



CIS 9340 SECTION QMWA STUDENT REGISTRATION SYSTEM DATABASE PROJECT

Group 3

Name	Email
Tzuyi Li	tzuyi.li@baruchmail.cuny.edu
Sitong Li	sitong.li@baruchmail.cuny.edu
Xinyue Chen	xinyue.chen@baruchmail.cuny.edu
Yue Zhan	yue.zhan@baruchmail.cuny.edu
Yining Yao	yining.yao@baruchmail.cuny.edu



Table of Contents

I.	3
II.	4
·	5
·	5
III.	6
·	6
IV.	7
·	7
·	7
·	8
·	8
·	9
·	9
·	9
V.	10
·	10
·	12
·	13
·	14
VI.	31
·	31
·	32
·	38
·	40
VII.	46

I. Business Scenario Introduction

ABC College is a public school located in the lower midtown Manhattan. With the retirement of the old system after years of usage, ABC College needs a new student registration system that enables students to get enrolled in different courses and get their grades.

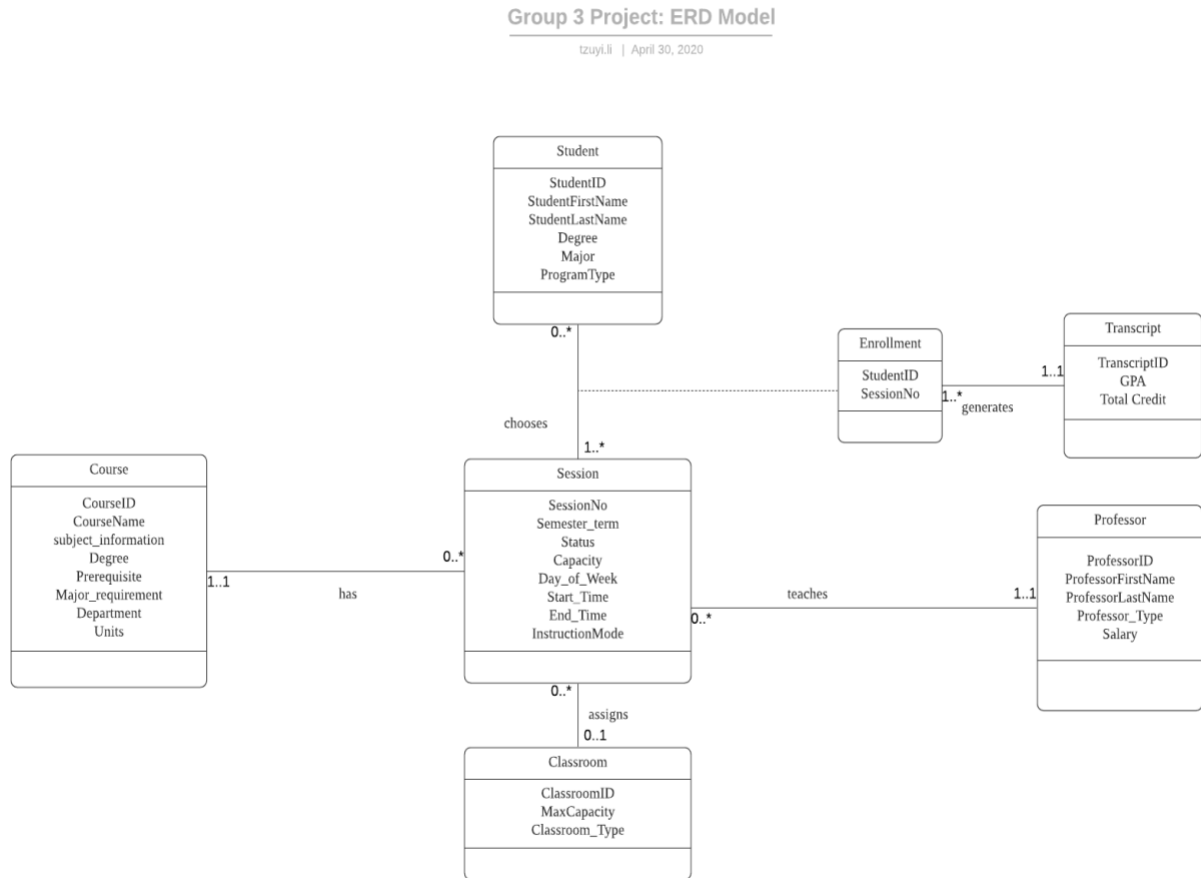
The College offers multiple courses with different sections every semester. The registrar office needs to allocate classrooms, instructors, and schedules for all the course sections, which creates excessive data entries and heavy managing workload.

ABC College would like to se

Ultimately, the system should help improve efficiency, avoid time clashes and monitor operations.

II. Entity Relationship Model Diagram

Based on the above background, we developed an Entity Relationship Model, using UML notion.



● Assumptions

1. This database only includes current students.
2. We assume one session can be selected by zero to more students, if it's not selected by any student, the session status will be marked as 'closed', but the record remains.
3. We assume some sessions may not be assigned with any classroom if the course is on online mode.
4. We assume "GPA" in the Transcript Table is the cumulative result of all past semesters.

● Relationship Sentences

A course may have many sessions.

A session must belong to one course.

A student may choose many sessions.

A session may be chosen by many students.

A session may assign to one classroom.

A classroom may be assigned to many sessions.

A professor may teach many sessions.

A session must be taught by one professor.

An enrollment must be in one transcript.

A transcript may be generated by many enrollments.

III. Conversion to Relational Model

After creating the Entity Relationship Model, the next step is to convert it to the Relational Model. We have created the following relational model according to the seven entities.

- Relational Model

Course(CourseID(PK), CourseName, Subject_information, Degree, Prerequisite, Major_requirement, Department, Units)

Student(StudentID(PK), StudentName, DegreeType, Major, ProgramType)

Session(SessionNo(PK), Semester_term, Status, Capacity, Day_of_Week, Start_Time, End_Time, InstructionMode, ProfessorID(FK), ClassroomID(FK), CourseID(FK))

Classroom(ClassroomID(PK), Classroom_Type, MaxCapacity)

Transcript(TranscriptID(PK), GPA, TotalCredit)

Professor(ProfessorID(PK), ProfessorName, Professor_Type, Salary)

Enrollment(StudentID(FK), SessionNo(FK), TranscriptID(FK))

IV. Normalization

● Course

(CourseID(PK), CourseName, Subject_information, Degree, Prerequisite, Major_requirement, Department, Units)

FD1: CourseID → CourseName, Subject_information, Degree, Prerequisite, Major_requirement, Department, Units

FD2: CourseName, Degree → Subject_information, Prerequisite, Major_requirement, Department, Units

How we normalize it?

Step1: Is there a primary key? Yes, CourseID is the unique identifier for the table. It is in 1NF.

Step2: There is no composite key, so it is in 2NF.

Step3: Are there any transitive dependencies?

Yes, CourseID → CourseName, Degree → Subject_information, Prerequisite, Major_requirement, Department, Units

It is not in 3NF.

Step4: Split Course into CourseDetail and Course.

CourseDetail (CourseName(pk), Degree(pk), Subject_information, Prerequisite, Major_requirement, Department, Units)

Course (CourseID(pk), CourseName(FK), Degree(FK))

Step5: Check if there is any transitive dependency remained. No, it is 3NF.

Step6: Is any determinant non-candidate key? The answer is no. It is in BCNF.

● Student

(StudentID(PK), StudentName, DegreeType, Major, ProgramType)

FD1: StudentID → StudentName, DegreeType, Major, ProgramType

How we normalize it?

Step 1: Is there a primary key? Yes, StudentID is the unique identifier for the table. It is in 1NF.

Step 2: Are there any partial dependencies? No, it is in 2NF.

Step 3: Are there any transitive dependencies? No, it is in 3NF.

Step 4: Is any determinant non-candidate key? The answer is no. It is in BCNF.

● Session

(SessionNo, Semester_term, Status, Capacity, Day_of_Week, Start_Time, End_Time, InstructionMode, ProfessorID, ClassroomID, CourseID)

FD1: SessionNo → Semester_term, Status, Capacity, Day_of_Week, Start_Time, End_Time, InstructionMode, ProfessorID, ClassroomID, CourseID

FD2: Day_of_Week, Start_Time, End_Time, ProfessorID → ClassroomID, CourseID, Semester_term, Status, Capacity, InstructionMode

How we normalize it?

Step1: Is there a primary key? Yes, SessionID is the unique identifier for the table. It is in 1NF.

Step2: Are there any partial dependencies? No, it is in 2NF.

Step3: Are there any transitive dependencies?

Yes, SessionNo → Day_of_Week, Start_Time, End_Time, ProfessorID → ClassroomID, CourseID, Semester_term, Status, Capacity, InstructionMod

Step4: Split Session into Session and SessionDetail.

Session (SessionNo, Day_of_Week(FK), Start_Time(FK), End_Time(FK), ProfessorID(FK))

SessionDetail (Day of Week, Start Time, End Time, ProfessorID, ClassroomID, CourseID, Semester_term, Status, Capacity, InstructionMod)

Step5: Check if there is any transitive dependency remained. No, it is in 3NF.

Step6: Is any determinant non-candidate key? The answer is no. It is in BCNF.

● Classroom

(ClassroomID, Classroom_Type, MaxCapacity)

FD1: Classroom ID (PK) → Classroom_Type, MaxCapacity

How we normalize it?

Step 1: Is there a primary key? Yes, ClassroomID is the unique identifier for the table. It is in 1NF.

Step 2: Are there any partial dependencies? No, it is in 2NF.

Step 3: Are there any transitive dependencies? No, it is in 3NF.

Step 4: Is any determinant non-candidate key? The answer is no. It is in BCNF.

● Transcript

(TranscriptID, GPA, TotalCredit)

FD1: TranscriptID → GPA, TotalCredit

How we normalize it?

Step 1: Is there a primary key? Yes, TranscriptID is the unique identifier for the table. It is in 1NF.

Step 2: Are there any partial dependencies? No, it is in 2NF.

Step 3: Are there any transitive dependencies? No, it is in 3NF.

Step 4: Is any determinant non-candidate key? The answer is no. It is in BCNF.

● Professor

(ProfessorID, ProfessorName, Professor_Type, Salary)

FD1: ProfessorID(PK) → ProfessorName, Professor_Type, Salary

Step 1: Is there a primary key? Yes, ProfessorID is the unique identifier for the table. It is in 1NF.

Step 2: Are there any partial dependencies? No, it is in 2NF.

Step 3: Are there any transitive dependencies? No, it is in 3NF.

Step 4: Is any determinant non-candidate key? The answer is no. It is in BCNF.

● Enrollment

(StudentID(FK), SessionNo(FK), TranscriptID(FK))

FD1: StudentID, SessionNo → TranscriptID

How we normalize it?

Step 1: Is there a primary key? Yes, StudentID and SessionNo are the composite key for the table. It is in 1NF.

Step 2: Are there any partial dependencies? No, it is in 2NF.

Step 3: Are there any transitive dependencies? No, it is in 3NF.

Step 4: Is any determinant non-candidate key? The answer is no. It is in BCNF.

V. Create the Database Schema with SQL

We use the following SQL codes to create the seven tables and add Primary Key and Foreign Key to each one:

- Create tables

CourseDetail

```
CREATE TABLE CourseDetail (  
  CourseName VARCHAR(50) NOT NULL,  
  Degree VARCHAR(50) NOT NULL,  
  Subject_Information VARCHAR(5000) NOT NULL,  
  Prerequisite VARCHAR(3000) NOT NULL,  
  Major_requirement VARCHAR(1000) NOT NULL,  
  Department VARCHAR(30) NOT NULL,  
  Units INT NOT NULL,  
);
```

Course

```
CREATE TABLE Course(  
  CourseID VARCHAR(30) NOT NULL,  
  CourseName varchar(50) NOT NULL,  
  Degree varchar(50) NOT NULL,  
);
```

Session

```
CREATE TABLE Session(  
  SessionNo VARCHAR(50) NOT NULL,  
  Day_of_Week VARCHAR(10) NOT NULL,  
  Start_Time TIME NOT NULL,  
  End_Time TIME NOT NULL,  
  ProfessorID VARCHAR(10) NOT NULL  
);
```

SessionDetail

```
CREATE TABLE SessionDetail(  
  Day_of_Week VARCHAR(10) NOT NULL,  
  Start_Time TIME NOT NULL,  
  End_Time TIME NOT NULL,  
  ProfessorID VARCHAR(10) NOT NULL,  
  ClassroomID VARCHAR(15) NOT NULL,  
  CourseID VARCHAR(30) NOT NULL,  
  Semester_term VARCHAR(8) NOT NULL,  
  Status VARCHAR(10) NOT NULL,
```

Capacity NUMBER NOT NULL,
InstructionMod varchar (10) NOT NULL
);

Student

```
CREATE TABLE Student (  
    StudentID varchar(8) NOT NULL UNIQUE,  
    StudentFirstName Varchar(50) NOT NULL,  
    StudentLastName Varchar(50) NOT NULL,  
    Degree varchar(50) NOT NULL,  
    Major varchar (30) NOT NULL,  
    ProgramType varchar(15) NOT NULL  
);
```

Classroom

```
CREATE TABLE Classroom(  
    ClassroomID VARCHAR(15) NOT NULL UNIQUE,  
    Classroom_Type VARCHAR(15) NOT NULL,  
    MaxCapacity NUMBER NOT NULL  
);
```

Transcript

```
CREATE TABLE Transcript(  
    TranscriptID VARCHAR(10) NOT NULL,  
    GPA NUMBER NOT NULL,  
    TotalCredit NUMBER NOT NULL  
);
```

Professor

```
CREATE TABLE professor (  
    professorid VARCHAR(10) NOT NULL UNIQUE,  
    professor_first_name VARCHAR (50) NOT NULL,  
    professor_last_name VARCHAR (50) NOT NULL,  
    professor_type VARCHAR(8) NOT NULL,  
    salary NUMBER NOT NULL  
);
```

Enrollment

```
CREATE TABLE enrollment (  
    StudentID VARCHAR(8) NOT NULL UNIQUE,  
    SessionNo VARCHAR (15) NOT NULL,  
    TranscriptID VARCHAR (10) NOT NULL  
);
```

- Adding Primary Keys

```
ALTER TABLE CourseDetail  
ADD CONSTRAINT pk_CourseDetail  
PRIMARY KEY (CourseName, Degree);
```

```
ALTER TABLE Course  
ADD CONSTRAINT pk_Course  
PRIMARY KEY (CourseID);
```

```
ALTER TABLE student  
ADD CONSTRAINT pk_student  
PRIMARY KEY (StudentID);
```

```
ALTER TABLE SessionDetail  
ADD CONSTRAINT pk_sessiondetail  
PRIMARY KEY (Day_of_Week, Start_Time, End_Time, ProfessorID);
```

```
ALTER TABLE Session  
ADD CONSTRAINT pk_session  
PRIMARY KEY (SessionNo);
```

```
ALTER TABLE Classroom  
ADD CONSTRAINT pk_Classroom  
PRIMARY KEY (ClassroomID);
```

```
ALTER TABLE Transcript  
ADD CONSTRAINT pk_Transcript  
PRIMARY KEY (TranscriptID);
```

```
ALTER TABLE professor  
ADD CONSTRAINT pk_professor  
PRIMARY KEY (professorid);
```

```
ALTER TABLE enrollment  
ADD CONSTRAINT pk_enrollment  
PRIMARY KEY (StudentID, SessionNo);
```

- Adding Foreign Keys

```
ALTER TABLE Session
ADD CONSTRAINT fk_session_sessiondetail
FOREIGN KEY (Day_of_Week, Start_Time, End_Time, ProfessorID)
REFERENCES SessionDetail (Day_of_Week, Start_Time, End_Time, ProfessorID)
```

```
ALTER TABLE SessionDetail
ADD CONSTRAINT fk_sessiondetail_course
FOREIGN KEY (CourseID)
REFERENCES Course (CourseID)
```

```
ALTER TABLE SessionDetail
ADD CONSTRAINT fk_sessiondetail_professor
FOREIGN KEY (ProfessorID)
REFERENCES Professor (ProfessorID)
```

```
ALTER TABLE SessionDetail
ADD CONSTRAINT fk_sessiondetail_classroom
FOREIGN KEY (ClassroomID)
REFERENCES Classroom (ClassroomID)
```

```
ALTER TABLE Course
ADD CONSTRAINT fk_CourseDetail_Course
FOREIGN KEY (CourseName, Degree)
REFERENCES CourseDetail (CourseName, Degree)
```

```
ALTER TABLE enrollment
ADD CONSTRAINT fk_enrollment_student
FOREIGN KEY (StudentID)
REFERENCES Student (StudentID);
```

```
ALTER TABLE enrollment
ADD CONSTRAINT fk_enrollment_session
FOREIGN KEY (SessionNo)
REFERENCES Session (SessionNo);
```

```
ALTER TABLE enrollment
ADD CONSTRAINT fk_enrollment_transcript
FOREIGN KEY (TranscriptID)
REFERENCES Transcript (TranscriptID);
```

- Insert Data to the Tables

Course Detail

```
INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite,  
Major_requirement, Department, Units)
```

```
VALUES ('Programming for Analytics', 'Undergraduate',
```

```
'This course introduces the aspects of programming that can support business analytics.
```

```
The course introduces students to programming (using a language such as python) and its  
uses in business analytics. The course covers hands-on issues in programming for analytics  
which include accessing data, creating informative data graphics, writing functions,  
debugging, and organizing and commenting code.', 'Prerequisites: CIS 2300 or Equivalent',  
'CIS/CS/CE',
```

```
'Computer Information Systems', 3.00);
```

```
INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite,  
Major_requirement, Department, Units)
```

```
VALUES ('Principles of FinTech', 'Graduate',
```

```
'The course is designed to give students a general understanding of  
Financial Technology (FinTech) in the context of the capital markets and  
financial services industry. The students will learn how IT affects the  
financial services industry and how to assess the potential for strategic  
advantage based on information technology. The course will survey  
emerging issues in the FinTech arena such as cryptocurrencies, blockchains,  
cybersecurity and machine learning and assess their application to an array  
of financial services.', 'N/A', 'CIS/FIN/BUS/ECO', 'Computer Information Systems', 3.00);
```

```
INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite,  
Major_requirement, Department, Units)
```

```
VALUES ('Data Mining for Business Analytics', 'Graduate',
```

```
'Data Mining is the process by which useful information is extracted from large amounts of  
data. This course is designed to provide students with the necessary tools and techniques to  
perform data mining and business analytics. The topics will include essentials of data  
management, data preparation, and model development. In addition, students will learn  
about model assessment and validation. Emphasis will be placed on careful presentation of  
quantitative aspects of data mining and business analytics, as well as on applications to big  
data.',
```

```
'Prerequisites: STA 9708','CIS/BA','Computer Information Systems',3.00);
```

```
INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite,  
Major_requirement, Department, Units)
```

```
VALUES ('Corporate Finance','Undergraduate',
```

```
'This course introduces students to the fundamental financial issues of the corporation. It  
covers basic concepts of debt and equity sources of financing and valuation; capital  
budgeting methods; cash flow forecasting and risk analysis; and the cost of capital. It  
introduces students to the process of securities issuance and techniques of financial  
planning and forecasting.',
```

'Prerequisite: FIN 300','FIN/ACC/BUS/ECO','Economics and Finance',3.00);

INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite, Major_requirement, Department, Units)

VALUES ('Futures and Forward Markets','Graduate',

'Study of derivative securities: interest, foreign currency, and equity swaps; the spot and futures markets; caps, floors, collars, and corridors; forward rate agreements (FRAs), and program trading. Market structure and valuation methods are examined.',

'Pre/Co-requisite: FIN 9783 or FIN 9773','FIN/ACC/BUS/ECO','Economics and Finance',3.00);

INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite, Major_requirement, Department, Units)

VALUES ('Investment Analysis','Undergraduate',

'This course introduces students to the fundamental principles and theories of financial asset pricing and valuation. It provides students with a rigorous analysis of modern portfolio theory, the capital asset pricing model, and the valuation of common stocks and bonds. It includes an introduction to the main financial markets, their organization and functional characteristics.',

'Pre/Co-requisite: FIN 3000','FIN/ACC/BUS/ECO','Economics and Finance',3.00);

INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite, Major_requirement, Department, Units)

VALUES ('Cost Accounting','Undergraduate',

'This course examines the measurement of costs, the compilation of cost data, and the impact of accounting data on the allocation of resources within an organization. Topics discussed include systems for cost accumulation, joint and by-products, budgeting, standard costs, and direct costing. The course integrates materials from accounting with economic analysis, quantitative methods, and behavioral science as the course also covers capital budgeting, cost-volume-profit analysis, profit performance, regression analysis, and linear programming. Credit will not be granted for both ACC 2203 and ACC

3200.','N/A','ACC','Accountancy',4.00);

INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite, Major_requirement, Department, Units)

VALUES ('Auditing and Accounting Information Systems','Graduate',

'This course covers in-depth audits of accounting information systems and their internal controls. Common accounting and auditing problems encountered in computer-based information systems are identified and analyzed along with their solutions or other means of managing their effects. There is also consideration of ways of using the computer as an audit tool.',

'Prerequisite: ACC 9110 or 9112.','ACC/CIS/ECO/FIN','Accountancy',3.00);

INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite, Major_requirement, Department, Units)

VALUES ('Data Analytics in Accounting','Graduate','Data Analytics in Accounting','Prerequisite: ACC 9112 or ACC 9110 Corequisites: ACC 9811 and ACC 9818','ACC/CIS/ECO/BA', 'Accountancy',4.00);

INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite, Major_requirement, Department, Units)
VALUES ('Business Statistics I','Undergraduate',
'A one-semester broad-based introductory business statistics course that focuses on descriptive statistics, control charts, regression, and inferential statistics. Topics covered include graphical methods, descriptive statistics with exploratory data analysis, an introduction to control charts (with a focus on special cause and common cause variation), linear regression and correlation, the normal distribution and sampling distribution of the mean, estimation for means and proportions, and hypothesis testing for one and two groups. Students will use a microcomputer statistical package for analyzing selected data sets. This course is required for all BBA students.','Prerequisite: Sophomore status, MTH 2003 or MTH 2009 or equivalent. Prerequisite or Corequisite: CIS 2200. Not open to students who have completed STA 2100.',
'STA','Computer Information Systems',3.00);

INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite, Major_requirement, Department, Units)
VALUES ('Managerial Statistics','Graduate',
'This course provides MBA students with the statistical tools and concepts needed in business applications. Students will learn quantitative analysis critical for business decision making. Emphasis will be placed on understanding data analysis and interpretation. Topics include: interpretation of descriptive measures; applications of probability and the normal distribution; confidence interval estimation; hypothesis testing; simple linear regression models; and multiple regression models. The students will use standard spreadsheet software to work with data and apply the concepts learned. Discussions of ethical issues are integrated throughout the course. This course will enhance skills in critical thinking, as well as oral and written communication. Techniques learned in this course can be immediately put to use by the student.',
'Not open to students who have completed FIN 9762.',
'STA/CIS/BA/MBA/ACC/FIN/ECO','Computer Information Systems',3.00);

INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite, Major_requirement, Department, Units)
VALUES ('Introduction to SAS Programming','Undergraduate',
'This course is a hands-on introduction to modern computer-intensive statistical methods, such as the bootstrap and Monte Carlo methods. It will be taught on platforms, such as Microsoft Excel and Visual Basic for Applications (VBA). Emphasis will be placed on fundamental programming rather than the use of canned routines. Applications will be taken from the fields of finance and accounting.','PREREQUISITE: STA 2000 or ECO 4000','STA/CIS/BA/MBA/ACC/FIN/ECO','Computer Information Systems',3.00);

INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite, Major_requirement, Department, Units)

VALUES ('Marketing Research','Undergraduate',
 'Training in the basic techniques of research in marketing, including problems definition,
 research design, questionnaire construction, sampling, and data collection and analysis, and
 report preparation. The student will design and will analyze cases based on real-world
 business problems and provide a written report for each.',
 'Prerequisite: MKT 3000 AND STA 2000','MKT','Marketing and Intl Business',4.00);

INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite,
 Major_requirement, Department, Units)
 VALUES ('Media Planning','Undergraduate',
 'This course examines the development of effective media strategy for advertising and other
 areas of marketing communication. Each of the major media channels for promotion are
 covered in depth. Students develop their own media plans and conduct a variety of
 computer-assisted quantitative analyses to assess competitive spending, set objectives, and
 evaluate the audience delivery of alternative media schedules.',
 'Prerequisite: MKT 3000 AND MKT 3520','MKT','Marketing and Intl Business',3.00);

INSERT INTO CourseDetail (CourseName, Degree, Subject_information, Prerequisite,
 Major_requirement, Department, Units)
 VALUES ('Data-driven Marketing Strategy','Graduate',
 'This course will focus on developing marketing mix and resource allocation decisions driven
 by quantitative analysis. The course will specifically focus on building quantitative models of
 pricing, channels of distribution, branding, and promotions. Topics covered include market
 response models, conjoint analysis techniques, resource allocation models, forecasting
 models, customer profitability analysis, value pricing, product line decisions, and other
 significant strategic marketing issues facing today's managers.',
 'Prerequisite: MKT 9703','MKT/BA','Marketing and Intl Business',3.00);

Course

INSERT INTO Course (CourseID, CourseName, Degree)
 VALUES ('CIS 3140', 'Programming for Analytics', 'Undergraduate');
 INSERT INTO Course (CourseID, CourseName, Degree)
 VALUES ('CIS 9555', 'Principles of FinTech', 'Graduate');
 INSERT INTO Course (CourseID, CourseName, Degree)
 VALUES ('CIS 9660', 'Data Mining for Business Analytics', 'Graduate');
 INSERT INTO Course (CourseID, CourseName, Degree)
 VALUES ('FIN 3610', 'Corporate Finance', 'Undergraduate');
 INSERT INTO Course (CourseID, CourseName, Degree)
 VALUES ('FIN 9782', 'Futures and Forward Markets', 'Graduate');
 INSERT INTO Course (CourseID, CourseName, Degree)
 VALUES ('FIN 3710', 'Investment Analysis', 'Undergraduate');
 INSERT INTO Course (CourseID, CourseName, Degree)
 VALUES ('ACC 3200', 'Cost Accounting', 'Undergraduate');
 INSERT INTO Course (CourseID, CourseName, Degree)
 VALUES ('ACC 9818', 'Auditing and Accounting Information Systems', 'Graduate');
 INSERT INTO Course (CourseID, CourseName, Degree)

```

VALUES ('ACC 9886', 'Data Analytics in Accounting', 'Graduate');
INSERT INTO Course (CourseID, CourseName, Degree)
VALUES ('STA 2000', 'Business Statistics I', 'Undergraduate');
INSERT INTO Course (CourseID, CourseName, Degree)
VALUES ('STA 9708', 'Managerial Statistics', 'Graduate');
INSERT INTO Course (CourseID, CourseName, Degree)
VALUES ('STA 4000', 'Introduction to SAS Programming', 'Undergraduate');
INSERT INTO Course (CourseID, CourseName, Degree)
VALUES ('MKT 3600', 'Marketing Research', 'Undergraduate');
INSERT INTO Course (CourseID, CourseName, Degree)
VALUES ('MKT 4120', 'Media Planning', 'Undergraduate');
INSERT INTO Course (CourseID, CourseName, Degree)
VALUES ('MKT 9740', 'Data-driven Marketing Strategy', 'Graduate');

```

Session

```

INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(8756,"We","5:50pm","7:10pm","1061046671")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(9875,"MoWe","9:05am","10:20am","1735156751")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(1288,"Fr","2:30pm","5:25pm","1314842598")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(9754,"Th","7:30pm","8:45pm","1904605352")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(2901,"MoWe","9:05am","10:20am","1680104634")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(8755,"TuTh","7:30pm","8:45pm","1698078080")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(9011,"We","11:10am","2:05pm","1997443771")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(5999,"TuTh","11:10am","2:05pm","1733708699")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(8655,"Sa","7:50am","10:45am","1970551178")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(4217,"Fr","2:30pm","5:25pm","1240328719")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(7580,"Sa","9:05am","10:20am","1775030773")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(3446,"We","11:10am","2:05pm","1738826610")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(8855,"Th","7:50am","10:45am","1146203408")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(3255,"Th","7:50am","10:45am","1698078080")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(2021,"TuTh","11:10am","2:05pm","1942648288")

```

```

INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(7657,"Sa","2:30pm","5:25pm","1477000521")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(0777,"MoWe","5:50pm","7:10pm","1997443771")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(7566,"Sa","2:30pm","5:25pm","1114667337")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(7647,"Fr","11:10am","2:05pm","1148529568")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(5445,"TuTh","7:50am","10:45am","1114667337")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(6434,"Tu ","2:30pm","5:25pm","1574751191")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(7976,"Mo","7:50am","10:45am","1131774660")
INSERT INTO Session(SessionNo,Day_of_Week, Start_Time,End_Time,ProfessorID)
VALUES(0988,"Fr","5:50pm","7:10pm","1588901445")

```

SessionDetail

```

INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("We","5:50pm","7:10pm","1061046671","B7152","ACC 3200","Summer","Wait
List",25,"Regular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("MoWe","9:05am","10:20am","1735156751","C6252","STA9708","Spring","Closed"
,25,"Regular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("Fr","2:30pm","5:25pm","1314842598","C6253","ACC 3200","Fall","WaitList",45 ,
"Online")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("Th","7:30pm","8:45pm","1904605352","B7153","ACC9818","Winter","Closed",50,"
Regular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("MoWe","9:05am","10:20am","1680104634","D8250","ACC9818","Fall","Open",30,
"Hybrid")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)

```

```

VALUES("TuTh","7:30pm","8:45pm","1698078080","A3151","CIS9555","Spring","Closed",35,
,"Hybrid")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("We","11:10am","2:05pm","1997443771","D8252","CIS 9555","Fall","Wait
List",25,"Regular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("TuTh","11:10am","2:05pm","1733708699","C6253","STA 4000","Summer","Wait
List",75,"Regular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("Sa","7:50am","10:45am","1970551178","C6250","ACC9886","Spring","Closed",30,
"Regular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("Fr","2:30pm","5:25pm","1240328719","C6251","STA2000","Fall","Closed",35,"Reg
ular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("Sa","9:05am","10:20am","1775030773","A3152","CIS9660","Fall","Open",30,"Reg
ular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("We","11:10am","2:05pm","1738826610","D8250","MKT 3600","Fall","Wait
List",35,"Regular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("Th","7:50am","10:45am","1146203408","D8251","MKT4120","Fall","Open",30,"Re
gular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("Th","7:50am","10:45am","1698078080","D8253","CIS9660","Fall","Open",85,"Reg
ular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("TuTh","11:10am","2:05pm","1942648288","D8251","CIS3140","Summer","Open",
33,"Regular")

```

```

INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("Sa","2:30pm","5:25pm","1477000521","D8252","MKT9740","Fall","Open",30,"Reg
ular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("MoWe","5:50pm","7:10pm","1997443771","A3150","CIS3140","Fall","Open",25,"O
nline")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("Sa","2:30pm","5:25pm","1114667337","D8253","FIN
3610","Fall","Open",65,"Regular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("Fr","11:10am","2:05pm","1148529568","A3153","FIN3610","Winter","Closed",85,"
Regular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("TuTh","7:50am","10:45am","1114667337","C6252","FIN
3710","Spring","Closed",30,"Hybrid")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("Tu","2:30pm","5:25pm","1574751191","B7151","FIN3710","Summer","Open",25,"
Regular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("Mo","7:50am","10:45am","1131774660","B7150","FIN9782","Spring","Closed",35,"
Regular")
INSERT INTO SessionDetail
(Day_of_Week,Start_Time,End_Time,ProfessorID,ClassroomID,CourseID,Semester_term,S
tatus,Capacity,InstructionMod)
VALUES("Fr","5:50pm","7:10pm","1588901445","D8253","FIN9782","Winter","Closed",35,"O
nline")

```

Classroom

```

INSERT INTO Classroom(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('A3150', 'Regular', 50);
INSERT INTO Classroom(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('A3151', 'Regular', 50);

```

```

INSERT INTO Classroom(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('A3152', 'Lab', 30);
INSERT INTO Classroo(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('A3153', 'Regular', 100);
INSERT INTO Classroom(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('B7150', 'Regular', 50);
INSERT INTO Classroom(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('B7151', 'Regular', 50);
INSERT INTO Classroom(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('B7152', 'Lab', 30);
INSERT INTO Classroom(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('B7153', 'Regular', 100);
INSERT INTO Classroom(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('C6250', 'Regular', 50);
INSERT INTO Classroom(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('C6251', 'Regular', 50);
INSERT INTO Classroom(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('C6252', 'Lab', 30);
INSERT INTO Classroom(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('C6253', 'Regular', 100);
INSERT INTO Classroom (ClassroomID, Classroom_Type, MaxCapacity)
VALUES('D8250', 'Regular', 50);
INSERT INTO Classroom(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('D8251', 'Regular', 50);
INSERT INTO Classroom(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('D8252', 'Lab', 30);
INSERT INTO Classroom(ClassroomID, Classroom_Type, MaxCapacity)
VALUES('D8253', 'Regular', 100);

```

Transcript

```

INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1001', '3.3', '30');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1002', '3.6', '12');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1003', '3.5', '22');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1004', '3.7', '34');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1005', '3.1', '45');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1006', '2.8', '23');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1007', '2.3', '18');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1008', '3.3', '15');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)

```

```

VALUES('1009', '3.9', '14');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1010', '2.1', '29');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1011', '3.4', '50');
INSERT INTO Transcript (TranscriptID, GPA, TotalCredit)
VALUES('1012', '3.5', '33');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1013', '3.9', '28');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1014', '4.0', '36');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1015', '3.3', '33');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1016', '3.8', '37');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1017', '3.6', '35');
INSERT INTO Transcript (TranscriptID, GPA, TotalCredit)
VALUES('1018', '3.7', '34');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1019', '2.5', '25');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1020', '2.9', '28');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1021', '3.3', '40');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1022', '3.5', '44');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1023', '3.3', '39');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1024', '2.9', '44');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1025', '4.0', '28');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1026', '3.8', '46');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1027', '3.3', '38');
INSERT INTO Transcript (TranscriptID, GPA, TotalCredit)
VALUES('1028', '3.7', '46');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1029', '3.2', '36');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1030', '3.6', '9');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1031', '3.9', '15');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1032', '3.7', '20');

```

```

INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1033', '3.9', '18');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1034', '2.6', '40');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1035', '2.9', '38');
INSERT INTO Transcript (TranscriptID, GPA, TotalCredit)
VALUES('1036', '3.0', '36');
INSERT INTO Transcript (TranscriptID, GPA, TotalCredit)
VALUES('1037', '3.0', '35');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1038', '3.7', '28');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1039', '4.0', '22');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1040', '3.6', '12');
INSERT INTO Transcript (TranscriptID, GPA, TotalCredit)
VALUES('1041', '3.6', '9');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1042', '3.7', '12');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1043', '3.2', '18');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1044', '3.3', '26');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1045', '3.4', '38');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1046', '3.1', '39');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1047', '3.7', '40');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1048', '3.7', '38');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1049', '3.8', '47');
INSERT INTO Transcript (TranscriptID, GPA, TotalCredit)
VALUES('1050', '3.5', '45');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1051', '2.9', '50');
INSERT INTO Transcript(TranscriptID, GPA, TotalCredit)
VALUES('1052', '2.4', '38');
INSERT INTO Transcript (TranscriptID, GPA, TotalCredit)
VALUES('1053', '2.3', '46');

```

Professor

```

INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)

```



```

VALUES ('1997443771', 'Kelly', 'Barrera', 'fulltime', 90000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1698078080', 'Mari', 'Mclean', 'fulltime', 100000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1775030773', 'Carter', 'Jackson', 'fulltime', 130000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1148529568', 'Quin', 'Ballard', 'fulltime', 90000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1131774660', 'Steel', 'Mayer', 'fulltime', 125000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1574751191', 'Marsden', 'Hale', 'fulltime', 80000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1061046671', 'Stuart', 'Donaldson', 'fulltime', 85000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1904605352', 'Cadman', 'Pennington', 'parttime', 30000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1970551178', 'Farrah', 'Washington', 'parttime', 50000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1240328719', 'Yoko', 'Church', 'parttime', 30000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1735156751', 'Uriah', 'Mccall', 'fulltime', 92000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1733708699', 'Amity', 'Gordon', 'fulltime', 105000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1738826610', 'Julian', 'Parks', 'fulltime', 128000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1146203408', 'Kameko', 'Webb', 'fulltime', 150000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1477000521', 'Xanthus', 'Mckee', 'fulltime', 125000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1588901445', 'Jane', 'Beck', 'fulltime', 85000);
INSERT INTO professor

```

```

(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1114667337', 'Cody', 'Holder', 'fulltime', 85000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1314842598', 'Carl', 'Lawrence', 'parttime', 30000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1680104634', 'Sacha', 'Mcclure', 'parttime', 45000);
INSERT INTO professor
(professorid, professor_first_name, professor_last_name, professor_type, salary)
VALUES ('1942648288', 'Rhea', 'Lester', 'parttime', 30000);

```

Enrollment

```

INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("42012237", "8756", "1001");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("60189043", "9875", "1002");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("77942801", "8756", "1003");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("28505188", "8756", "1004");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("60417231", "8756", "1005");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("49436677", "9875", "1006");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("69122657", "7580", "1007");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("28076079", "7580", "1008");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("36935144", "4217", "1009");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("94600626", "4217", "1010");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("62225314", "4217", "1011");

```

```

INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("51327795", "7580", "1012");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("47158329", "3446", "1013");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("48484110", "8855", "1014");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("82718168", "3255", "1015");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("29176473", "2021", "1016");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("72773514", "3255", "1017");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("19137998", "2021", "1018");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("47158329", "3255", "1013");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("48484110", "2021", "1014");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("82718168", "3446", "1015");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("29176473", "8855", "1016");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("72773514", "3446", "1017");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("19137998", "8855", "1018");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("32322720", "7566", "1019");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("17604969", "7566", "1020");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)

```

```

VALUES("47615957", "7566", "1021");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("32310351", "7566", "1022");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("63926611", "7566", "1023");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("43705448", "6434", "1024");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("92973915", "6434", "1025");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("22216825", "7976", "1026");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("47200958", "6434", "1027");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("63744655", "7976", "1028");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("43705448", "7976", "1024");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("92973915", "7976", "1025");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("47559331", "1288", "1029");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("91028171", "9754", "1030");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("36030525", "1288", "1031");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("32387051", "9754", "1032");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("96727279", "2901", "1033");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("36030525", "9754", "1031");
INSERT INTO enrollment

```

```

(StudentID, SessionNo, TranscriptID)
VALUES("32387051", "2901", "1032");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("47355073", "8755", "1034");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("51156575", "9011", "1035");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("52905878", "5999", "1036");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("50492937", "8655", "1037");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("58316683", "8655", "1038");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("51156575", "8755", "1035");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("52905878", "8755", "1036");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("50492937", "9011", "1037");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("58316683", "5999", "1038");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("21819872", "7657", "1039");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("92723200", "0777", "1040");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("23883576", "7657", "1041");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("70920631", "0777", "1042");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("72061949", "0777", "1043");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("21819872", "0777", "1039");

```

```

INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("92723200", "7657", "1040");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("39311602", "5445", "1044");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("70423802", "5445", "1045");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("15234147", "5445", "1046");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("18610448", "5445", "1047");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("25548455", "5445", "1048");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("20430070", "988", "1049");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("53103408", "988", "1050");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("13272764", "0988", "1051");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("10903178", "988", "1052");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("87733041", "988", "1053");
INSERT INTO enrollment
(StudentID, SessionNo, TranscriptID)
VALUES("47200958", "7976", "1027");

```

VI. Database Application

- Navigation Form

The database application consists of a set of Forms, Reports and Queries that are linked together on a Navigation Form.

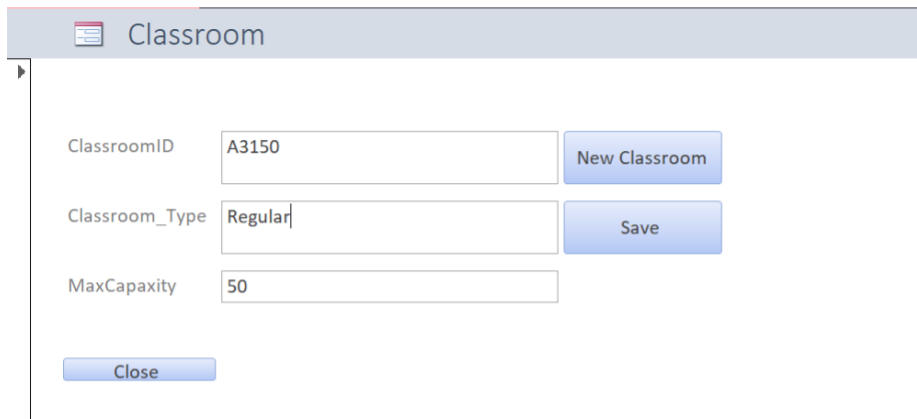
The Navigation Form is the first form that appears when the database is opened.

Different data entry forms and reports can be displayed by clicking on the selection on the left-hand side.

The screenshot displays a database application interface. At the top, a window titled "Navigation Form" contains a "Register System Menu" bar. Below this, a vertical menu on the left lists various options: Classroom, Course, CourseDetail, Enrollment, Professor, Session, SessionDetail, Student, Transcript (highlighted), Professor Detail, Student Schedule, and Course info. The main area of the application shows a "Transcript" form. This form includes three input fields: "TranscriptID" (empty), "GPA" (containing the value "3.3"), and "TotalCredit" (containing the value "30"). To the right of the "TranscriptID" field is a "New Enrollment" button. To the right of the "GPA" field is a "Save" button. Below the input fields is a "Close" button.

- Data Entry Forms

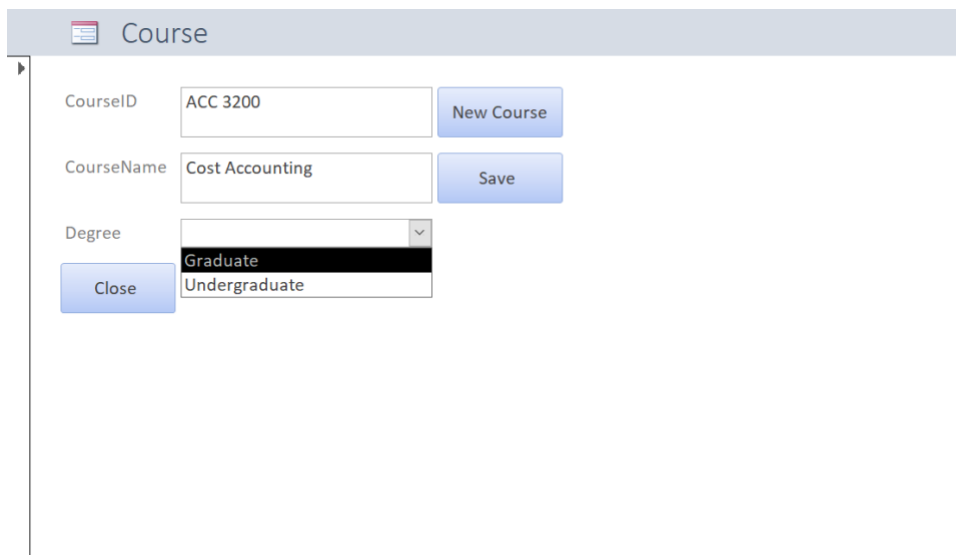
1. Classroom Data Entry Form



The Classroom Data Entry Form is a web-based interface for adding new classroom information. It features a title bar with a document icon and the word "Classroom". Below the title bar, there are three input fields: "ClassroomID" with the value "A3150", "Classroom_Type" with the value "Regular", and "MaxCapacity" with the value "50". To the right of the "ClassroomID" field is a blue button labeled "New Classroom". To the right of the "Classroom_Type" field is a blue button labeled "Save". Below the "MaxCapacity" field is a blue button labeled "Close".

The Classroom data entry form is to enter new classroom information. By entering information into those attributes and clicking on “Save”, a new classroom entry can be added.

2. Course Data Entry Form



The Course Data Entry Form is a web-based interface for adding new course information. It features a title bar with a document icon and the word "Course". Below the title bar, there are three input fields: "CourseID" with the value "ACC 3200", "CourseName" with the value "Cost Accounting", and "Degree" with a dropdown menu showing "Graduate" and "Undergraduate". To the right of the "CourseID" field is a blue button labeled "New Course". To the right of the "CourseName" field is a blue button labeled "Save". Below the "Degree" field is a blue button labeled "Close".

The Course data entry form is to query, update and add course instances.

3. CourseDetail Data Entry Form

CourseDetail

Coursell

CIS 3140

Units

3

Course Name

Programming for Analytics

New Course

Degree

Graduate

Save

Subject_Information

Undergraduate
that can support business analytics. The course
introduces students to programming (using a language

Prerequisite

Prerequisites: CIS 2300 or Equivalent

Major_require

CIS/CS/CE

Department

Computer Information Systems

Close

The CourseDetail Data Entry Form is to look up details of currently existing courses in the system, and at the same time users can also add or update the form.

4. Enrollment Data Entry Form

Enrollment

StudentID

10903178

New Enrollment

SessionNo

0988

Save

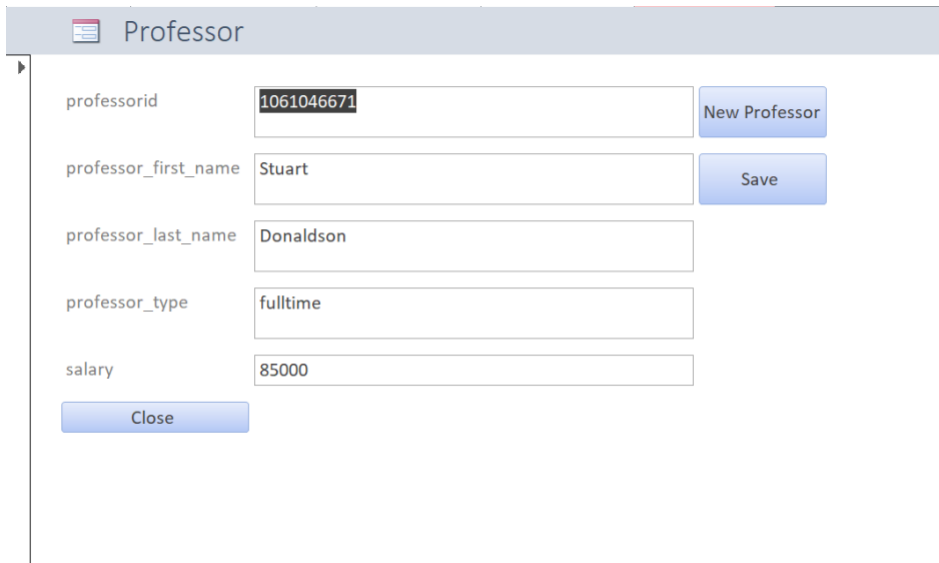
TranscriptID

1052

Close

If there's any new students enrolled, users can use this data entry form to add new student's information including Student ID, Session No. and Transcript ID.

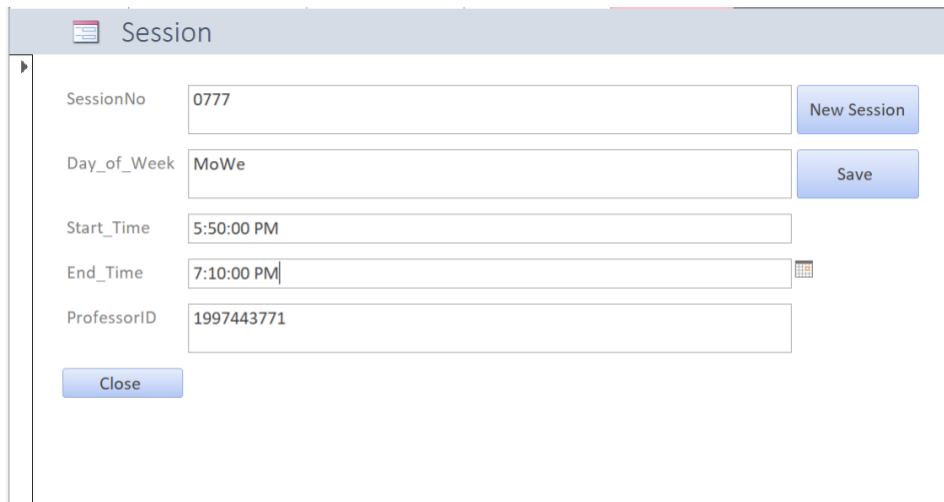
5. Professor Data Entry Form



The screenshot shows a web form titled "Professor" with a header bar containing a calendar icon and the title. The form contains five input fields: "professorid" with the value "1061046671", "professor_first_name" with "Stuart", "professor_last_name" with "Donaldson", "professor_type" with "fulltime", and "salary" with "85000". To the right of the "professorid" field is a "New Professor" button. To the right of the "professor_first_name" field is a "Save" button. At the bottom left of the form is a "Close" button.

Professor Data Entry Form is to query, update and add Professor records.

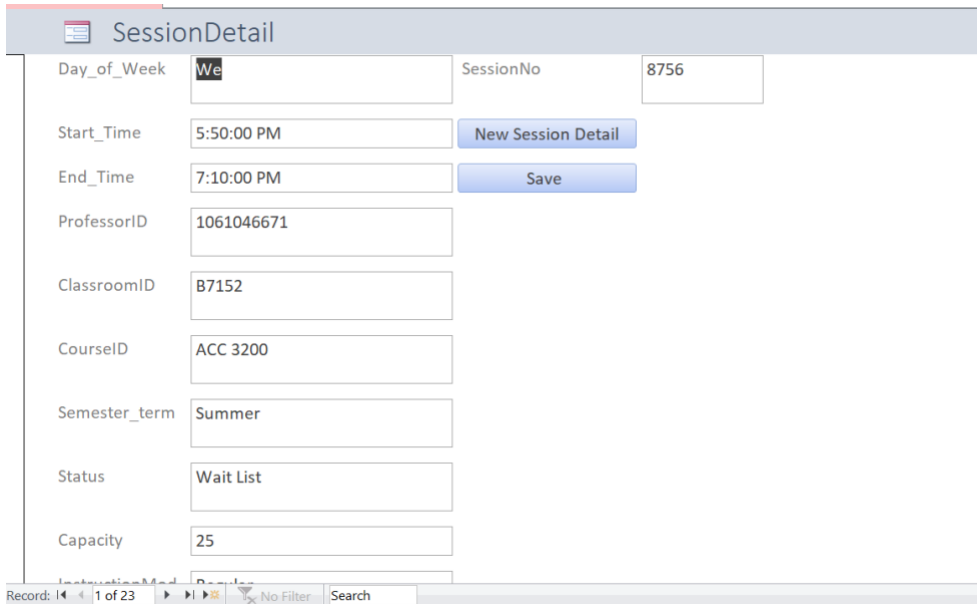
6. Session Data Entry Form



The screenshot shows a web form titled "Session" with a header bar containing a calendar icon and the title. The form contains five input fields: "SessionNo" with "0777", "Day_of_Week" with "MoWe", "Start_Time" with "5:50:00 PM", "End_Time" with "7:10:00 PM", and "ProfessorID" with "1997443771". To the right of the "SessionNo" field is a "New Session" button. To the right of the "Day_of_Week" field is a "Save" button. At the bottom left of the form is a "Close" button.

Session Data Entry Form is for data entry in the Session table

7. SessionDetail Data Entry Form



The screenshot shows the 'SessionDetail' form with the following fields and values:

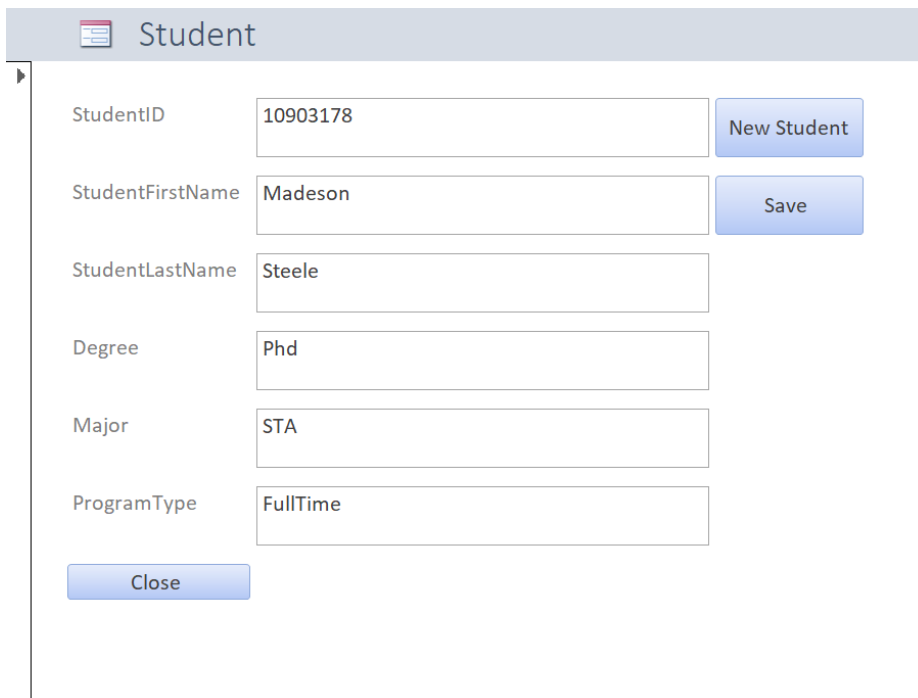
Field	Value
Day_of_Week	We
SessionNo	8756
Start_Time	5:50:00 PM
End_Time	7:10:00 PM
ProfessorID	1061046671
ClassroomID	B7152
CourseID	ACC 3200
Semester_term	Summer
Status	Wait List
Capacity	25

Buttons: 'New Session Detail' and 'Save'.

Footer: Record: 1 of 23, No Filter, Search.

This form is to look up and enter details for certain sessions.

8. Student Data Entry Form



The screenshot shows the 'Student' form with the following fields and values:

Field	Value
StudentID	10903178
StudentFirstName	Madeson
StudentLastName	Steele
Degree	Phd
Major	STA
ProgramType	FullTime

Buttons: 'New Student', 'Save', and 'Close'.

If a new student is enrolled, the Student table would be updated with new student information. Users can update student information by this form.

9. Transcript Data Entry Form

Transcript

TranscriptID

1001

New Transcript

GPA

3.3

Save

TotalCredit

30

Close

This Transcript Data Entry Form is to query, update or add new grade records to the Transcript table.

10. Student Schedule Form

Student Schedule

StudentID

47200958

GPA

3.3

StudentFirstName

Jaden

StudentLastName

Roth

Degree

undergraduate

Major

STA

	Day_of_Week	Start_Tim	End_Time	professor_	professor_last_nam	Classrooml
	Mo	7:50:00 AM	10:45:00 AM	Steel	Mayer	B7150
	Tu	2:30:00 PM	5:25:00 PM	Marsden	Hale	B7151
	*					

By entering the information of certain students, this form can link to relevant tables and present the student schedule information including class start time and professor name.

11. Professor Detail Form

very handy as users don't need to generate and aggregate tables together but attendance sheets can be pulled out directly.

13. Course Info Form

Course info

CourselD

ACC 3200

CourseName

Cost Accounting

Degree

Undergraduate

Day_of_Week	Start_Time	End_Time	ProfessorID	ClassroomID	Semester_term	Status	Capacit	Instructio
We	5:50:00 PM	7:10:00 PM	1061046671	B7152	Summer	Wait List	25 Regular	
Sa	2:30:00 PM	5:25:00 PM	1477000521	D8252	Fall	Open	30 Regular	

Course Info Form is the Course table collaborated with CourseDetail table, Session table and others. By the input from the Course table, the details of certain courses will be generated directly.

- Reports

The core functions of our system are course selection and session allocation. We derived three reports that summarized these functions.

1. Student Information Summary Report

List out the basic information of all students in all programs, degrees and majors and summarize their academic history.

Student Information Summary								Thursday, May 14, 2020	
								5:57:05 PM	
StudentID	StudentFirstName	StudentLastName	Degree	Major	ProgramType	Number_Of_Courses	Cummulative_GPA	TotalCreditObtained	
32387051	Geraldine	Herrera	graduate	ACC	FullTime	2	3.7	20	
36030525	Olympia	Melton	graduate	ACC	FullTime	2	3.9	15	
47559331	Brennan	Holden	graduate	ACC	FullTime	1	3.2	36	
91028171	Ross	Dominguez	graduate	ACC	FullTime	1	3.6	9	
96727279	Abra	Madden	graduate	ACC	FullTime	1	3.9	18	
47355073	Hamish	Richardson	graduate	CIS	FullTime	1	2.6	40	
50492937	Keely	Cole	graduate	CIS	FullTime	2	3	35	
51156575	Gary	Gaines	graduate	CIS	FullTime	2	2.9	38	
52905878	Quinn	Case	graduate	CIS	FullTime	2	3	36	
58316683	Carissa	Cameron	graduate	CIS	FullTime	2	3.7	28	
21819872	Dolan	Cantu	graduate	FIN	FullTime	2	4	22	
23883576	Iris	Randolph	graduate	FIN	FullTime	1	3.6	9	
70920631	Zephr	Bridges	graduate	FIN	FullTime	1	3.7	12	
72061949	Regina	Mann	graduate	FIN	FullTime	1	3.2	18	
92723200	Chester	Alvarado	graduate	FIN	FullTime	2	3.6	12	
15234147	Melissa	Lynn	graduate	MKT	PartTime	1	3.1	39	

Based on the query:

```
SELECT s.*, Number_Of_Courses, Cummulative_GPA, TotalCreditObtained
FROM Student AS s LEFT JOIN (SELECT StudentID, COUNT(*) AS Number_Of_Courses,
AVG(GPA) AS Cummulative_GPA, AVG(TotalCredit) AS TotalCreditObtained FROM Enrollment AS
e LEFT JOIN Transcript AS t ON e.TranscriptID = t.TranscriptID GROUP BY StudentID) AS g ON
s.StudentID = g.StudentID
ORDER BY s.StudentID;
```

2. Professor Information Summary Report

List out the basic information of all professors and summarize their teaching activities.

Professor Information Summary						Thursday, May 14, 2020	
						5:57:23 PM	
professorid	professor_first_name	professor_last_name	professor_type	salary	umber_of_Courses		
1061046671	Stuart	Donaldson	fulltime	85000	1		
1114667337	Cody	Holder	fulltime	85000	2		
1131774660	Steel	Mayer	fulltime	125000	1		
1146203408	Kameko	Webb	fulltime	150000	1		
1148529568	Quin	Ballard	fulltime	90000	1		
1477000521	Xanthus	Mckee	fulltime	125000	1		
1574751191	Marsden	Hale	fulltime	80000	1		
1588901445	Jane	Beck	fulltime	85000	1		
1698078080	Mari	Mclean	fulltime	100000	2		
1733708699	Amity	Gordon	fulltime	105000	1		
1735156751	Uriah	Mccall	fulltime	92000	1		
1738826610	Julian	Parks	fulltime	128000	1		

Based on the query:

```
SELECT p.*, a.Number_of_Courses
FROM Professor AS p LEFT JOIN (SELECT ProfessorID, COUNT(SessionNo) AS
Number_of_Courses FROM [Session] AS s GROUP BY ProfessorID) AS a ON p.ProfessorID =
a.ProfessorID
ORDER BY p.ProfessorID;
```

3. Session Summary

List out the basic information of class sessions and summarize their current enrollment numbers.

Session Summary											
						Thursday, May 14, 2020					
						5:57:39 PM					
SessionNo	Day_of_Week	Start_Time	End_Time	ProfessorID	ClassroomID	CourseID	Semester_t erm	Status	Capacity	InstructionMod	Number_of_ Students
8756	We	5:50:00 PM	7:10:00 PM	1061046671	B7152	ACC 3200	Summer	Wait List	25	Regular	4
9875	MoWe	9:05:00 AM	10:20:00 AM	1735156751	C6252	ACC 9818	Spring	Closed	25	Regular	2
1288	Fr	2:30:00 PM	5:25:00 PM	1314842598	C6253	ACC 9886	Fall	Wait List	45	Online	2
9754	Th	7:30:00 PM	8:45:00 PM	1904605352	B7153	CIS 3140	Winter	Closed	50	Regular	3
2901	MoWe	9:05:00 AM	10:20:00 AM	1680104634	D8250	CIS 9555	Fall	Open	30	Hybrid	2
8755	TuTh	7:30:00 PM	8:45:00 PM	1698078080	A3151	CIS 9660	Spring	Closed	35	Hybrid	3
9011	We	11:10:00 AM	2:05:00 PM	1997443771	D8252	FIN 3610	Fall	Wait List	25	Regular	2
5999	TuTh	11:10:00 AM	2:05:00 PM	1733708699	C6253	FIN 3710	Summer	Wait List	75	Regular	2
8655	Sa	7:50:00 AM	10:45:00 AM	1970551178	C6250	FIN 9782	Spring	Closed	30	Regular	2

Based on the query:

```
SELECT sa.*, Number_of_Students
FROM (SELECT s.SessionNo, sd.* FROM [Session] AS s LEFT JOIN SessionDetail AS sd
ON (s.ProfessorID = sd.ProfessorID) AND (s.End_Time = sd.End_Time) AND (s.Start_Time =
sd.Start_Time) AND (s.Day_of_Week = sd.Day_of_Week)) AS sa LEFT JOIN (SELECT SessionNo,
COUNT(*) AS Number_of_Students FROM Enrollment GROUP BY SessionNo) AS sn ON
sa.SessionNo = sn.SessionNo
ORDER BY sa.SessionNo;
```

● Scenarios

1. The university wants to know which teacher is the most and least popular among students in order to take this into consideration the raise range in their salary.

From the query result, we know the most popular professor is Cody Holder with 10 students. Carl Lawrence, Sacha McClure, Amity Gordon, Uriah McCall, and Farrah Washington are the least popular professors with only 2 students.

professorid	professor_first_name	professor_last_name	total_students
1114667337	Cody	Holder	10
1698078080	Mari	McClean	6
1997443771	Kelly	Barrera	6
1131774660	Steel	Mayer	5
1588901445	Jane	Beck	5
1061046671	Stuart	Donaldson	4
1146203408	Kameko	Webb	3
1240328719	Yoko	Church	3
1477000521	Xanthus	McKee	3
1574751191	Marsden	Hale	3
1738826610	Julian	Parks	3
1775030773	Carter	Jackson	3
1904605352	Cadman	Pennington	3
1942648288	Rhea	Lester	3
1314842598	Carl	Lawrence	2
1680104634	Sacha	McClure	2
1733708699	Amity	Gordon	2
1735156751	Uriah	McCall	2
1970551178	Farrah	Washington	2

Based on the following query:

```
SELECT DISTINCT p.professorid, p.professor_first_name, p.professor_last_name,
A.total_students
FROM professor p, SessionDetail sd,
(SELECT s.professorid, COUNT(e.StudentID) AS total_students
FROM Enrollment e, session s
WHERE s.SessionNo = e.SessionNo
GROUP BY s.professorID) A
WHERE sd.professorid = s.professorid
AND sd.professorid = p.professorid
ORDER BY A.total_students DESC;
```

2. Some professors do not have lots of courses so they complain that if the university takes total student numbers into consideration, it's not fair to them. Therefore, the university also wants to know the average student numbers of all sessions.

In the last scenario, we know Cody Holder has the most students. However, if we look at the average students per session, Jane Beck, Cody Holder, and Steel Mayer have the same amount of average students.

professorid	professor_first_name	professor_last_name	total_students	total_sessions	Average_Students
1588901445	Jane	Beck	5	1	5
1114667337	Cody	Holder	10	2	5
1131774660	Steel	Mayer	5	1	5
1061046671	Stuart	Donaldson	4	1	4
1146203408	Kameko	Webb	3	1	3
1240328719	Yoko	Church	3	1	3
1477000521	Xanthus	McKee	3	1	3
1574751191	Marsden	Hale	3	1	3
1698078080	Mari	McLean	6	2	3
1997443771	Kelly	Barrera	6	2	3
1738826610	Julian	Parks	3	1	3
1775030773	Carter	Jackson	3	1	3
1904605352	Cadman	Pennington	3	1	3
1942648288	Rhea	Lester	3	1	3
1733708699	Amity	Gordon	2	1	2
1735156751	Uriah	McCall	2	1	2
1314842598	Carl	Lawrence	2	1	2
1970551178	Farrah	Washington	2	1	2
1680104634	Sacha	McClure	2	1	2

Based on the following query:

```

SELECT DISTINCT p.professorid, p.professor_first_name, p.professor_last_name,
A.total_students, B.total_sessions, (A.total_students/B.total_sessions) AS
Average_Students
FROM professor p, SessionDetail sd,
(SELECT s.professorid, COUNT(e.StudentID) AS total_students
FROM Enrollment e, session s
WHERE s.SessionNo = e.SessionNo
GROUP BY s.professorID) A,
(SELECT professorid, COUNT(professorID) AS total_sessions
FROM Session
GROUP BY professorid) B
WHERE sd.professorid = s.professorid
AND sd.professorid = p.professorid
AND A.professorid = B.professorid
ORDER BY (A.total_students/B.total_sessions) DESC;

```

3. The professors spend lots of time on teaching. Do those professors with more students have more salary as well?

From our analysis, we can know salary has no relationship to total students or average students.

professorid	professor_first_name	professor_last_name	total_students	total_sessions	Average_Students	salary
1588901445	Jane	Beck	5	1	5	85000
1114667337	Cody	Holder	10	2	5	85000
1131774660	Steel	Mayer	5	1	5	125000
1061046671	Stuart	Donaldson	4	1	4	85000
1146203408	Kameko	Webb	3	1	3	150000
1240328719	Yoko	Church	3	1	3	30000
1477000521	Xanthus	McKee	3	1	3	125000
1574751191	Marsden	Hale	3	1	3	80000
1698078080	Mari	McClean	6	2	3	100000
1997443771	Kelly	Barrera	6	2	3	90000
1738826610	Julian	Parks	3	1	3	128000
1775030773	Carter	Jackson	3	1	3	130000
1904605352	Cadman	Pennington	3	1	3	30000
1942648288	Rhea	Lester	3	1	3	30000
1733708699	Amity	Gordon	2	1	2	105000
1735156751	Uriah	McCall	2	1	2	92000
1314842598	Carl	Lawrence	2	1	2	30000
1970551178	Farrah	Washington	2	1	2	50000
1680104634	Sacha	McClure	2	1	2	45000

Based on the following query:

```

SELECT DISTINCT p.professorid, p.professor_first_name, p.professor_last_name,
               A.total_students, B.total_sessions, (A.total_students/B.total_sessions) AS
               Average_Students, p.salary
FROM professor p, SessionDetail sd,
(SELECT s.professorid, COUNT(e.StudentID) AS total_students
FROM Enrollment e, session s
WHERE s.SessionNo = e.SessionNo
GROUP BY s.professorID) A,
(SELECT professorid, COUNT(sessionID) AS total_sessions
FROM Session
GROUP BY professorid) B
WHERE sd.professorid = s.professorid
AND sd.professorid = p.professorid
AND A.professorid = B.professorid
ORDER BY (A.total_students/B.total_sessions) DESC;

```

4. The university wants to know the average GPA performance to understand the study situation of students.

The Average_GPA is 3.35, which is above expectation of the university.

Average_GPA
3.35

```

SELECT ROUND(AVG(GPA),2) AS Average_GPA FROM Transcript;

```

5. The university wants to know students in which degree has a higher GPA.

From the analysis, we can know the graduate has highest average GPA. On the other hand, PHD students has lowest GPA.

Probably the coursework in PHD is too difficult, so students cannot get very high GPA. The undergraduate students may have more club events, so their average GPA is lower than the GPA of graduate students.

Degree	Average_GPA
graduate	3.45
undergraduate	3.42
Phd	2.98

```
SELECT s.Degree, ROUND(AVG(GPA),2) AS Average_GPA
FROM Transcript t, Enrollment e, Student s
WHERE t.Transcriptid = e.Transcriptid
AND e.StudentID = s.StudentID
GROUP BY s.Degree
ORDER BY AVG(GPA) DESC;
```

6. Do students in different majors have lots of differences in their GPA performance?

From this analysis, to the university's surprise, the highest average is the undergraduate students in the Financial department.

Also, graduate students in the Financial department also rank 3 in average GPA.

Degree	Major	Average_GPA
undergraduate	FIN	3.72
graduate	ACC	3.7
graduate	FIN	3.67
undergraduate	STA	3.49
graduate	MKT	3.44
undergraduate	ACC	3.33
undergraduate	MKT	3.1
graduate	CIS	3.09
undergraduate	CIS	3.08
Phd	STA	2.98

Based on the following query:

```
SELECT s.Degree, s.Major, ROUND(AVG(GPA),2) AS Average_GPA
FROM Transcript t, Enrollment e, Student s
WHERE t.Transcriptid = e.Transcriptid
AND e.StudentID = s.StudentID
GROUP BY s.Degree, s.Major
ORDER BY AVG(GPA) DESC;
```

7. An alumnus wants to sponsor scholarship to students with the highest GPA in each department and degree. Therefore, the university needs to generate the student lists and give them scholarships.

StudentID	Degree	Major	StudentFirstName	StudentLastName	GPA
96727279	graduate	ACC	Abra	Madden	3.9
36030525	graduate	ACC	Olympia	Melton	3.9
58316683	graduate	CIS	Carissa	Cameron	3.7
21819872	graduate	FIN	Dolan	Cantu	4
25548455	graduate	MKT	Quintessa	Lindsey	3.7
18610448	graduate	MKT	Hadassah	Norton	3.7
20430070	Phd	STA	Paul	Welch	3.8
28505188	undergraduate	ACC	Giselle	Dennis	3.7
36935144	undergraduate	CIS	Rigel	Delacruz	3.9
48484110	undergraduate	FIN	Moses	Beach	4
32310351	undergraduate	MKT	Wyoming	Goodman	3.5
92973915	undergraduate	STA	Knox	Stein	4

Based on the following query:

```

SELECT M.StudentID, M.Degree, M.Major, M.StudentFirstName, M.StudentLastName,
M.GPA
FROM (SELECT DISTINCT g.StudentID, s.Degree, s.Major, s.StudentFirstName,
s.StudentLastName, g.GPA
FROM student s,
(SELECT DISTINCT e.StudentID, t.GPA
FROM enrollment e
INNER JOIN Transcript t
ON e.TranscriptID = t.TranscriptID) g
WHERE g.StudentID = s.StudentID
ORDER BY s.Degree, s.Major, g.GPA DESC) M
INNER JOIN
(SELECT A.DEGREE, A.MAJOR, Max(A.GPA) AS MaxOfGPA
FROM (SELECT DISTINCT g.StudentID, s.Degree, s.Major, s.StudentFirstName,
s.StudentLastName, g.GPA
FROM student s,
(SELECT DISTINCT e.StudentID, t.GPA
FROM enrollment e
INNER JOIN Transcript t
ON e.TranscriptID = t.TranscriptID) g
WHERE g.StudentID = s.StudentID
ORDER BY s.Degree, s.Major, g.GPA DESC) AS A
GROUP BY A.DEGREE, A.MAJOR) N
ON M.DEGREE=N.DEGREE AND M.MAJOR=N.MAJOR AND M.GPA=N.MaxOfGPA
ORDER BY M.Degree, M.Major;

```

VII. Conclusions

Our project of developing a database system follows the system development procedures. We started from detecting the problem and information needs, figuring out the entity names and attributes information. We recognized that building the Entity Relationship Model is the most important part of the whole project, because all of the following steps and designs are based on this model. Our Entity Relationship Model was first built naturally from the system description and problems we want to solve, and was later adjusted when we were creating the database schema to make it more logically reasonable. All the assignments and adjustments are completed by group discussion. Duties are clearly distributed, and every group member is fully contributed.

Our Student Registration System can be broadly applied to different scenarios as discussed in the above part. We realized the function of our database system through the steps of converting to relational models, normalization, creating database schema, and database application. It can help relieve the burden of data management for the school registration office.