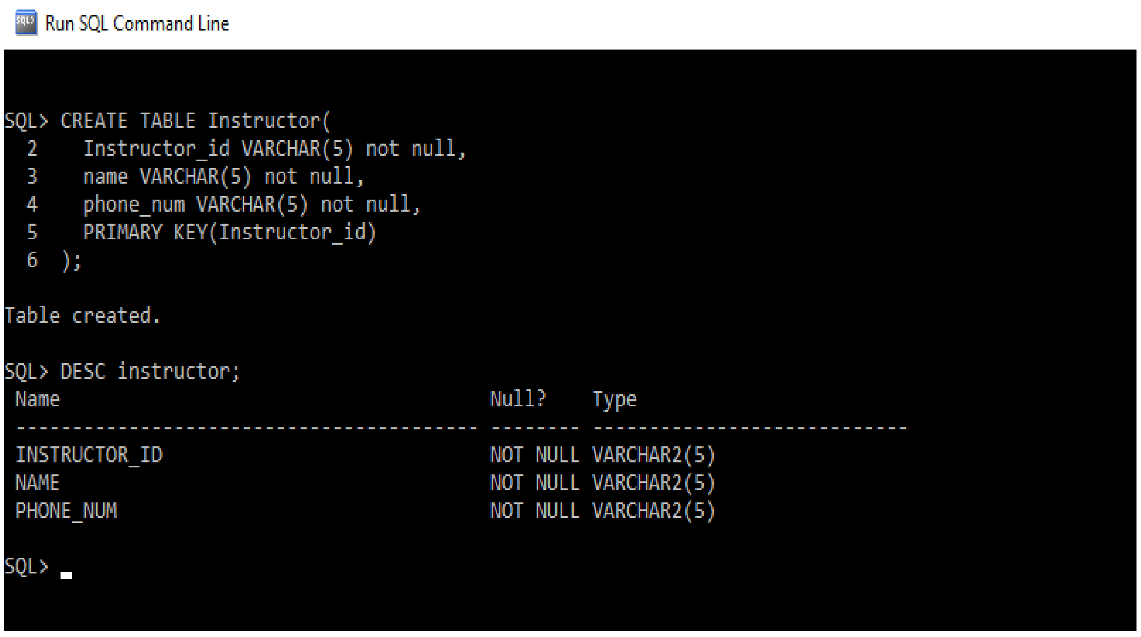
```
SQL> CREATE TABLE course_type(
    salary number(12) not null,
    course_type varchar(5) references course(course_type)
  );

Table created.

SQL> desc course_type;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SALARY                                    NOT NULL NUMBER(12)
 COURSE_TYPE                                        VARCHAR2(5)

SQL>
```

Figure 9: create table course_type



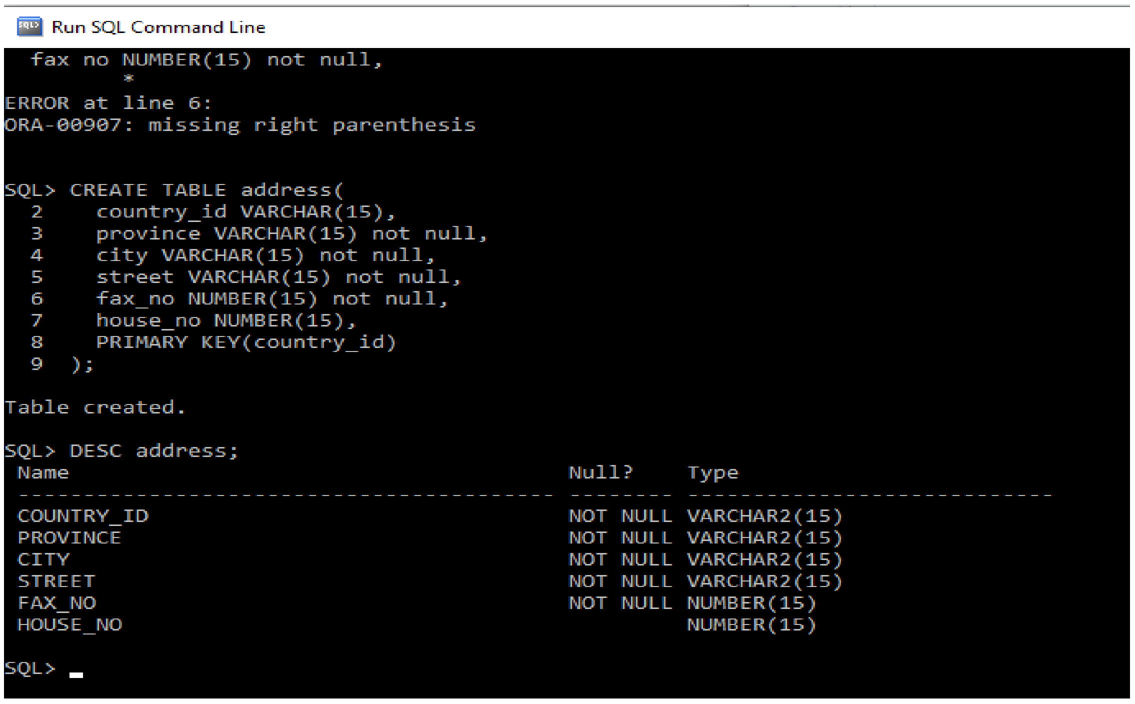
```
SQL> CREATE TABLE Course (
    course_type varchar(15) not null,
    fees number(10) not null,
    total_students number(5) not null,
    primary key(course_type),
    instructor_id varchar(15) not null,
    Constraints instructor_id_FK
    Foreign key(instructor_id)
    references instructor(instructor_id)
  );

Table created.

SQL> DESC course;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 COURSE_TYPE                               NOT NULL VARCHAR2(15)
 FEES                                      NOT NULL NUMBER(10)
 TOTAL_STUDENTS                            NOT NULL NUMBER(5)
 INSTRUCTOR_ID                             NOT NULL VARCHAR2(15)

SQL>
```

Figure 10: create table course



```
SQL> CREATE TABLE Instructor(
       Instructor_id VARCHAR(5) not null,
       name VARCHAR(5) not null,
       phone_num VARCHAR(5) not null,
       PRIMARY KEY(Instructor_id)
     );

Table created.

SQL> DESC instructor;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 INSTRUCTOR_ID                             NOT NULL VARCHAR2(5)
 NAME                                      NOT NULL VARCHAR2(5)
 PHONE_NUM                                 NOT NULL VARCHAR2(5)

SQL> _
```

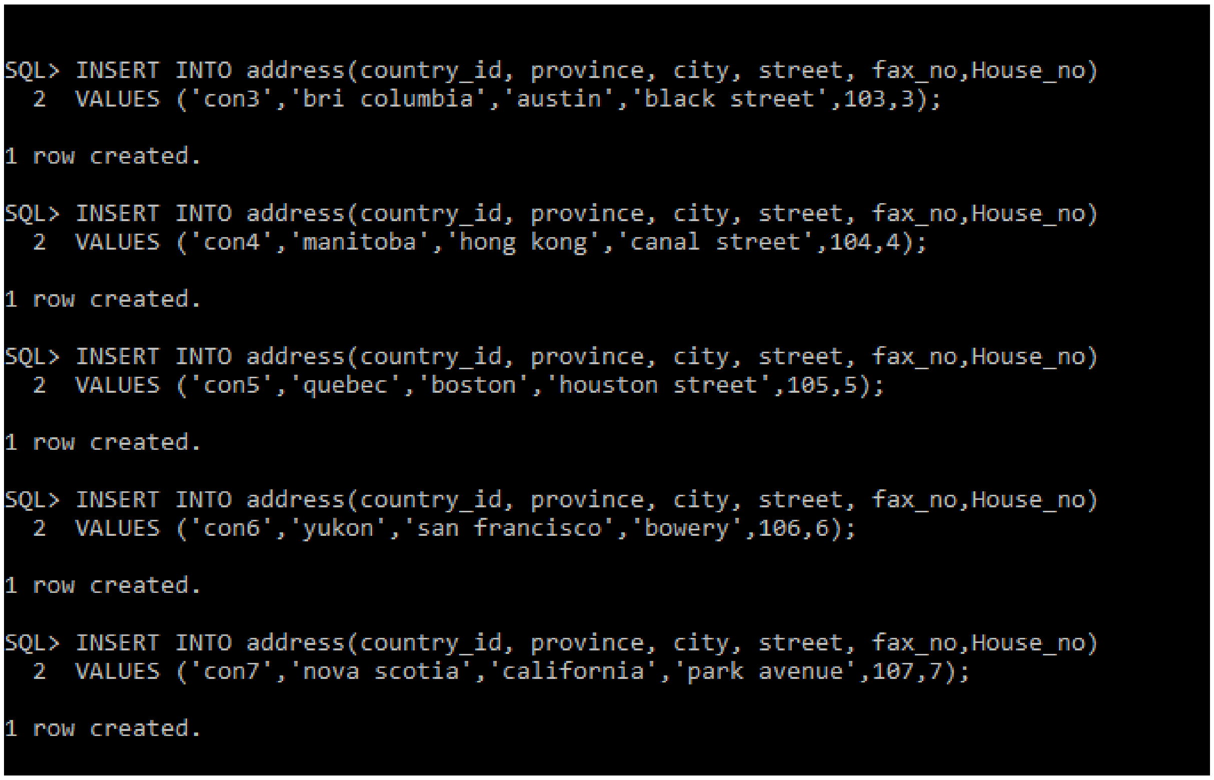
Figure 11: create table instructor



```
  fax no NUMBER(15) not null,
          *
ERROR at line 6:
ORA-00907: missing right parenthesis


SQL> CREATE TABLE address(
       country_id VARCHAR(15),
       province VARCHAR(15) not null,
       city VARCHAR(15) not null,
       street VARCHAR(15) not null,
       fax_no NUMBER(15) not null,
       house_no NUMBER(15),
       PRIMARY KEY(country_id)
     );

Table created.

SQL> DESC address;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 COUNTRY_ID                                NOT NULL VARCHAR2(15)
 PROVINCE                                  NOT NULL VARCHAR2(15)
 CITY                                      NOT NULL VARCHAR2(15)
 STREET                                    NOT NULL VARCHAR2(15)
 FAX_NO                                    NOT NULL NUMBER(15)
 HOUSE_NO                                           NUMBER(15)

SQL> _
```

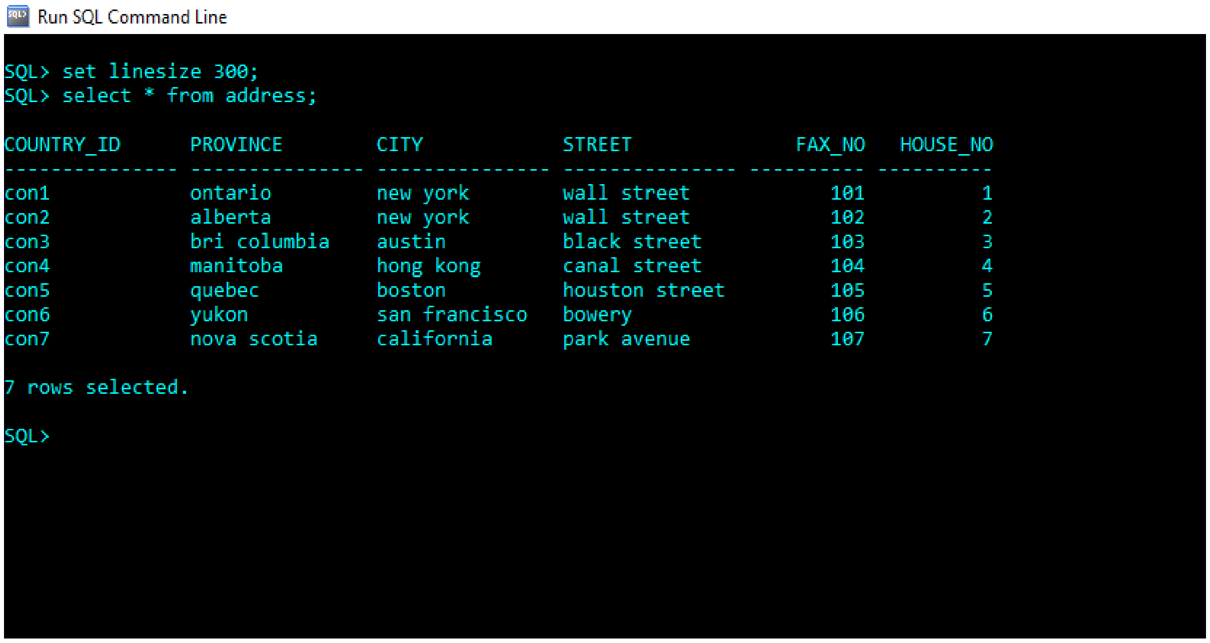
Figure 12: create table address

4.2 Populating Database



```
SQL> INSERT INTO address(country_id, province, city, street, fax_no,House_no)
  2  VALUES ('con3','bri columbia','austin','black street',103,3);

1 row created.

SQL> INSERT INTO address(country_id, province, city, street, fax_no,House_no)
  2  VALUES ('con4','manitoba','hong kong','canal street',104,4);

1 row created.

SQL> INSERT INTO address(country_id, province, city, street, fax_no,House_no)
  2  VALUES ('con5','quebec','boston','houston street',105,5);

1 row created.

SQL> INSERT INTO address(country_id, province, city, street, fax_no,House_no)
  2  VALUES ('con6','yukon','san francisco','bowery',106,6);

1 row created.

SQL> INSERT INTO address(country_id, province, city, street, fax_no,House_no)
  2  VALUES ('con7','nova scotia','california','park avenue',107,7);

1 row created.
```

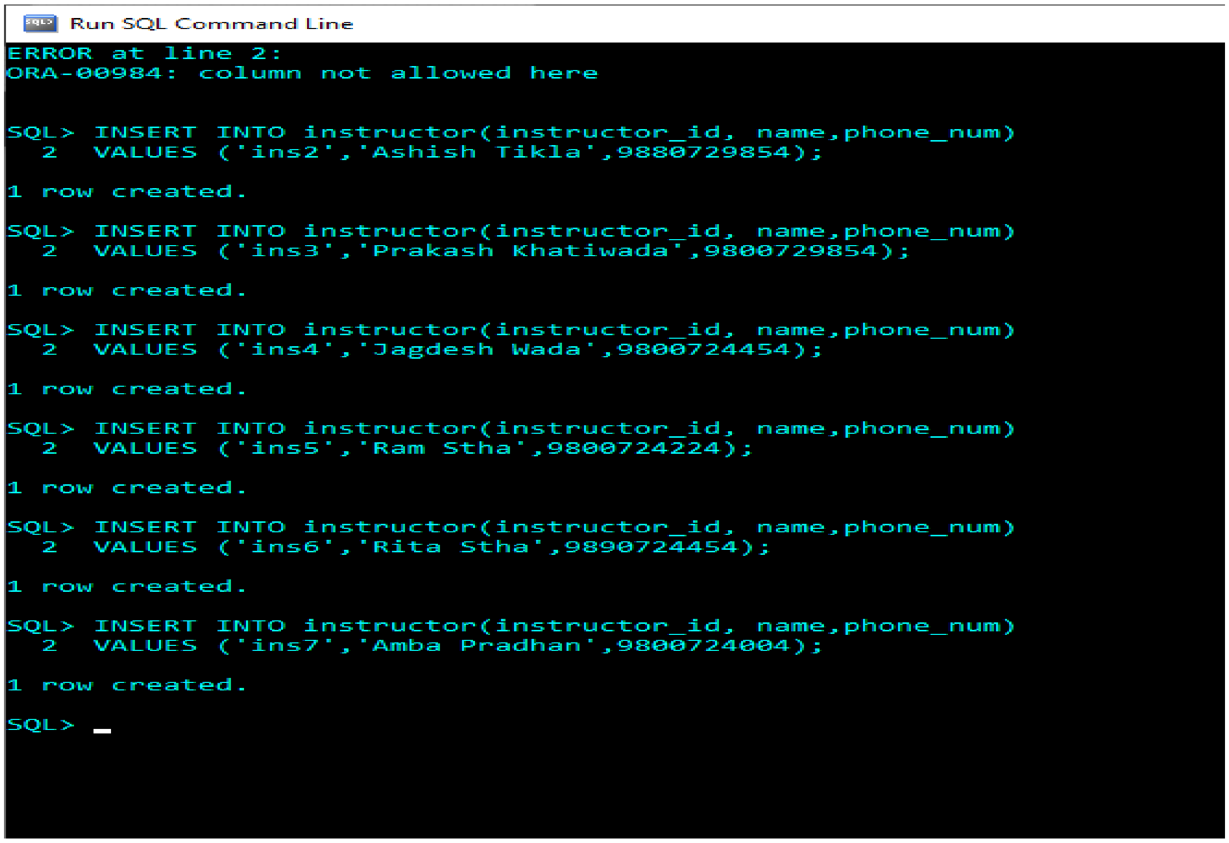
Figure 13: insert in table address



Run SQL Command Line

```
SQL> set linesize 300;
SQL> select * from address;

COUNTRY_ID      PROVINCE        CITY            STREET              FAX_NO   HOUSE_NO
--------------- --------------- --------------- --------------- ---------- ----------
con1            ontario         new york        wall street            101          1
con2            alberta         new york        wall street            102          2
con3            bri columbia    austin          black street           103          3
con4            manitoba        hong kong       canal street           104          4
con5            quebec          boston          houston street         105          5
con6            yukon           san francisco   bowery                 106          6
con7            nova scotia     california      park avenue            107          7

7 rows selected.

SQL>
```

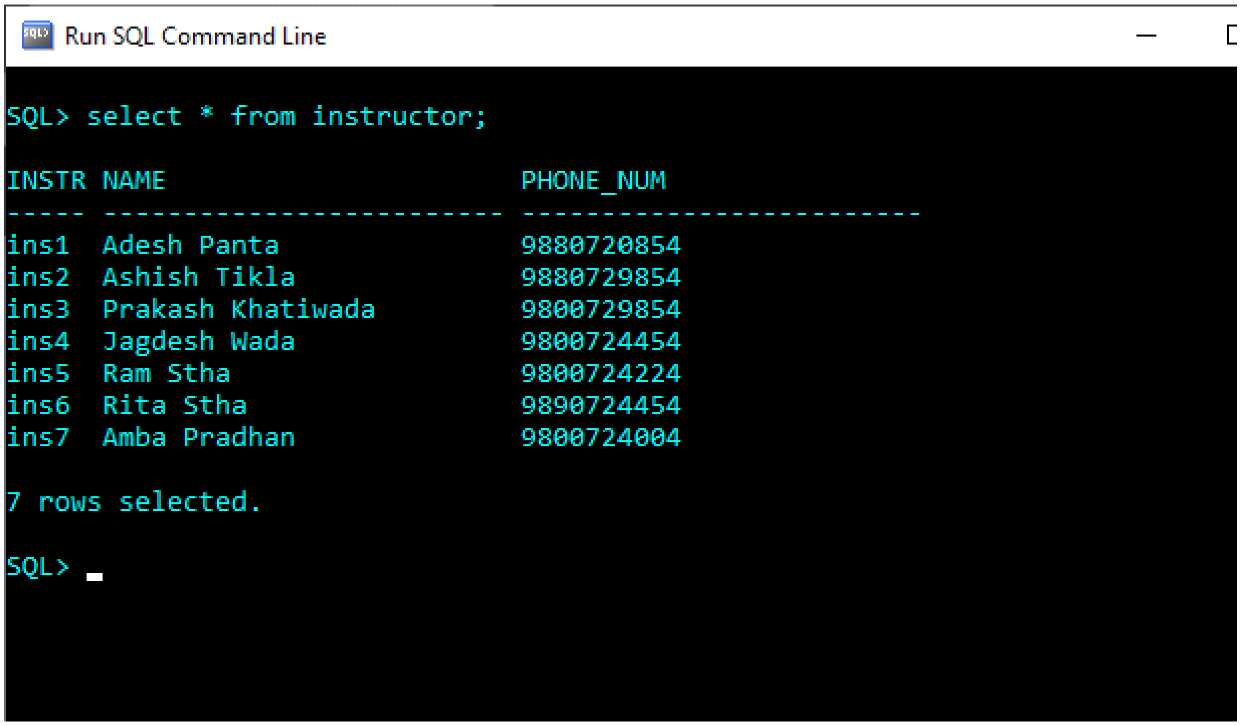
Figure 14: table address



Run SQL Command Line

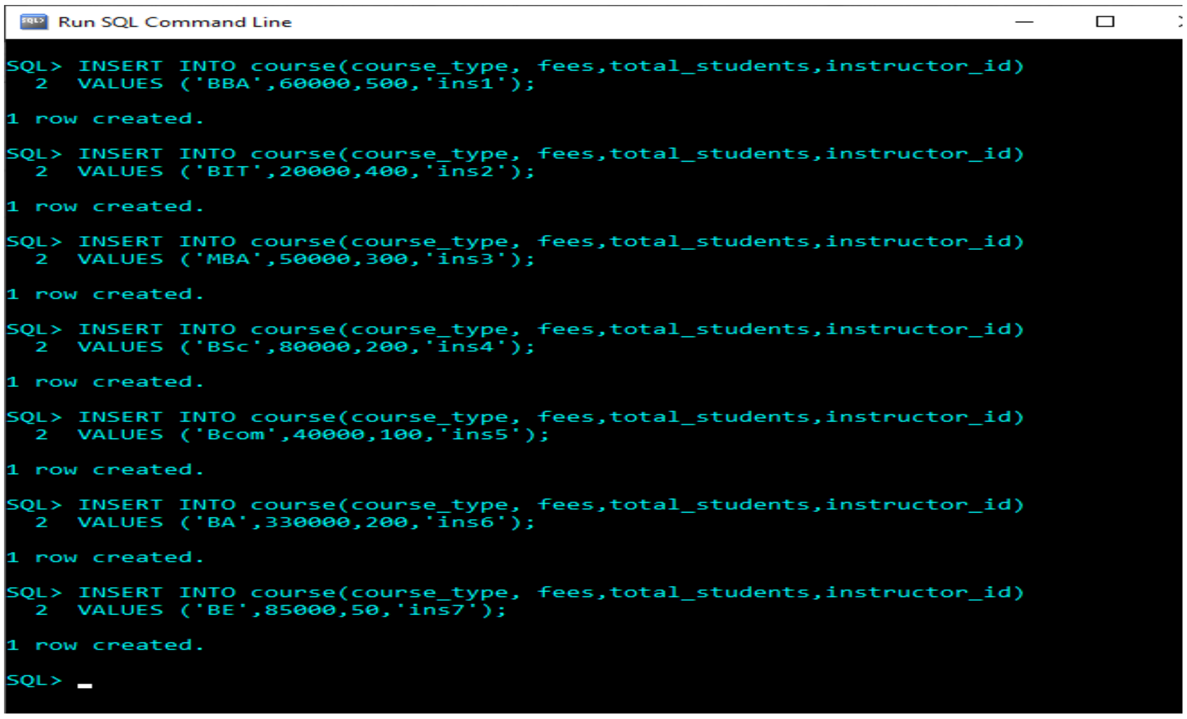
```
ERROR at line 2:
ORA-00984: column not allowed here


SQL> INSERT INTO instructor(instructor_id, name,phone_num)
  2  VALUES ('ins2','Ashish Tikla',9880729854);

1 row created.

SQL> INSERT INTO instructor(instructor_id, name,phone_num)
  2  VALUES ('ins3','Prakash Khatiwada',9800729854);

1 row created.

SQL> INSERT INTO instructor(instructor_id, name,phone_num)
  2  VALUES ('ins4','Jagdesh Wada',9800724454);

1 row created.

SQL> INSERT INTO instructor(instructor_id, name,phone_num)
  2  VALUES ('ins5','Ram Stha',9800724224);

1 row created.

SQL> INSERT INTO instructor(instructor_id, name,phone_num)
  2  VALUES ('ins6','Rita Stha',9890724454);

1 row created.

SQL> INSERT INTO instructor(instructor_id, name,phone_num)
  2  VALUES ('ins7','Amba Pradhan',9800724004);

1 row created.

SQL> _
```

Figure 15: insert in table instructor



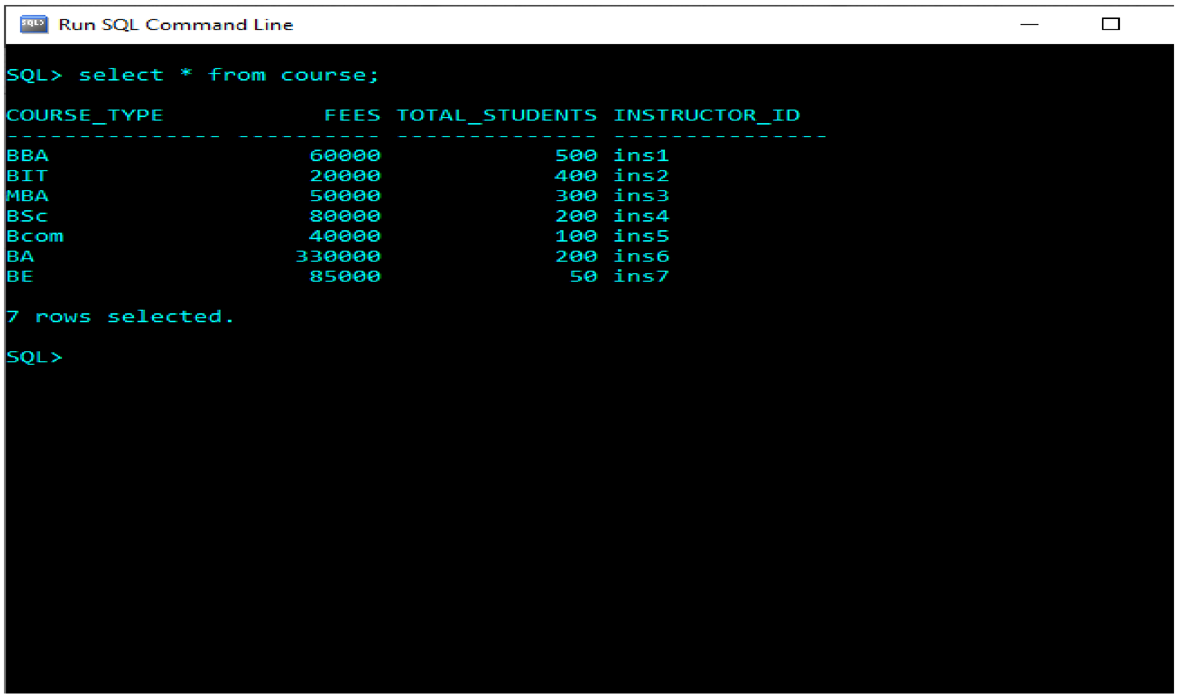
```
SQL> select * from instructor;

INSTR NAME                      PHONE_NUM
----- ------------------------- -------------------------
ins1  Adesh Panta               9880720854
ins2  Ashish Tikla              9880729854
ins3  Prakash Khatiwada         9800729854
ins4  Jagdesh Wada              9800724454
ins5  Ram Stha                  9800724224
ins6  Rita Stha                 9890724454
ins7  Amba Pradhan              9800724004

7 rows selected.

SQL>
```

Figure 16: table instructor



```
SQL> INSERT INTO course(course_type, fees,total_students,instructor_id)
  2  VALUES ('BBA',60000,500,'ins1');

1 row created.

SQL> INSERT INTO course(course_type, fees,total_students,instructor_id)
  2  VALUES ('BIT',20000,400,'ins2');

1 row created.

SQL> INSERT INTO course(course_type, fees,total_students,instructor_id)
  2  VALUES ('MBA',50000,300,'ins3');

1 row created.

SQL> INSERT INTO course(course_type, fees,total_students,instructor_id)
  2  VALUES ('BSc',80000,200,'ins4');

1 row created.

SQL> INSERT INTO course(course_type, fees,total_students,instructor_id)
  2  VALUES ('Bcom',40000,100,'ins5');

1 row created.

SQL> INSERT INTO course(course_type, fees,total_students,instructor_id)
  2  VALUES ('BA',330000,200,'ins6');

1 row created.

SQL> INSERT INTO course(course_type, fees,total_students,instructor_id)
  2  VALUES ('BE',85000,50,'ins7');

1 row created.

SQL>
```

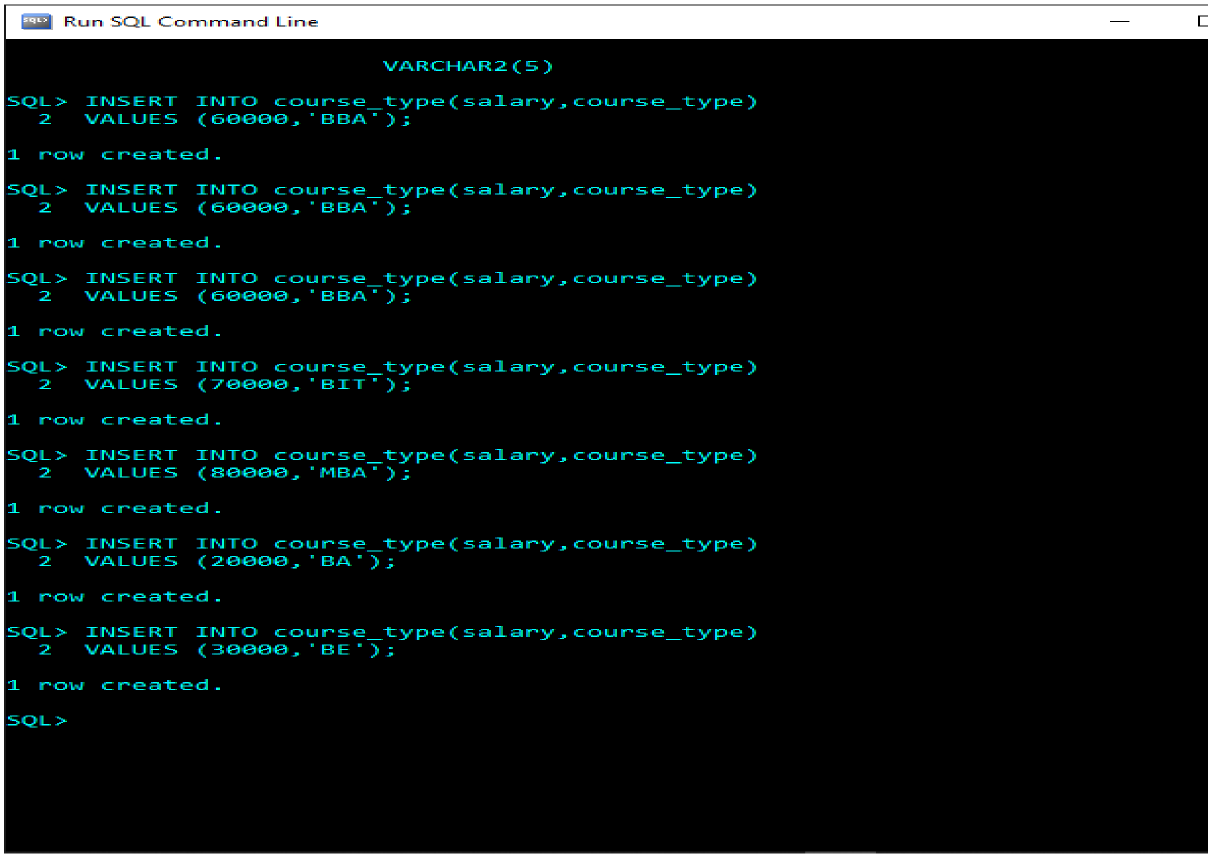
Figure 17: insert in table course



Run SQL Command Line

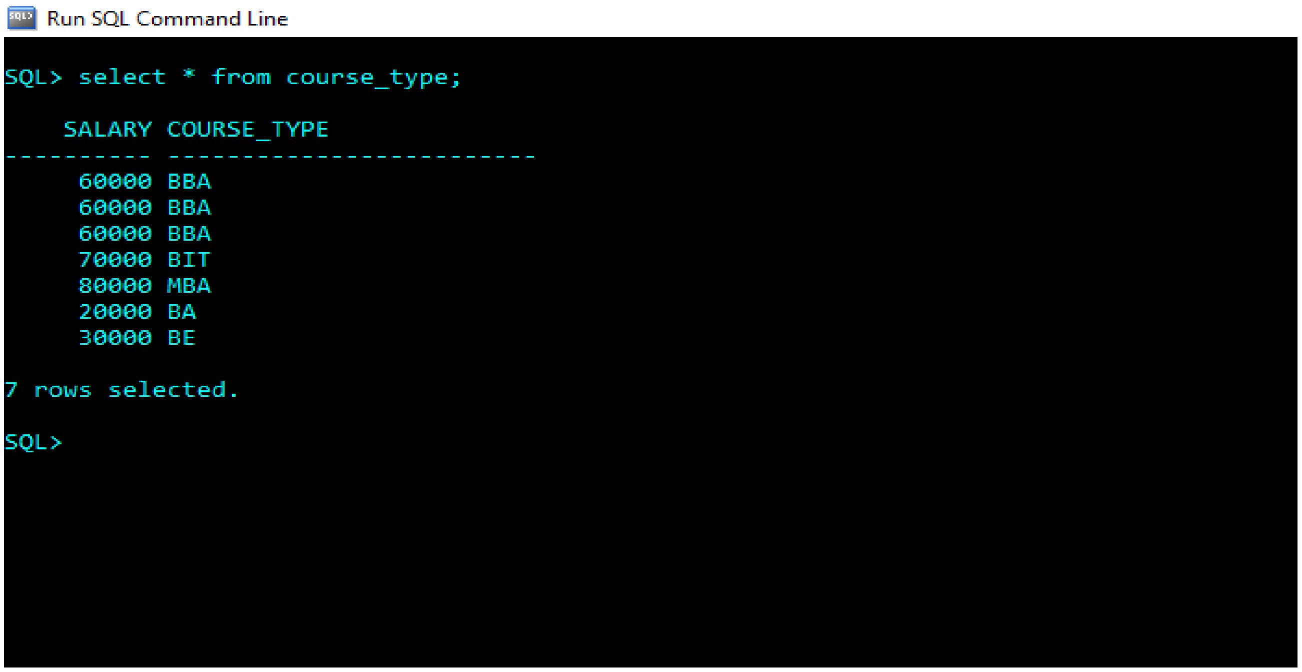
```
SQL> select * from course;

COURSE_TYPE                FEES TOTAL_STUDENTS INSTRUCTOR_ID
--------------- ---------- -------------- ---------------
BBA                       60000            500 ins1
BIT                       20000            400 ins2
MBA                       50000            300 ins3
BSc                       80000            200 ins4
Bcom                      40000            100 ins5
BA                       330000            200 ins6
BE                        85000             50 ins7

7 rows selected.

SQL>
```

Figure 18: table course



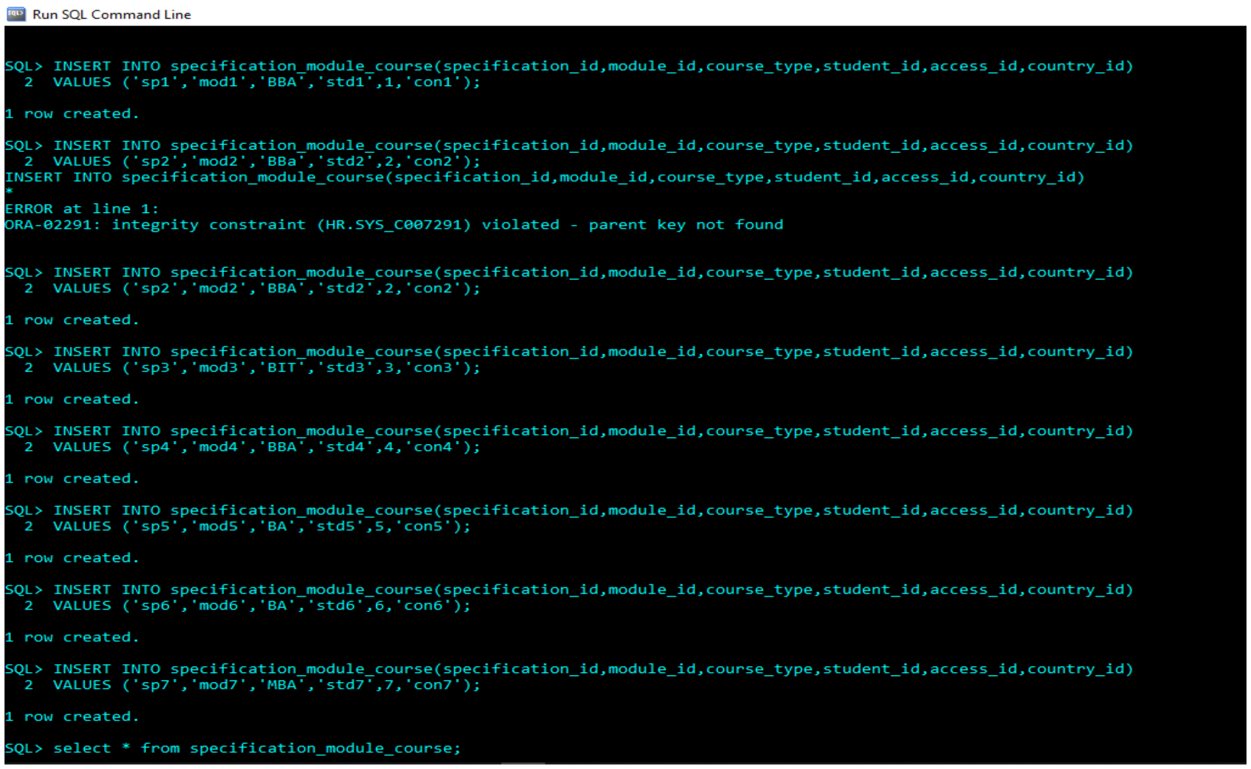
```
Run SQL Command Line
                                    VARCHAR2(5)

SQL> INSERT INTO course_type(salary,course_type)
  2  VALUES (60000,'BBA');

1 row created.

SQL> INSERT INTO course_type(salary,course_type)
  2  VALUES (60000,'BBA');

1 row created.

SQL> INSERT INTO course_type(salary,course_type)
  2  VALUES (60000,'BBA');

1 row created.

SQL> INSERT INTO course_type(salary,course_type)
  2  VALUES (70000,'BIT');

1 row created.

SQL> INSERT INTO course_type(salary,course_type)
  2  VALUES (80000,'MBA');

1 row created.

SQL> INSERT INTO course_type(salary,course_type)
  2  VALUES (20000,'BA');

1 row created.

SQL> INSERT INTO course_type(salary,course_type)
  2  VALUES (30000,'BE');

1 row created.

SQL>
```

Figure 19: insert in table course_type



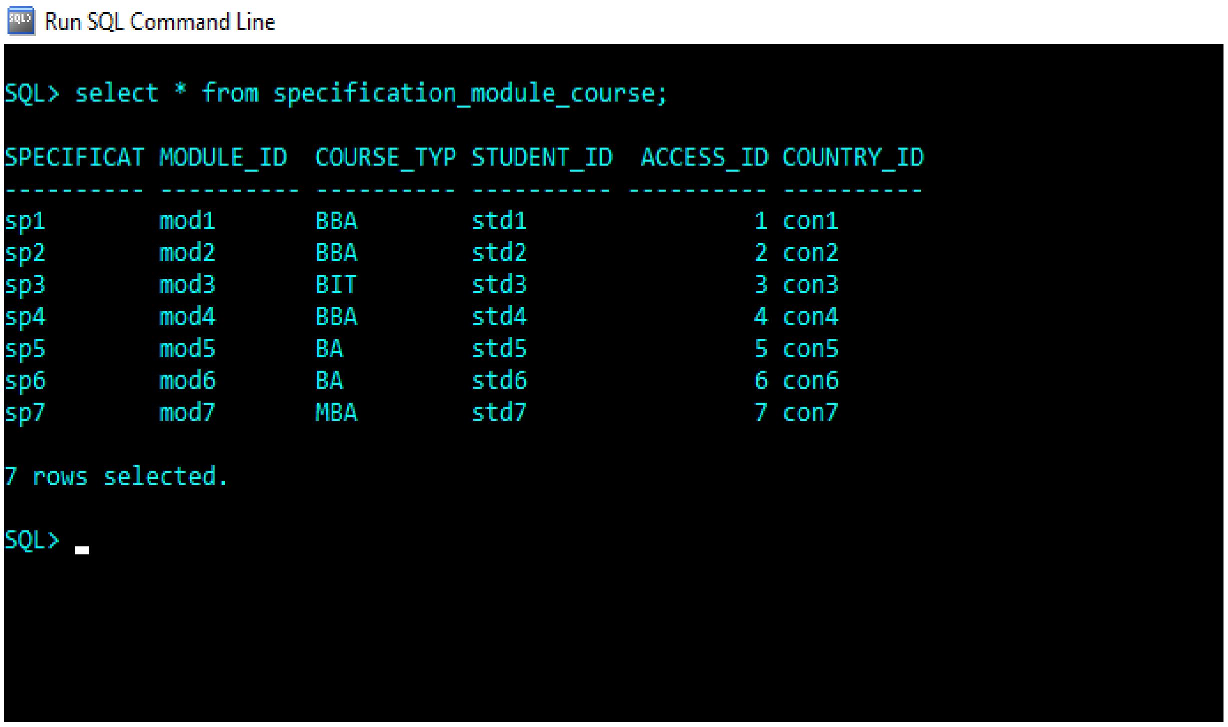
```
SQL> select * from course_type;

    SALARY COURSE_TYPE
---------- -------------------------
     60000 BBA
     60000 BBA
     60000 BBA
     70000 BIT
     80000 MBA
     20000 BA
     30000 BE

7 rows selected.

SQL>
```

Figure 20: table course_type



```
SQL> INSERT INTO specification_module_course(specification_id,module_id,course_type,student_id,access_id,country_id)
  2  VALUES ('sp1','mod1','BBA','std1',1,'con1');

1 row created.

SQL> INSERT INTO specification_module_course(specification_id,module_id,course_type,student_id,access_id,country_id)
  2  VALUES ('sp2','mod2','BBa','std2',2,'con2');
INSERT INTO specification_module_course(specification_id,module_id,course_type,student_id,access_id,country_id)
*
ERROR at line 1:
ORA-02291: integrity constraint (HR.SYS_C007291) violated - parent key not found


SQL> INSERT INTO specification_module_course(specification_id,module_id,course_type,student_id,access_id,country_id)
  2  VALUES ('sp2','mod2','BBA','std2',2,'con2');

1 row created.

SQL> INSERT INTO specification_module_course(specification_id,module_id,course_type,student_id,access_id,country_id)
  2  VALUES ('sp3','mod3','BIT','std3',3,'con3');

1 row created.

SQL> INSERT INTO specification_module_course(specification_id,module_id,course_type,student_id,access_id,country_id)
  2  VALUES ('sp4','mod4','BBA','std4',4,'con4');

1 row created.

SQL> INSERT INTO specification_module_course(specification_id,module_id,course_type,student_id,access_id,country_id)
  2  VALUES ('sp5','mod5','BA','std5',5,'con5');

1 row created.

SQL> INSERT INTO specification_module_course(specification_id,module_id,course_type,student_id,access_id,country_id)
  2  VALUES ('sp6','mod6','BA','std6',6,'con6');

1 row created.

SQL> INSERT INTO specification_module_course(specification_id,module_id,course_type,student_id,access_id,country_id)
  2  VALUES ('sp7','mod7','MBA','std7',7,'con7');

1 row created.

SQL> select * from specification_module_course;
```

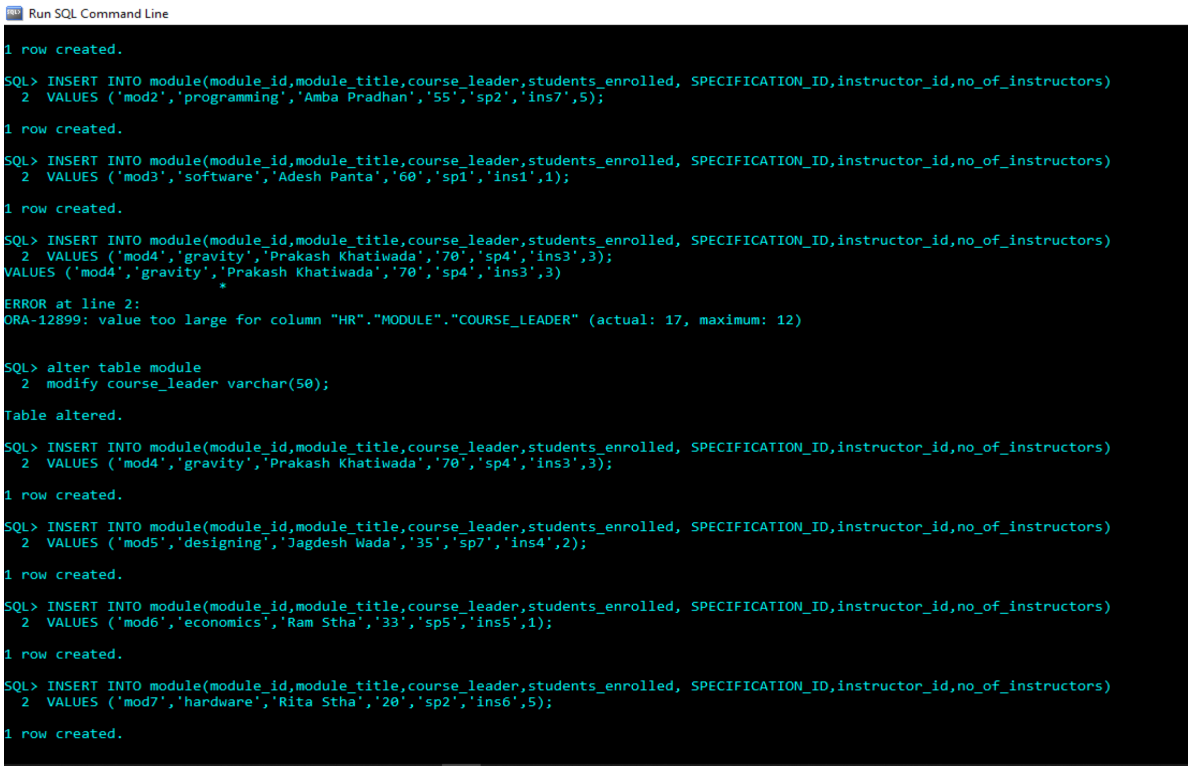
Figure 21: insert in specification_module_course



Run SQL Command Line

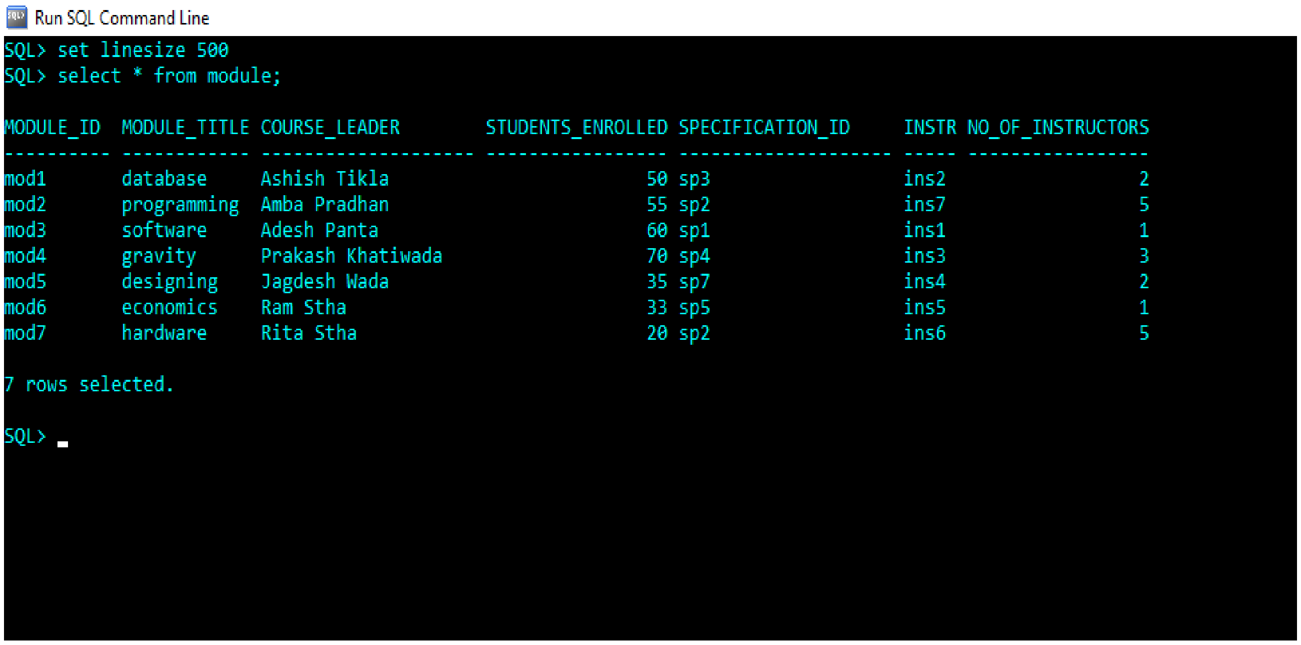
```
SQL> select * from specification_module_course;

SPECIFICAT MODULE_ID  COURSE_TYP STUDENT_ID  ACCESS_ID COUNTRY_ID
---------- ---------- ---------- ---------- ---------- ----------
sp1        mod1       BBA        std1                1 con1
sp2        mod2       BBA        std2                2 con2
sp3        mod3       BIT        std3                3 con3
sp4        mod4       BBA        std4                4 con4
sp5        mod5       BA         std5                5 con5
sp6        mod6       BA         std6                6 con6
sp7        mod7       MBA        std7                7 con7

7 rows selected.

SQL>
```

Figure 22: table specification_module_course



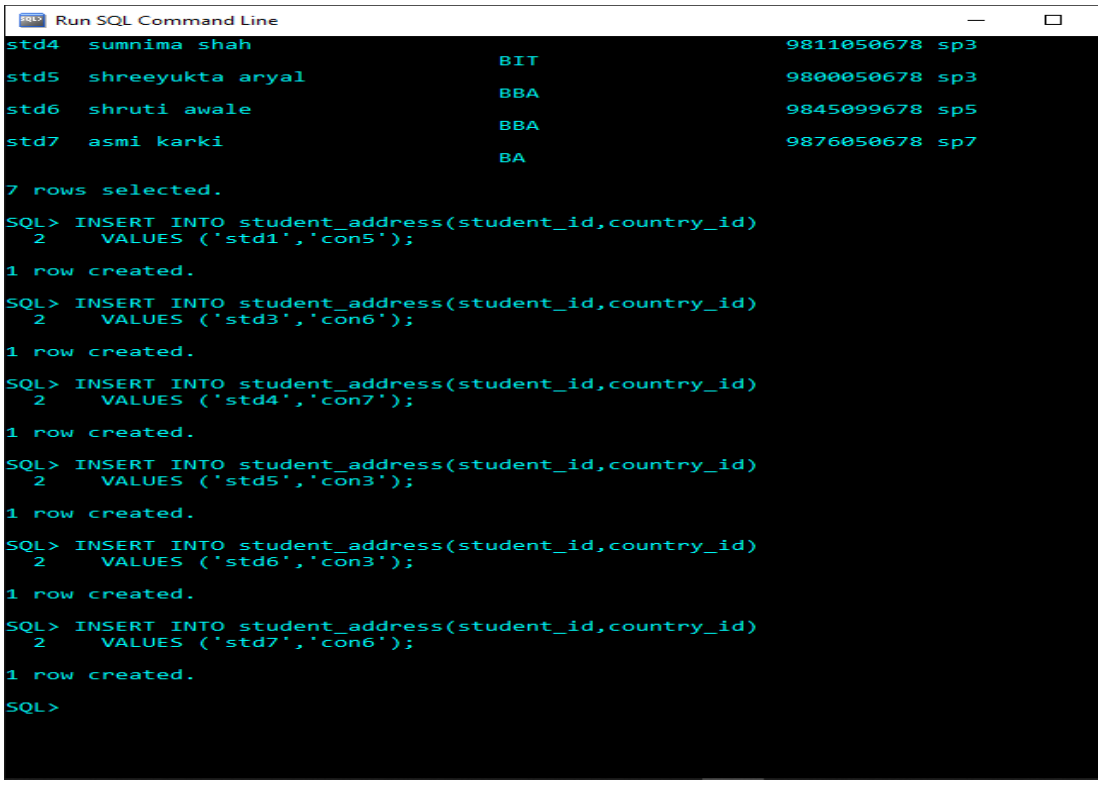
```
Run SQL Command Line
1 row created.

SQL> INSERT INTO module(module_id,module_title,course_leader,students_enrolled, SPECIFICATION_ID,instructor_id,no_of_instructors)
  2  VALUES ('mod2','programming','Amba Pradhan','55','sp2','ins7',5);

1 row created.

SQL> INSERT INTO module(module_id,module_title,course_leader,students_enrolled, SPECIFICATION_ID,instructor_id,no_of_instructors)
  2  VALUES ('mod3','software','Adesh Panta','60','sp1','ins1',1);

1 row created.

SQL> INSERT INTO module(module_id,module_title,course_leader,students_enrolled, SPECIFICATION_ID,instructor_id,no_of_instructors)
  2  VALUES ('mod4','gravity','Prakash Khatiwada','70','sp4','ins3',3);
VALUES ('mod4','gravity','Prakash Khatiwada','70','sp4','ins3',3)
                         *
ERROR at line 2:
ORA-12899: value too large for column "HR"."MODULE"."COURSE_LEADER" (actual: 17, maximum: 12)


SQL> alter table module
  2  modify course_leader varchar(50);

Table altered.

SQL> INSERT INTO module(module_id,module_title,course_leader,students_enrolled, SPECIFICATION_ID,instructor_id,no_of_instructors)
  2  VALUES ('mod4','gravity','Prakash Khatiwada','70','sp4','ins3',3);

1 row created.

SQL> INSERT INTO module(module_id,module_title,course_leader,students_enrolled, SPECIFICATION_ID,instructor_id,no_of_instructors)
  2  VALUES ('mod5','designing','Jagdesh Wada','35','sp7','ins4',2);

1 row created.

SQL> INSERT INTO module(module_id,module_title,course_leader,students_enrolled, SPECIFICATION_ID,instructor_id,no_of_instructors)
  2  VALUES ('mod6','economics','Ram Stha','33','sp5','ins5',1);

1 row created.

SQL> INSERT INTO module(module_id,module_title,course_leader,students_enrolled, SPECIFICATION_ID,instructor_id,no_of_instructors)
  2  VALUES ('mod7','hardware','Rita Stha','20','sp2','ins6',5);

1 row created.
```

Figure 23: insert in table module



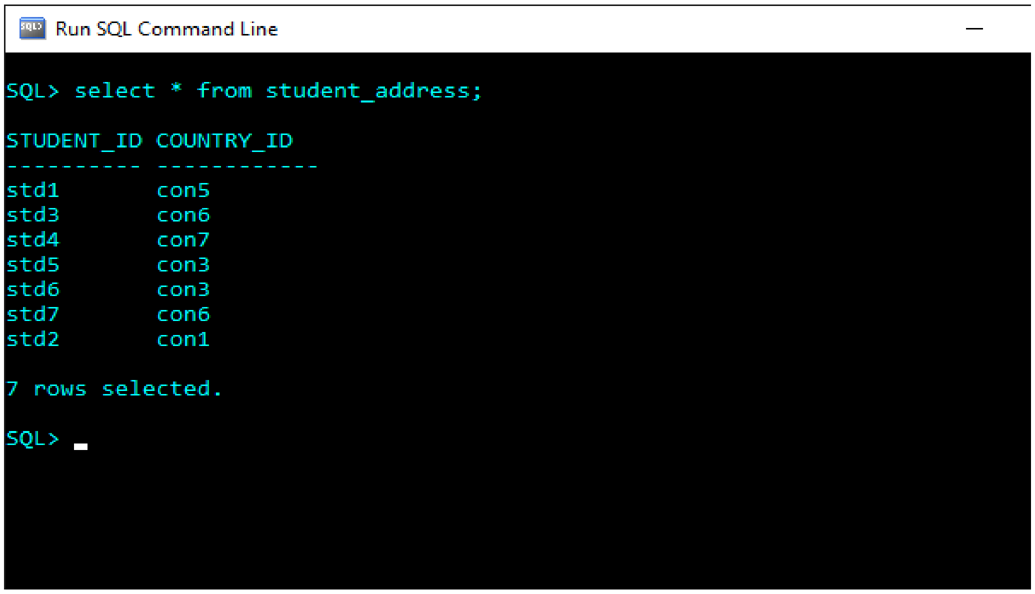
```
Run SQL Command Line
SQL> set linesize 500
SQL> select * from module;

MODULE_ID  MODULE_TITLE COURSE_LEADER        STUDENTS_ENROLLED SPECIFICATION_ID     INSTR NO_OF_INSTRUCTORS
---------- ------------ -------------------- ----------------- -------------------- ----- -----------------
mod1       database     Ashish Tikla                        50 sp3                  ins2                  2
mod2       programming  Amba Pradhan                        55 sp2                  ins7                  5
mod3       software     Adesh Panta                         60 sp1                  ins1                  1
mod4       gravity      Prakash Khatiwada                   70 sp4                  ins3                  3
mod5       designing    Jagdesh Wada                        35 sp7                  ins4                  2
mod6       economics    Ram Stha                            33 sp5                  ins5                  1
mod7       hardware     Rita Stha                           20 sp2                  ins6                  5

7 rows selected.

SQL> _
```

Figure 24: table module



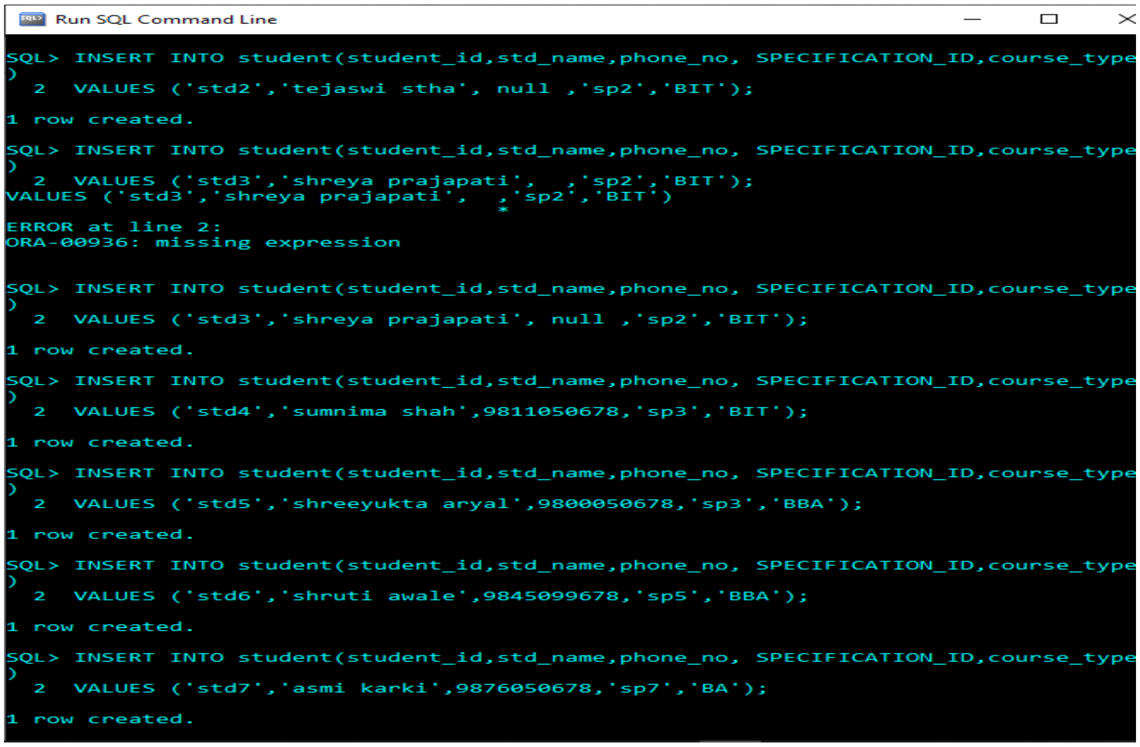
```
std4   sumnima shah                                                    9811050678 sp3
                                         BIT
std5   shreeyukta aryal                                                9800050678 sp3
                                         BBA
std6   shruti awale                                                    9845099678 sp5
                                         BBA
std7   asmi karki                                                      9876050678 sp7
                                         BA

7 rows selected.

SQL> INSERT INTO student_address(student_id,country_id)
  2    VALUES ('std1','con5');

1 row created.

SQL> INSERT INTO student_address(student_id,country_id)
  2    VALUES ('std3','con6');

1 row created.

SQL> INSERT INTO student_address(student_id,country_id)
  2    VALUES ('std4','con7');

1 row created.

SQL> INSERT INTO student_address(student_id,country_id)
  2    VALUES ('std5','con3');

1 row created.

SQL> INSERT INTO student_address(student_id,country_id)
  2    VALUES ('std6','con3');

1 row created.

SQL> INSERT INTO student_address(student_id,country_id)
  2    VALUES ('std7','con6');

1 row created.

SQL>
```

Figure 25: insert in table student_address



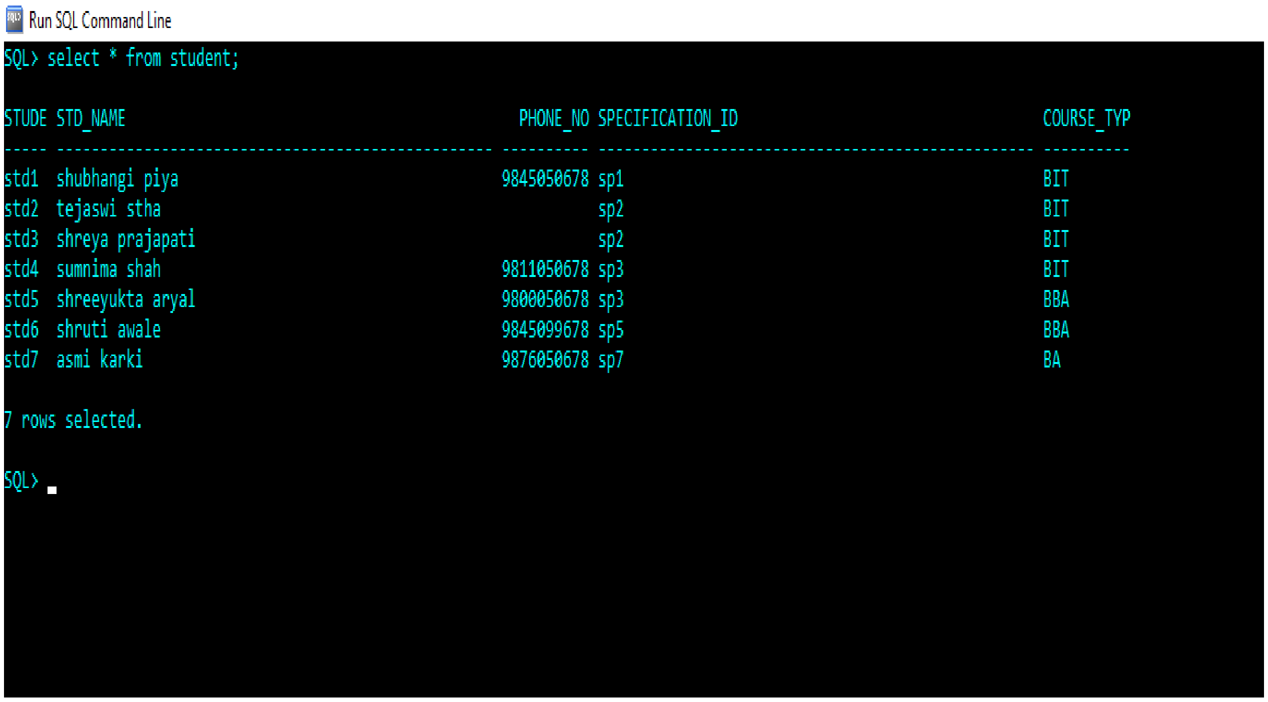
```
SQL> select * from student_address;

STUDENT_ID COUNTRY_ID
---------- ------------
std1       con5
std3       con6
std4       con7
std5       con3
std6       con3
std7       con6
std2       con1

7 rows selected.

SQL>
```

Figure 26: table student_address



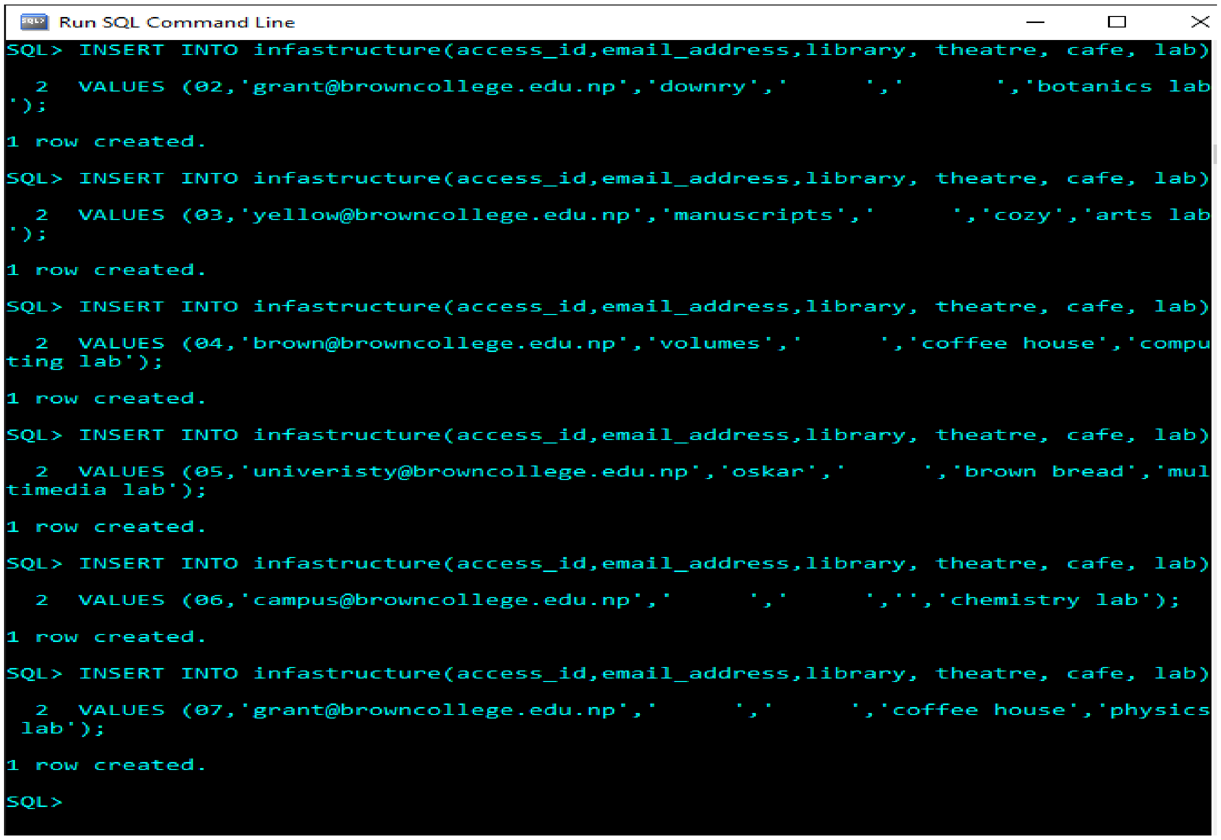
```
SQL> INSERT INTO student(student_id,std_name,phone_no, SPECIFICATION_ID,course_type
)
  2  VALUES ('std2','tejaswi stha', null ,'sp2','BIT');

1 row created.

SQL> INSERT INTO student(student_id,std_name,phone_no, SPECIFICATION_ID,course_type
)
  2  VALUES ('std3','shreya prajapati',   ,'sp2','BIT');
VALUES ('std3','shreya prajapati',   ,'sp2','BIT')
                                     *
ERROR at line 2:
ORA-00936: missing expression


SQL> INSERT INTO student(student_id,std_name,phone_no, SPECIFICATION_ID,course_type
)
  2  VALUES ('std3','shreya prajapati', null ,'sp2','BIT');

1 row created.

SQL> INSERT INTO student(student_id,std_name,phone_no, SPECIFICATION_ID,course_type
)
  2  VALUES ('std4','sumnima shah',9811050678,'sp3','BIT');

1 row created.

SQL> INSERT INTO student(student_id,std_name,phone_no, SPECIFICATION_ID,course_type
)
  2  VALUES ('std5','shreeyukta aryal',9800050678,'sp3','BBA');

1 row created.

SQL> INSERT INTO student(student_id,std_name,phone_no, SPECIFICATION_ID,course_type
)
  2  VALUES ('std6','shruti awale',9845099678,'sp5','BBA');

1 row created.

SQL> INSERT INTO student(student_id,std_name,phone_no, SPECIFICATION_ID,course_type
)
  2  VALUES ('std7','asmi karki',9876050678,'sp7','BA');

1 row created.
```

Figure 27: insert in table student



```
SQL> select * from student;

STUDE STD_NAME                                            PHONE_NO SPECIFICATION_ID                                   COURSE_TYP
----- -------------------------------------------------- ---------- -------------------------------------------------- ----------
std1  shubhangi piya                                     9845050678 sp1                                                BIT
std2  tejaswi stha                                                  sp2                                                BIT
std3  shreya prajapati                                              sp2                                                BIT
std4  sumnima shah                                       9811050678 sp3                                                BIT
std5  shreeyukta aryal                                   9800050678 sp3                                                BBA
std6  shruti awale                                       9845099678 sp5                                                BBA
std7  asmi karki                                         9876050678 sp7                                                BA

7 rows selected.

SQL>
```

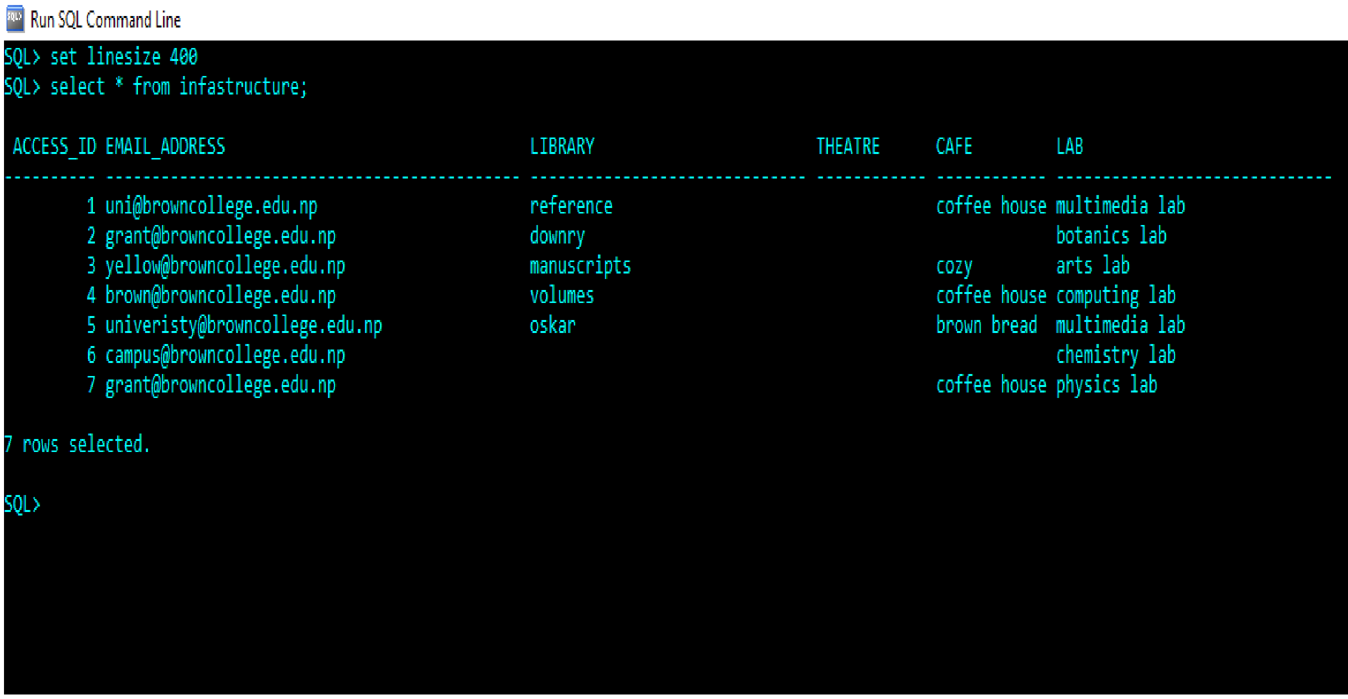
Figure 28: table student



Run SQL Command Line

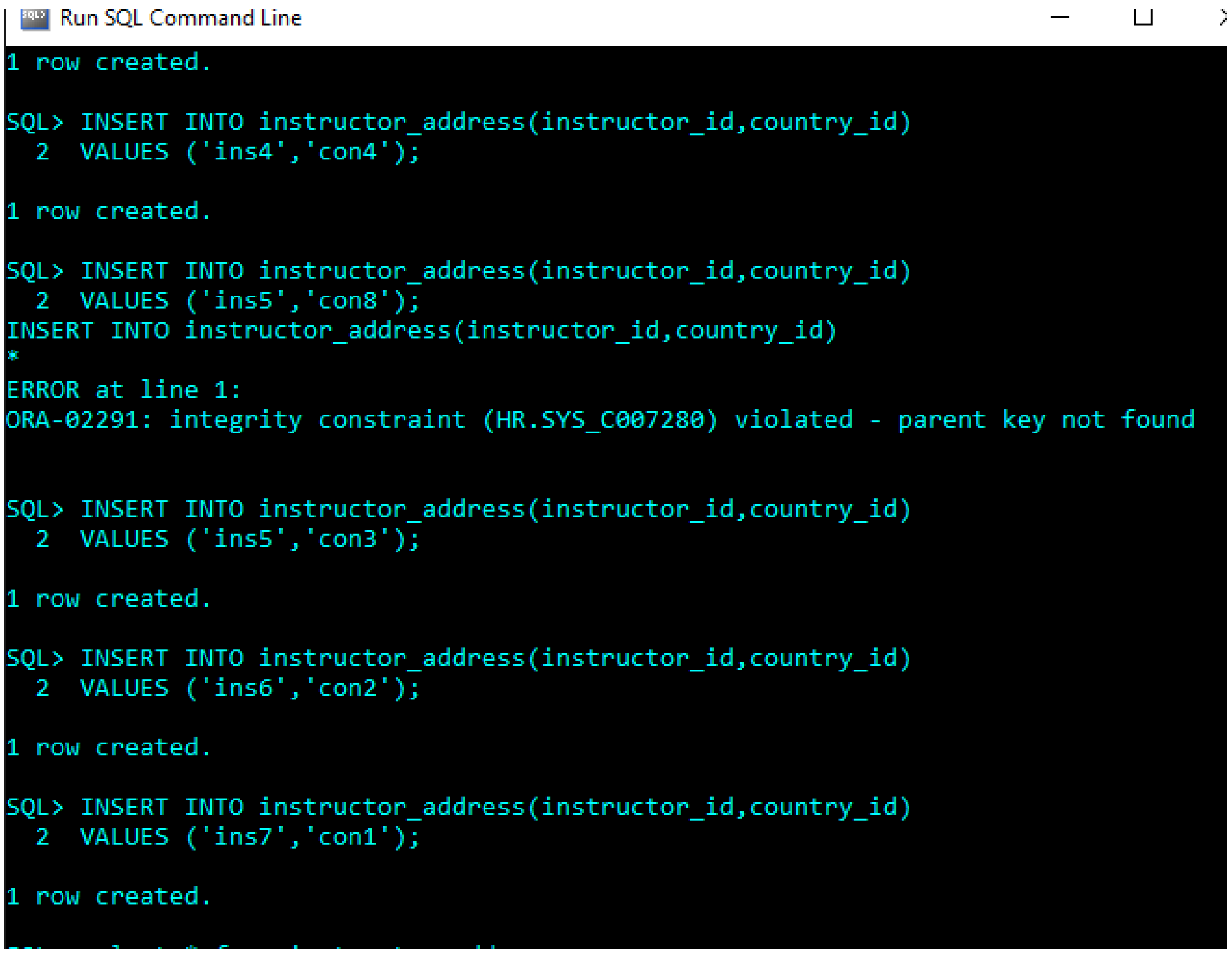
```
SQL> INSERT INTO infastructure(access_id,email_address,library, theatre, cafe, lab)
  2  VALUES (02,'grant@browncollege.edu.np','downry','      ','       ','botanics lab
');

1 row created.

SQL> INSERT INTO infastructure(access_id,email_address,library, theatre, cafe, lab)
  2  VALUES (03,'yellow@browncollege.edu.np','manuscripts','      ','cozy','arts lab
');

1 row created.

SQL> INSERT INTO infastructure(access_id,email_address,library, theatre, cafe, lab)
  2  VALUES (04,'brown@browncollege.edu.np','volumes','      ','coffee house','compu
ting lab');

1 row created.

SQL> INSERT INTO infastructure(access_id,email_address,library, theatre, cafe, lab)
  2  VALUES (05,'univeristy@browncollege.edu.np','oskar','      ','brown bread','mul
timedia lab');

1 row created.

SQL> INSERT INTO infastructure(access_id,email_address,library, theatre, cafe, lab)
  2  VALUES (06,'campus@browncollege.edu.np','      ','      ','','chemistry lab');

1 row created.

SQL> INSERT INTO infastructure(access_id,email_address,library, theatre, cafe, lab)
  2  VALUES (07,'grant@browncollege.edu.np','      ','      ','coffee house','physics
 lab');

1 row created.

SQL>
```

Figure 29: insert in table infastructure


Run SQL Command Line

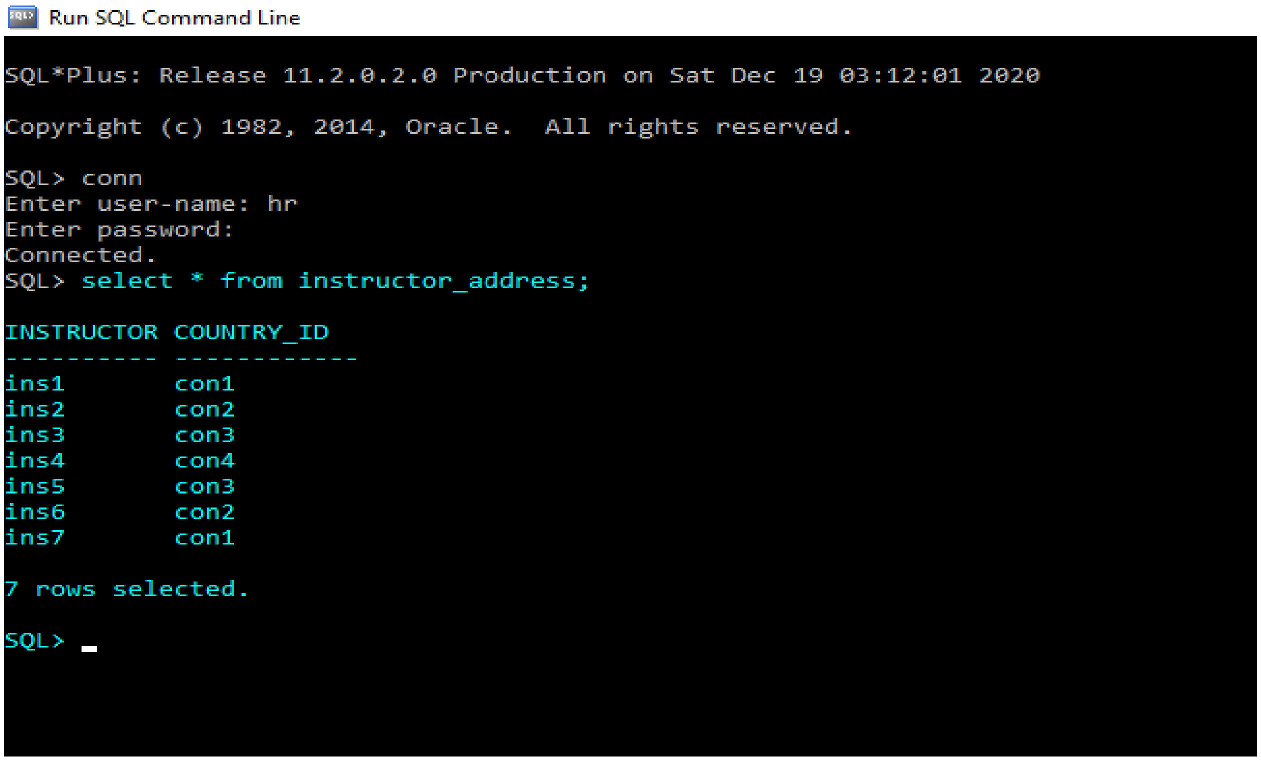
```
SQL> set linesize 400
SQL> select * from infastructure;

 ACCESS_ID EMAIL_ADDRESS                                   LIBRARY                        THEATRE     CAFE        LAB
---------- ----------------------------------------------- ------------------------------ ----------- ----------- ------------------------------
         1 uni@browncollege.edu.np                         reference                                  coffee house multimedia lab
         2 grant@browncollege.edu.np                       downry                                                  botanics lab
         3 yellow@browncollege.edu.np                      manuscripts                                cozy         arts lab
         4 brown@browncollege.edu.np                       volumes                                    coffee house computing lab
         5 univeristy@browncollege.edu.np                  oskar                                      brown bread  multimedia lab
         6 campus@browncollege.edu.np                                                                              chemistry lab
         7 grant@browncollege.edu.np                                                                  coffee house physics lab

7 rows selected.

SQL>
```

Figure 30: table infastructure



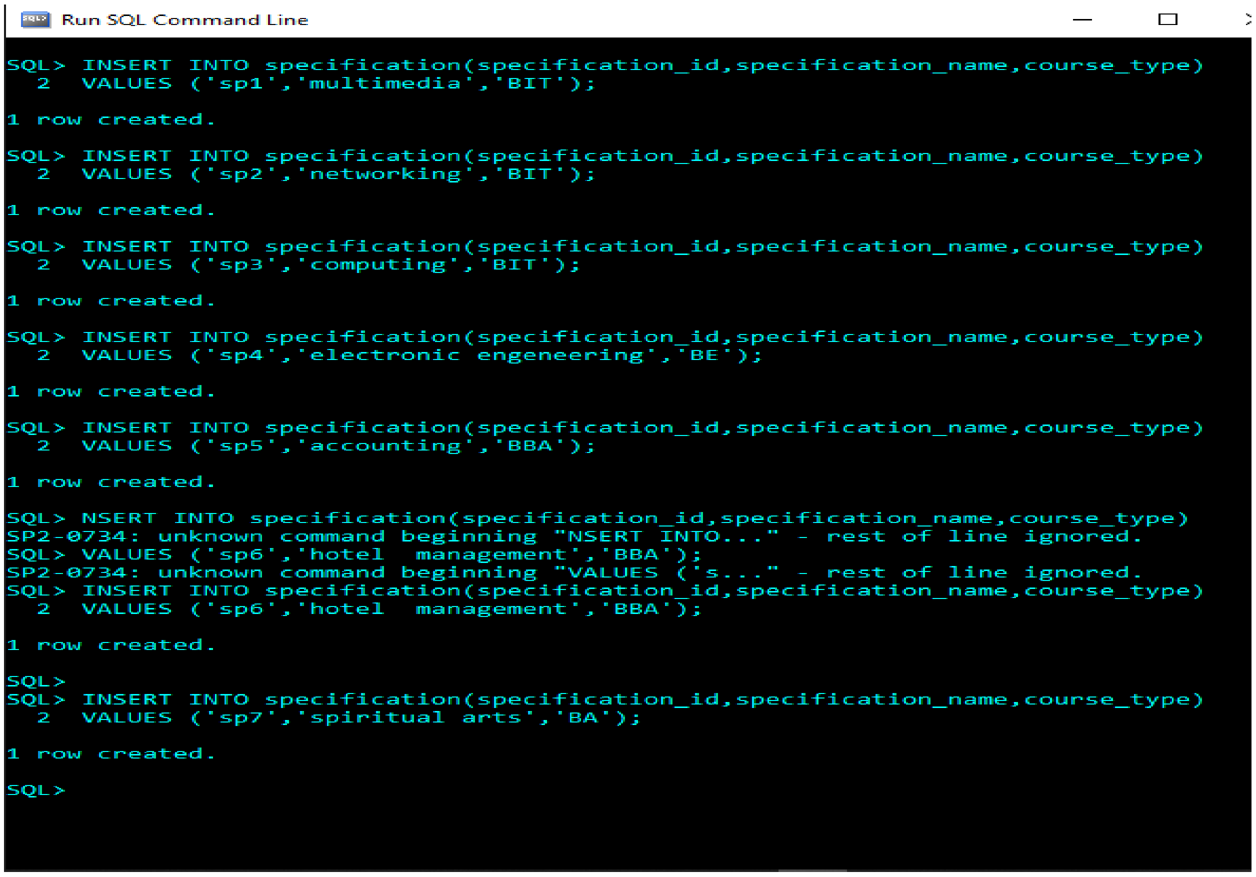
```
1 row created.

SQL> INSERT INTO instructor_address(instructor_id,country_id)
  2  VALUES ('ins4','con4');

1 row created.

SQL> INSERT INTO instructor_address(instructor_id,country_id)
  2  VALUES ('ins5','con8');
INSERT INTO instructor_address(instructor_id,country_id)
*
ERROR at line 1:
ORA-02291: integrity constraint (HR.SYS_C007280) violated - parent key not found


SQL> INSERT INTO instructor_address(instructor_id,country_id)
  2  VALUES ('ins5','con3');

1 row created.

SQL> INSERT INTO instructor_address(instructor_id,country_id)
  2  VALUES ('ins6','con2');

1 row created.

SQL> INSERT INTO instructor_address(instructor_id,country_id)
  2  VALUES ('ins7','con1');

1 row created.
```

Figure 31: insert in table instructor_address


Run SQL Command Line

```
SQL*Plus: Release 11.2.0.2.0 Production on Sat Dec 19 03:12:01 2020

Copyright (c) 1982, 2014, Oracle.  All rights reserved.

SQL> conn
Enter user-name: hr
Enter password:
Connected.
SQL> select * from instructor_address;

INSTRUCTOR COUNTRY_ID
---------- ------------
ins1       con1
ins2       con2
ins3       con3
ins4       con4
ins5       con3
ins6       con2
ins7       con1

7 rows selected.

SQL> _
```

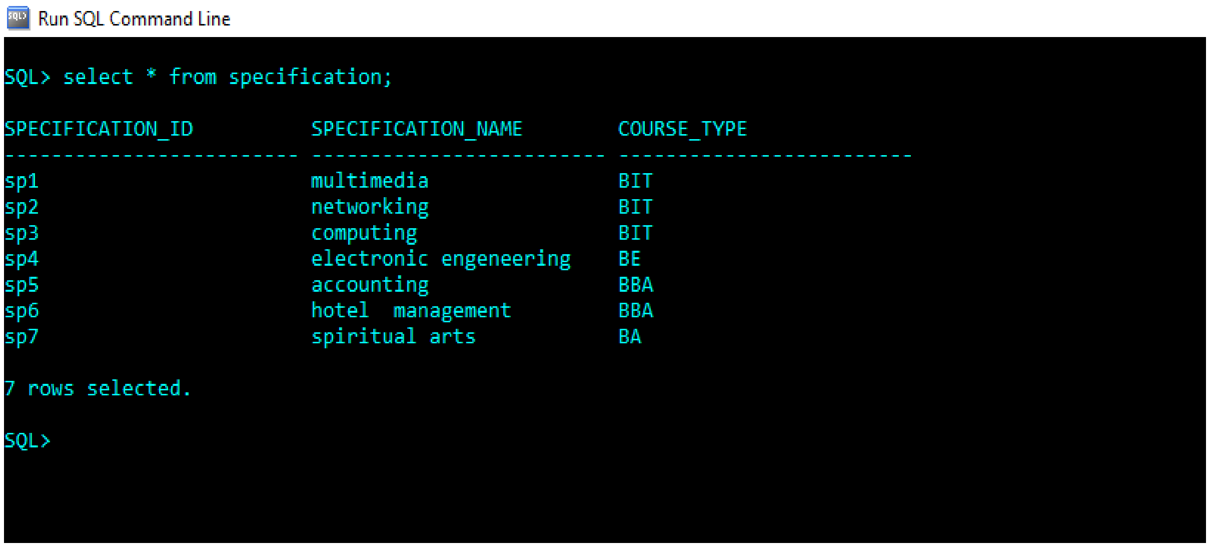
Figure 32: table instructor_address



Run SQL Command Line

```
SQL> INSERT INTO specification(specification_id,specification_name,course_type)
  2  VALUES ('sp1','multimedia','BIT');

1 row created.

SQL> INSERT INTO specification(specification_id,specification_name,course_type)
  2  VALUES ('sp2','networking','BIT');

1 row created.

SQL> INSERT INTO specification(specification_id,specification_name,course_type)
  2  VALUES ('sp3','computing','BIT');

1 row created.

SQL> INSERT INTO specification(specification_id,specification_name,course_type)
  2  VALUES ('sp4','electronic engeneering','BE');

1 row created.

SQL> INSERT INTO specification(specification_id,specification_name,course_type)
  2  VALUES ('sp5','accounting','BBA');

1 row created.

SQL> NSERT INTO specification(specification_id,specification_name,course_type)
SP2-0734: unknown command beginning "NSERT INTO..." - rest of line ignored.
SQL> VALUES ('sp6','hotel  management','BBA');
SP2-0734: unknown command beginning "VALUES ('s..." - rest of line ignored.
SQL> INSERT INTO specification(specification_id,specification_name,course_type)
  2  VALUES ('sp6','hotel  management','BBA');

1 row created.

SQL>
SQL> INSERT INTO specification(specification_id,specification_name,course_type)
  2  VALUES ('sp7','spiritual arts','BA');

1 row created.

SQL>
```

Figure 33: insert in table specification



Run SQL Command Line

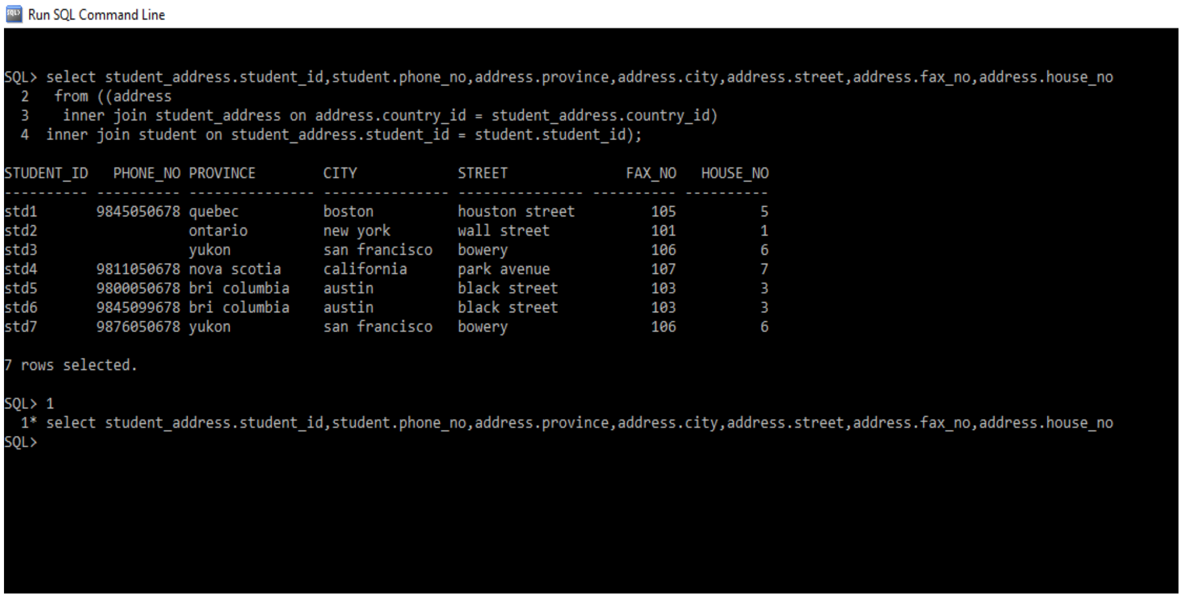
```
SQL> select * from specification;

SPECIFICATION_ID          SPECIFICATION_NAME        COURSE_TYPE
------------------------- ------------------------- -------------------------
sp1                       multimedia                BIT
sp2                       networking                BIT
sp3                       computing                 BIT
sp4                       electronic engeneering    BE
sp5                       accounting                BBA
sp6                       hotel  management         BBA
sp7                       spiritual arts            BA

7 rows selected.

SQL>
```

Figure 34: table specification

5. Database Querying

5.1 Information Queries with screenshots

5.1.1 List all the students with all their addresses with their phone numbers.

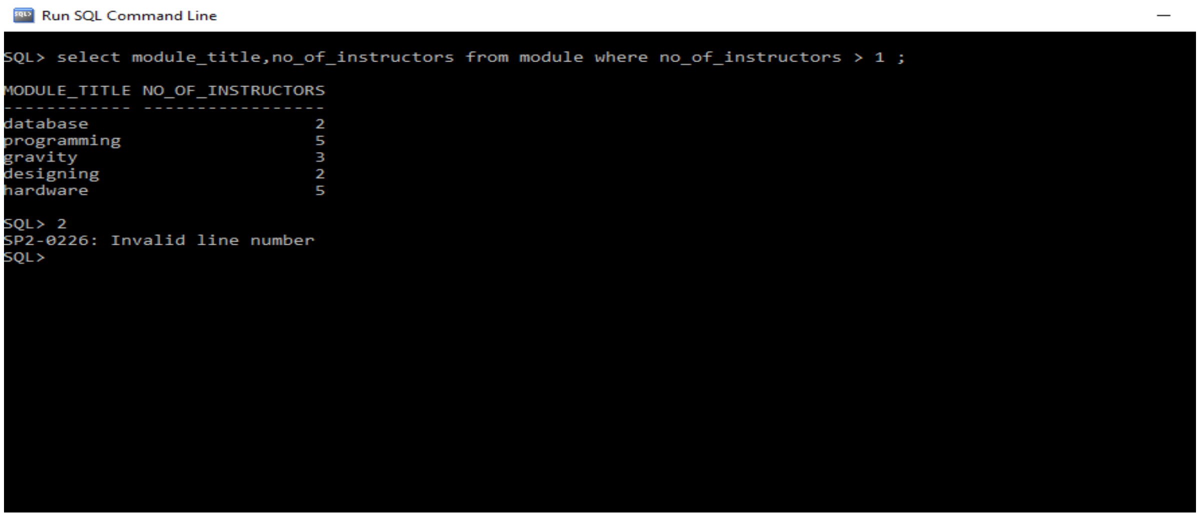


Run SQL Command Line

```
SQL> select student_address.student_id,student.phone_no,address.province,address.city,address.street,address.fax_no,address.house_no
  2   from ((address
  3    inner join student_address on address.country_id = student_address.country_id)
  4  inner join student on student_address.student_id = student.student_id);

STUDENT_ID   PHONE_NO PROVINCE        CITY           STREET             FAX_NO   HOUSE_NO
--------- ---------- -------------- -------------- -------------- ---------- ----------
std1      9845050678 quebec         boston         houston street        105          5
std2                 ontario        new york       wall street           101          1
std3                 yukon          san francisco  bowery                106          6
std4      9811050678 nova scotia    california     park avenue           107          7
std5      9800050678 bri columbia   austin         black street          103          3
std6      9845099678 bri columbia   austin         black street          103          3
std7      9876050678 yukon          san francisco  bowery                106          6

7 rows selected.

SQL> 1
  1* select student_address.student_id,student.phone_no,address.province,address.city,address.street,address.fax_no,address.house_no
SQL>
```

Figure 35: information query number 1

5.1.2 List all the modules which are taught by more than one instructor.

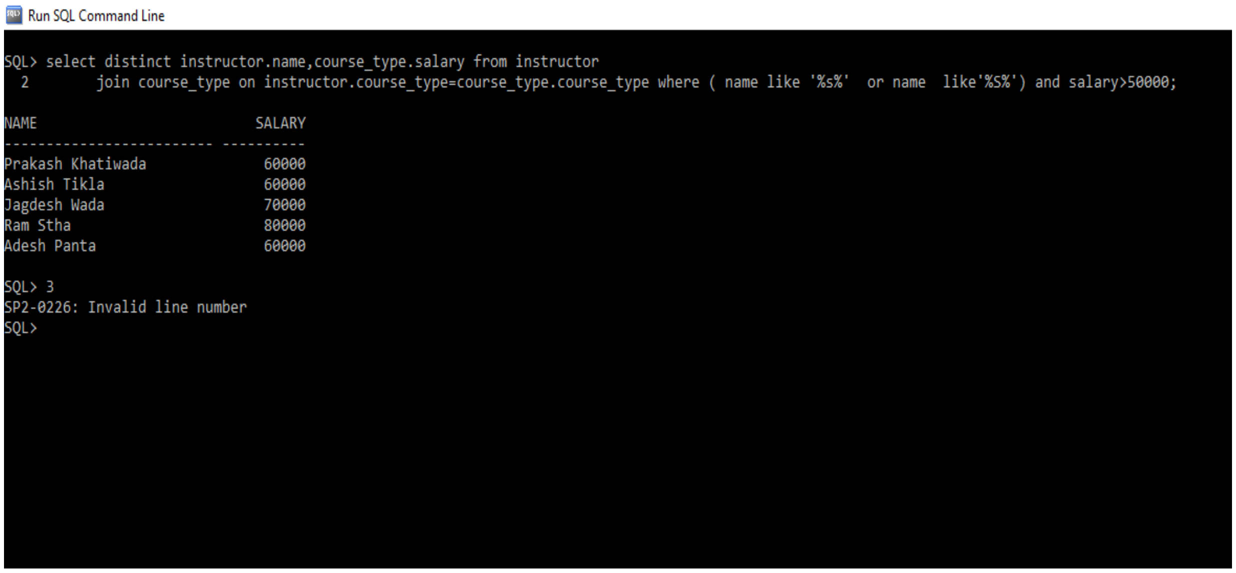


Run SQL Command Line

```
SQL> select module_title,no_of_instructors from module where no_of_instructors > 1 ;

MODULE_TITLE NO_OF_INSTRUCTORS
------------ -----------------
database                     2
programming                  5
gravity                      3
designing                    2
hardware                     5

SQL> 2
SP2-0226: Invalid line number
SQL>
```

Figure 36: information query number 2

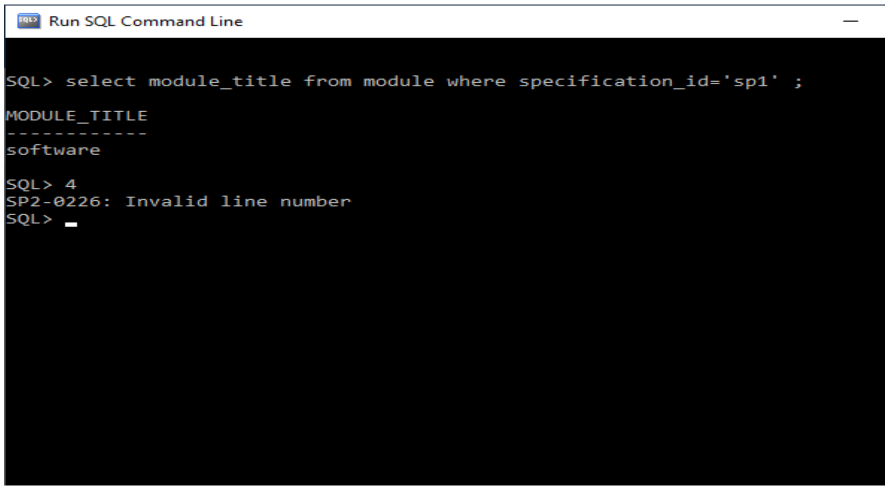
5.1.3 List the name of all the instructors whose name contains ‘s’ and salary is above 50,000.



```
SQL> select distinct instructor.name,course_type.salary from instructor
  2       join course_type on instructor.course_type=course_type.course_type where ( name like '%s%'  or name  like'%S%') and salary>50000;

NAME                         SALARY
------------------------ ----------
Prakash Khatiwada             60000
Ashish Tikla                  60000
Jagdesh Wada                  70000
Ram Stha                      80000
Adesh Panta                   60000

SQL> 3
SP2-0226: Invalid line number
SQL>
```

Figure 37: information query number 3

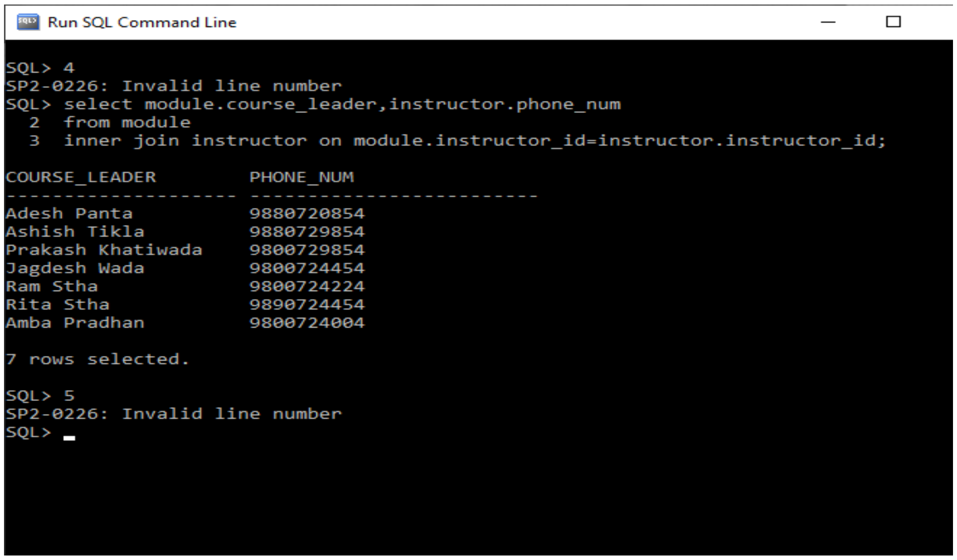
5.1.4 List the modules comes under the ‘Multimedia’ specification.



```
SQL> select module_title from module where specification_id='sp1' ;

MODULE_TITLE
------------
software

SQL> 4
SP2-0226: Invalid line number
SQL>
```

Figure 38: information query number 4

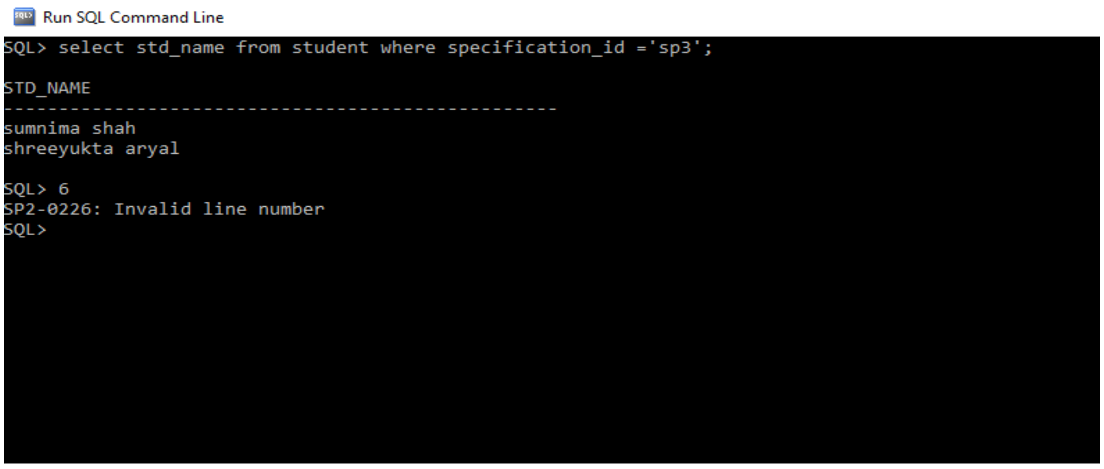
5.1.5 List the name of the head of modules with the list of his phone number.


Run SQL Command Line

```
SQL> 4
SP2-0226: Invalid line number
SQL> select module.course_leader,instructor.phone_num
  2  from module
  3  inner join instructor on module.instructor_id=instructor.instructor_id;

COURSE_LEADER        PHONE_NUM
-------------------- -------------------------
Adesh Panta          9880720854
Ashish Tikla         9880729854
Prakash Khatiwada    9800729854
Jagdesh Wada         9800724454
Ram Stha             9800724224
Rita Stha            9890724454
Amba Pradhan         9800724004

7 rows selected.

SQL> 5
SP2-0226: Invalid line number
SQL>
```

Figure 39: information query number 5

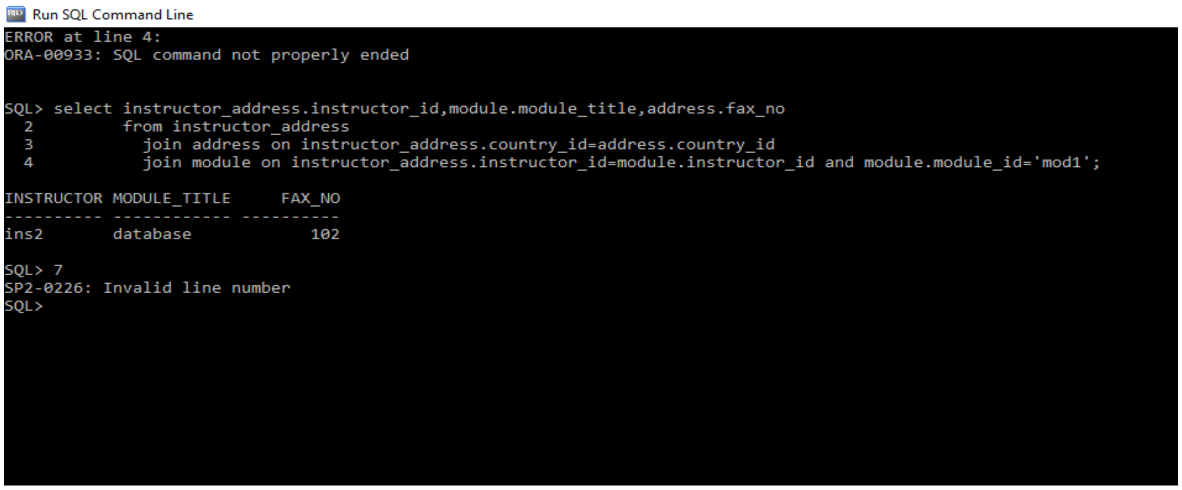
5.1.6 List all Students who have enrolled in ‘networking’ specifications.


Run SQL Command Line

```
SQL> select std_name from student where specification_id ='sp3';

STD_NAME
--------------------------------------------------
sumnima shah
shreeyukta aryal

SQL> 6
SP2-0226: Invalid line number
SQL>
```

Figure 40: information query number 6

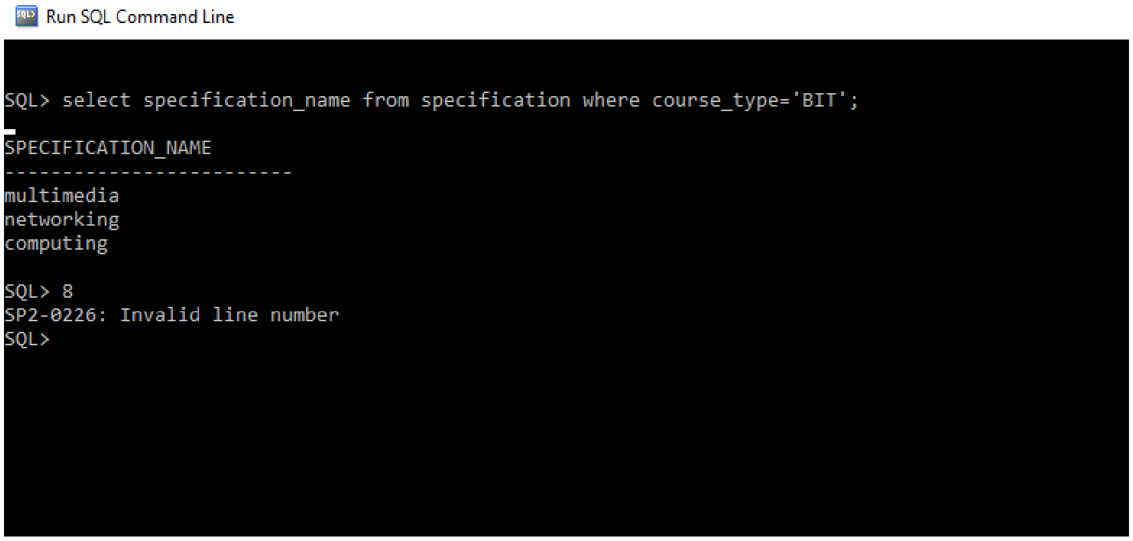
5.1.7 List the fax number of the instructor who teaches the 'database' module.


Run SQL Command Line

```
ERROR at line 4:
ORA-00933: SQL command not properly ended


SQL> select instructor_address.instructor_id,module.module_title,address.fax_no
  2         from instructor_address
  3           join address on instructor_address.country_id=address.country_id
  4           join module on instructor_address.instructor_id=module.instructor_id and module.module_id='mod1';

INSTRUCTOR MODULE_TITLE     FAX_NO
---------- ------------ ----------
ins2       database            102

SQL> 7
SP2-0226: Invalid line number
SQL>
```

Figure 41: information query number 7

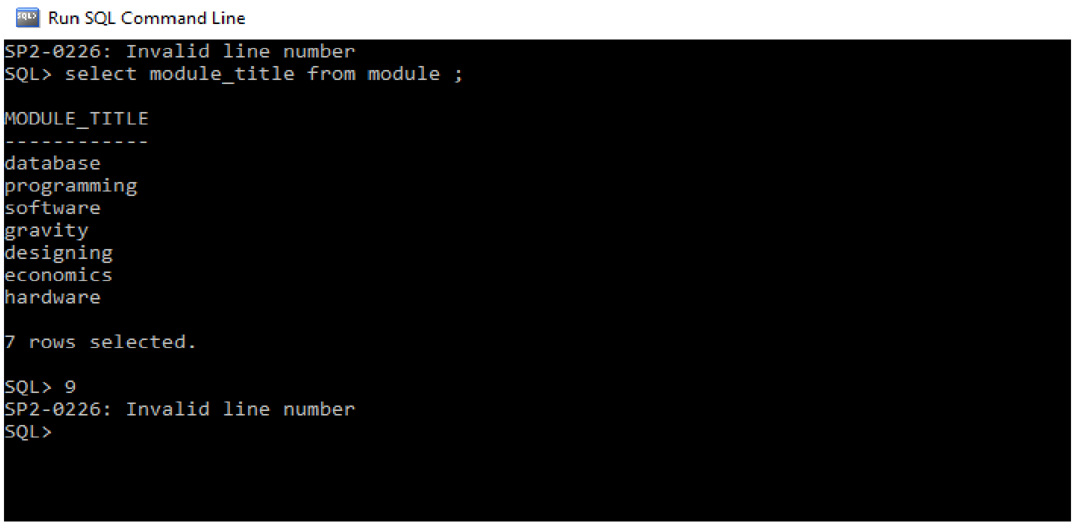
5.1.8 List the specification falls under the BIT course.


Run SQL Command Line

```
SQL> select specification_name from specification where course_type='BIT';

SPECIFICATION_NAME
-------------------------
multimedia
networking
computing

SQL> 8
SP2-0226: Invalid line number
SQL>
```

Figure 42: information query number 8

5.1.9 List all the modules taught in any one particular class.

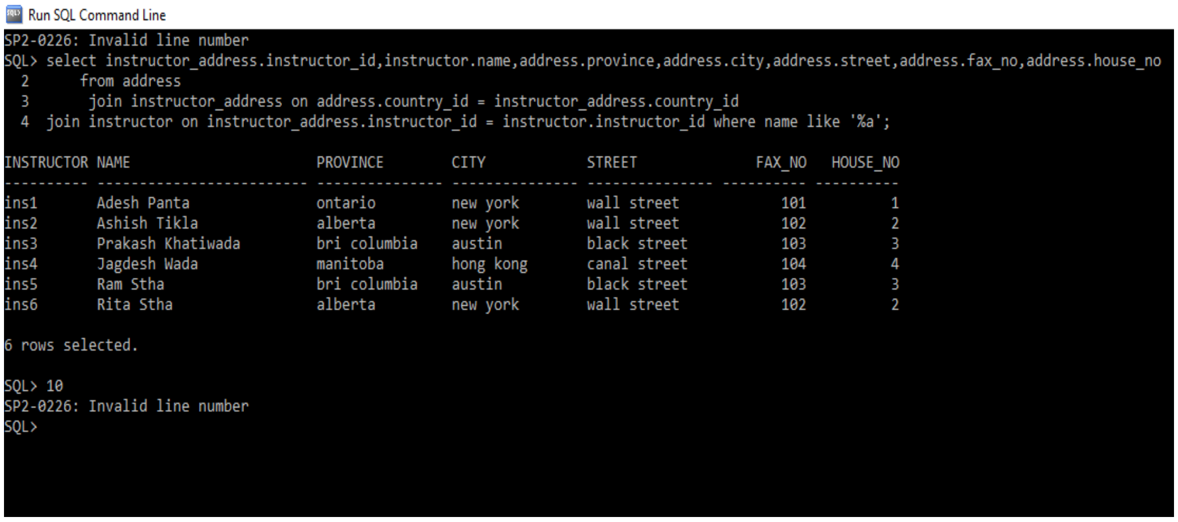


Run SQL Command Line

```
SP2-0226: Invalid line number
SQL> select module_title from module ;

MODULE_TITLE
------------
database
programming
software
gravity
designing
economics
hardware

7 rows selected.

SQL> 9
SP2-0226: Invalid line number
SQL>
```

Figure 43: information query number 9

5.1.10 List all the teachers with all their addresses who have ‘a’ at the end of their first names.



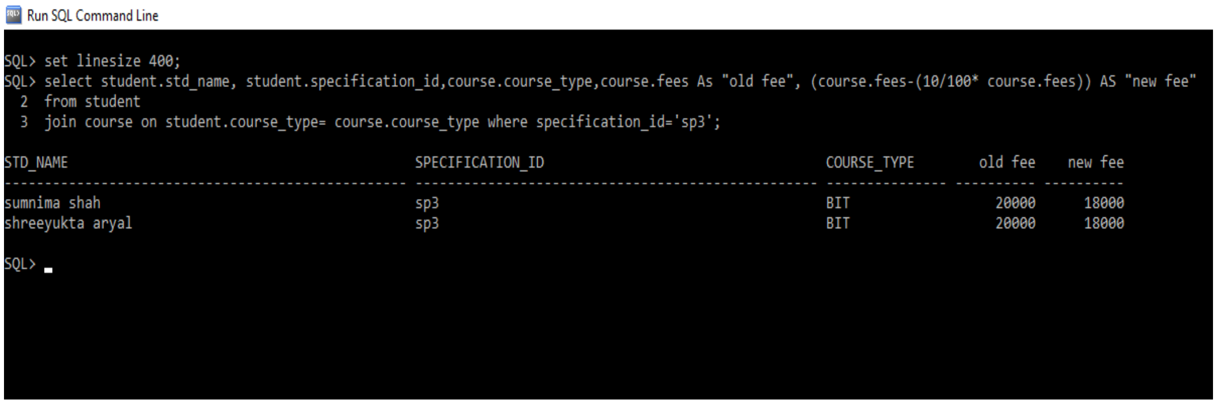
Run SQL Command Line

```
SP2-0226: Invalid line number
SQL> select instructor_address.instructor_id,instructor.name,address.province,address.city,address.street,address.fax_no,address.house_no
  2      from address
  3       join instructor_address on address.country_id = instructor_address.country_id
  4  join instructor on instructor_address.instructor_id = instructor.instructor_id where name like '%a';

INSTRUCTOR NAME                     PROVINCE       CITY           STREET              FAX_NO   HOUSE_NO
---------- ------------------------ -------------- -------------- -------------- ---------- ----------
ins1       Adesh Panta              ontario        new york       wall street           101          1
ins2       Ashish Tikla             alberta        new york       wall street           102          2
ins3       Prakash Khatiwada        bri columbia   austin         black street          103          3
ins4       Jagdesh Wada             manitoba       hong kong      canal street          104          4
ins5       Ram Stha                 bri columbia   austin         black street          103          3
ins6       Rita Stha                alberta        new york       wall street           102          2

6 rows selected.

SQL> 10
SP2-0226: Invalid line number
SQL>
```

Figure 44: information query number 10

5.2 Transaction Queries with screenshots

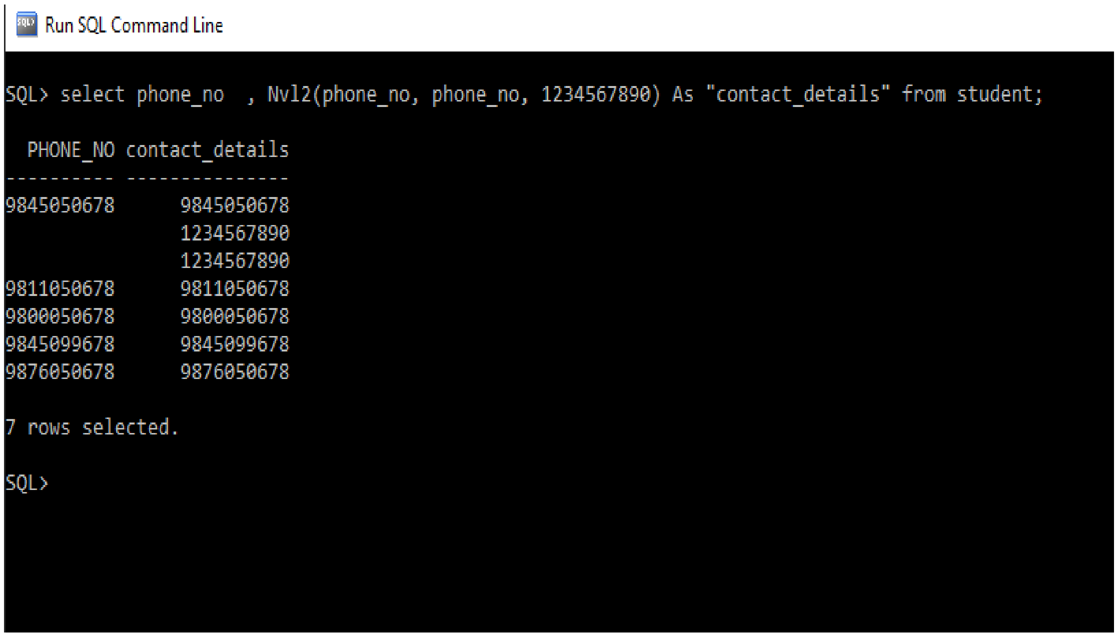
5.2.1 Show the students, course they enroll in and their fees. Reduce 10% of the fees if they are enrolled in a computing course.



```
SQL> set linesize 400;
SQL> select student.std_name, student.specification_id,course.course_type,course.fees As "old fee", (course.fees-(10/100* course.fees)) AS "new fee"
  2  from student
  3  join course on student.course_type= course.course_type where specification_id='sp3';

STD_NAME                                          SPECIFICATION_ID                                  COURSE_TYPE       old fee    new fee
------------------------------------------------- ------------------------------------------------- -------------- ---------- ----------
sumnima shah                                      sp3                                               BIT                 20000      18000
shreeyukta aryal                                  sp3                                               BIT                 20000      18000

SQL>
```

Figure 45: transaction query no 1

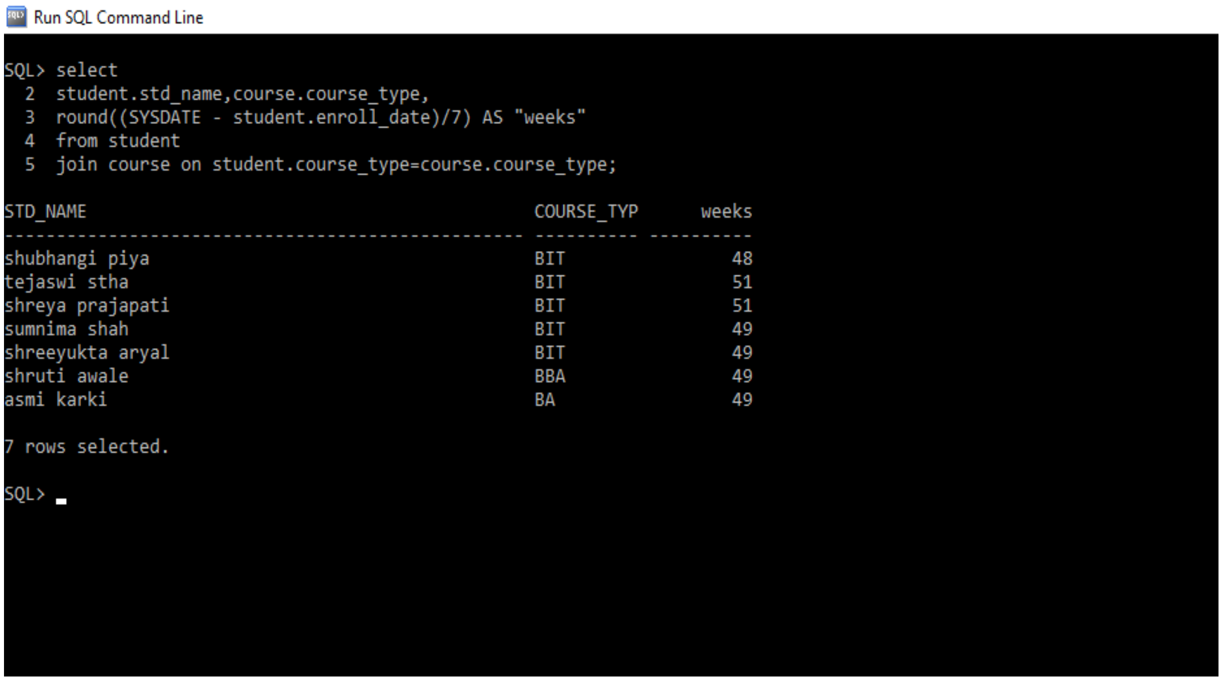
5.2.2 Place the default Number 1234567890 if the list of phone numbers to the location of the address is empty and give the column name as 'Contact details.



```
SQL> select phone_no  , Nvl2(phone_no, phone_no, 1234567890) As "contact_details" from student;

  PHONE_NO contact_details
---------- ---------------
9845050678      9845050678
                1234567890
                1234567890
9811050678      9811050678
9800050678      9800050678
9845099678      9845099678
9876050678      9876050678

7 rows selected.

SQL>
```

Figure 46transaction query no 2

5.2.3 Show the name of all the students with the number of weeks since they have enrolled in the course.



Run SQL Command Line

```
SQL> select
  student.std_name,course.course_type,
  round((SYSDATE - student.enroll_date)/7) AS "weeks"
  from student
  join course on student.course_type=course.course_type;

STD_NAME                                          COURSE_TYP      weeks
-------------------------------------------------- ---------- ----------
shubhangi piya                                    BIT                48
tejaswi stha                                      BIT                51
shreya prajapati                                  BIT                51
sumnima shah                                      BIT                49
shreeyukta aryal                                  BIT                49
shruti awale                                      BBA                49
asmi karki                                        BA                 49

7 rows selected.

SQL>
```

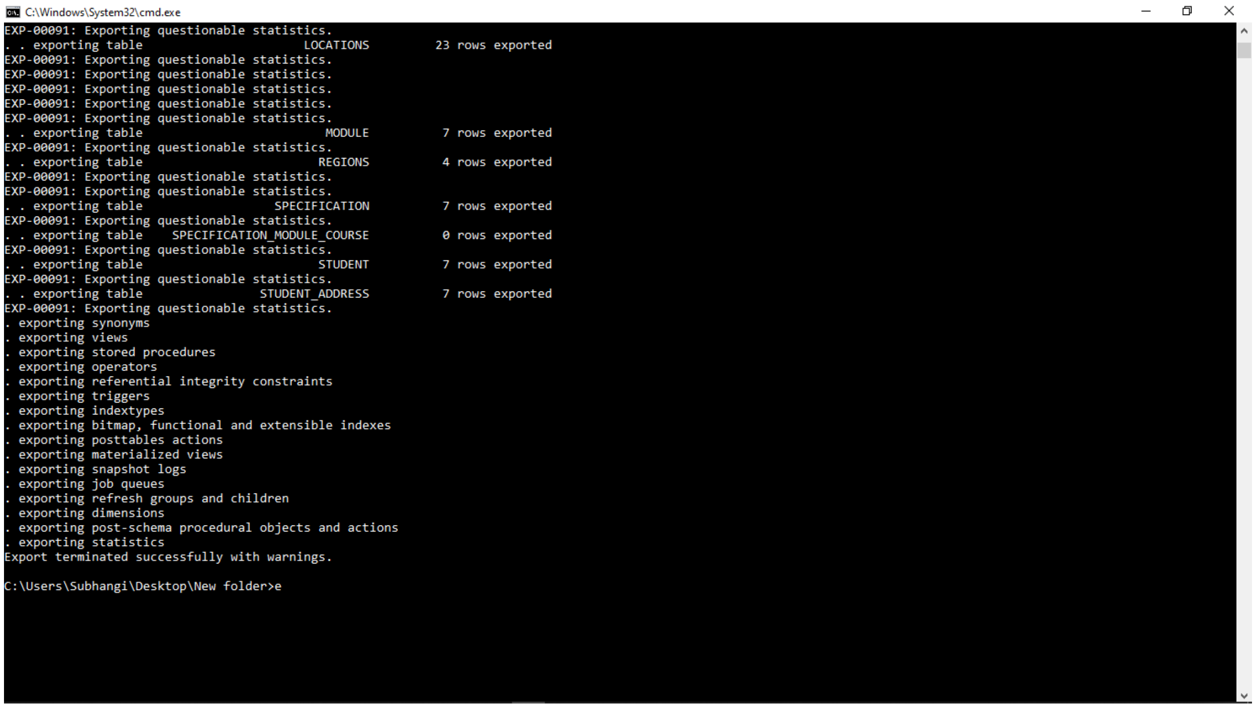
Figure 47transaction query no 3

5.2.4 Show the name of the instructors who got equal salary and work in the same specification.

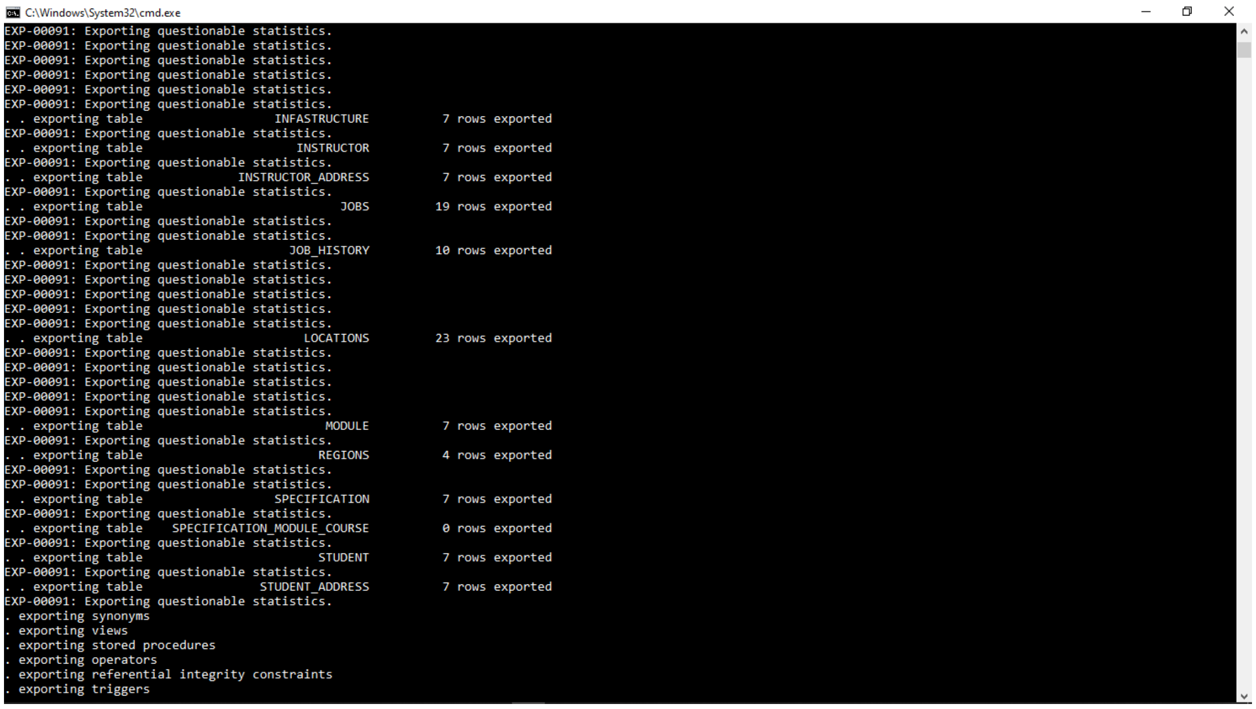
5.2.5 List all the courses with the total number of students enrolled course name and the highest marks obtained.

5.2.6 List all the instructors who are also a course leader.

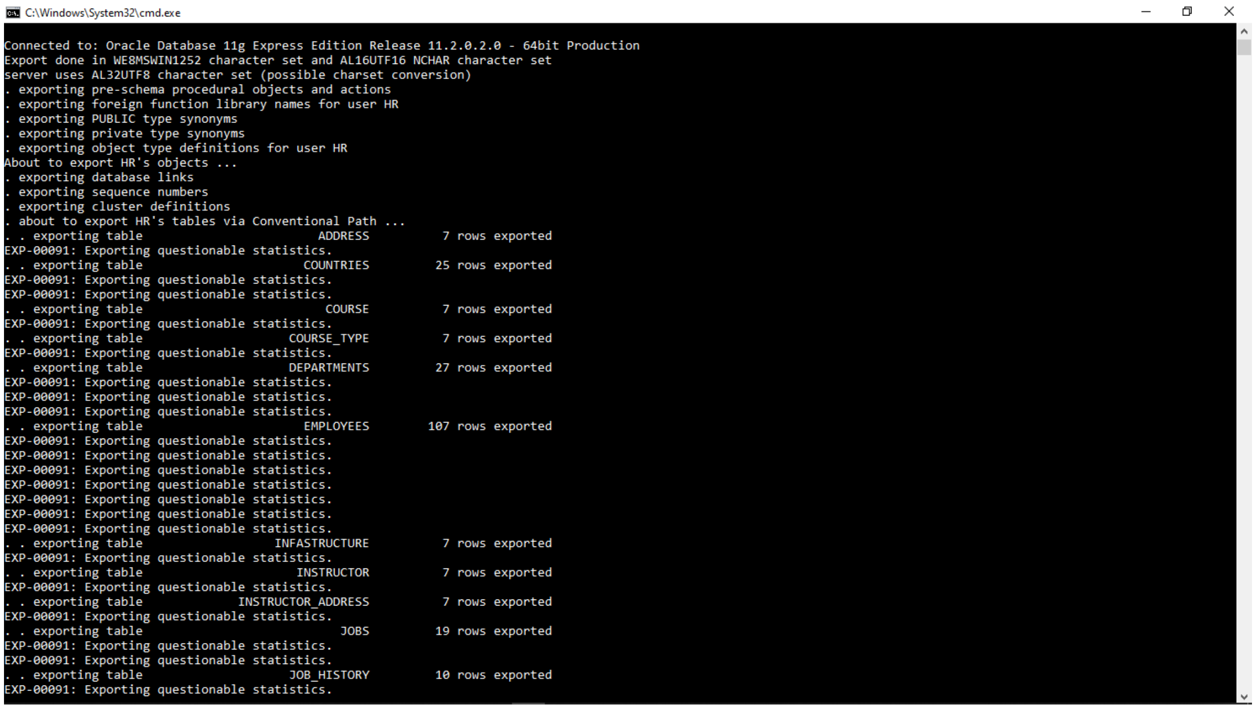
Dump file



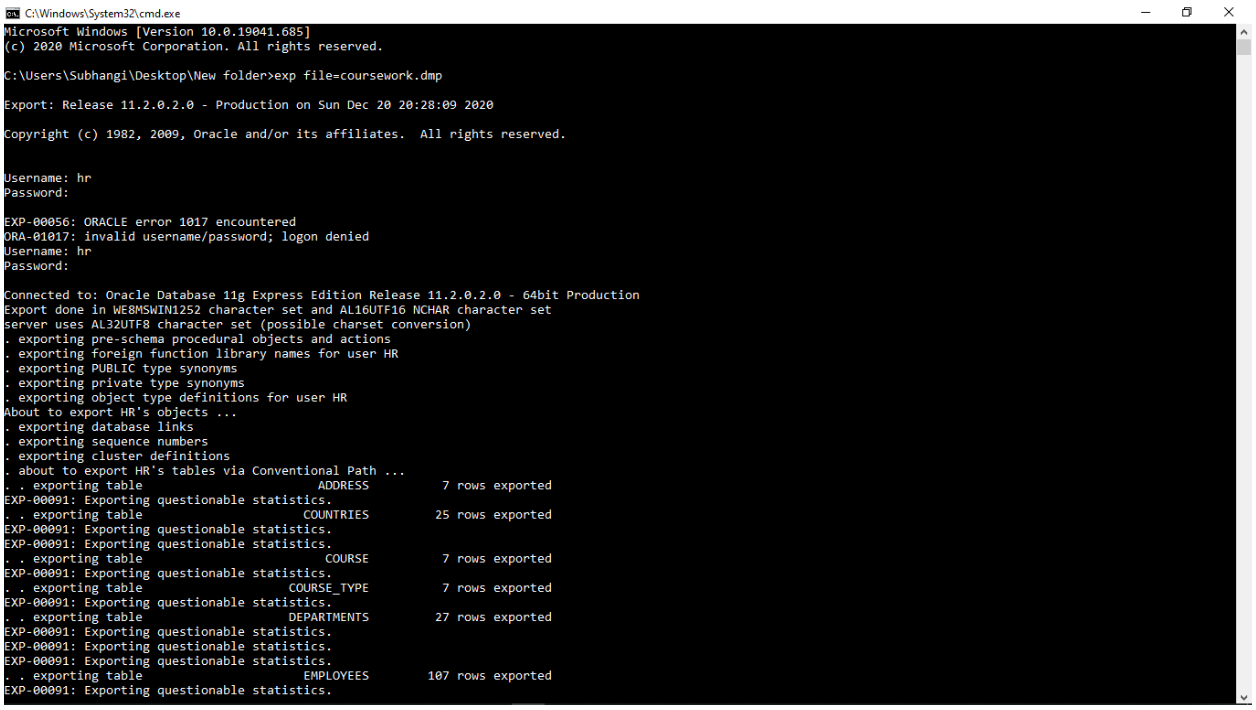
```
EXP-00091: Exporting questionable statistics.
. . exporting table                      LOCATIONS         23 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                         MODULE          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                        REGIONS          4 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                  SPECIFICATION          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table    SPECIFICATION_MODULE_COURSE          0 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                        STUDENT          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                STUDENT_ADDRESS          7 rows exported
EXP-00091: Exporting questionable statistics.
. exporting synonyms
. exporting views
. exporting stored procedures
. exporting operators
. exporting referential integrity constraints
. exporting triggers
. exporting indextypes
. exporting bitmap, functional and extensible indexes
. exporting posttables actions
. exporting materialized views
. exporting snapshot logs
. exporting job queues
. exporting refresh groups and children
. exporting dimensions
. exporting post-schema procedural objects and actions
. exporting statistics
Export terminated successfully with warnings.

C:\Users\Subhangi\Desktop\New folder>e
```



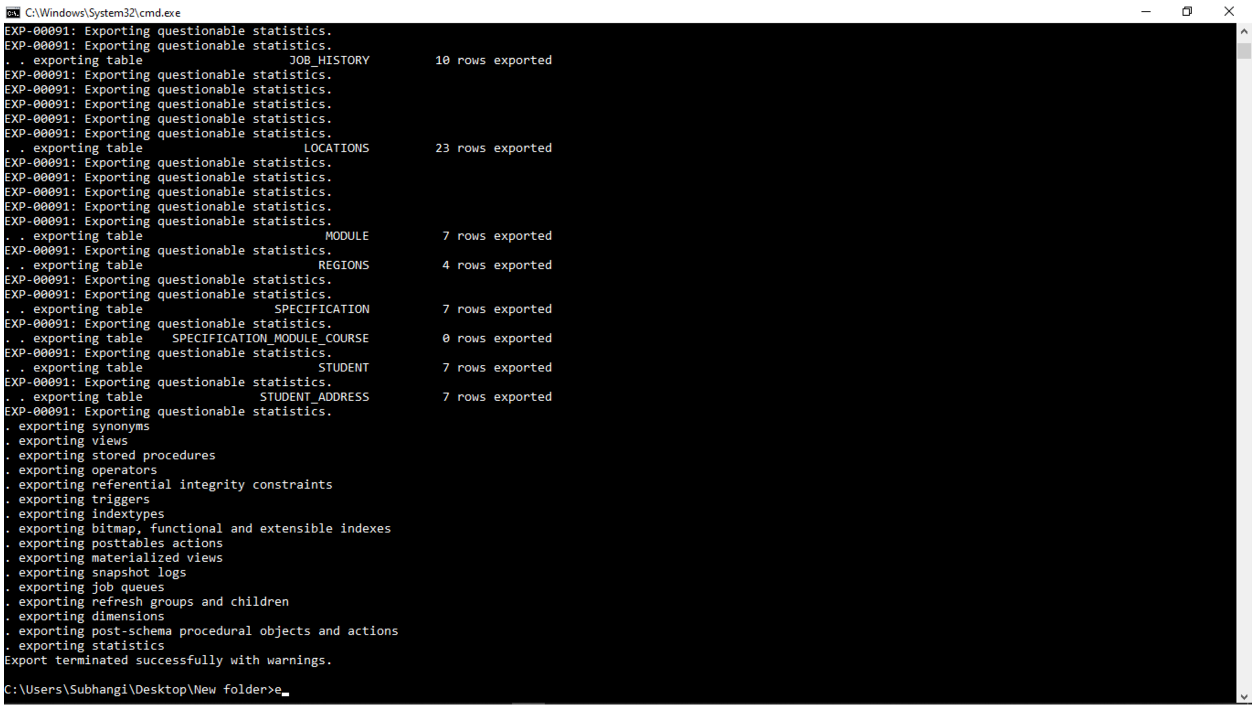
```
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                  INFASTRUCTURE          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                     INSTRUCTOR          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table             INSTRUCTOR_ADDRESS          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                           JOBS         19 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                    JOB_HISTORY         10 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                      LOCATIONS         23 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                         MODULE          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                        REGIONS          4 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                  SPECIFICATION          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table    SPECIFICATION_MODULE_COURSE          0 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                        STUDENT          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                STUDENT_ADDRESS          7 rows exported
EXP-00091: Exporting questionable statistics.
. exporting synonyms
. exporting views
. exporting stored procedures
. exporting operators
. exporting referential integrity constraints
. exporting triggers
```


C:\Windows\System32\cmd.exe

```
Connected to: Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production
Export done in WE8MSWIN1252 character set and AL16UTF16 NCHAR character set
server uses AL32UTF8 character set (possible charset conversion)
. exporting pre-schema procedural objects and actions
. exporting foreign function library names for user HR
. exporting PUBLIC type synonyms
. exporting private type synonyms
. exporting object type definitions for user HR
About to export HR's objects ...
. exporting database links
. exporting sequence numbers
. exporting cluster definitions
. about to export HR's tables via Conventional Path ...
. . exporting table                        ADDRESS          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                      COUNTRIES         25 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                         COURSE          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                    COURSE_TYPE          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                    DEPARTMENTS         27 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                      EMPLOYEES        107 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                 INFASTRUCTURE          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                     INSTRUCTOR          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table             INSTRUCTOR_ADDRESS          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                           JOBS         19 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                    JOB_HISTORY         10 rows exported
EXP-00091: Exporting questionable statistics.
```


C:\Windows\System32\cmd.exe

```
Microsoft Windows [Version 10.0.19041.685]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\Subhangi\Desktop\New folder>exp file=coursework.dmp

Export: Release 11.2.0.2.0 - Production on Sun Dec 20 20:28:09 2020

Copyright (c) 1982, 2009, Oracle and/or its affiliates.  All rights reserved.


Username: hr
Password:

EXP-00056: ORACLE error 1017 encountered
ORA-01017: invalid username/password; logon denied
Username: hr
Password:

Connected to: Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production
Export done in WE8MSWIN1252 character set and AL16UTF16 NCHAR character set
server uses AL32UTF8 character set (possible charset conversion)
. exporting pre-schema procedural objects and actions
. exporting foreign function library names for user HR
. exporting PUBLIC type synonyms
. exporting private type synonyms
. exporting object type definitions for user HR
About to export HR's objects ...
. exporting database links
. exporting sequence numbers
. exporting cluster definitions
. about to export HR's tables via Conventional Path ...
. . exporting table                        ADDRESS          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                      COUNTRIES         25 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                         COURSE          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                    COURSE_TYPE          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                    DEPARTMENTS         27 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                      EMPLOYEES        107 rows exported
EXP-00091: Exporting questionable statistics.
```



C:\Windows\System32\cmd.exe

```
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                    JOB_HISTORY         10 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                      LOCATIONS         23 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                         MODULE          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                        REGIONS          4 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                  SPECIFICATION          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table    SPECIFICATION_MODULE_COURSE          0 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                        STUDENT          7 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                STUDENT_ADDRESS          7 rows exported
EXP-00091: Exporting questionable statistics.
. exporting synonyms
. exporting views
. exporting stored procedures
. exporting operators
. exporting referential integrity constraints
. exporting triggers
. exporting indextypes
. exporting bitmap, functional and extensible indexes
. exporting posttables actions
. exporting materialized views
. exporting snapshot logs
. exporting job queues
. exporting refresh groups and children
. exporting dimensions
. exporting post-schema procedural objects and actions
. exporting statistics
Export terminated successfully with warnings.

C:\Users\Subhangi\Desktop\New folder>e
```

8. Conclusion

This coursework accounted for 50% of the module's total grade and was assigned to us at week 7. After the corona virus pandemic took over the world, the beginning of online classes made a huge impact in our education system. Of course, it was a privilege to be able to sit in front of our desktop and join the classes but at the same time I realized that taking classes through zoom meetings made me slothful and less interested as they were not that interactive.

Being a part of the IT world means advancing with the rapidly growing technology and swift moving world. Limiting ourselves to the online classes would not be adequate. An actual IT student will always keep himself updated with everyday news and resources. When the course work was assigned I was terrified because we could not sit together with our classmates to solve the problems or get a practical help from our tutors. I knew I had to do this all on my own and it would not be easy. Only after reading the question paper for about four times I got a clear picture of what the coursework was about. Slowly I started drawing rough sketches and building blocks in my head. I noted down the blueprint so I had a clear vision of how I wanted my coursework to go.

I started first by analyzing the scenario then I created the entities and attributes after which I drew an initial ER diagram. The normalization part was really tough. I had an awareness back in my mind what normalization was about and how it was done but in my assignment I was finding it really difficult to put pieces together and even after several tries nothing was making sense to me. Then I decided to slow down and go through the recording and slides once again that were uploaded by the tutors. I also did a lot of research and watched a lot of videos on normalization. My tutors as well as my seniors guided me to complete the normalization process. We had to convert UNF to 3NF. I finalized the normalization after properly identifying the entity and attributes and analyzing the scenarios and queries. It took me few days to complete the normalization and draw the tables and Er diagram

After the data normalization, I inserted data in the created tables according to the queries provided to us and also wrote the informational and transactional queries from the tables. The querying part was a bit challenging so as usual I did some research on the web and revised the lecture slides. At last I included the screenshots and wrapped up the project with further discussion on the learning experience.

I would sincerely like to thank my tutors, seniors, my friends for guiding me in this journey and of course myself for incessant hard work and dedication.

WHAT I LEARNED FROM THE COURSEWORK:

- To organize data in a database
- Retrieve, add, update delete from database
- Deal with data redundancy, data inconsistency & anomalies using normalization
- Partial and transitive dependencies
- Create tables in SQL and draw ER diagram
- Normalize data from UNF to 3NF which makes it possible to organize huge amount of data.
- Broadens knowledge on SQL queries

Bibliography

guru99, 2005. *What is Normalization? 1NF, 2NF, 3NF, BCNF Database Example.* [Online]
Available at: https://www.guru99.com/database-normalization.html
[Accessed 2020].

Kaula, R., 2007. *Normalizing with Entity Relationship Diagramming – TDAN.com.* [Online]
Available at: https://tdan.com/normalizing-with-entity-relationship-diagramming/4583#:~:text=One%20of%20the%20ways%20an,tenets%20in%20relational%20model%20design.&text=Normalization%20utilizes%20association%20among%20attributes,table%20to%20accomplish%20its%20objective
[Accessed 2020].

krishnaswamy, 2019. *Number of possible Superkeys in DBMS - GeeksforGeeks.* [Online]
Available at: https://www.geeksforgeeks.org/number-of-possible-superkeys-in-dbms/
[Accessed 2020].

rdbms.opengrass, n.d. *SQA.* [Online]
Available at:
http://rdbms.opengrass.net/2_Database%20Design/2.2_Normalisation/r/UNF%20Create%20Model%20Data.pdf
[Accessed 2020].

SINGH, C., 2015. *Normalization in DBMS: 1NF, 2NF, 3NF and BCNF in Database.* [Online]
Available at: https://beginnersbook.com/2015/05/normalization-in-dbms/
[Accessed 2020].

Stephens, R. P. a. R., 2003. *The Database Normalization Process | Normalizing a Database | InformIT.* [Online]

Available at: https://www.informit.com/articles/article.aspx?p=30646 [Accessed 2020].