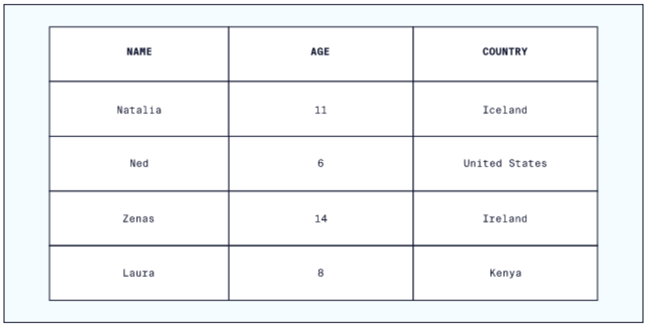
CSC 4402 Honors Option Report (Spring 2023)

Maggie Stewart

For this honors option assignment, I was tasked to choose a different database software other than MySQL to solve the questions from Assignment 2. After researching different database softwares like SQLite and PostgreSQL, I decided to use Microsoft SQL Server to complete the assignment questions.

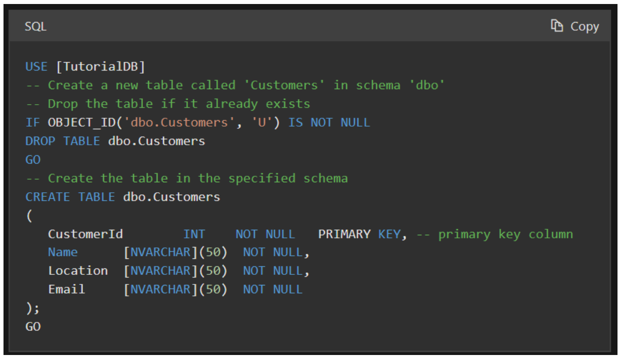
Although SQL Server and MySQL are different database softwares, there are many similarities between the two. Both MySQL and SQL Server are relational database management systems and use Structured Query Language (SQL) to access data in the database tables (Code Academy). Both softwares have high performance in regards to how quickly it returns the data queries and use a tabular form (left) to store data in rows and columns (UpWork). In my experience with using both MySQL and SQL Server, the commands like *'SELECT'*, *'FROM'*, and *'ORDER BY'*, among others, have the same function when querying the data, though syntax differs.



NAME	AGE	COUNTRY
Natalia	11	Iceland
Ned	6	United States
Zenas	14	Ireland
Laura	8	Kenya

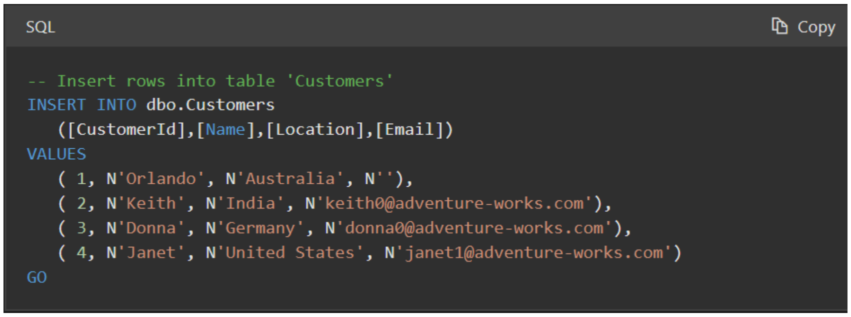
There are also a few differences between MySQL and SQL Server. Since Microsoft owns SQL Server, the code is closed source and typically used by large businesses (Code Academy). MySQL, on the other hand, is an open source SQL database that is easy to use and inexpensive.

Based on the similarities and differences between MySQL and SQL Server, there are a few advantages and disadvantages to using each software that depend on the users' purpose. For MySQL, since it is an open-source software that is easy to use, easy to set up, and has many forums to help with various issues, I think this software is good for beginners who are trying to gain more experience writing queries and understanding the SQL language. For SQL Server, it is a closed-source software owned by Microsoft and is used more in large-scale industry practices. It might be harder to gain access to a license or set up the environment for personal use. The company where I am currently an intern uses Microsoft SQL Server, so I think this software is better for developers who are looking to gain industry experience in an internship or full-time position.



```sql
USE [TutorialDB]
-- Create a new table called 'Customers' in schema 'dbo'
-- Drop the table if it already exists
IF OBJECT_ID('dbo.Customers', 'U') IS NOT NULL
DROP TABLE dbo.Customers
GO
-- Create the table in the specified schema
CREATE TABLE dbo.Customers
(
   CustomerId        INT    NOT NULL   PRIMARY KEY, -- primary key column
   Name      [NVARCHAR](50)  NOT NULL,
   Location  [NVARCHAR](50)  NOT NULL,
   Email     [NVARCHAR](50)  NOT NULL
);
GO
```

In my experience with SQL Server, it was a bit difficult to set-up SQL Server on my personal computer. It required a server to be connected to use the software, including the server type, name, and type of authentication. It took a bit of research to determine how to properly set up this connection for personal use since MySQL did not require those parameters in the initial setup. MySQL only required the connection name, hostname, and port number, and we had guidance in class documentation available. After the initial setup of the environment, there were a few syntax differences, specifically to create the table (above) and insert rows into the table (below). I think it was very interesting to see and use different syntax for SQL since the respective commands in both SQL Server and MySQL express the same concept. It did take some practice and patience to get the first question of the assignment completed in SQL Server because it was difficult to tell if it was a syntax error or if the initial setup failed in some aspect. This delay was primarily due to my lack of experience

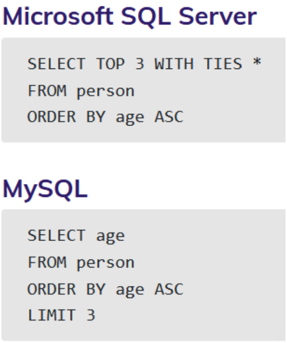


```sql
-- Insert rows into table 'Customers'
INSERT INTO dbo.Customers
   ([CustomerId],[Name],[Location],[Email])
VALUES
   ( 1, N'Orlando', N'Australia', N''),
   ( 2, N'Keith', N'India', N'keith0@adventure-works.com'),
   ( 3, N'Donna', N'Germany', N'donna0@adventure-works.com'),
   ( 4, N'Janet', N'United States', N'janet1@adventure-works.com')
GO
```

with SQL Server, but after finding a few resources from Microsoft shown in the photos above, I found it was much easier to complete the rest of the question and the assignment. However, there was one issue I found that I was not able to fully resolve. You could not rename or create a new database in SQL Server like you could in MySQL or at least this was not an option for the free/personal use version I used for SQL Server. Therefore, I used one of the default databases called *‘model’* in SQL Server rather than using the database named ‘*assignment2’* as stated in the original assignment.

The SQL Server syntax for normal commands with *‘SELECT’* and *‘WHERE’* was the same as MySQL for all except Question #4 where we set the initial value of the row to be 100. In that case, I used the alternative solution from Assignment 2 using *‘100 + 5 * (ROW_NUMBER() OVER (ORDER BY sect.building, sect.room_number) - 1)’* instead of *‘@row_num:=100’* and *‘(@row_num:=@row_num+5)’* since setting the row_num to 100 in that manner was invalid syntax in SQL Server.

In a different article, I noticed that there were other syntax differences when queries would be more particular. In this case (right), to select the three smallest values, which is essentially the three youngest ages, from the person table, MySQL uses *‘LIMIT 3’* while SQL Server uses *‘TOP 3 WITH TIES *’*. Again, the functionality of each query is the same, but it all comes down to syntactical differences. Similarly in other programming languages that are not SQL based, the differences in defining a variable or a function can be particular to a programming language design, which is based on a variety of factors like readability, writability, reliability, and cost.



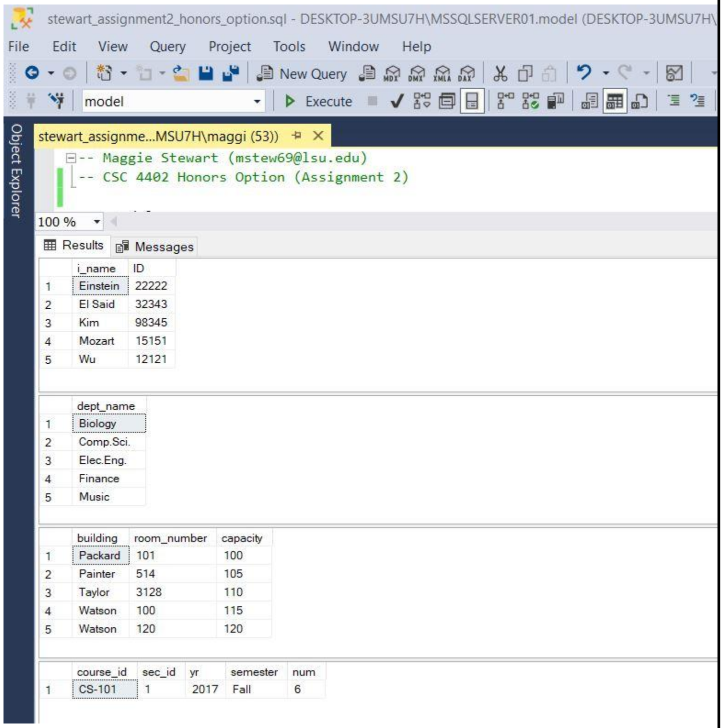
Microsoft SQL Server

```
SELECT TOP 3 WITH TIES *
FROM person
ORDER BY age ASC
```

MySQL

```
SELECT age
FROM person
ORDER BY age ASC
LIMIT 3
```

SQL Server Output Questions 2-5 (top to bottom)


stewart_assignment2_honors_option.sql - DESKTOP-3UMSU7H\MSSQLSERVER01.model (DESKTOP-3UMSU7H\

```
-- Maggie Stewart (mstew69@lsu.edu)
-- CSC 4402 Honors Option (Assignment 2)
```

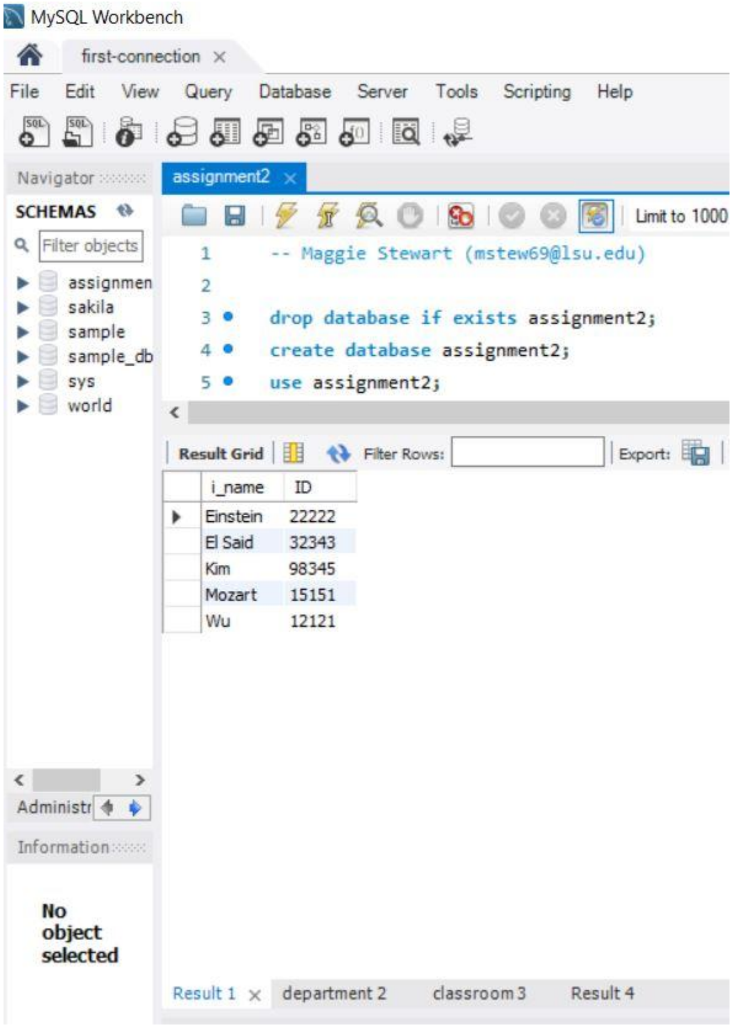
```
   i_name    ID
1  Einstein  22222
2  El Said   32343
3  Kim       98345
4  Mozart    15151
5  Wu        12121
```

```
   dept_name
1  Biology
2  Comp.Sci.
3  Elec.Eng.
4  Finance
5  Music
```

```
   building  room_number  capacity
1  Packard   101          100
2  Painter   514          105
3  Taylor    3128         110
4  Watson    100          115
5  Watson    120          120
```

```
   course_id  sec_id  yr    semester  num
1  CS-101     1       2017  Fall      6
```

MySQL Output Question #2

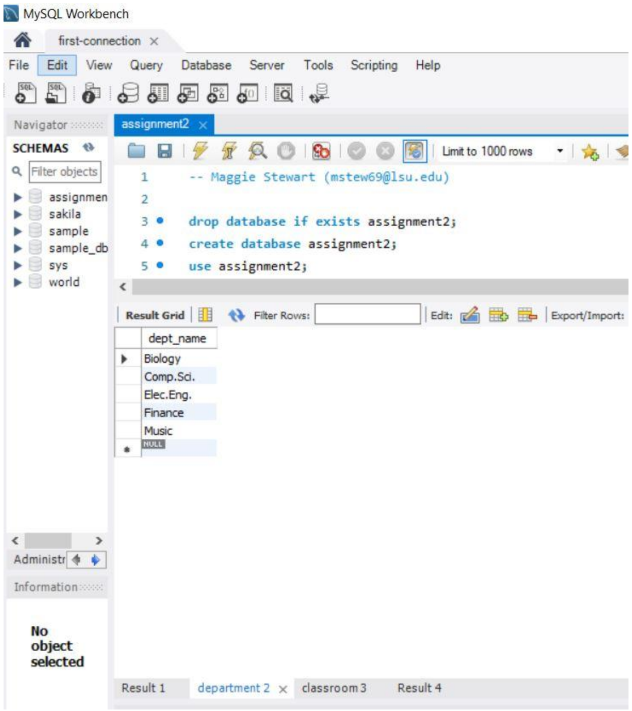


```
-- Maggie Stewart (mstew69@lsu.edu)

drop database if exists assignment2;
create database assignment2;
use assignment2;
```

```
  i_name    ID
▶ Einstein  22222
  El Said   32343
  Kim       98345
  Mozart    15151
  Wu        12121
```

MySQL Output Question #3

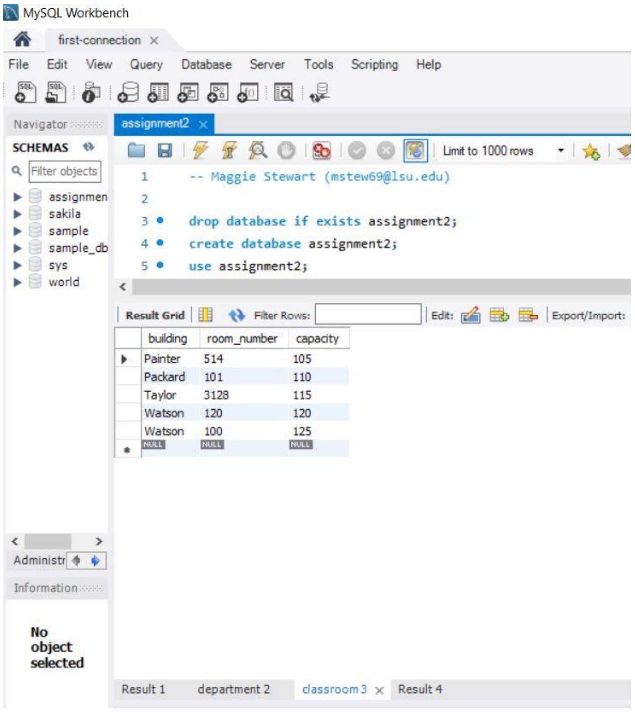


```
-- Maggie Stewart (mstew69@lsu.edu)

drop database if exists assignment2;
create database assignment2;
use assignment2;
```

```
dept_name
Biology
Comp.Sci.
Elec.Eng.
Finance
Music
NULL
```

MySQL Output Question #4

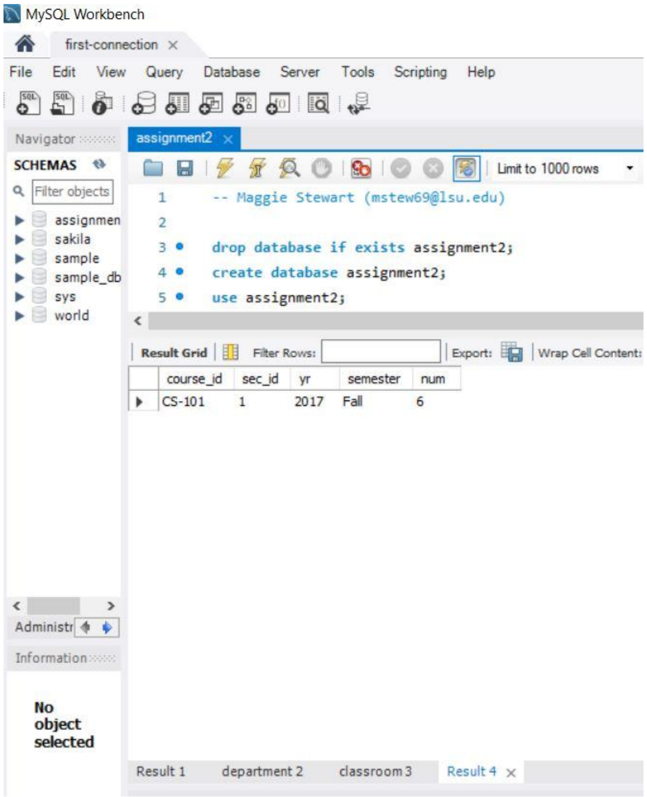


```
-- Maggie Stewart (mstew69@lsu.edu)

drop database if exists assignment2;
create database assignment2;
use assignment2;
```

```
building  room_number  capacity
Painter   514          105
Packard   101          110
Taylor    3128         115
Watson    120          120
Watson    100          125
NULL      NULL         NULL
```

MySQL Output Question #5



```
-- Maggie Stewart (mstew69@lsu.edu)

drop database if exists assignment2;
create database assignment2;
use assignment2;
```

```
course_id  sec_id  yr    semester  num
CS-101     1       2017  Fall      6
```

MySQL Code

```
-- Maggie Stewart

drop database if exists assignment2;
create database assignment2;
use assignment2;

-- #1 creating the 11 tables as defined in the assignment, adding values into those tables, and declaring
primary/foreign keys
create table department (
   dept_name varchar(20),
          building varchar(20),
          budget numeric(8,0),
          primary key (dept_name)
);

insert into department values ('Biology','Watson',90000);
insert into department values('Comp.Sci.','Taylor',100000);
insert into department values ('Elec.Eng.','Taylor',85000);
insert into department values ('Finance','Painter',120000);
insert into department values ('History','Painter',50000);
insert into department values ('Music','Packard',80000);
insert into department values ('Physics','Watson',70000);

create table instructor (
   ID char(5),
          i_name varchar(20),
          dept_name varchar(20),
          salary numeric(8,0),
          primary key(ID),
          foreign key (dept_name) references department(dept_name)
);

insert into instructor values ('10101','Srinivasan','Comp.Sci.',65000);
insert into instructor values ('12121','Wu','Finance',90000);
insert into instructor values ('15151','Mozart','Music',40000);
insert into instructor values ('22222','Einstein','Physics',95000);
insert into instructor values ('32343','El Said','History',60000);
insert into instructor values ('45565','Katz','Comp.Sci.',75000);
insert into instructor values ('58583','Califieri','History',62000);
insert into instructor values ('76543','Singh','Finance',80000);
insert into instructor values ('76766','Crick','Biology',72000);
insert into instructor values ('83821','Brandt','Comp.Sci.',92000);
insert into instructor values ('98345','Kim','Elec.Eng.',80000);

create table classroom (
   building varchar(20),
          room_number numeric(8,0),
		capacity numeric(8,0),
		primary key (building, room_number)
		-- foreign key (building) references department(building)
);

create table course (
   course_id varchar(20),
		title varchar(40),
		dept_name varchar(20),
		credits numeric(2,0),
		primary key (course_id),
		foreign key(dept_name) references department(dept_name)
);

insert into course values ('BIO-101','Intro. to Biology','Biology',4);
insert into course values ('BIO-301','Genetics','Biology',4);
insert into course values ('BIO-399','Computational Biology','Biology',3);
insert into course values ('CS-101','Intro. to Computer Science','Comp.Sci.',5);
insert into course values ('CS-190','Game Design','Comp.Sci.',4);
insert into course values ('CS-315','Robotics','Comp.Sci.',3);
insert into course values ('CS-319','Image Processing','Comp.Sci.',3);
insert into course values ('CS-347','Database System Concepts','Comp.Sci.',3);
insert into course values ('EE-181','Intro. to Digital Systems','Elec.Eng.',3);
insert into course values ('FIN-201','Investment Banking','Finance',3);
insert into course values ('HIS-351','World History','History',3);
insert into course values ('MU-199','Music Video Production','Music',3);
insert into course values ('PHY-101','Physical Principles','Physics',4);

create table section (
   course_id varchar(20),
		sec_id numeric(2,0),
		semester varchar(20),
		yr numeric(4,0),
		building varchar(20),
		room_number numeric(8,0),
		time_slot_id char(1),
		primary key (course_id, sec_id, semester, yr),
		foreign key (course_id) references course(course_id)
		-- omit foreign key for classrooms
);

insert into section values ('BIO-101',1,'Summer',2017,'Painter',514,'B');
insert into section values ('BIO-301',1,'Summer',2018,'Painter',514,'A');
insert into section values ('CS-101',1,'Fall',2017,'Packard',101,'H');
insert into section values ('CS-101',1,'Spring',2018,'Packard',101,'F');
insert into section values ('CS-190',1,'Spring',2017,'Taylor',3128,'E');
insert into section values ('CS-190',2,'Spring',2017,'Taylor',3128,'A');
insert into section values ('CS-315',1,'Spring',2018,'Watson',120,'D');
insert into section values ('CS-319',1,'Spring',2018,'Watson',100,'B');
insert into section values ('CS-319',2,'Spring',2018,'Taylor',3128,'C');
insert into section values ('CS-347',1,'Fall',2017,'Taylor',3128,'A');
insert into section values ('EE-181',1,'Spring',2017,'Taylor',3128,'C');
insert into section values ('FIN-201',1,'Spring',2018,'Packard',101,'B');
insert into section values ('HIS-351',1,'Spring',2018,'Painter',514,'C');
insert into section values ('MU-199',1,'Spring',2018,'Packard',101,'D');
insert into section values ('PHY-101',1,'Fall',2017,'Watson',100,'A');

create table teaches (
  ID char(5),
  course_id varchar(20),
  sec_id numeric(2,0),
  semester varchar(20),
  yr numeric(4,0),
  primary key (ID, course_id, sec_id, semester, yr),
  foreign key (course_id, sec_id, semester, yr) references section(course_id, sec_id, semester, yr)
);

insert into teaches values('10101','CS-101',1,'Fall',2017);
insert into teaches values('10101','CS-315',1,'Spring',2018);
insert into teaches values('10101','CS-347',1,'Fall',2017);
insert into teaches values('12121','FIN-201',1,'Spring',2018);
insert into teaches values('15151','MU-199',1,'Spring',2018);
insert into teaches values('22222','PHY-101',1,'Fall',2017);
insert into teaches values('32343','HIS-351',1,'Spring',2018);
insert into teaches values('45565','CS-101',1,'Spring',2018);
insert into teaches values('45565','CS-319',1,'Spring',2018);
insert into teaches values('76766','BIO-101',1,'Summer',2017);
insert into teaches values('76766','BIO-301',1,'Summer',2018);
insert into teaches values('83821','CS-190',1,'Spring',2017);
insert into teaches values('83821','CS-190',2,'Spring',2017);
insert into teaches values('83821','CS-319',2,'Spring',2018);
insert into teaches values('98345','EE-181',1,'Spring',2017);

create table student (
  ID char(5),
  s_name varchar(20),
  dept_name varchar(20),
  tot_cred numeric(8,0),
  primary key (ID),
  foreign key (dept_name) references department(dept_name)
);

insert into student values('00128','Zhang','Comp.Sci.',102);
insert into student values('12345','Shankar','Comp.Sci.',32);
insert into student values('19991','Brandt','History',80);
insert into student values('23121','Chavez','Finance',110);
insert into student values('44553','Peltier','Physics',56);
insert into student values('45678','Levy','Physics',46);
insert into student values('54321','Williams','Comp.Sci.',54);
insert into student values('55739','Sanchez','Music',38);
insert into student values('70557','Snow','Physics',0);
insert into student values('76543','Brown','Comp.Sci.',58);
insert into student values('76653','Aoi','Elec.Eng.',60);
insert into student values('98765','Bourikas','Elec.Eng.',98);
insert into student values('98988','Tanaka','Biology',120);

create table takes (
  ID char(5),
  course_id varchar(20),
  sec_id numeric(2,0),
  semester varchar(20),
  yr numeric(4,0),
  grade varchar(2),
  primary key (ID, course_id, sec_id, semester, yr),
  foreign key (course_id, sec_id, semester, yr) references section(course_id, sec_id, semester, yr)
);

insert into takes values('00128','CS-101',1,'Fall',2017,'A');
insert into takes values('00128','CS-347',1,'Fall',2017,'A-');
insert into takes values('12345','CS-101',1,'Fall',2017,'C');
insert into takes values('12345','CS-190',2,'Spring',2017,'A');
insert into takes values('12345','CS-315',1,'Spring',2018,'A');
insert into takes values('12345','CS-347',1,'Fall',2017,'A');
insert into takes values('19991','HIS-351',1,'Spring',2018,'B');
insert into takes values('23121','FIN-201',1,'Spring',2018,'C+');
insert into takes values('44553','PHY-101',1,'Fall',2017,'B-');
insert into takes values('45678','CS-101',1,'Fall',2017,'F');
insert into takes values('45678','CS-101',1,'Spring',2018,'B+');
insert into takes values('45678','CS-319',1,'Spring',2018,'B');
insert into takes values('54321','CS-101',1,'Fall',2017,'A-');
insert into takes values('54321','CS-190',2,'Spring',2017,'B+');
insert into takes values('55739','MU-199',1,'Spring',2018,'A-');
insert into takes values('76543','CS-101',1,'Fall',2017,'A');
insert into takes values('76543','CS-319',2,'Spring',2018,'A');
insert into takes values('76653','EE-181',1,'Spring',2017,'C');
insert into takes values('98765','CS-101',1,'Fall',2017,'C-');
insert into takes values('98765','CS-315',1,'Spring',2018,'B');
insert into takes values('98988','BIO-101',1,'Summer',2017,'A');
insert into takes values('98988','BIO-301',1,'Summer',2018,'');

create table advisor (
  s_ID varchar(20),
  i_ID varchar(20),
  primary key (s_ID),
  foreign key (s_ID) references student(ID),
  foreign key (i_ID) references instructor(ID)
);
```

```
-- #2 find names & ids of instructors who teach every course in their department; order by instructor name
-- (select distinct i_name, ID from instructor) INTERSECT (select distinct i_name, ID from course) order by i_name
-- asc;
select i_name, ID
    from instructor as I
            where not exists (
             (select C.course_id from course as C where C.dept_name = I.dept_name)
            except
            (select T.course_id from teaches as T where I.ID = T.ID)
    )
            order by i_name asc;

-- #3 find names of departments whose total budget is higher than that of Physics; order alphabetically
select dept_name
    from department
            where budget > (select budget from department where dept_name = 'Physics')
            order by dept_name asc;

-- #4 insert values to the classroom relation using the section relation; start capacity value as 100 and increase by 5
for each insertion
set @row_num:=100;
insert into classroom
    select distinct
             building, room_number,
             (select (@row_num:=@row_num+5) as capacity)
    from section;

select * from classroom order by capacity asc;

-- #5 create relation takes (see above); find sections with maximum enrollment; result should be courseid, secid,
year, semester, num

with enrollment as (
    select
             course_id, sec_id, yr, semester,
            COUNT(*) as num from takes
            group by course_id, sec_id, yr, semester
)     select section.course_id, section.sec_id, section.yr, section.semester, enrollment.num
            from section
            join enrollment on
                    section.course_id = enrollment.course_id
            and section.sec_id = enrollment.sec_id
            and section.semester = enrollment.semester
            and section.yr = enrollment.yr
            where enrollment.num =
                    (select MAX(num) from enrollment)
            order by section.course_id, section.sec_id, section.yr, section.semester, enrollment.num desc;
```

SQL Server Code

```
-- Maggie Stewart
-- CSC 4402 Honors Option (Assignment 2)

use model;

-- #1 creating the 11 tables as defined in the assignment, adding values into those tables, and declaring
primary/foreign keys
IF OBJECT_ID('dbo.department', 'U') IS NOT NULL
DROP TABLE dbo.department
GO
CREATE TABLE dbo.department
(
   dept_name varchar(20), --PRIMARY KEY,
           building varchar(20), -- NOT NULL,
           budget numeric(8,0), -- NOT NULL
);
GO

INSERT INTO dbo.department
   ([dept_name], [building], [budget])
VALUES
   ('Biology','Watson',90000),
   ('Comp.Sci.','Taylor',100000),
   ('Elec.Eng.','Taylor',85000),
   ('Finance','Painter',120000),
   ('History','Painter',50000),
   ('Music','Packard',80000),
   ('Physics','Watson',70000)
GO

IF OBJECT_ID('dbo.instructor', 'U') IS NOT NULL
DROP TABLE dbo.instructor
GO
CREATE TABLE instructor
(
   ID char(5), --NOT NULL PRIMARY KEY,
           i_name varchar(20), -- NOT NULL,
           dept_name varchar(20), -- NOT NULL, --FOREIGN KEY,
           salary numeric(8,0), -- NOT NULL
)
GO

INSERT INTO dbo.instructor
   ([ID], [i_name], [dept_name], [salary])
VALUES
   ('10101','Srinivasan','Comp.Sci.',65000),
   ('12121','Wu','Finance',90000),
   ('15151','Mozart','Music',40000),
   ('22222','Einstein','Physics',95000),
   ('32343','El Said','History',60000),
   ('45565','Katz','Comp.Sci.',75000),
   ('58583','Califieri','History',62000),
   ('76543','Singh','Finance',80000),
   ('76766','Crick','Biology',72000),
   ('83821','Brandt','Comp.Sci.',92000),
   ('98345','Kim','Elec.Eng.',80000)
GO

IF OBJECT_ID('dbo.classroom', 'U') IS NOT NULL
DROP TABLE dbo.classroom
GO
CREATE TABLE classroom
(
   building varchar(20),
   room_number numeric(8,0),
   capacity numeric(8,0),
)
GO

IF OBJECT_ID('dbo.course', 'U') IS NOT NULL
DROP TABLE dbo.course
GO
CREATE TABLE course (
   course_id varchar(20),
          title varchar(40),
          dept_name varchar(20),
          credits numeric(2,0)
)
GO

INSERT INTO dbo.course ([course_id], [title], [dept_name], [credits])
VALUES
('BIO-101','Intro. to Biology','Biology',4),
('BIO-301','Genetics','Biology',4),
('BIO-399','Computational Biology','Biology',3),
('CS-101','Intro. to Computer Science','Comp.Sci.',5),
('CS-190','Game Design','Comp.Sci.',4),
('CS-315','Robotics','Comp.Sci.',3),
('CS-319','Image Processing','Comp.Sci.',3),
('CS-347','Database System Concepts','Comp.Sci.',3),
('EE-181','Intro. to Digital Systems','Elec.Eng.',3),
('FIN-201','Investment Banking','Finance',3),
('HIS-351','World History','History',3),
('MU-199','Music Video Production','Music',3),
('PHY-101','Physical Principles','Physics',4)
```

```
GO

IF OBJECT_ID('dbo.section', 'U') IS NOT NULL
DROP TABLE dbo.section
GO
CREATE TABLE section (
    course_id varchar(20),
              sec_id numeric(2,0),
              semester varchar(20),
              yr numeric(4,0),
              building varchar(20),
              room_number numeric(8,0),
              time_slot_id char(1)
)
GO

INSERT INTO dbo.section ([course_id],[sec_id],[semester],[yr],[building],[room_number],[time_slot_id])
VALUES
('BIO-101',1,'Summer',2017,'Painter',514,'B'),
('BIO-301',1,'Summer',2018,'Painter',514,'A'),
('CS-101',1,'Fall',2017,'Packard',101,'H'),
('CS-101',1,'Spring',2018,'Packard',101,'F'),
('CS-190',1,'Spring',2017,'Taylor',3128,'E'),
('CS-190',2,'Spring',2017,'Taylor',3128,'A'),
('CS-315',1,'Spring',2018,'Watson',120,'D'),
('CS-319',1,'Spring',2018,'Watson',100,'B'),
('CS-319',2,'Spring',2018,'Taylor',3128,'C'),
('CS-347',1,'Fall',2017,'Taylor',3128,'A'),
('EE-181',1,'Spring',2017,'Taylor',3128,'C'),
('FIN-201',1,'Spring',2018,'Packard',101,'B'),
('HIS-351',1,'Spring',2018,'Painter',514,'C'),
('MU-199',1,'Spring',2018,'Packard',101,'D'),
('PHY-101',1,'Fall',2017,'Watson',100,'A')
GO

IF OBJECT_ID('dbo.teaches', 'U') IS NOT NULL
DROP TABLE dbo.teaches
GO
CREATE TABLE teaches (
  ID char(5),
  course_id varchar(20),
  sec_id numeric(2,0),
  semester varchar(20),
  yr numeric(4,0)
)
GO

INSERT INTO dbo.teaches
([ID],[course_id],[sec_id],[semester],[yr])
```

```
VALUES
('10101','CS-101',1,'Fall',2017),
('10101','CS-315',1,'Spring',2018),
('10101','CS-347',1,'Fall',2017),
('12121','FIN-201',1,'Spring',2018),
('15151','MU-199',1,'Spring',2018),
('22222','PHY-101',1,'Fall',2017),
('32343','HIS-351',1,'Spring',2018),
('45565','CS-101',1,'Spring',2018),
('45565','CS-319',1,'Spring',2018),
('76766','BIO-101',1,'Summer',2017),
('76766','BIO-301',1,'Summer',2018),
('83821','CS-190',1,'Spring',2017),
('83821','CS-190',2,'Spring',2017),
('83821','CS-319',2,'Spring',2018),
('98345','EE-181',1,'Spring',2017)

GO

IF OBJECT_ID('dbo.student', 'U') IS NOT NULL
DROP TABLE dbo.student
GO
CREATE TABLE student (
  ID char(5),
  s_name varchar(20),
  dept_name varchar(20),
  tot_cred numeric(8,0)
)
GO

INSERT INTO dbo.student
           ([ID],[s_name],[dept_name],[tot_cred])
VALUES
('00128','Zhang','Comp.Sci.',102),
('12345','Shankar','Comp.Sci.',32),
('19991','Brandt','History',80),
('23121','Chavez','Finance',110),
('44553','Peltier','Physics',56),
('45678','Levy','Physics',46),
('54321','Williams','Comp.Sci.',54),
('55739','Sanchez','Music',38),
('70557','Snow','Physics',0),
('76543','Brown','Comp.Sci.',58),
('76653','Aoi','Elec.Eng.',60),
('98765','Bourikas','Elec.Eng.',98),
('98988','Tanaka','Biology',120)
GO

IF OBJECT_ID('dbo.takes', 'U') IS NOT NULL
```

```
DROP TABLE dbo.takes
GO
CREATE TABLE takes (
  ID char(5),
  course_id varchar(20),
  sec_id numeric(2,0),
  semester varchar(20),
  yr numeric(4,0),
  grade varchar(2)
)
GO

INSERT INTO dbo.takes
([ID],[course_id],[sec_id],[semester],[yr],[grade])

VALUES
('00128','CS-101',1,'Fall',2017,'A'),
('00128','CS-347',1,'Fall',2017,'A-'),
('12345','CS-101',1,'Fall',2017,'C'),
('12345','CS-190',2,'Spring',2017,'A'),
('12345','CS-315',1,'Spring',2018,'A'),
('12345','CS-347',1,'Fall',2017,'A'),
('19991','HIS-351',1,'Spring',2018,'B'),
('23121','FIN-201',1,'Spring',2018,'C+'),
('44553','PHY-101',1,'Fall',2017,'B-'),
('45678','CS-101',1,'Fall',2017,'F'),
('45678','CS-101',1,'Spring',2018,'B+'),
('45678','CS-319',1,'Spring',2018,'B'),
('54321','CS-101',1,'Fall',2017,'A-'),
('54321','CS-190',2,'Spring',2017,'B+'),
('55739','MU-199',1,'Spring',2018,'A-'),
('76543','CS-101',1,'Fall',2017,'A'),
('76543','CS-319',2,'Spring',2018,'A'),
('76653','EE-181',1,'Spring',2017,'C'),
('98765','CS-101',1,'Fall',2017,'C-'),
('98765','CS-315',1,'Spring',2018,'B'),
('98988','BIO-101',1,'Summer',2017,'A'),
('98988','BIO-301',1,'Summer',2018,'')
GO

IF OBJECT_ID('dbo.advisor', 'U') IS NOT NULL
DROP TABLE dbo.advisor
GO
CREATE TABLE advisor (
  s_ID varchar(20),
  i_ID varchar(20)
)
GO
```

```
-- #2 find names & ids of instructors who teach every course in their department; order by instructor name
SELECT i_name, ID
   FROM instructor AS I
        WHERE not exists (
         (SELECT C.course_id
                FROM course AS C
                WHERE C.dept_name = I.dept_name)
        EXCEPT
        (SELECT T.course_id
                FROM teaches AS T
                WHERE I.ID = T.ID)
   )
        ORDER BY i_name ASC;

-- #3 find names of departments whose total budget is higher than that of Physics; order alphabetically
SELECT dept_name
   FROM department
        WHERE budget > (
         SELECT budget
                FROM department
                WHERE dept_name = 'Physics')
        ORDER BY dept_name ASC;

-- #4 insert values to the classroom relation using the section relation; start capacity value as 100 and increase by 5
for each insertion
INSERT INTO classroom
   (building, room_number, capacity)
SELECT
   sect.building, sect.room_number, 100 + 5 * (ROW_NUMBER() OVER (ORDER BY sect.building,
sect.room_number) - 1) AS capacity
   FROM (
     SELECT DISTINCT building, room_number
     FROM section
     WHERE building IS NOT NULL AND room_number IS NOT NULL
   ) sect
LEFT JOIN classroom class
   ON sect.building = class.building AND sect.room_number = class.room_number
   WHERE class.building IS NULL AND class.room_number IS NULL;

SELECT * FROM classroom ORDER BY capacity ASC;

-- #5 create relation takes (see above); find sections with maximum enrollment; result should be courseid, secid,
year, semester, num
WITH enrollment AS (
   SELECT
         course_id, sec_id, yr, semester,
        COUNT(*) AS num FROM takes
        GROUP BY course_id, sec_id, yr, semester
)    SELECT section.course_id, section.sec_id, section.yr, section.semester, enrollment.num
```

```
FROM section
JOIN enrollment ON
        section.course_id = enrollment.course_id
AND section.sec_id = enrollment.sec_id
AND section.semester = enrollment.semester
AND section.yr = enrollment.yr
WHERE enrollment.num =
        (SELECT MAX(num) FROM enrollment)
ORDER BY section.course_id, section.sec_id, section.yr, section.semester, enrollment.num DESC;
```

Sources

Codecademy Team. “What Is a Relational Database Management System?” Codecademy, Codecademy, www.codecademy.com/article/what-is-rdbms-sql.

G., Domantas. “MySQL VS SQL: Overview, Similarities, Differences.” MySQL vs SQL: Overview, Similarities, Differences, Hostinger Tutorials, 2 Mar. 2023, www.hostinger.com/tutorials/difference-between-mysql-and-sql-server.

Markingmyname, et al. “Connect and Query a SQL Server Instance Using SQL Server Management Studio (SSMS) - SQL Server Management Studio (SSMS).” Connect and Query a SQL Server Instance Using SQL Server Management Studio (SSMS) - SQL Server Management Studio (SSMS) | Microsoft Learn, 3 Mar. 2023, learn.microsoft.com/en-us/sql/ssms/quickstarts/ssms-connect-query-sql-server?view=sql-server-ver16#create-a-table.

The Upwork Team. “MS SQL vs. Mysql: Which Relational Database Is Right for You?” MS SQL vs. MySQL: Which Relational Database Is Right for You?, UpWork, 16 July 2021, www.upwork.com/resources/ms-sql-vs-mysql-which-relational-database-is-right-for-you.