DATABASE BUILD | PROJECT DOCUMENTATION

ACCESS & MYSQL WORKBENCH ENVIRONMENTS

MELISSA LAMELZA | FULL-STACK DEVELOPER

TECHNICAL COLLEGE DATABASE MANAGEMENT SYSTEM

PROJECT DESCRIPTION & REQUIREMENTS

The South Carolina Technical College System needs an updated database management system. I will be implementing a robust RDBMS, both in Access and MySQL Workbench, which will house all of the necessary information for each school. Initially, I will create a profile for 11 of the total 16 South Carolina Technical Schools located throughout the Palmetto State. This will give users the option to enter more data as the database continues to grow. The database will house information for all users involved within the school system, including students, academic professionals such as professors and departmental personnel, and general staff employees as well. Eventually, this database will evolve into a front-end site (website) the users will have access to, but for right now I will be concentrating on the main database framework. The minimum requirements for the RDBMS will include the following:

- Each South Carolina Technical School will house user data including first name, last name, address, phone number, email, start date (acceptance date, hire date), user title (student, instructor, staff member), department association (academic or non-academic), status (active, inactive), and campus affiliation depending on the technical school chosen.
- Every user will be assigned a department within the system that will emulate their position at the school. Students will be assigned a department based on their major of choice. Professors, instructors, and department heads will also be assigned to an academic department based on their job title, and which courses they teach. Staff members and administrators will be assigned a non-academic department based on their job title and description as well.
- Each user will be assigned a title upon entry into the database system, which will include both academic and non-academic titles, depending on each user's role at the college.
- With each title, there will be a level of permission assigned to each user as well. This will determine their administrative privileges to view and modify data. For example, a student will have the ability to change his/her own personal data upon logging into the system, and have the ability to register for classes if their academic requirements are met.
- Each object contained within the database will also have a status assigned. For example, a user in the system can be made active or inactive depending on their employment and/or degree status. Classes and courses will also have status requirements assigned (such as open or closed), depending on whether or not one may be offered during a particular school term or not.
- There will be courses assigned to each academic department, along with the classes associated with each course. These can be viewed and chosen during registration as well.

- Each class will be assigned a reference number, a unique course identifier including course name and number, a specific instructor, course section, date and time offered, day of the week, building and room number, and any specific attributes indicating term offerings and availability.
- Each college will be associated with at least one campus location. Most of the technical schools, however, have multiple campus locations, which will all be included.
- The database will house data regarding the campus buildings which will be utilized, especially when displaying information for classes offered.
- There will be a separate table for days of the week, academic terms the school year is based on, course sections, section type (online or on-campus instruction, for example), and department type (degree-based or staff-related) as well.
- These tables will have relationships assigned to each other depending on the data contained within each one. The column information and data type requirements will be shown below.
- I will create 16 tables to begin with, which will house the minimum amount of data each one requires in order to establish strong relationships with each other. I will also create at least five queries (views) that will be utilized in viewing important data within the school system.
- The primary key (PK) fields are auto-numbered, and the foreign key (FK) fields are displayed.

TABLE STRUCTURE & COLUMN DATA TYPE SPECIFICATIONS

BUILDING: HOLDS DATA IDENTIFYING CAMPUS BUILDINGS

FIELD NAME	DATA TYPE	FIELD SIZE	REQUIRED?
dbBuildingID (PK)	AutoNumber	Long Integer	
dbBuildingName	Short Text	10	

CAMPUS: HOLDS DATA IDENTIFYING EACH CAMPUS LOCATION

FIELD NAME	DATA TYPE	FIELD SIZE	REQUIRED?
dbCampusID (PK)	AutoNumber	Long Integer	
dbCampusName	Short Text	40	
dbLocationID (FK)	Number	Long Integer	*(no default value)*
dbCampusAbbreviation	Short Text	10	

CLASS: HOLDS DATA FOR INDIVIDUAL CLASSES OFFERED WITHIN EACH COURSE

FIELD NAME	DATA TYPE	FIELD SIZE	REQUIRED?
dbClassID (PK)	AutoNumber	Long Integer	
dbStatusID (FK)	Number	Long Integer	*(no default value)*
dbClassReferenceNumber	Short Text	10	
dbDepartmentID (FK)	Number	Long Integer	*(no default value)*
dbCourseID (FK)	Number	Long Integer	*(no default value)*
dbSectionID (FK)	Number	Long Integer	*(no default value)*
dbCampusID (FK)	Number	Long Integer	*(no default value)*

CLASS (CONT'D)

FIELD NAME	**DATA TYPE**	**FIELD SIZE**	**REQUIRED?**
dbDayID (FK)	Number	Long Integer	*(no default value)*
dbClassTimeStart	Date/Time	Medium Time	*(no show date picker)*
dbClassTimeEnd	Date/Time	Medium Time	*(no show date picker)*
dbUserID (FK)	Number	Long Integer	*(no default value)*
dbTermID (FK)	Number	Long Integer	*(no default value)*
dbBuildingID (FK)	Number	Long Integer	*(no default value)*
dbRoomNumber	Short Text	10	
dbClassAttributeID (FK)	Number	Long Integer	*(no default value)*

CLASS_ATTRIBUTE: HOLDS DATA IDENTIFYING CLASS SPECIFICIFICATIONS

FIELD NAME	**DATA TYPE**	**FIELD SIZE**	**REQUIRED?**
dbClassAttributeID (PK)	AutoNumber	Long Integer	
dbClassAttributeName	Short Text	30	

COURSE: HOLDS DATA ABOUT EACH COURSE AVAILABLE WITHIN THE DEPARTMENTS

FIELD NAME	**DATA TYPE**	**FIELD SIZE**	**REQUIRED?**
dbCourseID (PK)	AutoNumber	Long Integer	
dbCourseName	Short Text	100	
dbCourseNumber	Short Text	10	
dbDepartmentID (FK)	Number	Long Integer	*(no default value)*
dbCourseCredit	Number	Long Integer	*(no default value)*

DAY: HOLDS DATA IDENTIFYING DAYS WHEN CLASSES ARE HELD

FIELD NAME	**DATA TYPE**	**FIELD SIZE**	**REQUIRED?**
dbDayID (PK)	AutoNumber	Long Integer	
dbDayName	Short Text	75	
dbDayAbbreviation	Short Text	10	

DEPARTMENT: HOLDS DATA IDENTIFYING EACH SCHOOL'S DEPARTMENTS

FIELD NAME	**DATA TYPE**	**FIELD SIZE**	**REQUIRED?**
dbDepartmentID (PK)	AutoNumber	Long Integer	
dbDepartmentName	Short Text	50	
dbDepartmentTypeID (FK)	Number	Long Integer	*(no default value)*
dbDepartmentAbbreviation	Short Text	6	

DEPARTMENT_TYPE: HOLDS DATA IDENTIFYING DEPARTMENT SPECIFICATIONS

FIELD NAME	DATA TYPE	FIELD SIZE	REQUIRED?
dbDepartmentTypeID (PK)	AutoNumber	Long Integer	
dbDepartmentTypeName	Short Text	30	

LOCATION: HOLDS DATA IDENTIFYING EACH SCHOOL LOCATION

FIELD NAME	DATA TYPE	FIELD SIZE	REQUIRED?
dbLocationID (PK)	AutoNumber	Long Integer	
dbLocationName	Short Text	40	

PERMISSION: HOLDS DATA IDENTIFYING EACH TYPE OF PERMISSION LEVEL

FIELD NAME	DATA TYPE	FIELD SIZE	REQUIRED?
dbPermissionID (PK)	AutoNumber	Long Integer	
dbPermissionName	Short Text	20	
dbPermissionDescription	Short Text	50	

SECTION: HOLDS DATA IDENTIFYING EACH COURSE SECTION

FIELD NAME	DATA TYPE	FIELD SIZE	REQUIRED?
dbSectionID (PK)	AutoNumber	Long Integer	
dbSectionName	Short Text	20	
dbSectionTypeID (FK)	Number	Long Integer	*(no default value)*

SECTION_TYPE: HOLDS DATA IDENTIFYING SECTION SPECIFICATIONS

FIELD NAME	DATA TYPE	FIELD SIZE	REQUIRED?
dbSectionTypeID (PK)	AutoNumber	Long Integer	
dbSectionTypeName	Short Text	30	
dbSectionTypeAbbreviation	Short Text	10	

STATUS: HOLDS DATA IDENTIFYING EACH STATUS TYPE

FIELD NAME	DATA TYPE	FIELD SIZE	REQUIRED?
dbStatusID (PK)	AutoNumber	Long Integer	
dbStatusName	Short Text	20	
dbStatusAbbreviation	Short Text	7	

TERM: HOLDS DATA IDENTIFYING EACH ACADEMIC SCHOOL TERM

FIELD NAME	DATA TYPE	FIELD SIZE	REQUIRED?
dbTermID (PK)	AutoNumber	Long Integer	
dbTermName	Short Text	50	
dbClassTimeStart	Date/Time	Medium Time	*(no show date picker)*
dbClassTimeEnd	Date/Time	Medium Time	*(no show date picker)*

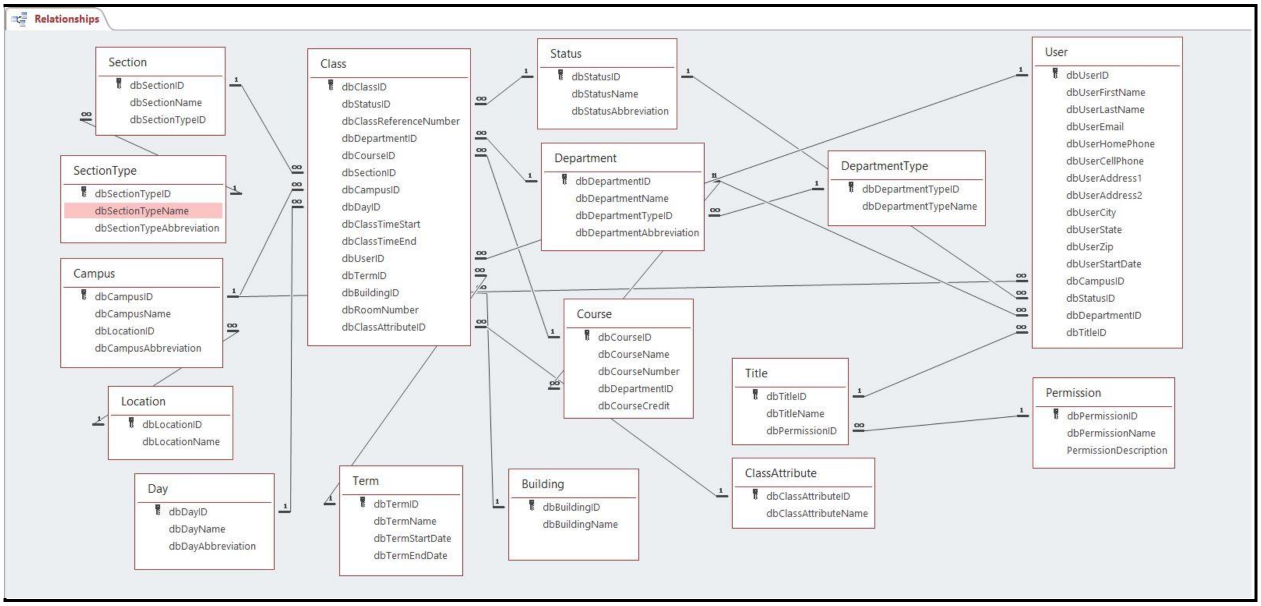
TITLE: HOLDS DATA IDENTIFYING EACH USER TITLE

FIELD NAME	DATA TYPE	FIELD SIZE	REQUIRED?
dbTitleID (PK)	AutoNumber	Long Integer	
dbTitleName	Short Text	30	
dbPermissionID (FK)	Number	Long Integer	*(no default value)*

USER: HOLDS DATA IDENTIFYING USER SPECIFICATIONS

FIELD NAME	DATA TYPE	FIELD SIZE	REQUIRED?
dbUserID (PK)	AutoNumber	Long Integer	
dbUserFirstName	Short Text	50	Yes
dbUserLastName	Short Text	50	Yes
dbUserEmail	Short Text	30	
dbUserHomePhone	Short Text	13	
dbUserCellPhone	Short Text	13	
dbUserAddress1	Short Text	40	
dbUserAddress2	Short Text	40	
dbUserCity	Short Text	20	
dbUserState	Short Text	2	
dbUserZip	Short Text	10	
dbUserStartDate	Date/Time		
dbCampusID (FK)	Number	Long Integer	*(no default value)*
dbStatusID (FK)	Number	Long Integer	*(no default value)*
dbDepartmentID (FK)	Number	Long Integer	*(no default value)*
dbTitleID (FK)	Number	Long Integer	*(no default value)*

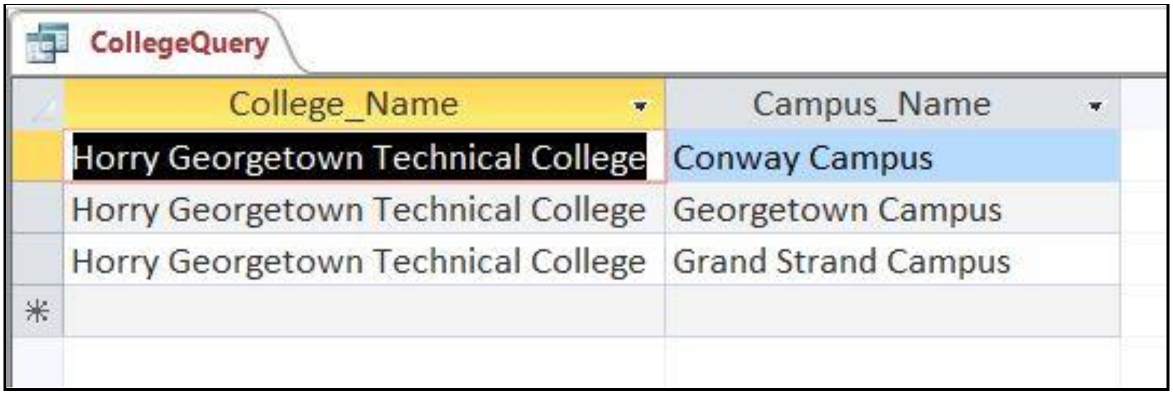
ERD: ESTABLISHING RELATIONSHIPS BETWEEN EACH DATABASE TABLE



- Notice the one-to-many relationships shown above between the tables. I utilized the same name for each one of the foreign keys that relate to primary keys within their associated tables.
- Notice that the *Class* table is primarily made up of a number of foreign key fields that simply pull data from all other associated tables. Similar structure can be seen within the *User* table as well.
- In order to reduce data duplication, avoid data abnormalities, and keep the integrity of the data while ensuring referential integrity, I divided the data in such a way using a number of tables that may hold only one or two fields. This simplifies data management, while creating strong relationships between the tables, making queries and data modification more viable.

ACCESS QUERIES AND SQL CODE

COLLEGE_QUERY: DISPLAYS ALL CAMPUS LOCATIONS FOR A SPECIFIC COLLEGE


CollegeQuery

College_Name	Campus_Name
Horry Georgetown Technical College	Conway Campus
Horry Georgetown Technical College	Georgetown Campus
Horry Georgetown Technical College	Grand Strand Campus

CollegeQuery view created *(see above image)*

- Notice the query is fairly simple, but a realistic and functional one. It gives the user the ability to view data regarding all of the campus locations associated with a specific school. In this case, I chose the school with a primary key *(dbLocationID)* value of 9. If you take a look at the SQL code below, it is identical to the SQL statements we have been utilizing in MySQL Workbench.
- The SELECT statement chooses the data from the *Campus* and *Location* tables, creating a JOIN with the key field *(dbLocationID)* they have in common. Notice that I also created aliases for the column names as the data was output as well. Finally, I sorted the results first by the location name, and then by the campus name associated with the particular school.

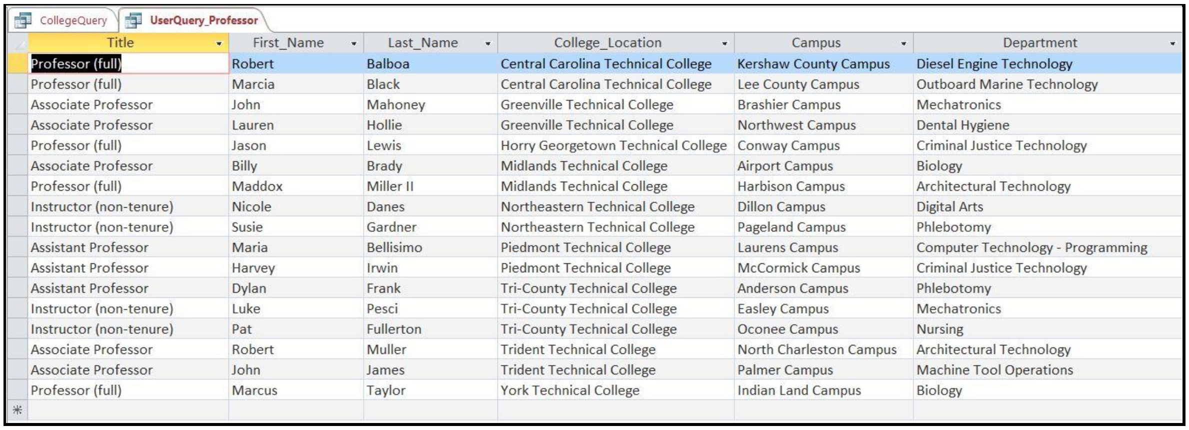
```
SELECT Location.dbLocationName AS College_Name, Campus.dbCampusName AS Campus_Name

FROM Location INNER JOIN Campus ON Location.dbLocationID = Campus.dbLocationID

WHERE Location.dbLocationID=9

ORDER BY Location.dbLocationName, Campus.dbCampusName;
```

USER_QUERY_PROFESSOR: DISPLAYS PROFESSORS AT ALL CAMPUS LOCATIONS



Title	First_Name	Last_Name	College_Location	Campus	Department
Professor (full)	Robert	Balboa	Central Carolina Technical College	Kershaw County Campus	Diesel Engine Technology
Professor (full)	Marcia	Black	Central Carolina Technical College	Lee County Campus	Outboard Marine Technology
Associate Professor	John	Mahoney	Greenville Technical College	Brashier Campus	Mechatronics
Associate Professor	Lauren	Hollie	Greenville Technical College	Northwest Campus	Dental Hygiene
Professor (full)	Jason	Lewis	Horry Georgetown Technical College	Conway Campus	Criminal Justice Technology
Associate Professor	Billy	Brady	Midlands Technical College	Airport Campus	Biology
Professor (full)	Maddox	Miller II	Midlands Technical College	Harbison Campus	Architectural Technology
Instructor (non-tenure)	Nicole	Danes	Northeastern Technical College	Dillon Campus	Digital Arts
Instructor (non-tenure)	Susie	Gardner	Northeastern Technical College	Pageland Campus	Phlebotomy
Assistant Professor	Maria	Bellisimo	Piedmont Technical College	Laurens Campus	Computer Technology - Programming
Assistant Professor	Harvey	Irwin	Piedmont Technical College	McCormick Campus	Criminal Justice Technology
Assistant Professor	Dylan	Frank	Tri-County Technical College	Anderson Campus	Phlebotomy
Instructor (non-tenure)	Luke	Pesci	Tri-County Technical College	Easley Campus	Mechatronics
Instructor (non-tenure)	Pat	Fullerton	Tri-County Technical College	Oconee Campus	Nursing
Associate Professor	Robert	Muller	Trident Technical College	North Charleston Campus	Architectural Technology
Associate Professor	John	James	Trident Technical College	Palmer Campus	Machine Tool Operations
Professor (full)	Marcus	Taylor	York Technical College	Indian Land Campus	Biology

UserQuery_Professor view created *(see above image)*

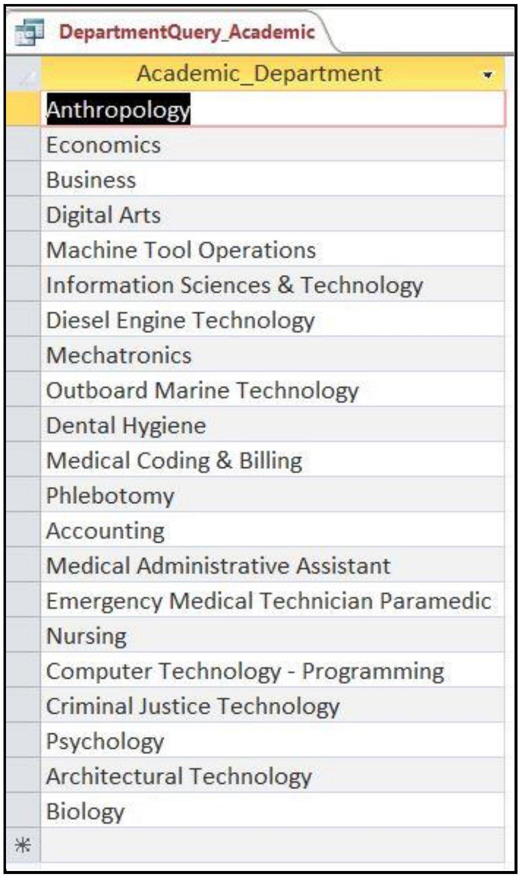
- This query is a bit more complicated, but something I found very useful when I was assigning courses to particular users in the system. I needed to know which users had a title assigned to them that was related to some sort of teaching position. This *includes Instructor (non-tenure)*, *Professor (full)*, *Associate Professor*, and *Assistant Professor*.
- What I then needed to find out was the department that the particular instructor was associated with. Once I knew that information, I was then able to pull up the courses within these particular departments. Once this was established, I was finally able to assign these instructors to their respective classes.
- Obviously, I needed to know which school locations and which campuses they were associated with as well, in order to match the data correctly in the database. You can see here that I pulled, not only the instructor names, but their school locations, particular campus associations, and finally, the departments they were tied to. If you look at the SQL

code below, you can see that the title ids I defined in my database associated with teaching are *2*, *3*, *13*, and *14*.

- Finally, I sorted the data, first by school location, then by campus location, and finally, by the user's last name. You can see how easy it was for me to pull all of this data because of all the key fields each table had in common.

```
SELECT Title.dbTitleName AS Title, User.dbUserFirstName AS First_Name, User.dbUserLastName AS Last_Name,
Location.dbLocationName AS College_Location, Campus.dbCampusName AS Campus,
Department.dbDepartmentName AS Department
FROM Title INNER JOIN (Location INNER JOIN (Department INNER JOIN (Campus INNER JOIN [User] ON
Campus.dbCampusID = User.dbCampusID) ON Department.dbDepartmentID = User.dbDepartmentID) ON
Location.dbLocationID = Campus.dbLocationID) ON Title.dbTitleID = User.dbTitleID
WHERE Title.dbTitleID=2 OR Title.dbTitleID=3 OR Title.dbTitleID=13 OR Title.dbTitleID=14
ORDER BY Location.dbLocationName, Campus.dbCampusName, User.dbUserLastName;
```

DEPARTMENT_QUERY_ACADEMIC: DISPLAYS ONLY ACADEMIC-RELATED DEPARTMENTS



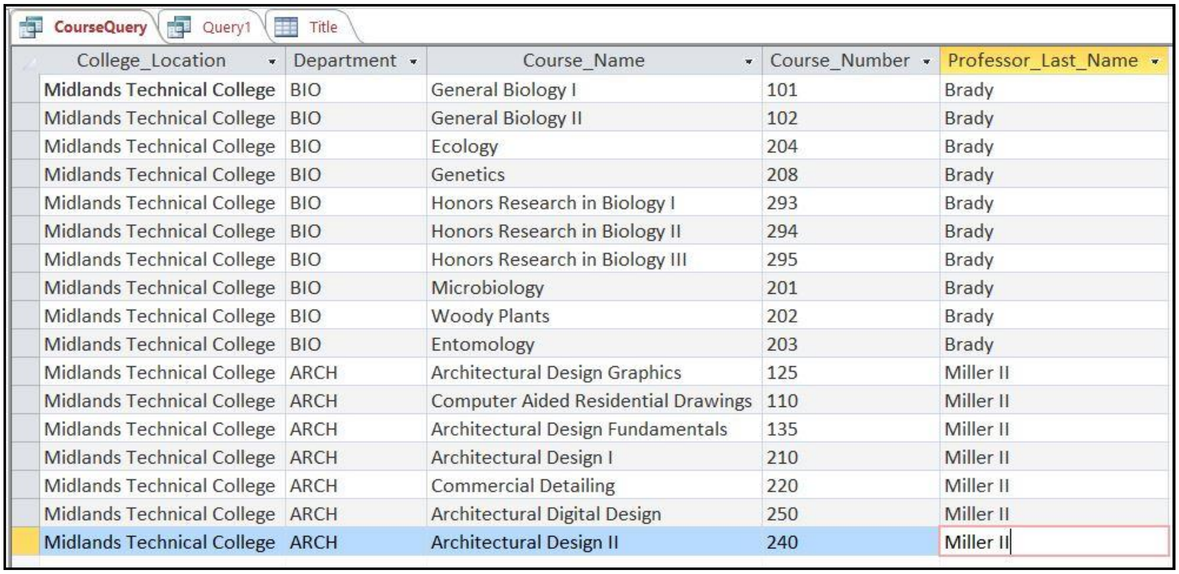
Academic_Department
Anthropology
Economics
Business
Digital Arts
Machine Tool Operations
Information Sciences & Technology
Diesel Engine Technology
Mechatronics
Outboard Marine Technology
Dental Hygiene
Medical Coding & Billing
Phlebotomy
Accounting
Medical Administrative Assistant
Emergency Medical Technician Paramedic
Nursing
Computer Technology - Programming
Criminal Justice Technology
Psychology
Architectural Technology
Biology

DepartmentQuery_Academic view created *(see above image)*

- Here I created a fairly simple query that displays the departments that are only academic-related. What this means is that it left out any of the departments that are defined as staff or administration-related.
- I also used this SQL code to pull data for my drop-down list box on my *Course* and *Class Entry Forms*. When I created these forms, I wanted to give the user the ability to enter a new course or class, but with the ability to associate it with a particular department that was purely degree-related. I did not want to offer the user the chance to inadvertently choose a department that was not degree-related.
- You can see how I defined these filters in the SQL code below. Notice that any department not related to an academic degree are not included in the list (defined by the id numbers *9*, *10*, *11*, *22*, *23*, and *24*).

```
SELECT Department.dbDepartmentName AS Academic_Department
FROM Department
WHERE Department.dbDepartmentID <> 9 AND Department.dbDepartmentID <> 10 AND
Department.dbDepartmentID <> 11 AND Department.dbDepartmentID <> 22 AND (Department.dbDepartmentID
<> 23 AND Department.dbDepartmentID <> 24;
```

COURSE_QUERY: DISPLAYS ALL COURSES OFFERED WITHIN ONE LOCATION



College_Location	Department	Course_Name	Course_Number	Professor_Last_Name
Midlands Technical College	BIO	General Biology I	101	Brady
Midlands Technical College	BIO	General Biology II	102	Brady
Midlands Technical College	BIO	Ecology	204	Brady
Midlands Technical College	BIO	Genetics	208	Brady
Midlands Technical College	BIO	Honors Research in Biology I	293	Brady
Midlands Technical College	BIO	Honors Research in Biology II	294	Brady
Midlands Technical College	BIO	Honors Research in Biology III	295	Brady
Midlands Technical College	BIO	Microbiology	201	Brady
Midlands Technical College	BIO	Woody Plants	202	Brady
Midlands Technical College	BIO	Entomology	203	Brady
Midlands Technical College	ARCH	Architectural Design Graphics	125	Miller II
Midlands Technical College	ARCH	Computer Aided Residential Drawings	110	Miller II
Midlands Technical College	ARCH	Architectural Design Fundamentals	135	Miller II
Midlands Technical College	ARCH	Architectural Design I	210	Miller II
Midlands Technical College	ARCH	Commercial Detailing	220	Miller II
Midlands Technical College	ARCH	Architectural Digital Design	250	Miller II
Midlands Technical College	ARCH	Architectural Design II	240	Miller II

CourseQuery view created *(see above image)*

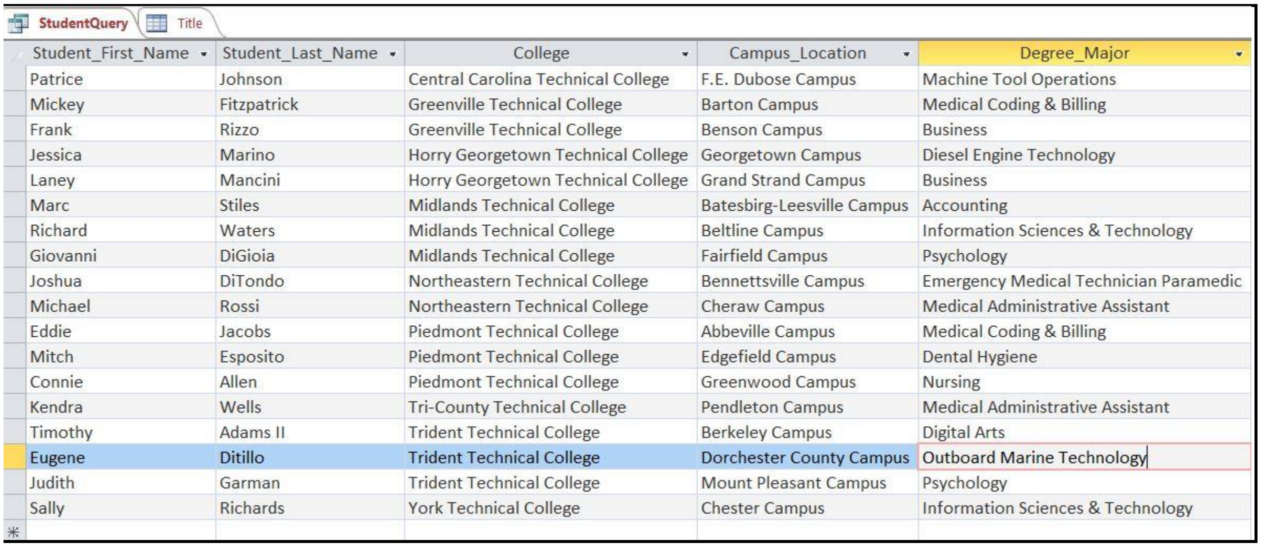
- Here you can see the course query I created which essentially displays all of the current courses offered at a particular college, within their associated academic departments. In this case, all of the current courses within the *Architecture* and *Biology* Departments at the *Midlands Technical College* location are shown, along with the name of the *Professor* associated with these courses.
- The SQL code below displays the columns with alias names, and you can see where I defined the users to be shown with a title id of *2*, *3*, *4*, *13*, and *14*, indicating an instructor-type of user. It also displays the location id with a value of *3,* indicating the particular college chosen.

```
SELECT Location.dbLocationName AS College_Location, Department.dbDepartmentAbbreviation AS Department,
Course.dbCourseName AS Course_Name, Course.dbCourseNumber AS Course_Number, User.dbUserLastName AS
Professor_Last_Name
```

```
FROM Title INNER JOIN (Location INNER JOIN ((Department INNER JOIN Course ON Department.dbDepartmentID =
Course.dbDepartmentID) INNER JOIN (Campus INNER JOIN [User] ON Campus.dbCampusID = User.dbCampusID)
ON Department.dbDepartmentID = User.dbDepartmentID) ON Location.dbLocationID = Campus.dbLocationID) ON
Title.dbTitleID = User.dbTitleID
```

```
WHERE (((Title.dbTitleID)=2) AND ((Location.dbLocationID)=3)) OR (((Title.dbTitleID)=3) AND
((Location.dbLocationID)=3)) OR (((Title.dbTitleID)=4) AND ((Location.dbLocationID)=3)) OR (((Title.dbTitleID)=13)
AND ((Location.dbLocationID)=3)) OR (((Title.dbTitleID)=14) AND ((Location.dbLocationID)=3));
```

STUDENT_QUERY: DISPLAYS CURRENT STUDENTS AT ALL COLLEGE LOCATIONS



Student_First_Name	Student_Last_Name	College	Campus_Location	Degree_Major
Patrice	Johnson	Central Carolina Technical College	F.E. Dubose Campus	Machine Tool Operations
Mickey	Fitzpatrick	Greenville Technical College	Barton Campus	Medical Coding & Billing
Frank	Rizzo	Greenville Technical College	Benson Campus	Business
Jessica	Marino	Horry Georgetown Technical College	Georgetown Campus	Diesel Engine Technology
Laney	Mancini	Horry Georgetown Technical College	Grand Strand Campus	Business
Marc	Stiles	Midlands Technical College	Batesbirg-Leesville Campus	Accounting
Richard	Waters	Midlands Technical College	Beltline Campus	Information Sciences & Technology
Giovanni	DiGioia	Midlands Technical College	Fairfield Campus	Psychology
Joshua	DiTondo	Northeastern Technical College	Bennettsville Campus	Emergency Medical Technician Paramedic
Michael	Rossi	Northeastern Technical College	Cheraw Campus	Medical Administrative Assistant
Eddie	Jacobs	Piedmont Technical College	Abbeville Campus	Medical Coding & Billing
Mitch	Esposito	Piedmont Technical College	Edgefield Campus	Dental Hygiene
Connie	Allen	Piedmont Technical College	Greenwood Campus	Nursing
Kendra	Wells	Tri-County Technical College	Pendleton Campus	Medical Administrative Assistant
Timothy	Adams II	Trident Technical College	Berkeley Campus	Digital Arts
Eugene	Ditillo	Trident Technical College	Dorchester County Campus	Outboard Marine Technology
Judith	Garman	Trident Technical College	Mount Pleasant Campus	Psychology
Sally	Richards	York Technical College	Chester Campus	Information Sciences & Technology

StudentQuery view created *(see above image)*

- Here I created a query that displays all of the current students within the system. Their associated college and campus locations are displayed as well. You can also see that the academic department assigned to each student as well, which is essentially their major of study. I sorted the results, first by location name, then by campus name, and finally, by the student's last name *(students are associated with a title id of 5).*

```
SELECT User.dbUserFirstName AS Student_First_Name, User.dbUserLastName AS Student_Last_Name,
Location.dbLocationName AS College, Campus.dbCampusName AS Campus_Location,
Department.dbDepartmentName AS Degree_Major
```

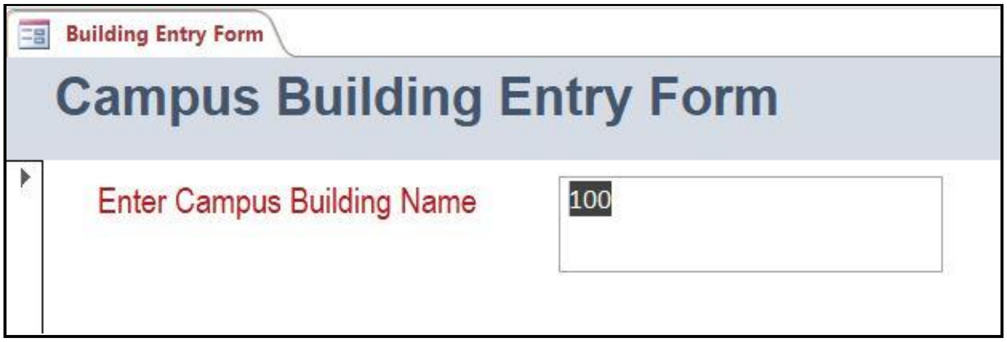
```
FROM Title INNER JOIN (Department INNER JOIN (Location INNER JOIN (Campus INNER JOIN [User] ON Campus.dbCampusID = User.dbCampusID) ON Location.dbLocationID = Campus.dbLocationID) ON Department.dbDepartmentID = User.dbDepartmentID) ON Title.dbTitleID = User.dbTitleID

WHERE Title.dbTitleID = 5

ORDER BY Location.dbLocationName, Campus.dbCampusName, User.dbUserLastName;
```

ACCESS ENTRY FORMS CREATED FOR ALL DATABASE TABLES

BUILDING ENTRY FORM

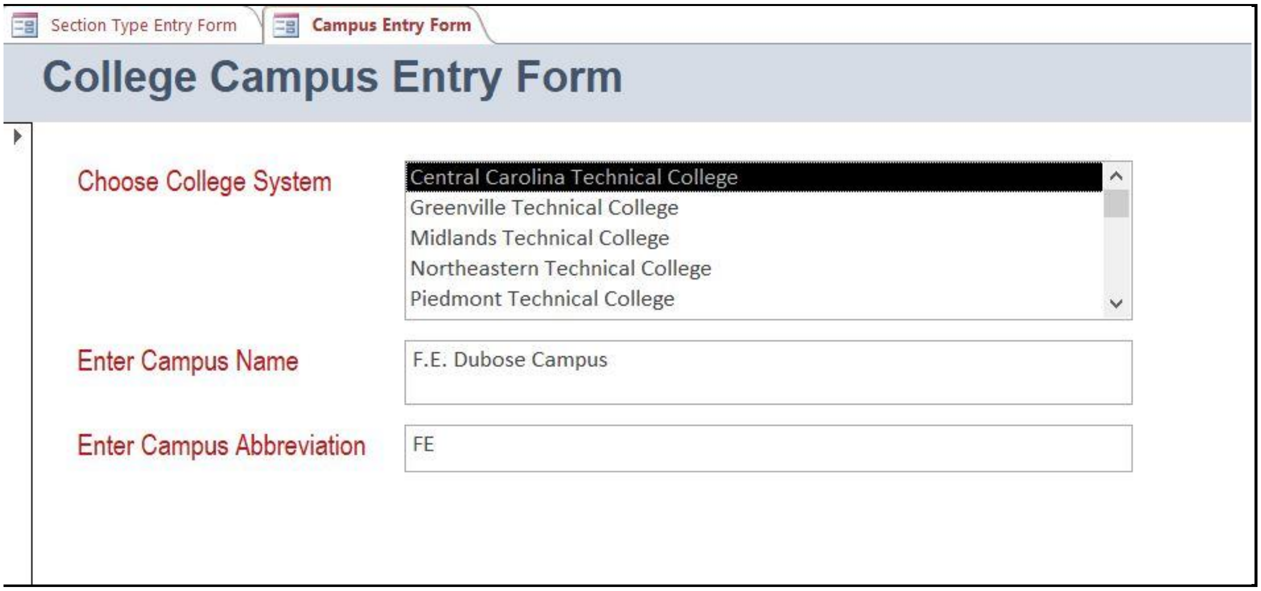


Campus Building Entry Form

Enter Campus Building Name: 100

Building Entry Form created *(see above image)*

CAMPUS ENTRY FORM



College Campus Entry Form

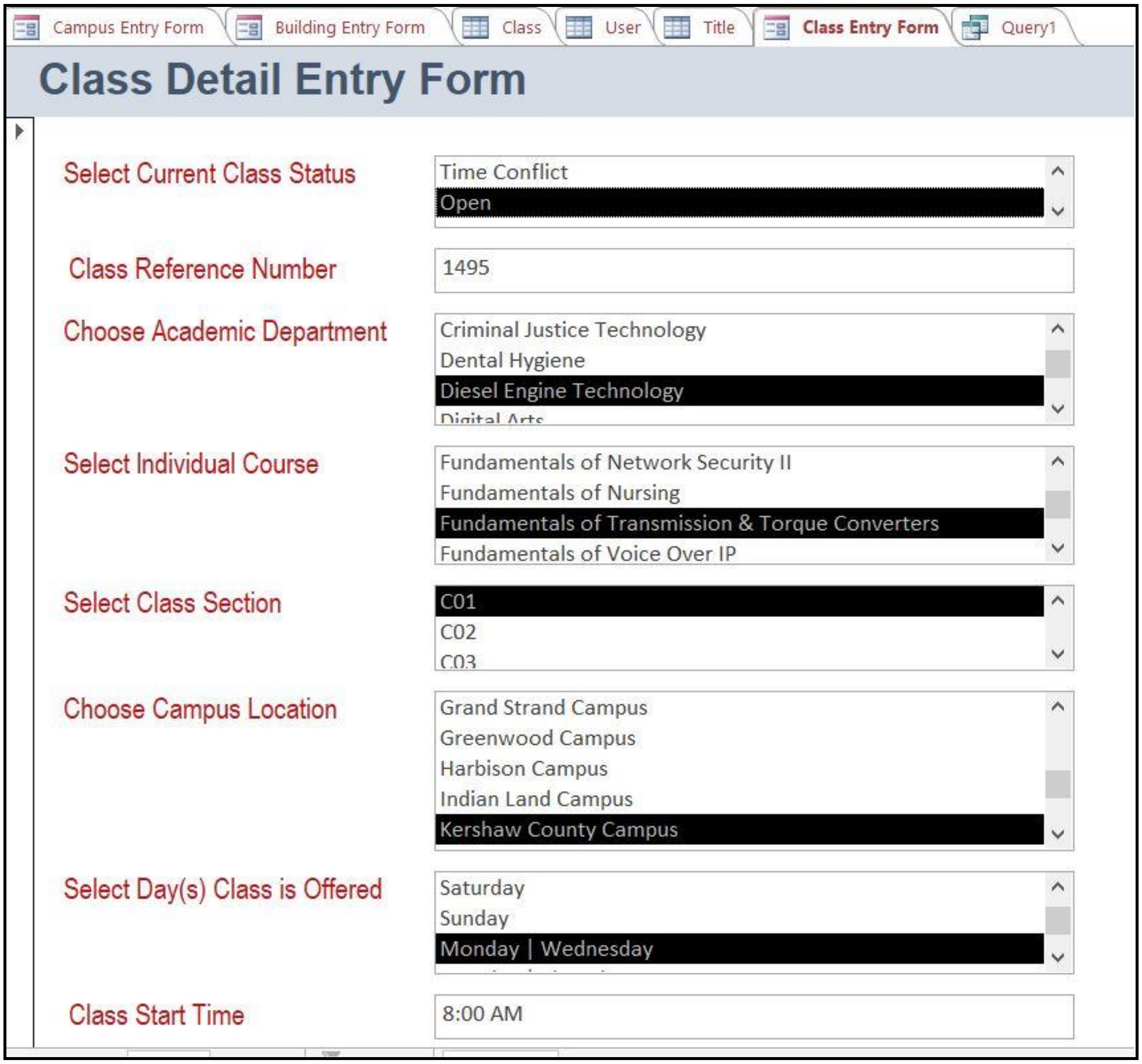
Choose College System:
- Central Carolina Technical College
- Greenville Technical College
- Midlands Technical College
- Northeastern Technical College
- Piedmont Technical College

Enter Campus Name: F.E. Dubose Campus

Enter Campus Abbreviation: FE

Campus Entry Form created *(see above image)*

CLASS ENTRY FORM



Class Detail Entry Form

Select Current Class Status
Time Conflict
Open

Class Reference Number
1495

Choose Academic Department
Criminal Justice Technology
Dental Hygiene
Diesel Engine Technology
Digital Arts

Select Individual Course
Fundamentals of Network Security II
Fundamentals of Nursing
Fundamentals of Transmission & Torque Converters
Fundamentals of Voice Over IP

Select Class Section
C01
C02
C03

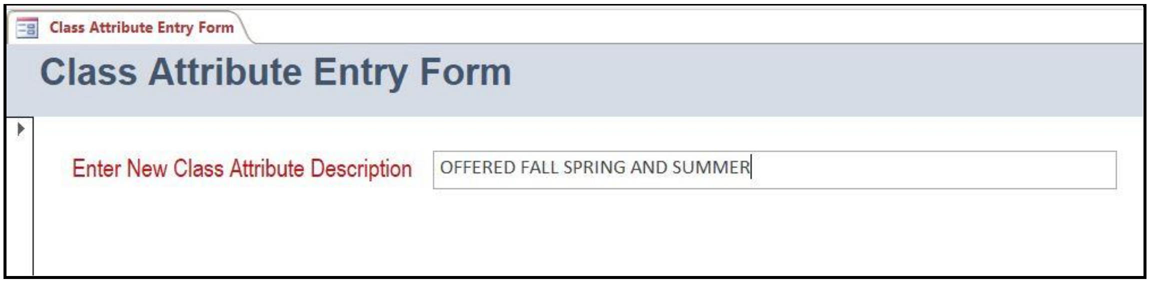
Choose Campus Location
Grand Strand Campus
Greenwood Campus
Harbison Campus
Indian Land Campus
Kershaw County Campus

Select Day(s) Class is Offered
Saturday
Sunday
Monday | Wednesday

Class Start Time
8:00 AM

Class Entry Form created *(see above image)*

CLASS_ATTRIBUTE ENTRY FORM

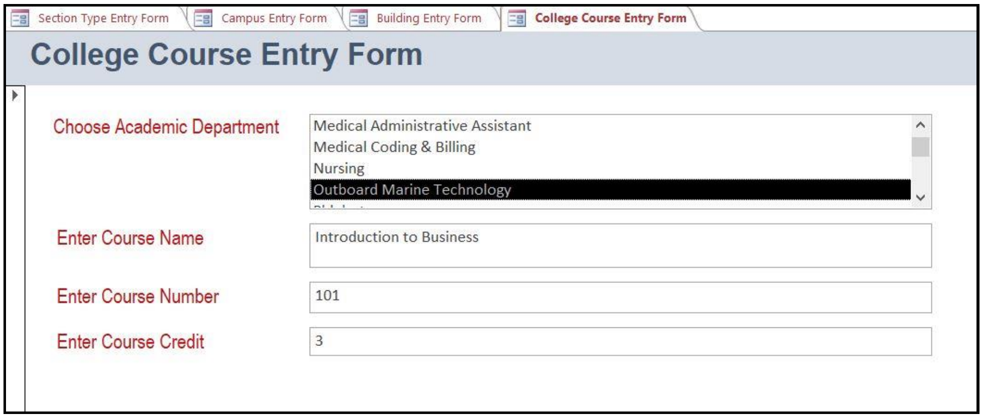


Class Attribute Entry Form

Enter New Class Attribute Description
OFFERED FALL SPRING AND SUMMER

Class Attribute Entry Form created *(see above image)*

COURSE ENTRY FORM



College Course Entry Form

Choose Academic Department: Medical Administrative Assistant / Medical Coding & Billing / Nursing / Outboard Marine Technology

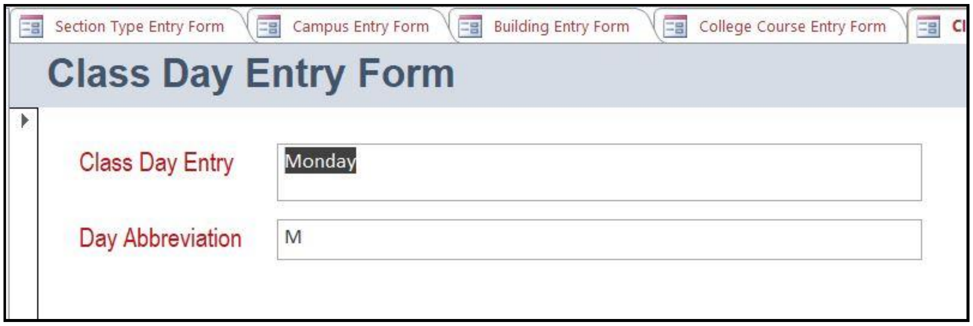
Enter Course Name: Introduction to Business

Enter Course Number: 101

Enter Course Credit: 3

Course Entry Form created *(see above image)*

DAY ENTRY FORM



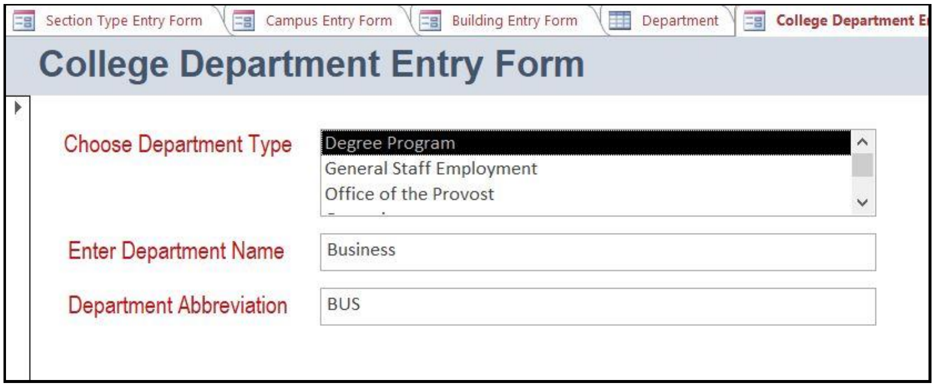
Class Day Entry Form

Class Day Entry: Monday

Day Abbreviation: M

Day Entry Form created *(see above image)*

DEPARTMENT ENTRY FORM



College Department Entry Form

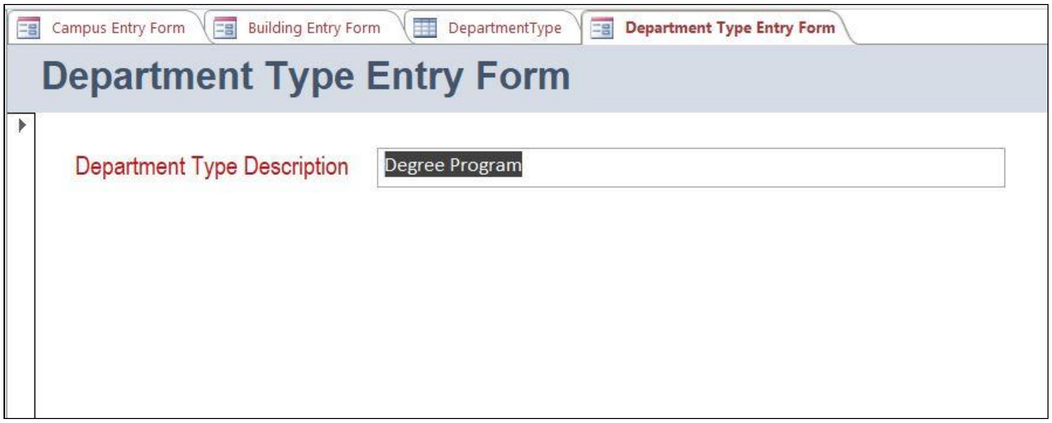
Choose Department Type: Degree Program / General Staff Employment / Office of the Provost

Enter Department Name: Business

Department Abbreviation: BUS

Department Entry Form created *(see above image)*

DEPARTMENT_TYPE ENTRY FORM

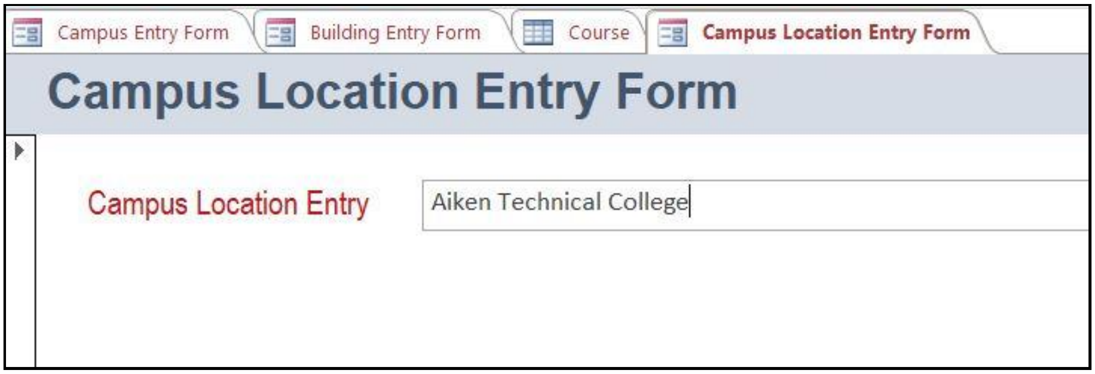


Department Type Entry Form

Department Type Description	Degree Program

Department Type Entry Form created *(see above image)*

LOCATION ENTRY FORM

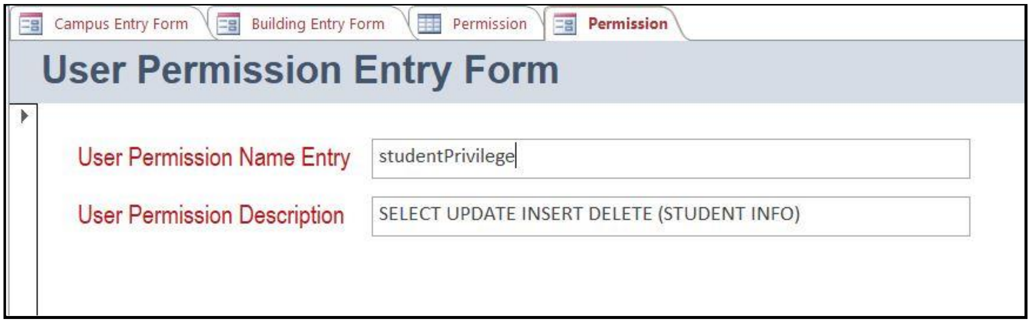


Campus Location Entry Form

Campus Location Entry	Aiken Technical College

Location Entry Form created *(see above image)*

PERMISSION ENTRY FORM

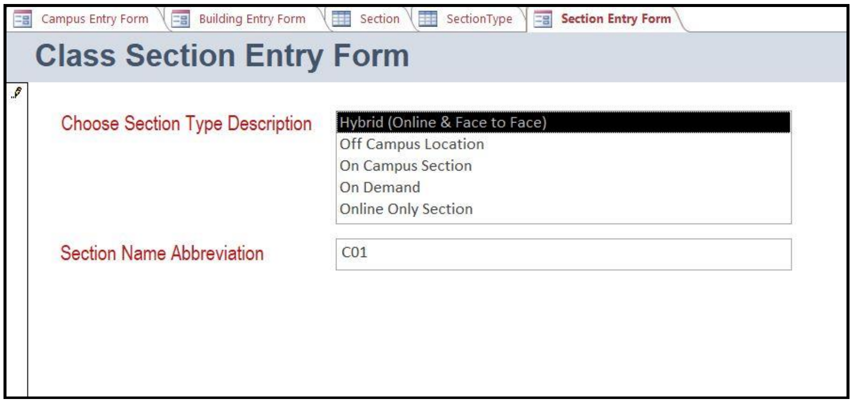


User Permission Entry Form

User Permission Name Entry	studentPrivilege
User Permission Description	SELECT UPDATE INSERT DELETE (STUDENT INFO)

Permission Entry Form created *(see above image)*

SECTION ENTRY FORM



Class Section Entry Form

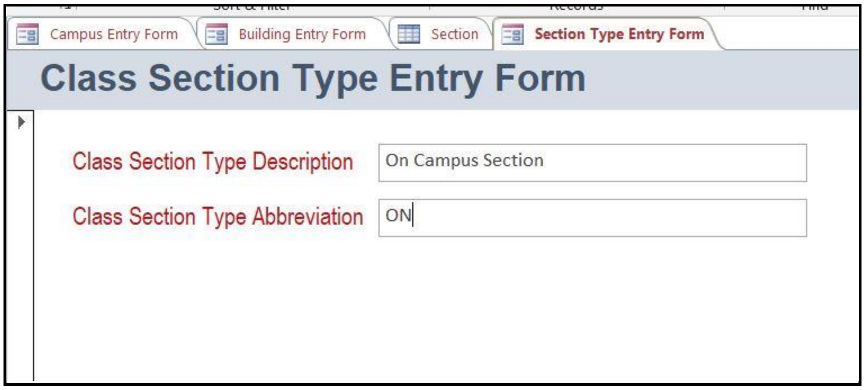
Choose Section Type Description

- Hybrid (Online & Face to Face)
- Off Campus Location
- On Campus Section
- On Demand
- Online Only Section

Section Name Abbreviation: C01

Section Entry Form created *(see above image)*

SECTION_TYPE ENTRY FORM



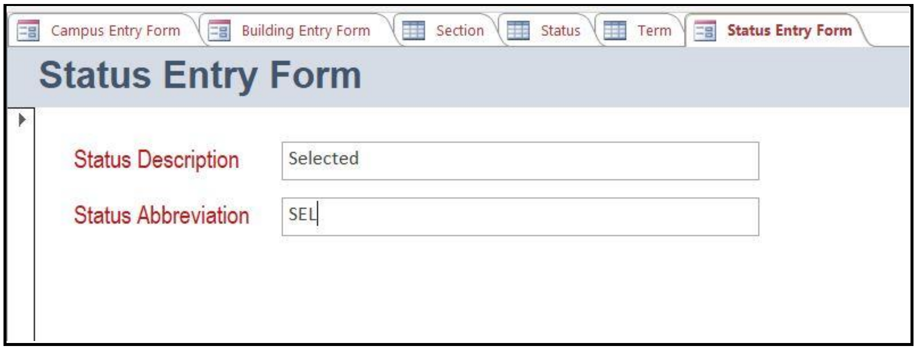
Class Section Type Entry Form

Class Section Type Description: On Campus Section

Class Section Type Abbreviation: ON

Section Type Entry Form created *(see above image)*

STATUS ENTRY FORM



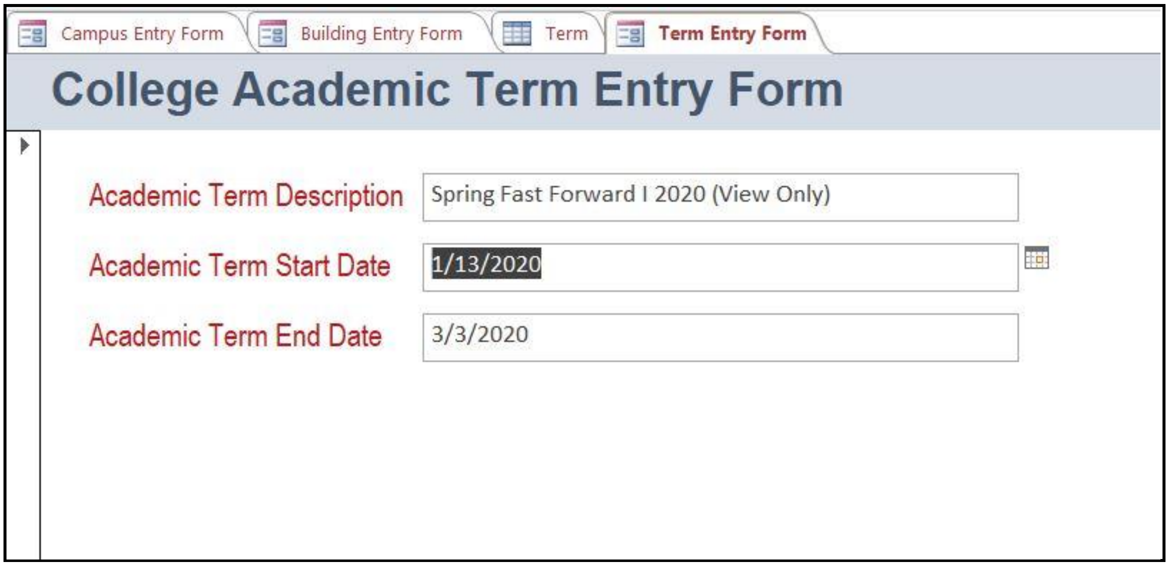
Status Entry Form

Status Description: Selected

Status Abbreviation: SEL

Status Entry Form created *(see above image)*

TERM ENTRY FORM



College Academic Term Entry Form

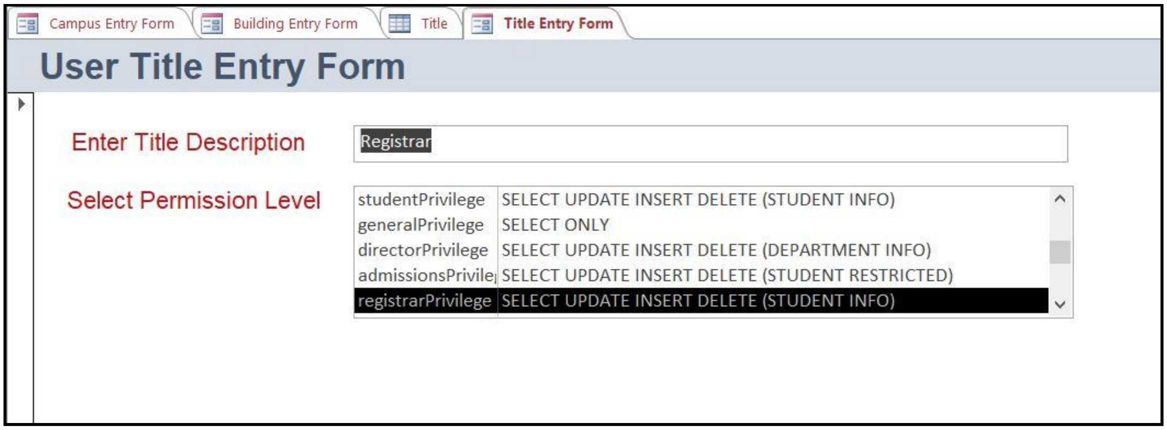
Academic Term Description: Spring Fast Forward I 2020 (View Only)

Academic Term Start Date: 1/13/2020

Academic Term End Date: 3/3/2020

Term Entry Form created *(see above image)*

TITLE ENTRY FORM



User Title Entry Form

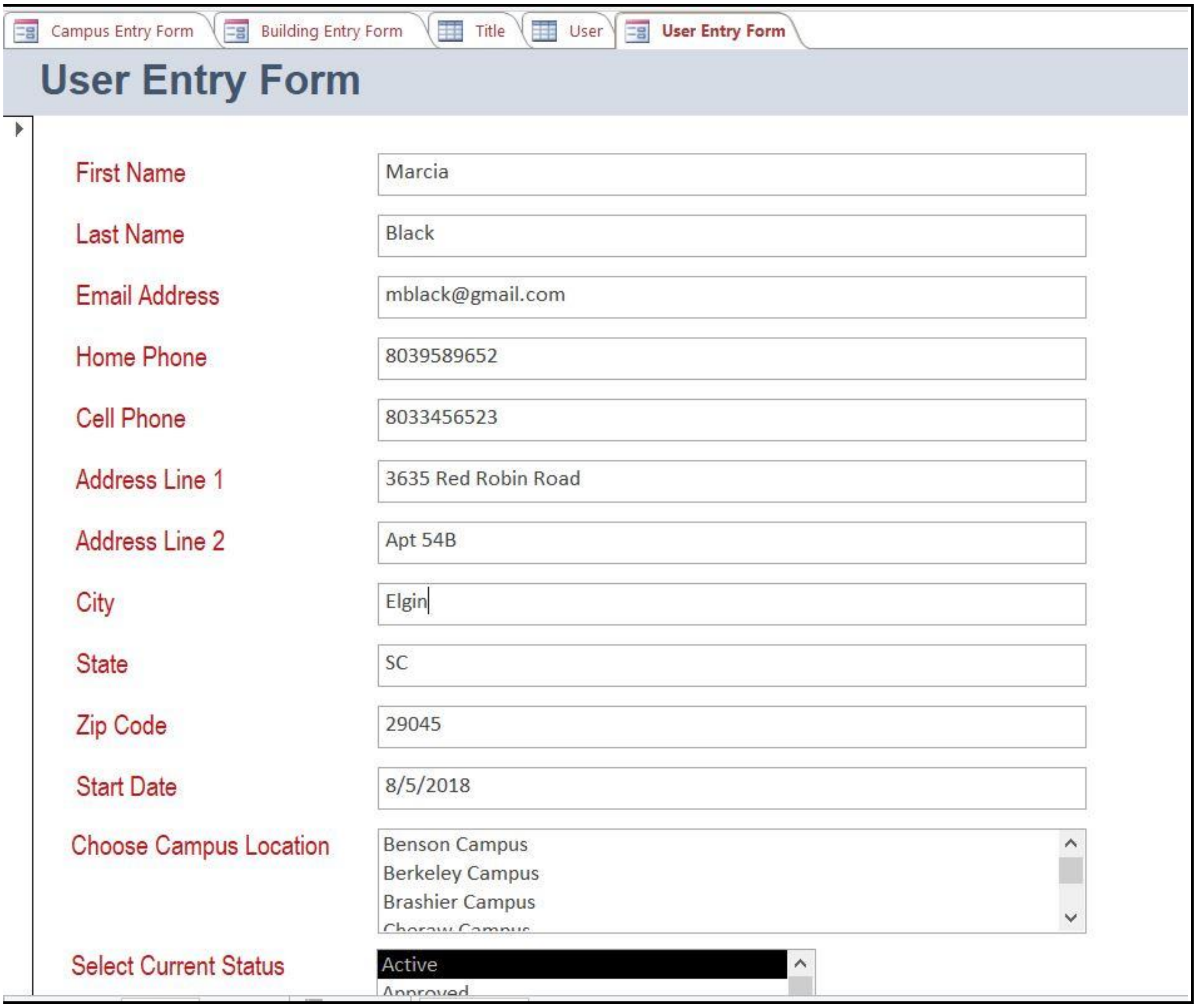
Enter Title Description: Registrar

Select Permission Level

studentPrivilege	SELECT UPDATE INSERT DELETE (STUDENT INFO)
generalPrivilege	SELECT ONLY
directorPrivilege	SELECT UPDATE INSERT DELETE (DEPARTMENT INFO)
admissionsPrivile	SELECT UPDATE INSERT DELETE (STUDENT RESTRICTED)
registrarPrivilege	SELECT UPDATE INSERT DELETE (STUDENT INFO)

Title Entry Form created *(see above image)*

USER ENTRY FORM



User Entry Form

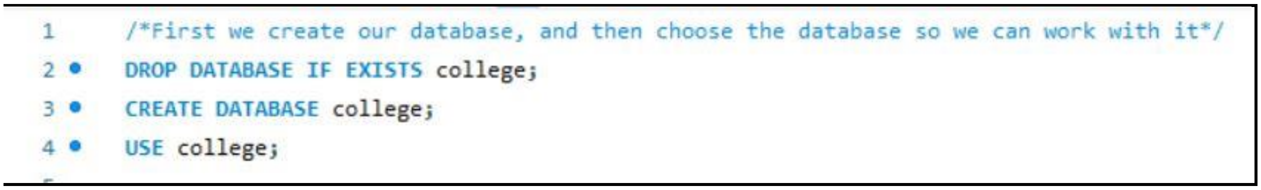
First Name	Marcia
Last Name	Black
Email Address	mblack@gmail.com
Home Phone	8039589652
Cell Phone	8033456523
Address Line 1	3635 Red Robin Road
Address Line 2	Apt 54B
City	Elgin
State	SC
Zip Code	29045
Start Date	8/5/2018
Choose Campus Location	Benson Campus Berkeley Campus Brashier Campus Cheraw Campus
Select Current Status	Active Approved

User Entry Form created *(see above image)*

- You will notice that almost all of the entry forms I created had some sort of drop-down box that pulled related data from another table. Some entry forms were very simple, consisting of only one or two entry fields, but there were several more complicated forms created as well.
- All of the entry forms I created work correctly, and all of the drop-down list boxes pull the correct data from their related tables as well.

MYSQL WORKBENCH SCHOOL DATABASE & TABLES CREATED

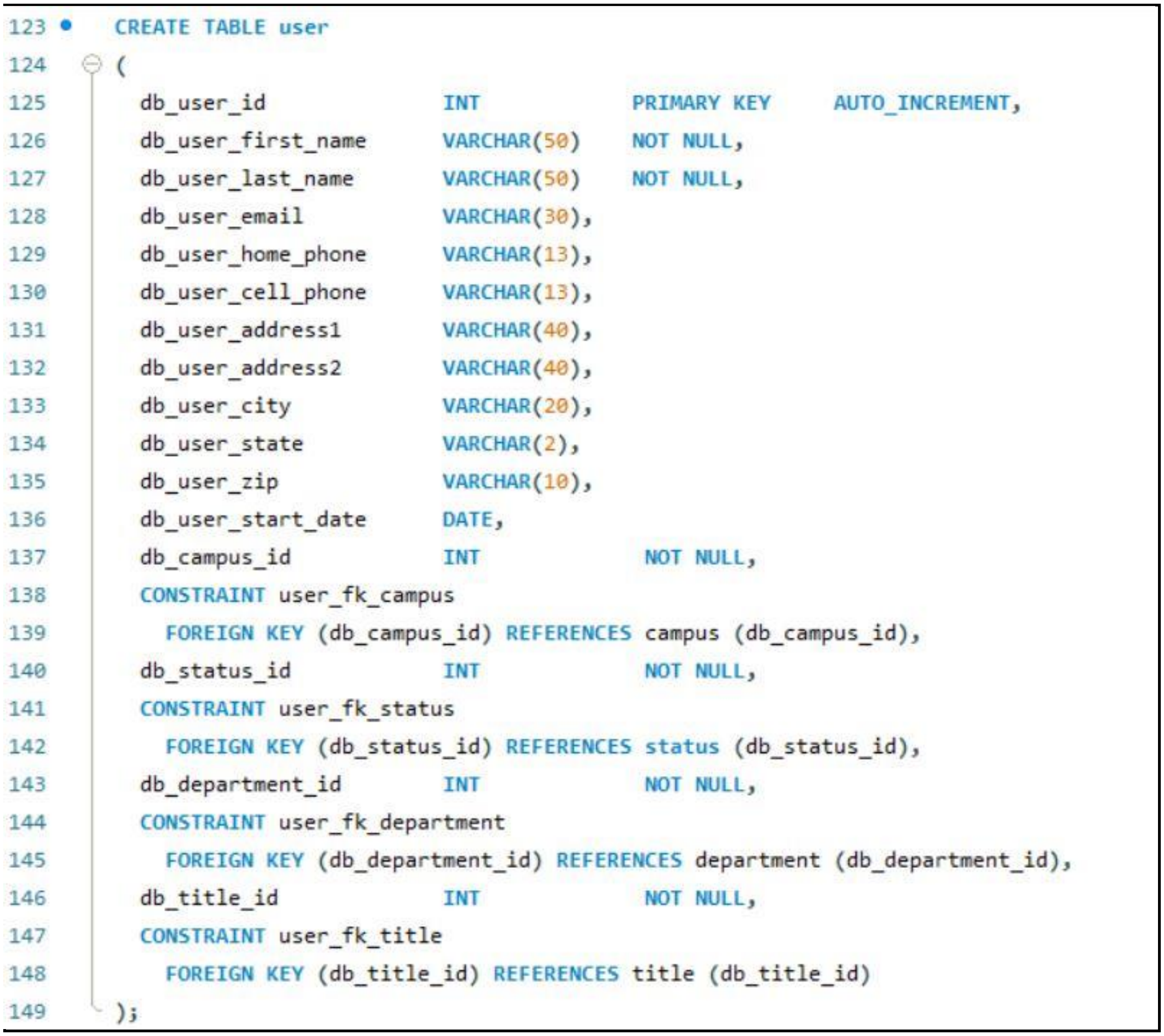
MYSQL WORKBENCH *COLLEGE* DATABASE



```sql
/*First we create our database, and then choose the database so we can work with it*/
DROP DATABASE IF EXISTS college;
CREATE DATABASE college;
USE college;
```

***College* Database** created *(see above image)*

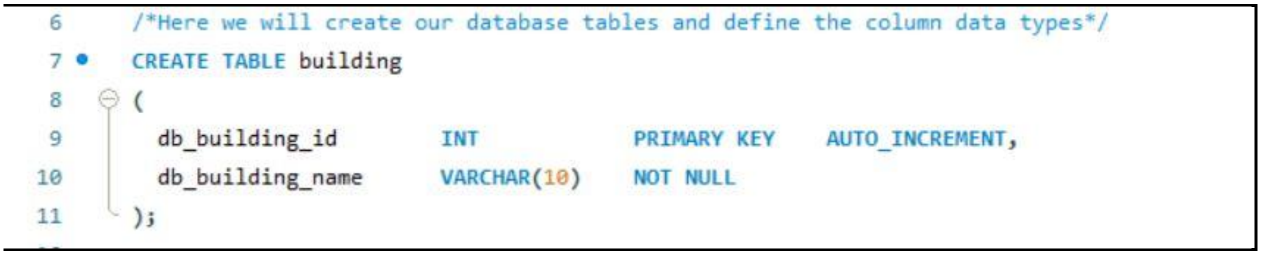
MYSQL USER TABLE



```sql
CREATE TABLE user
(
  db_user_id              INT            PRIMARY KEY     AUTO_INCREMENT,
  db_user_first_name      VARCHAR(50)    NOT NULL,
  db_user_last_name       VARCHAR(50)    NOT NULL,
  db_user_email           VARCHAR(30),
  db_user_home_phone      VARCHAR(13),
  db_user_cell_phone      VARCHAR(13),
  db_user_address1        VARCHAR(40),
  db_user_address2        VARCHAR(40),
  db_user_city            VARCHAR(20),
  db_user_state           VARCHAR(2),
  db_user_zip             VARCHAR(10),
  db_user_start_date      DATE,
  db_campus_id            INT             NOT NULL,
  CONSTRAINT user_fk_campus
    FOREIGN KEY (db_campus_id) REFERENCES campus (db_campus_id),
  db_status_id            INT             NOT NULL,
  CONSTRAINT user_fk_status
    FOREIGN KEY (db_status_id) REFERENCES status (db_status_id),
  db_department_id        INT             NOT NULL,
  CONSTRAINT user_fk_department
    FOREIGN KEY (db_department_id) REFERENCES department (db_department_id),
  db_title_id             INT             NOT NULL,
  CONSTRAINT user_fk_title
    FOREIGN KEY (db_title_id) REFERENCES title (db_title_id)
);
```

***User* Table** created *(see above image)*

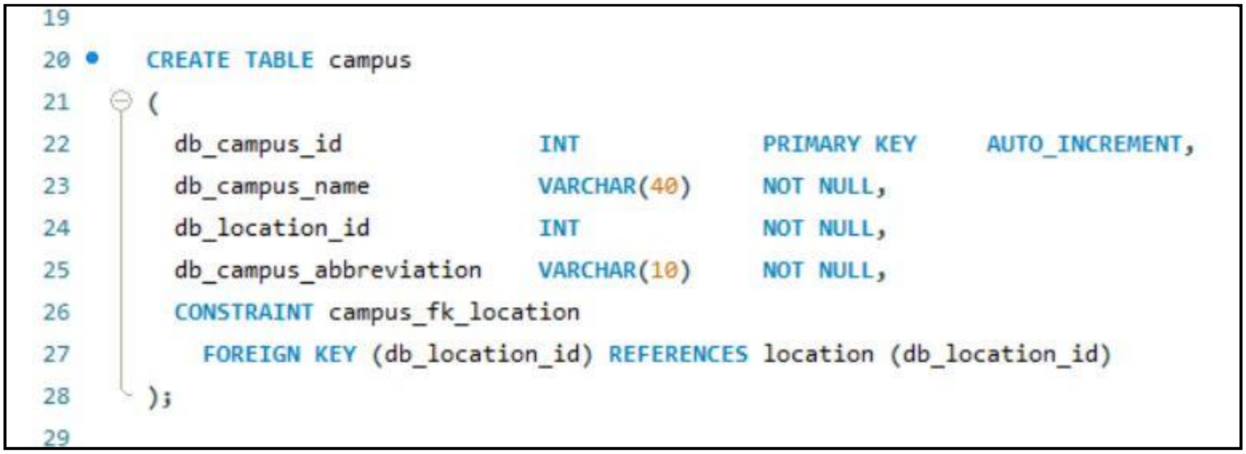
MYSQL BUILDING TABLE



```
/*Here we will create our database tables and define the column data types*/
CREATE TABLE building
(
  db_building_id       INT            PRIMARY KEY    AUTO_INCREMENT,
  db_building_name     VARCHAR(10)    NOT NULL
);
```

***Building* Table** created *(see above image)*

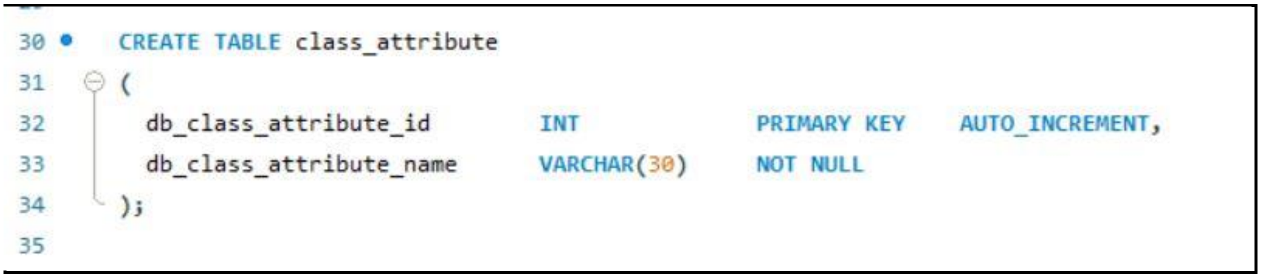
MYSQL CAMPUS TABLE



```
CREATE TABLE campus
(
  db_campus_id              INT            PRIMARY KEY    AUTO_INCREMENT,
  db_campus_name            VARCHAR(40)    NOT NULL,
  db_location_id            INT            NOT NULL,
  db_campus_abbreviation    VARCHAR(10)    NOT NULL,
  CONSTRAINT campus_fk_location
    FOREIGN KEY (db_location_id) REFERENCES location (db_location_id)
);
```

***Campus* Table** created *(see above image)*

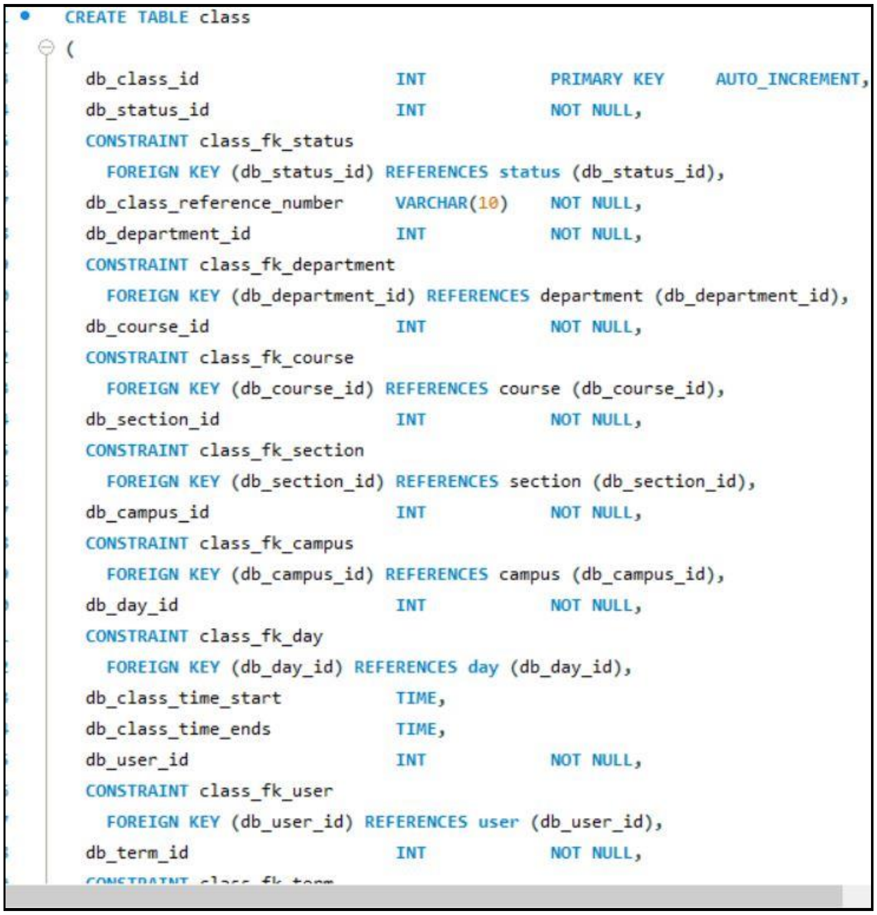
MYSQL CLASS_ATTRIBUTE TABLE



```
CREATE TABLE class_attribute
(
  db_class_attribute_id       INT            PRIMARY KEY    AUTO_INCREMENT,
  db_class_attribute_name     VARCHAR(30)    NOT NULL
);
```

***Class_Attribute* Table** created *(see above image)*

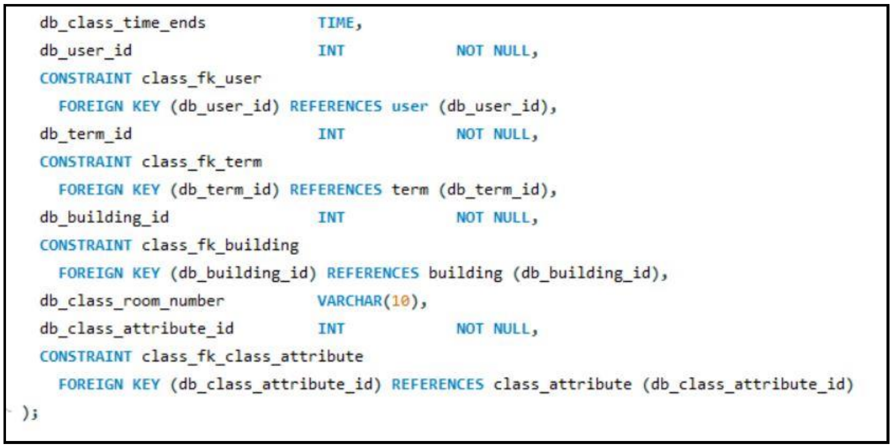
MYSQL CLASS TABLE



```
CREATE TABLE class
(
  db_class_id                    INT            PRIMARY KEY     AUTO_INCREMENT,
  db_status_id                   INT            NOT NULL,
  CONSTRAINT class_fk_status
    FOREIGN KEY (db_status_id) REFERENCES status (db_status_id),
  db_class_reference_number      VARCHAR(10)    NOT NULL,
  db_department_id               INT            NOT NULL,
  CONSTRAINT class_fk_department
    FOREIGN KEY (db_department_id) REFERENCES department (db_department_id),
  db_course_id                   INT            NOT NULL,
  CONSTRAINT class_fk_course
    FOREIGN KEY (db_course_id) REFERENCES course (db_course_id),
  db_section_id                  INT            NOT NULL,
  CONSTRAINT class_fk_section
    FOREIGN KEY (db_section_id) REFERENCES section (db_section_id),
  db_campus_id                   INT            NOT NULL,
  CONSTRAINT class_fk_campus
    FOREIGN KEY (db_campus_id) REFERENCES campus (db_campus_id),
  db_day_id                      INT            NOT NULL,
  CONSTRAINT class_fk_day
    FOREIGN KEY (db_day_id) REFERENCES day (db_day_id),
  db_class_time_start            TIME,
  db_class_time_ends             TIME,
  db_user_id                     INT            NOT NULL,
  CONSTRAINT class_fk_user
    FOREIGN KEY (db_user_id) REFERENCES user (db_user_id),
  db_term_id                     INT            NOT NULL,
  CONSTRAINT class_fk_term
```

***Class* Table** created *(see above image)*

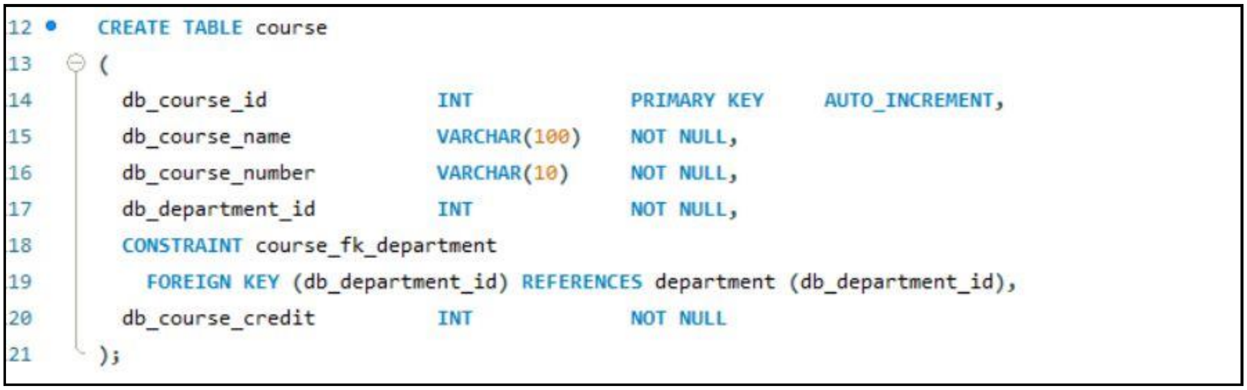
MYSQL CLASS TABLE (CONT'D)



```
  db_class_time_ends             TIME,
  db_user_id                     INT            NOT NULL,
  CONSTRAINT class_fk_user
    FOREIGN KEY (db_user_id) REFERENCES user (db_user_id),
  db_term_id                     INT            NOT NULL,
  CONSTRAINT class_fk_term
    FOREIGN KEY (db_term_id) REFERENCES term (db_term_id),
  db_building_id                 INT            NOT NULL,
  CONSTRAINT class_fk_building
    FOREIGN KEY (db_building_id) REFERENCES building (db_building_id),
  db_class_room_number           VARCHAR(10),
  db_class_attribute_id          INT            NOT NULL,
  CONSTRAINT class_fk_class_attribute
    FOREIGN KEY (db_class_attribute_id) REFERENCES class_attribute (db_class_attribute_id)
);
```

***Class* Table** created *(see above image)*

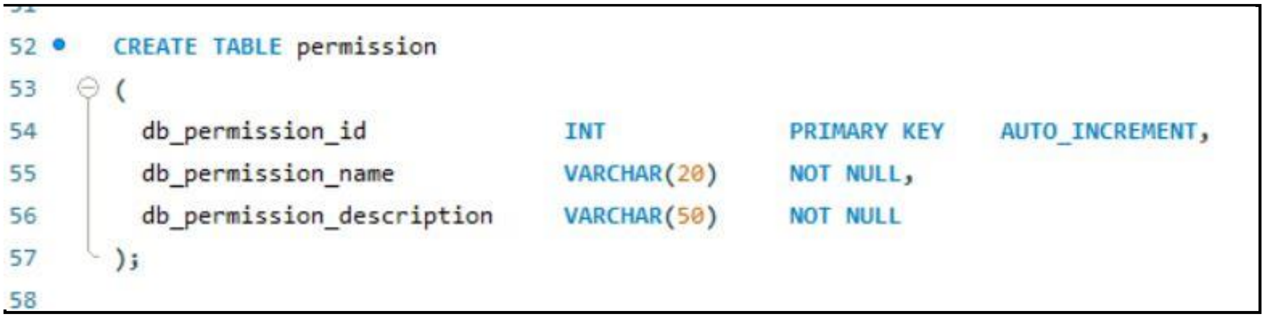
MYSQL COURSE TABLE



```
CREATE TABLE course
(
  db_course_id            INT             PRIMARY KEY     AUTO_INCREMENT,
  db_course_name          VARCHAR(100)    NOT NULL,
  db_course_number        VARCHAR(10)     NOT NULL,
  db_department_id        INT             NOT NULL,
  CONSTRAINT course_fk_department
    FOREIGN KEY (db_department_id) REFERENCES department (db_department_id),
  db_course_credit        INT             NOT NULL
);
```

***Course* Table** created *(see above image)*

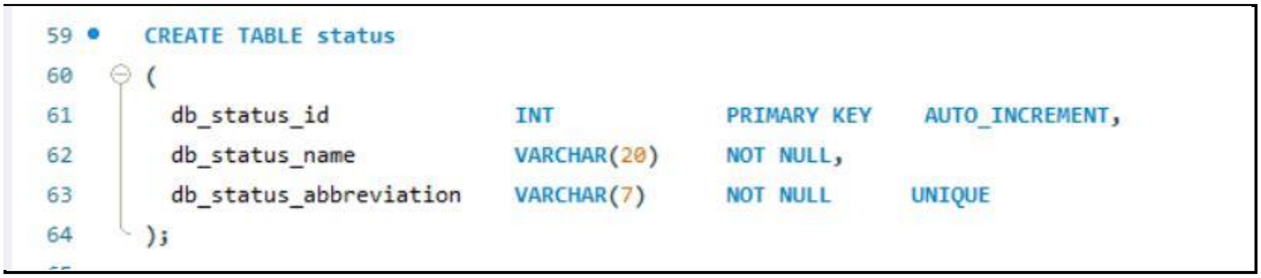
MYSQL PERMISSION TABLE



```
CREATE TABLE permission
(
  db_permission_id              INT            PRIMARY KEY     AUTO_INCREMENT,
  db_permission_name            VARCHAR(20)    NOT NULL,
  db_permission_description     VARCHAR(50)    NOT NULL
);
```

***Permission* Table** created *(see above image)*

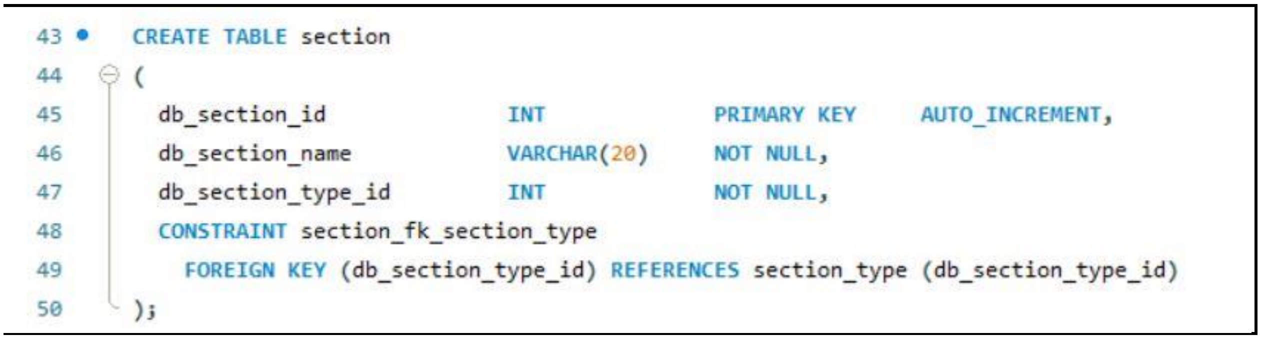
MYSQL STATUS TABLE



```
CREATE TABLE status
(
  db_status_id              INT            PRIMARY KEY     AUTO_INCREMENT,
  db_status_name            VARCHAR(20)    NOT NULL,
  db_status_abbreviation    VARCHAR(7)     NOT NULL       UNIQUE
);
```

***Status* Table** created *(see above image)*

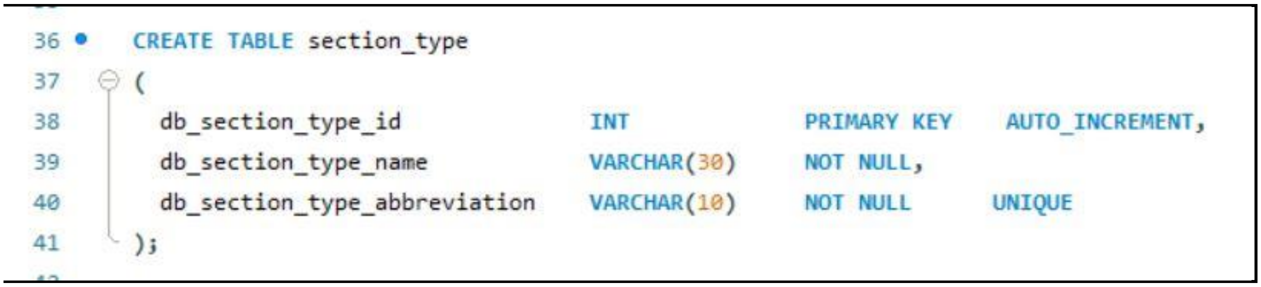
MYSQL SECTION TABLE



```
CREATE TABLE section
(
  db_section_id             INT           PRIMARY KEY    AUTO_INCREMENT,
  db_section_name           VARCHAR(20)   NOT NULL,
  db_section_type_id        INT           NOT NULL,
  CONSTRAINT section_fk_section_type
    FOREIGN KEY (db_section_type_id) REFERENCES section_type (db_section_type_id)
);
```

***Section* Table** created *(see above image)*

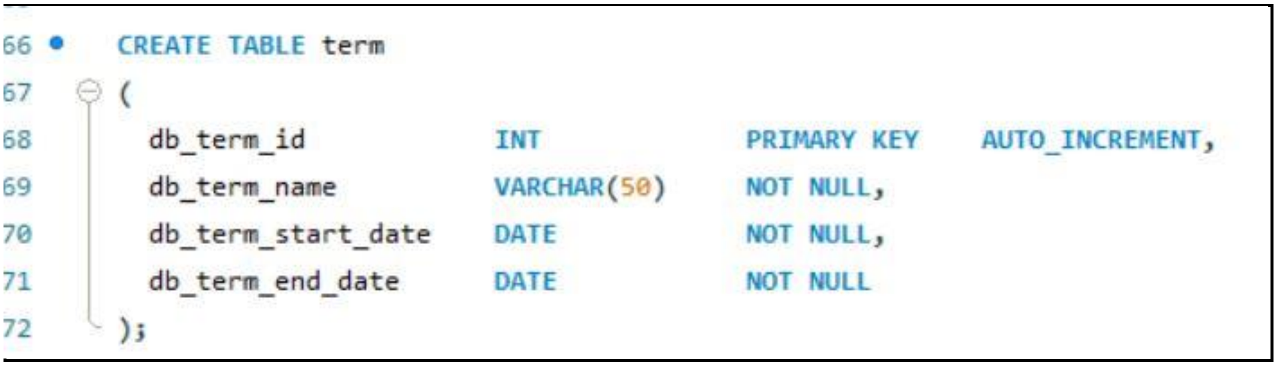
MYSQL SECTION_TYPE TABLE



```
CREATE TABLE section_type
(
  db_section_type_id              INT           PRIMARY KEY    AUTO_INCREMENT,
  db_section_type_name            VARCHAR(30)   NOT NULL,
  db_section_type_abbreviation    VARCHAR(10)   NOT NULL      UNIQUE
);
```

***Section_Type* Table** created *(see above image)*

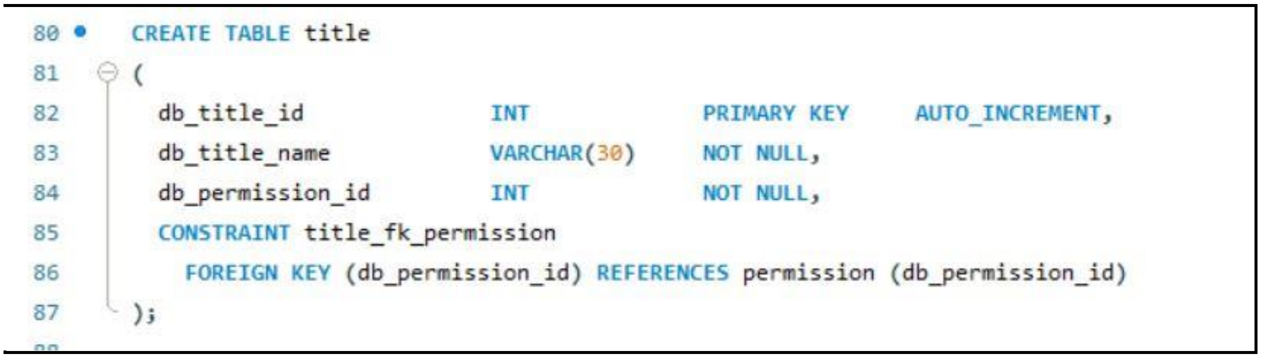
MYSQL TERM TABLE



```
CREATE TABLE term
(
  db_term_id            INT           PRIMARY KEY    AUTO_INCREMENT,
  db_term_name          VARCHAR(50)   NOT NULL,
  db_term_start_date    DATE          NOT NULL,
  db_term_end_date      DATE          NOT NULL
);
```

***Term* Table** created *(see above image)*

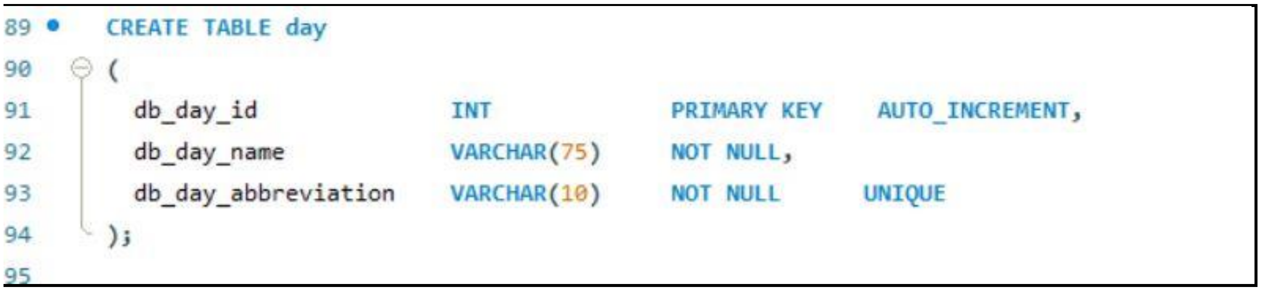
MYSQL TITLE TABLE



```
CREATE TABLE title
(
  db_title_id           INT            PRIMARY KEY     AUTO_INCREMENT,
  db_title_name         VARCHAR(30)    NOT NULL,
  db_permission_id      INT            NOT NULL,
  CONSTRAINT title_fk_permission
    FOREIGN KEY (db_permission_id) REFERENCES permission (db_permission_id)
);
```

***Title* Table** created *(see above image)*

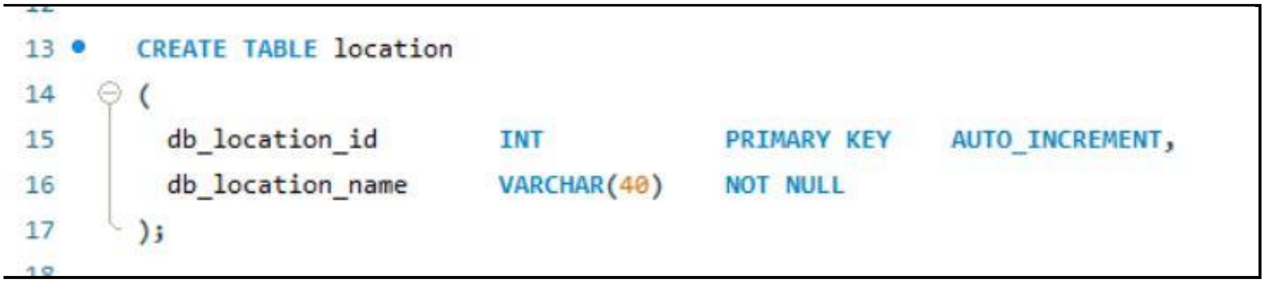
MYSQL DAY TABLE



```
CREATE TABLE day
(
  db_day_id              INT            PRIMARY KEY    AUTO_INCREMENT,
  db_day_name            VARCHAR(75)    NOT NULL,
  db_day_abbreviation    VARCHAR(10)    NOT NULL      UNIQUE
);
```

***Day* Table** created *(see above image)*

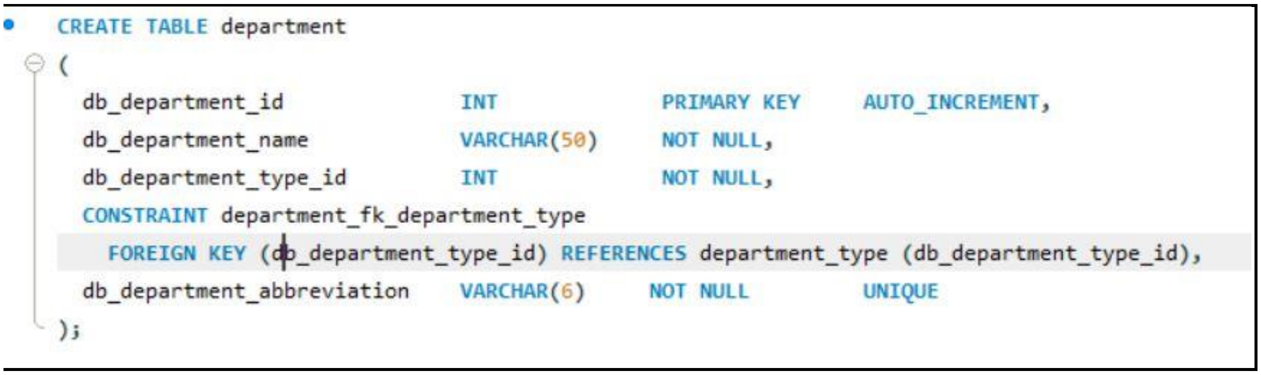
MYSQL LOCATION TABLE



```
CREATE TABLE location
(
  db_location_id       INT            PRIMARY KEY    AUTO_INCREMENT,
  db_location_name     VARCHAR(40)    NOT NULL
);
```

***Location* Table** created *(see above image)*

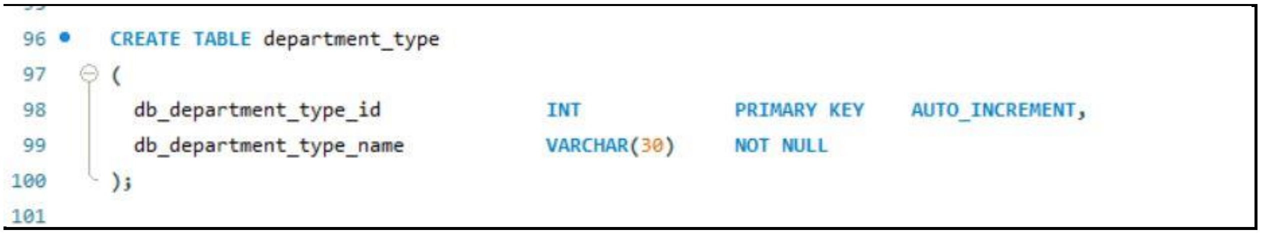
MYSQL DEPARTMENT TABLE



```sql
CREATE TABLE department
(
  db_department_id              INT            PRIMARY KEY     AUTO_INCREMENT,
  db_department_name            VARCHAR(50)    NOT NULL,
  db_department_type_id         INT            NOT NULL,
  CONSTRAINT department_fk_department_type
    FOREIGN KEY (db_department_type_id) REFERENCES department_type (db_department_type_id),
  db_department_abbreviation    VARCHAR(6)    NOT NULL         UNIQUE
);
```

***Department* Table** created *(see above image)*

MYSQL DEPARTMENT_TYPE TABLE



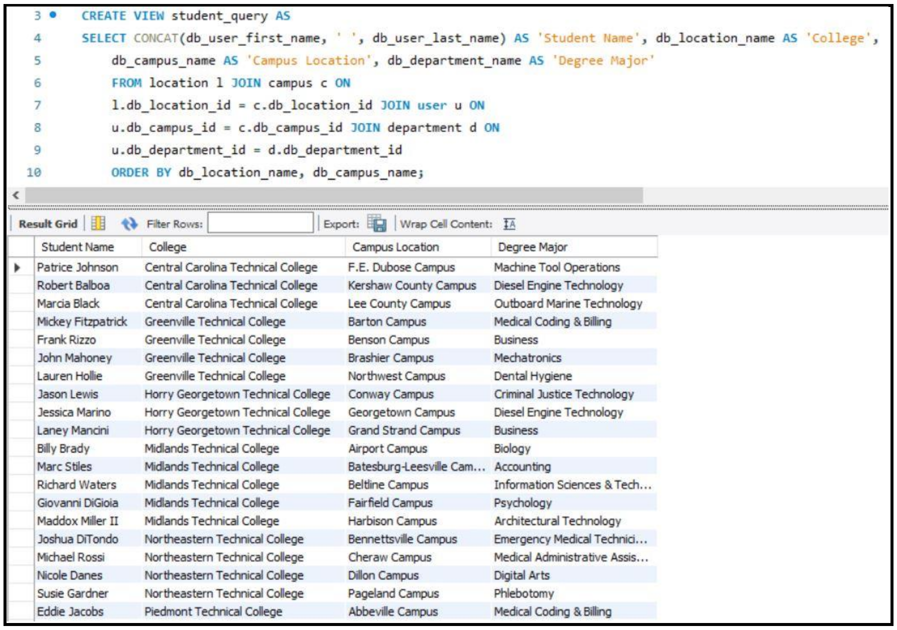
```sql
CREATE TABLE department_type
(
  db_department_type_id            INT            PRIMARY KEY    AUTO_INCREMENT,
  db_department_type_name          VARCHAR(30)    NOT NULL
);
```

***Department_Type* Table** created *(see above image)*

- The screenshots above display the syntax I used for creating the *college* database and all of the tables contained within. Notice the table structures I defined and the column data types that were specified as each table was created.
- You can see where the primary keys were defined as *auto-increment*, and the foreign keys that were properly established between the related tables.
- Once my tables were created, I inserted data in each one, while keeping the integrity of the relationships between the foreign keys. This can be seen in the my*sqldump* folder I have included with my exam files.

MYSQL WORKBENCH SCHOOL DATABASE VIEWS CREATED

MYSQL STUDENT_QUERY VIEW

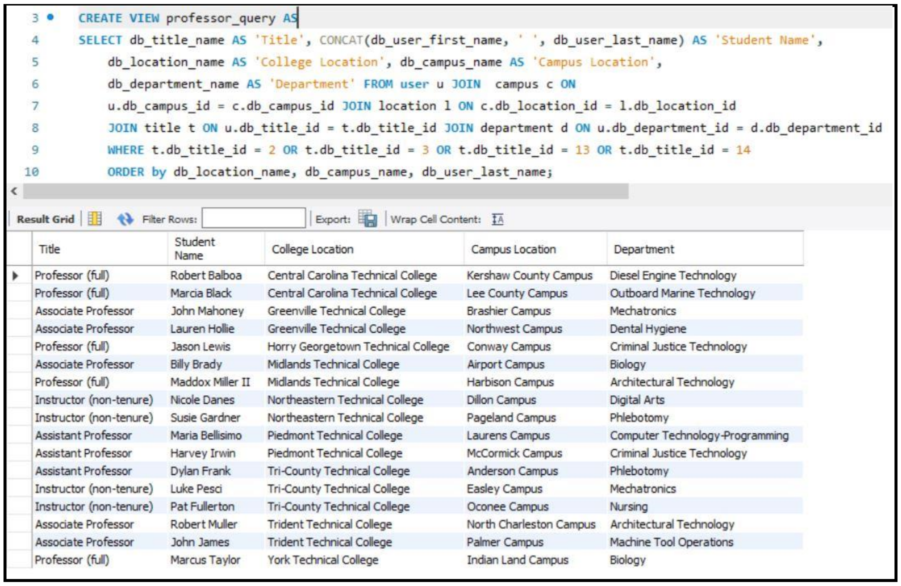


```sql
CREATE VIEW student_query AS
SELECT CONCAT(db_user_first_name, ' ', db_user_last_name) AS 'Student Name', db_location_name AS 'College',
    db_campus_name AS 'Campus Location', db_department_name AS 'Degree Major'
    FROM location l JOIN campus c ON
    l.db_location_id = c.db_location_id JOIN user u ON
    u.db_campus_id = c.db_campus_id JOIN department d ON
    u.db_department_id = d.db_department_id
    ORDER BY db_location_name, db_campus_name;
```

Student Name	College	Campus Location	Degree Major
Patrice Johnson	Central Carolina Technical College	F.E. Dubose Campus	Machine Tool Operations
Robert Balboa	Central Carolina Technical College	Kershaw County Campus	Diesel Engine Technology
Marcia Black	Central Carolina Technical College	Lee County Campus	Outboard Marine Technology
Mickey Fitzpatrick	Greenville Technical College	Barton Campus	Medical Coding & Billing
Frank Rizzo	Greenville Technical College	Benson Campus	Business
John Mahoney	Greenville Technical College	Brashier Campus	Mechatronics
Lauren Hollie	Greenville Technical College	Northwest Campus	Dental Hygiene
Jason Lewis	Horry Georgetown Technical College	Conway Campus	Criminal Justice Technology
Jessica Marino	Horry Georgetown Technical College	Georgetown Campus	Diesel Engine Technology
Laney Mancini	Horry Georgetown Technical College	Grand Strand Campus	Business
Billy Brady	Midlands Technical College	Airport Campus	Biology
Marc Stiles	Midlands Technical College	Batesburg-Leesville Cam...	Accounting
Richard Waters	Midlands Technical College	Beltline Campus	Information Sciences & Tech...
Giovanni DiGioia	Midlands Technical College	Fairfield Campus	Psychology
Maddox Miller II	Midlands Technical College	Harbison Campus	Architectural Technology
Joshua DiTondo	Northeastern Technical College	Bennettsville Campus	Emergency Medical Technici...
Michael Rossi	Northeastern Technical College	Cheraw Campus	Medical Administrative Assis...
Nicole Danes	Northeastern Technical College	Dillon Campus	Digital Arts
Susie Gardner	Northeastern Technical College	Pageland Campus	Phlebotomy
Eddie Jacobs	Piedmont Technical College	Abbeville Campus	Medical Coding & Billing

***Student_Query* View** created *(see above image)*

MYSQL PROFESSOR_QUERY VIEW

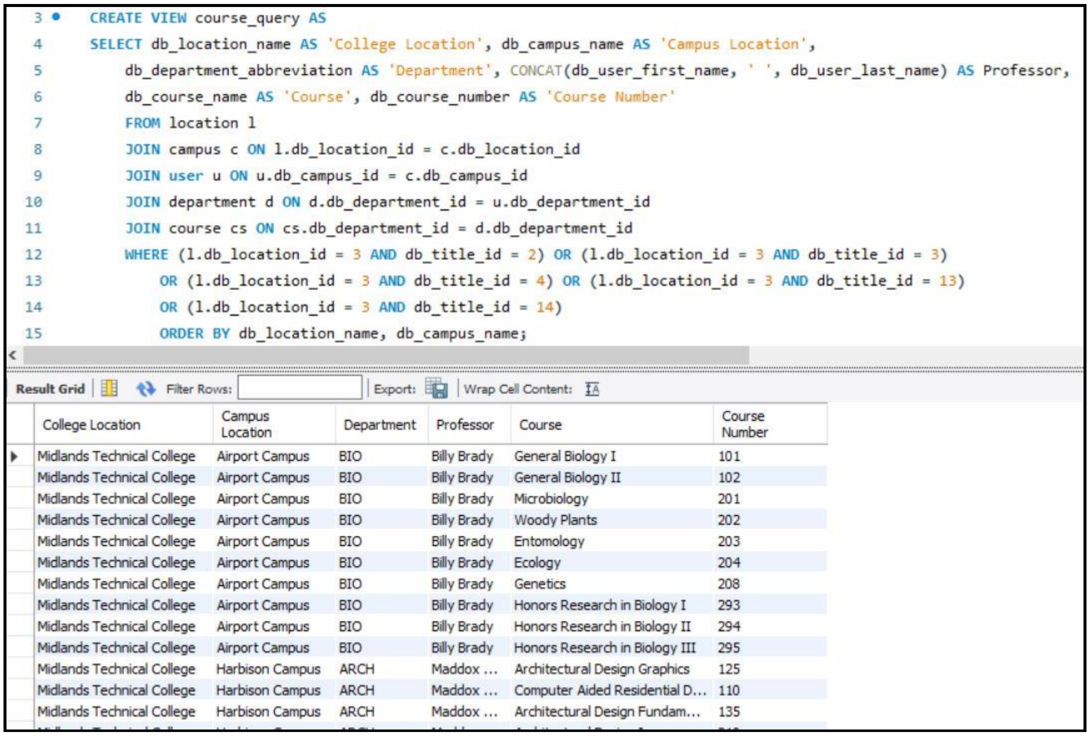


```sql
CREATE VIEW professor_query AS
SELECT db_title_name AS 'Title', CONCAT(db_user_first_name, ' ', db_user_last_name) AS 'Student Name',
    db_location_name AS 'College Location', db_campus_name AS 'Campus Location',
    db_department_name AS 'Department' FROM user u JOIN  campus c ON
    u.db_campus_id = c.db_campus_id JOIN location l ON c.db_location_id = l.db_location_id
    JOIN title t ON u.db_title_id = t.db_title_id JOIN department d ON u.db_department_id = d.db_department_id
    WHERE t.db_title_id = 2 OR t.db_title_id = 3 OR t.db_title_id = 13 OR t.db_title_id = 14
    ORDER by db_location_name, db_campus_name, db_user_last_name;
```

Title	Student Name	College Location	Campus Location	Department
Professor (full)	Robert Balboa	Central Carolina Technical College	Kershaw County Campus	Diesel Engine Technology
Professor (full)	Marcia Black	Central Carolina Technical College	Lee County Campus	Outboard Marine Technology
Associate Professor	John Mahoney	Greenville Technical College	Brashier Campus	Mechatronics
Associate Professor	Lauren Hollie	Greenville Technical College	Northwest Campus	Dental Hygiene
Professor (full)	Jason Lewis	Horry Georgetown Technical College	Conway Campus	Criminal Justice Technology
Associate Professor	Billy Brady	Midlands Technical College	Airport Campus	Biology
Professor (full)	Maddox Miller II	Midlands Technical College	Harbison Campus	Architectural Technology
Instructor (non-tenure)	Nicole Danes	Northeastern Technical College	Dillon Campus	Digital Arts
Instructor (non-tenure)	Susie Gardner	Northeastern Technical College	Pageland Campus	Phlebotomy
Assistant Professor	Maria Bellisimo	Piedmont Technical College	Laurens Campus	Computer Technology-Programming
Assistant Professor	Harvey Irwin	Piedmont Technical College	McCormick Campus	Criminal Justice Technology
Assistant Professor	Dylan Frank	Tri-County Technical College	Anderson Campus	Phlebotomy
Instructor (non-tenure)	Luke Pesci	Tri-County Technical College	Easley Campus	Mechatronics
Instructor (non-tenure)	Pat Fullerton	Tri-County Technical College	Oconee Campus	Nursing
Associate Professor	Robert Muller	Trident Technical College	North Charleston Campus	Architectural Technology
Associate Professor	John James	Trident Technical College	Palmer Campus	Machine Tool Operations
Professor (full)	Marcus Taylor	York Technical College	Indian Land Campus	Biology

***Professor_Query* View** created *(see above image)*

MYSQL COURSE_QUERY VIEW

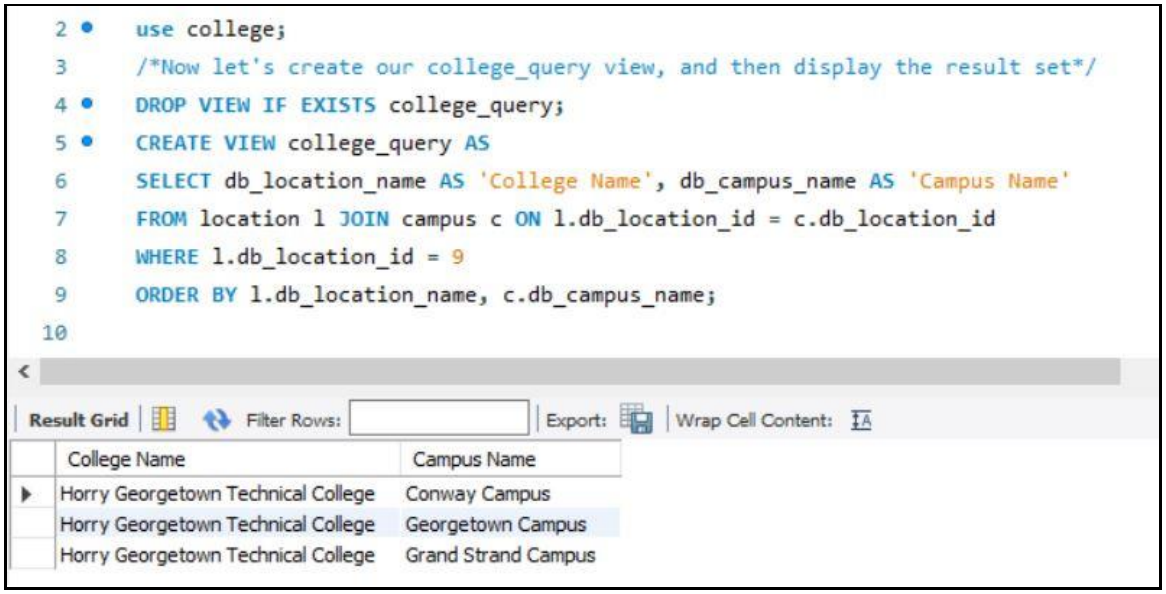


```sql
CREATE VIEW course_query AS
SELECT db_location_name AS 'College Location', db_campus_name AS 'Campus Location',
    db_department_abbreviation AS 'Department', CONCAT(db_user_first_name, ' ', db_user_last_name) AS Professor,
    db_course_name AS 'Course', db_course_number AS 'Course Number'
    FROM location l
    JOIN campus c ON l.db_location_id = c.db_location_id
    JOIN user u ON u.db_campus_id = c.db_campus_id
    JOIN department d ON d.db_department_id = u.db_department_id
    JOIN course cs ON cs.db_department_id = d.db_department_id
    WHERE (l.db_location_id = 3 AND db_title_id = 2) OR (l.db_location_id = 3 AND db_title_id = 3)
        OR (l.db_location_id = 3 AND db_title_id = 4) OR (l.db_location_id = 3 AND db_title_id = 13)
        OR (l.db_location_id = 3 AND db_title_id = 14)
        ORDER BY db_location_name, db_campus_name;
```

College Location	Campus Location	Department	Professor	Course	Course Number
Midlands Technical College	Airport Campus	BIO	Billy Brady	General Biology I	101
Midlands Technical College	Airport Campus	BIO	Billy Brady	General Biology II	102
Midlands Technical College	Airport Campus	BIO	Billy Brady	Microbiology	201
Midlands Technical College	Airport Campus	BIO	Billy Brady	Woody Plants	202
Midlands Technical College	Airport Campus	BIO	Billy Brady	Entomology	203
Midlands Technical College	Airport Campus	BIO	Billy Brady	Ecology	204
Midlands Technical College	Airport Campus	BIO	Billy Brady	Genetics	208
Midlands Technical College	Airport Campus	BIO	Billy Brady	Honors Research in Biology I	293
Midlands Technical College	Airport Campus	BIO	Billy Brady	Honors Research in Biology II	294
Midlands Technical College	Airport Campus	BIO	Billy Brady	Honors Research in Biology III	295
Midlands Technical College	Harbison Campus	ARCH	Maddox ...	Architectural Design Graphics	125
Midlands Technical College	Harbison Campus	ARCH	Maddox ...	Computer Aided Residential D...	110
Midlands Technical College	Harbison Campus	ARCH	Maddox ...	Architectural Design Fundam...	135

***Course_Query* View** created *(see above image)*

MYSQL COLLEGE_QUERY VIEW

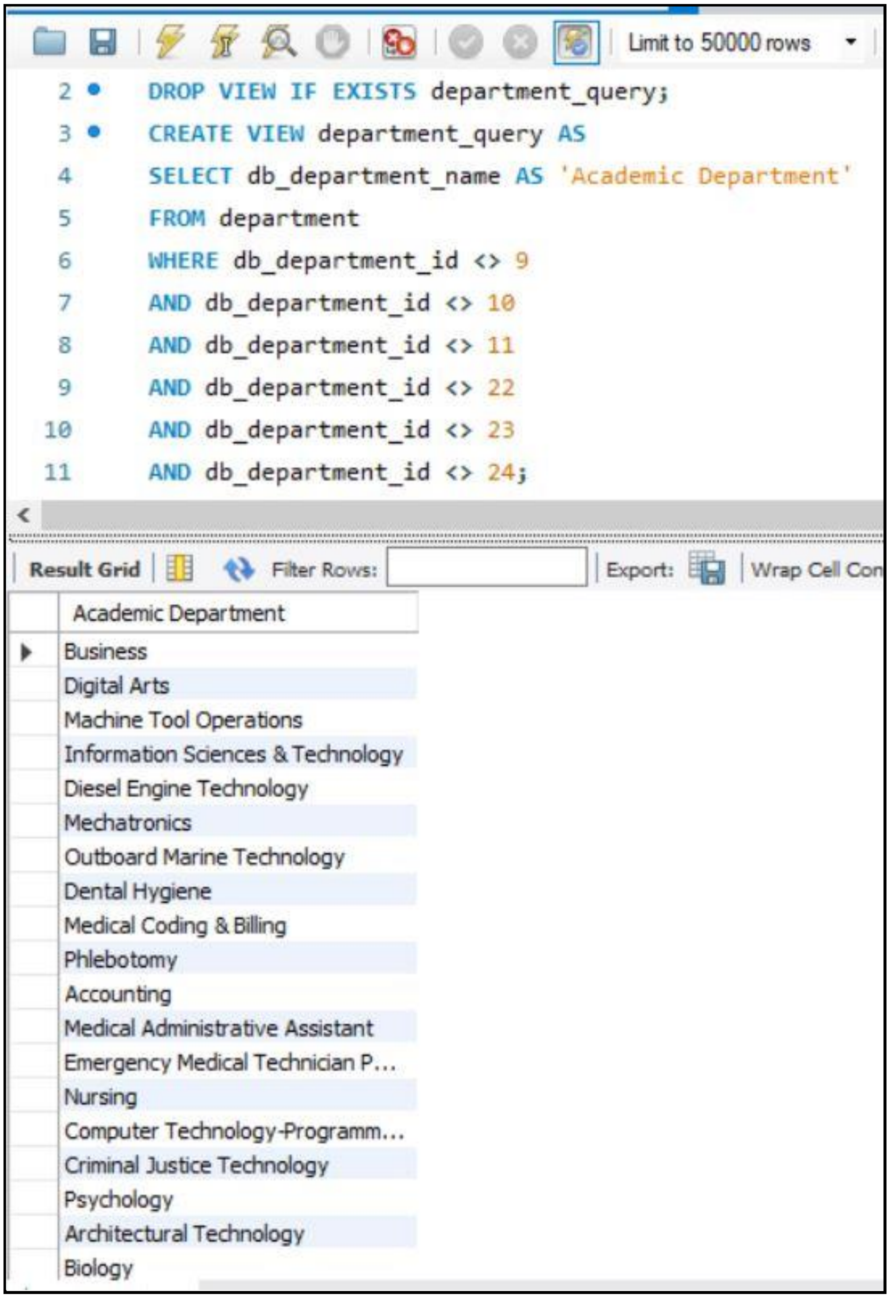


```sql
use college;
/*Now let's create our college_query view, and then display the result set*/
DROP VIEW IF EXISTS college_query;
CREATE VIEW college_query AS
SELECT db_location_name AS 'College Name', db_campus_name AS 'Campus Name'
FROM location l JOIN campus c ON l.db_location_id = c.db_location_id
WHERE l.db_location_id = 9
ORDER BY l.db_location_name, c.db_campus_name;
```

College Name	Campus Name
Horry Georgetown Technical College	Conway Campus
Horry Georgetown Technical College	Georgetown Campus
Horry Georgetown Technical College	Grand Strand Campus

***College_Query* View** created *(see above image)*

MYSQL DEPARTMENT_QUERY VIEW



```sql
DROP VIEW IF EXISTS department_query;
CREATE VIEW department_query AS
SELECT db_department_name AS 'Academic Department'
FROM department
WHERE db_department_id <> 9
AND db_department_id <> 10
AND db_department_id <> 11
AND db_department_id <> 22
AND db_department_id <> 23
AND db_department_id <> 24;
```

Academic Department
Business
Digital Arts
Machine Tool Operations
Information Sciences & Technology
Diesel Engine Technology
Mechatronics
Outboard Marine Technology
Dental Hygiene
Medical Coding & Billing
Phlebotomy
Accounting
Medical Administrative Assistant
Emergency Medical Technician P...
Nursing
Computer Technology-Programm...
Criminal Justice Technology
Psychology
Architectural Technology
Biology

***Department_Query* View** created *(see above image)*

- If you look at the screenshots above, you can see the syntax and result tables for the views I created in MySQL Workbench. The actual SQL view files can all be found in the *mysqldump* folder I included within the exam file folder, named *college_routines*.
- Items contained within *Final Exam* folder in addition to this file include:
  - *Lamelza_College_Database* Access file
  - *mysqldump* folder (contains all MySQL Workbench SQL files)