**Final Project – CS 3550**

**Relational database portion**

ESPN is starting a new division to cover E-Sports. A key element of this new product offering is the ability to display statistics and team information related to E-Sports. Your project manager has tasked you with creating a database to keep track of the following items:

* Information about each of the E-Sport athletes, including name, country, game alias, age, and university (if applicable).
* Roster information – what players are attached to what teams (current and historical). For example, Cloud9 is a League of Legends team that has six players currently (5 active and 1 substitute). Historically, there are more than 20 other players that were part of Cloud9.
* History of games played with win/loss information. For this exercise, I want you to track only team vs. team information (what two teams were involved, when they played, and what the outcome was).
* Other bits of information that you’ve been asked to track include team sponsor, country of the team, # people who viewed a match, and game genre. Feel free to add other bits of information you feel relevant to accomplish later tasks. For each of the tables, please add a field that auto populates when the record was created.
* Rules to consider. An athlete will only compete on a single team at any given time. Only one E-Sport athlete can have a given alias.

Using the above information, here is what I need turned in:

* CREATE TABLE scripts for your third normal form database. I expect you to use primary key/foreign key relationships and auto-incrementing identity fields where applicable. Make sure your scripts create things in the correct order (i.e. test your final script, running all your create table scripts at once and make sure you don’t get any errors).
* INSERT scripts to add a few records to each of your tables (at least one record in all your tables).
* Stored procedures to insert new records into your Games table and the table you create to track games played.
* Create a query that displays the win/loss record for all teams for 2016 and 2017. Your results should look something like the following:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Team** | **Game** | **2016 Wins** | **2016 Losses** | **2017 Wins** | **2017 Losses** |
| Cloud9 | League of Legends | 23 | 16 | 6 | 2 |
| Cloud9 | Halo | 82 | 42 | 21 | 2 |
| FlyQuest | League of Legends | 18 | 23 | 21 | 10 |

* Create a trigger that prevents a roster assignment from being deleted. This trigger should instead “deactivate” or “end date” that player’s association with a given team.
* Create a trigger that prevents new roster assignments if a team has more than six active players on it.
* Create a view that will show a team’s current roster of players. This view should include the name of the player, their alias, when they started on the team, their country of origin, team name, and game the team plays.
* Write a query that shows the win/loss record for each player. A player will be given credit for any win or loss that is for a team that they were active on a roster. This should include wins/losses for any team that the player was associated with.
* Create an update script that will invalidate all wins for a specific team for a specific year (i.e. turn all their wins for 2017 to losses because they cheated).

Points to consider for this part of the assignment –

* I want to be able to run all your scripts with a single execution, with the exception of the final update script (comment it out).
* Name your table and columns well – your names should match what is being stored.
* Put enough test data into your tables to test the above well. I’ll add data where I need to but the only way to know if things are working as expected is to have test data. Include your insert statements for all this test data.
* Good code formatting is a must. Whitespace and tabbed lines for code blocks are required!
* If you’re not sure about something, ask. We’ll use Canvas for this so everyone can benefit from the answers.
* If you get stuck, move on. I’m overly generous on giving partial credit where I see an honest effort.

**Mongo/JSON portion**

The goal of this portion is to make sure you can do some basic work with SQL Server JSON and MongoDB. You need to do the following –

* Using WideWorldImporters and only the Application.Person table, write a query that returns all ten sales people (isSalesPerson = 1). I want you to include their first name, last name, alias, email address, phone number, title, territory, and hire date. Kayla’s record would look like the following:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PersonID** | **FirstName** | **LastName** | **Alias** | **Email** | **Phone** |
| 2 | Kayla | Woodcock | Kayla | kaylaw@wideworldimporters.com | (415) 555-0102 |
|  |  |  |  |  |  |
| **Title** | **PrimaryTerritory** | **HireDate** |  |  |  |
| Team Member | Plains | 4/19/2008 |  |  |  |
|  |  |  |  |  |  |

* Leveraging the first query, write something that exports out each sales person record into a JSON string. Kayla’s record should look like this:

[

{

"PersonID": 2,

"FirstName": "Kayla",

"LastName": "Woodcock",

"Alias": "Kayla",

"Email": "kaylaw@wideworldimporters.com",

"Phone": "(415) 555-0102",

"CurrentJob":

{

"Title": "Team Member",

"Territory": "Plains",

"HireDate": "4/19/2008"

}

}

]

* Leveraging the JSON script, create a new database called WideWorldImporters, a collection called SalesPeople, and insert all the sales people.
* Write a script that returns just Kayla’s record
* Write a script that changes Kayla’s job title to “I am done”

Turn in the .sql and .js files from above.