# Instructions

In these tasks, existing database design is taken and expanded it using PL/SQL and various objects to better support a business application. Through the process of understanding the business requirements, the business rules, and how the database will work with the software application, learners will be able to identify ways, or solutions, in which the database design can be extended to be an integral part of the software architecture, and not just a storage and retrieval facility.

PL/SQL codes are written which creates the required objects in a database to support the business requirements, and business rules, as well as various objects to support the user interface and basic CRUD operations through a secure parameterized method.

# Tasks

The assignment is broken into a few tasks and are based on the SportLeagues database that has already been provided to you. For the purposes of this assignment, we will only cover the following tables:

* Games
* GoalScorers
* Players
* Teams
* Rosters
* slLocations

we will not be using the divs or teamsindivs tables.

1. For each table, create Stored Procedures to cover the 4 basic CRUD tasks.
   1. INSERT a new record and if the PK using autonumber, the SP returns the new PK
   2. UPDATE an existing record given the PK
   3. DELETE an existing record given the PK
   4. SELECT return all fields from a table given a PK

* Name the SPs using the following guide: spTableNameMethod (example spPlayersInsert)
* Do not use DBMS\_OUTPUT in the procedures in any way.
* All SPs must have appropriate exception handling specific to the method and table.
* Use error codes of the same type and size of the PK to return values that can be clearly determined to indicate an error (example: -1 might indicate no record was found)

1. For each table, create a Stored Procedure that outputs the contents of the table to the script window (using DBMS\_OUTPUT) for the standard SELECT \* FROM <tablename> statement.
2. Create a view which stores the “players on teams” information, called vwPlayerRosters which includes all fields from players, rosters, and teams in a single output table. You only need to include records that have exact matches.
3. Using the vwPlayerRosters view, create an SP, named spTeamRosterByID, that outputs, using DBMS\_OUTPUT, the team rosters, with names, for a team given a specific input parameter of teamID
4. Repeat task 4, by creating another similar stored procedure, named spTeamRosterByName, that receives a string parameter and returns the team roster, with names, for a team found through a search string. The entered parameter may be any part of the name.
5. Create a view that returns the number of players currently registered on each team, called vwTeamsNumPlayers.
6. Using vwTeamsNumPlayers create a user defined function, that given the team PK, will return the number of players currently registered, called fncNumPlayersByTeamID.
7. Create a view, called vwSchedule, that shows all games, but includes the written names for teams and locations, in addition to the PK/FK values. Do not worry about division here.
8. Create a stored procedure, spSchedUpcomingGames, using DBMS\_OUTPUT, that displays the games to be played in the next n days, where n is an input parameter. Make sure your code will work on any day of the year.
9. Create a stored procedure, spSchedPastGames, using DBMS\_OUTPUT, that displays the games that have been played in the past n days, where n is an input parameter. Make sure your code will work on any day of the year.
10. Each group must be creative and come up with an object (SP, UDF, or potentially trigger), of your own choosing, that will be built in the database to help support the same ideals of the above objects.