Introduction to Databases – Flipped Classroom THREE

Virtual Machines, SQL, Relational Calculus, EER Diagrams – NEED A LAPTOP FOR THIS!

General Instructions:

- You must have watched all the videos for Week 5 and Week 6, completed the Owl review exercises and Quiz 3 BEFORE doing the "flipped classroom" so that you are fully prepared to work on the exercises with your group.
- In your Owl group of students, TOGETHER as a group, discuss and complete the questions below on the paper.
- You will be given approximately 90 minutes to complete the questions. Once the 90 minutes are up, you will swap your paper with another group.
- We will then take up the questions together and you will write a score at the top of the paper and sign your group number as the markers and have one of your members sign the paper as the marker.
- Take your marked exercises and have one group member take a picture with his/her phone of each page. Make sure that for the first page, the group mark and marking groups signature is clear in the photo
- Have one group member upload and submit the images to owl for the flipped classroom THREE assignment AFTER IT WAS MARKED. This allows your group to use the flipped classrooms work for studying.
    - **FOR THIS WEEK → Take pictures of pages 1,5,6,7,8,9 & 10 and upload those pictures to OWL to share with your group AND REMEMBER TO TYPE GROUP MEMBERS WHO SHOWED IN THE OWL TEXTBOX.**
    - **TAKE A PICTURE OF THE FACES OF YOUR GROUP MEMBER AND UPLOAD THAT ALSO, so I can confirm which members of your group showed up (I have had people sign for other people when the students hadn't showed up. THIS IS AN ACADEMIC OFFENSE TO DO THIS. DO NOT SIGN THAT SOMEONE SHOWED UP WHEN THEY DID NOT SHOW UP!)**
- The goal is for your group to learn from each other, so it is fine, actually encourage, to brainstorm and discuss and problem solve! Feel free to surf the internet or watch the course videos again to figure out your answers.

YOUR GROUP NUMBER: ______________

Group Member Name (PRINT)	Present Today (Circle One)
1	YES NO
2	YES NO
3	YES NO
4	YES NO

THIS AREA IS ONLY TO BE FILLED IN BY THE MARKING GROUP **Marking Group Number:**		
Name of one of the marking group members (Printed)		
Signature of that group member		
SCORE OUT OF 1 (Total for this sheet is 40, so just divide by 40 and round to 1 decimal)		

Marking Instructions:

- The flipped classrooms are worth 2% each.
- Every group member who shows up to class and stays till the end will get 1% automatically.
- The other 1% is based on what you get on the questions below. The 2 values are then added together to get your flipped classroom mark.
- Each question is worth 1 or 2 marks but has several parts, if the group got MOST of the question right, give them the full mark, if they got more than 50% wrong, give them half marks. Add the marks together at the end and give them a score out of 1 rounded to 1 digit. (e.g. 0.7 or 0.8 or 1.0)
- **DON'T BE TOO EVIL WHILE MARKING, the goal is to understand the concepts better while learning together, not get everything perfect (but if a group everything wrong, don't give them perfect-try to find a nice balance!)**
- Pass the marked sheet back to the group you were marking.

Objectives:

- To practice creating databases in SQL
- To practice writing SQL queries
- To gain experience reading and writing Relational Calculus Expressions
- To practice drawing Enhanced Entity Relationship Diagrams
- To learn collaboratively how to problem solve

QUESTION 1 (5 Marks)

Part 1 (Creating the Database): Based on the ER diagram below, you are going to create a relational database, create tables, put data into the tables and then write some queries. Read the below scenario & look at the ER diagram CAREFULLY:

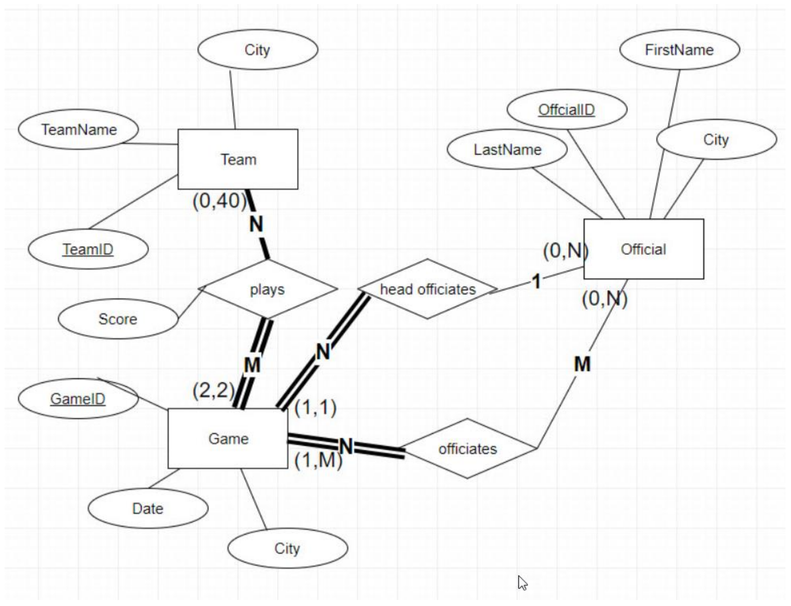
The NHL needs to keep track of it's games, officials (refs) and teams.

For teams, you must keep track of the city the team is from (e.g. Toronto), this field will be no longer than 15 letters, the team name (e.g. Maple Leafs), this field will be no longer than 20 letters, and a 2 digit team id (e.g. 12)

For the officials, you must keep track of their first and last names (each will be 20 letters) and their official id (a 2 digit field) and their home city (15 letter field).

For games, you must keep track of the game id (2 digits), the date of the game and the city the game took place in (15 letters field).

Teams play games. A team plays up to 40 games a season. Each game is played by 2 teams. You must keep track of the score for each team in a game. Teams may never play a game, but if a game is scheduled then it must have 2 teams play in it. Games are officiated by officials (refs). Every game has at least one official, but may have more. Every game has one official who acts as the head official (so every game will at least 2 officials, a regular one and a head one, HINT: keep track of both (officiates game and head officiates game) as 2 separate relationships). Some officials will be in our database who have not yet officiated any games.



Your group needs to:

1. Get into the virtual machine on the laptop of one of your group members → login with putty or terminal window using vm??? and then do:
   ssh centos@vm???
2. You should add each command, as soon as you are sure you are correct, into Notepad/textedit (format as plain text) as well and save that file occasionally because then you can reuse your commands from Notepad if you screw something up and have to drop the database. (OR USE THE UP ARROW to get back a command).
3. Get into mysql (**`mysql -u root -p`**)
4. Create a database, you can give it any name you want
5. Here is the SQL statement to create the ***official*** table, run this statement:
   ```
   CREATE TABLE official (officialid CHAR(2) NOT NULL, firstname VARCHAR(20), lastname
   VARCHAR(20), offcity VARCHAR(15), PRIMARY KEY (officialid));
   ```
6. Now write your own SQL to create the ***team*** table (make the teamid char(2) → do not make it int, so that the inserts work later). **IMPORTANT HINT: it will make your MUCH life easier later on if you put the columns in the following order: teamid, teamcity, teamname** because of a later step. Remember to also set the primary key for the team table. Once you are sure it works, write the command here:

   ____
7. Type the following two commands:
   ```
   SHOW TABLES;
   DESCRIBE team;
   ```
8. Here are the SQL statements to create the ***game*** table and the ***reffing*** table and the **playing table**. Copy them into your VM mysql window to run all 3 statements:

   ```
   CREATE TABLE game (gameid CHAR(2) NOT NULL, gamedate DATE, gamecity VARCHAR(15),
   headoff CHAR(2), FOREIGN KEY(headoff) REFERENCES official(officialid), PRIMARY KEY
   (gameid));

   CREATE TABLE reffing (gameid CHAR(2) NOT NULL, officialid CHAR(2) NOT NULL, FOREIGN
   KEY (gameid) REFERENCES game(gameid), FOREIGN KEY (officialid) REFERENCES
   official(officialid), PRIMARY KEY (gameid, officialid));

   CREATE TABLE playing (gameid char(2) NOT NULL, teamid char(2) NOT NULL, score INT,
   PRIMARY KEY (gameid, teamid), FOREIGN KEY (gameid) REFERENCES game(gameid), FOREIGN
   KEY(teamid) REFERENCES team(teamid) ON DELETE CASCADE);
   ```

9. Look at the statement that creates the many to many relationship between team and game (the *playing* table). What do you think will happen to this table when a team gets deleted from the team table?

Part 2 (Filling your tables): Now you are going to put rows of data into your database.

1. I have written most of the insert statements for you. Keep in mind that if you have called your tables with different names than me OR if your attributes weren't in the same order as mine, then you will need to modify these statements. For example, you might have to change this:
   ```
            INSERT INTO team VALUES('99','Colorado', 'Avalanche');
   ```
   to this (because your table was different and your order of columns was different, then include the column names):
   ```
            INSERT INTO myteam (teamid, teamcity, teamname) VALUES ('99','Colorado',
   'Avalanche');
   ```

   The insert statement are here:
   http://www.csd.uwo.ca/~lreid/blendedcs3319/flippedclassroom/three/insertdata.txt
   IMPORTANT: You might want to copy and paste these statements FIRST into NOTEPAD, save the NOTEPAD file and then copy them over to your virtual machine. As you test and fix them, put the fixed version back into NOTEPAD and occasionally save this NOTEPAD file so that just in case your screw up your database, you can use this NOTEPAD file to help you reload your tables.
2. Now write two more inserts:
   a. Insert one more team of your group's choice, any team name, any team city, any unused teamid.
   b. Insert one more official, pick your group's favourite actor, makeup an officialid, and use any city you want.
3. Make sure that your SQL INSERTS worked by writing 5 SELECT SQL statements for each of the 5 tables to see all the data in each table.

Part 3 (Updating your tables): Now you are going to modify the existing rows into your tables.

1. Type this command to see what the *team* table and the *game* table looks like:
   DESCRIBE team;
   DESCRIBE game;
   DESCRIBE is a very useful command! If you forget what your table looks like, use that command!
2. Write the SQL UPDATE statement to change the team name of the "Maple Leaves" to the "Maple Leafs"

3. For every game that the Leafs have played, change their score to 3. DO NOT do this by referencing the Leafs team id, rather do it by referencing their team name (this will require a join within your UPDATE statement, i.e. a SELECT inside the UPDATE – It will kind of look like this: UPDATE … WHERE … IN (SELECT …); Fill in the blanks below correctly and then run your statement to make sure it works:

   UPDATE playing SET ________ WHERE teamid in (SELECT teamid FROM team WHERE teamname = "______________");

QUESTION 2 (15 Marks – 1 Mark Each)

Use the database you just created to answer the following queries and write down the SQL for the queries as you solve them. HINT: use the describe command if you forget the column names:

1. Show the team name of all the teams:

   ____

2. Show the city of where all games were played with no repeats.

   ____

3. Show all the data in the officials' table, but show them in order of first name.

   ____

4. Show the First and Last Name of all officials from Ottawa or New York.

   ____

5. List the last name of all head officials.

   ____

6. List the first and last name and official's city of any official who comes from a city with the letters "on" anywhere in the city name.

   ____

7. List the gameid, the city the game took place in, the team's city and the teams name and the score for that team. For these results, make sure the 2 teams that played each other are listed immediately after each other. That is: do NOT put the team that played in game 15, then the team that played in game 31, then the other team that played in game 15, instead make sure you keep the 2 teams in each game right after each other in the results from the query.

   ____

8. Find the total number of goals scored for by the Rangers (when creating your query, make sure you reference the team name of 'Rangers' in your query).

   ____

   FOR THIS QUERY: WHAT WAS THE TOTAL NUMBER OF GOALS? ___________________

9. Find the average number of goals that each team scored, print out the team name and the average goals. HINT you will need to use GROUP BY

   ____

10. List the head official's first and last name for all head officials of Maple Leafs games.

    ____

11. Find the city name and team name of any teams who haven't played any games.

    ____

12. Find all officials (first and last name) who have refereed for more than 1 game (Hint: you will have to use the key words GROUP BY and HAVING).

    ____

13. To check for bias by the official, find the first name and last name and official's home city and the game id for any official who referred a game where the official comes from a city that is the same city as one of the teams playing the game (don't worry about head officials for this one, just officials in the reffing table).

    ____

" % on % "

Game

QUESTION 3 (5 Marks) CREATING VIEWS AND DELETING ROWS

Create and test the following SQL commands and then write them down in pencil below.

1. (1 Marks) Create a view that shows the first and last name and official's city of any official who comes from a city with an "on" in the city name (Query 6 from above only as a view) and select from the view to make sure it worked (NOTE: it is a good idea to names views starting with a "v"):

   City

   CREATE VIEW ViewName AS SELECT FName, LName, | FROM official WHERE city LIKE '%on%'

2. (1 Marks) Select all the officials, then delete the official that your group made up, then select again to make sure it worked. Write the DELETE statement below:

   DELETE FROM official WHERE id=55

3. (1 Marks) Select all the teams, then delete any teams who have not played any games, then select again to make sure it worked. Write the delete statement below:

   DELETE FROM team WHERE team. (SELECT team FROM play WHERE (COUNT(teamID)) FROM gameplay) =0

4. (2 Marks) Run the following query and figure out what it is giving you:

```
SELECT score, gameid, teamname FROM playing,team WHERE team.teamid=playing.teamid ORDER BY gameid;
```

   Then delete the Maple Leafs team (goodbye Matthew Austin and John Tavares and all of Toronto's hopes and dreams). Then run the above query again. Can your group figure out the problem that arose because of the DELETE CASCADE? If so, write it down here:

   ____

   ____

   ____

5. On your virtual machine, make a text file call ***instuff.txt*** In this file put the SQL statements required to show all the data in all of your 5 tables. Then use the following command to pipe it into MySQL and send the output to another file called ***outstuff.txt***. View the output file to make sure it shows all the data (to view type this command: *more outstuff.txt*).
   HINT: you need to add one more command to start of you instuff.txt in order to work with the correct database.

```
sudo mysql --verbose -pyourpassword < instuff.txt  > outstuff.txt 2>&1
```

QUESTION 4: Relational Calculus (8 Marks - 2 Marks Each)

Given the following tables:

q r s t u v w x y z

Table Name	Table Layout									
EMP	FName	Minit	LName	SSN	BDate	Address	Sex	Salary	SuperSSN*	DNO*
DEPT	DName	DNumber	MGRSSN *	MGRStartDate						
WORKSON	ESSN *	PNO *	Hours							
PROJ	PName	PNumber	PLocation	DNum*						
DEPENDENT	ESSN*	DependentName	Sex	BDate	Relationship					

a b c d e.

1. Get all the salary and birth date and address of employees with the first name of "Homer".

Tuple Calculus:

$\{t.Salary, t.BDate \mid EMP(t)$ and $t.FName = \text{"Homer"}\}$.

Domain Calculus: (write the letters over the columns)

$\{x, u \mid (\exists q)\, EMP(qrstuvwxyz)$ and $q = \text{"Homer"}\}$.

2. Get the first and last names of all employees who have a salary over 1000

Tuple Calculus:

$\{e.FName, e.LName \mid EMP(e)$ and $e.Salary > 1000\}$

Domain Calculus:

$\{q, s \mid (\exists q)\, EMP(qrstuvwxyz)$ and $x > 1000\}$.

3. Get the names of dependents who have a father (relationship) with the FName of 'Kramer'

Tuple Calculus:

$\{t.DependentName \mid DEPENDENT(t)$ and $((\exists q)(EMP(q)$ and $t.relationship = father.$ and $q.FName = \text{'kramer'})$

Domain Calculus:

$\{b \mid DEPENDENT(abcde)$ and $e = father$ and $(\exists q)((EMP(qrstuvwxyz)$ and $q = \text{'kramer'})$.

4. Find the last name of employees who work more than 30 hours on the project with the name of 'Alpha'

Tuple Calculus:

Domain Calculus:

QUESTION 5: EER Diagrams (5 Marks)

Draw an EER diagram for the following scenario. Make sure that you put the correct symbol (either O or D) in the oval and use the correct line (either single or double) from the superclass to the oval.

Scenario: Students must complete projects assigned to them by their professors. Just draw the EER for the project entity (you don't need to represent the students or profs for this example). Every projects have a due date and a project id. The projects can be research papers (if so, the research paper must have a title and subject), or the projects can be posters (posters have a width and height and topic) or the projects can be programs (programs have a language (e.g. C++, Java), number of lines of code and a platform (e.g. windows, mac, linux)). Projects must be one of the above 3 types, there are no other types of projects.