**Technical Guide to NBA Contract Evaluation Project**

Alex Grimm, Chinedu Okeke, Jordan Kakuyo

**Overview:** This document details the files, stored procedures, functions, and triggers associated with the NBA Contract Evaluation Project. There are many different files and tools necessary to gain access and understanding of the project. This document will explain how to run the files and the basic details behind every method that was implemented.

**Files:** Listed below are the necessary files that contain all of the SQL code to create our database, insert data, and generate our stored procedures, functions, triggers, and views.

***project\_create\_tables.sql***

This file should be run first. It contains the code that is responsible for making the database, creating and dropping tables as well as the table constraints.

***project\_triggers.sql***

This file should be run second. It creates and applies two triggers. The first being a trigger that limits the roster size to fifteen. The second limits the cap space taken up by players on a team’s roster

***project\_insert\_data.sql***

This file should be run second. It inserts the data into our databases. This data includes our initial data as well as the data we will be using for our demo.

***project\_stored\_procedures.sql***

There is no specific order in which this file should be run, but the stored procedures in this file can be used to insert data into each table. These procedures can be utilized as an alternative to insert statements.

***project \_views.sql***

This file should be run anytime after the data insertion file. It creates views regarding different areas that the database covers

***project\_functions.sql***

This file should be run anytime after the table creation file. It creates functions that can be used to filter the data.

***project\_demo.sql***

This file should be run last. It is a demo that shows how the database can be utilized to identify what a player's contract length and amount should be and how to assign that player to a certain team.

**Views:** Listed below are the views we have created for our project.

***v\_salary\_team\_breakdown***

This view returns information regarding every player in the database. This information includes their name, team, contract length in years, the total amount their contract is worth, the yearly value of their contract, and the percentage of the capspace that a player’s salary takes up.

***v\_team\_cap\_space***

This view provides a breakdown of a team's spending during a given season. The view shows the team, the year, the amount they have spent on their roster, and the amount that they still have available to spend.

***v\_team\_position\_breakdown***

This view provides information regarding each team and the number of players at each position in a given season

***v\_alternative\_position\_breakdown***

This view provides the same information as v\_team\_position\_breakdown. However, the table is in wide format, so it is easier for the user to read and understand.

***v\_2024\_player\_stats\_and\_salaries***

This view provides information regarding every player's name, the team they play for in the 2024 season, their season statistics, and their contract information.

***v\_2023\_player\_stats\_and\_salaries***

This view provides information regarding every player's name, the team they play for in the 2024 season, their season statistics, and their contract information.

***v\_team\_information***

Provides information regarding each team for both the 2023 and 2024 seasons.This information includes the teams location, name, statistics, and colors.

***v\_roster\_space***

Providesa breakdown of the number of players at each position for a team in a specific season.

**Functions:** Listed below are the functions included in our project

***f\_player\_info***

This function takes arguments:

@player\_firstname as varchar,

@player\_lastname as varchar

This table valued function takes the player’s first name and lastname as input and returns information regarding their previous teams, seasons, and related player statistics. It should be used when a user needs to see specific information regarding a certain player.

***f\_team\_info***

This function takes argument:

@team\_name

It returns all team information and statistics for every season for the specified team input.

***f\_get\_player\_salary***

This function takes arguments:

@player\_firstname as varchar,

@player\_lastname as varchar,

It returns information regarding the specified players' contract information.

***f\_filter\_ppg***

This function takes arguments:

@filter\_value

It returns the players in the database with average points per game values greater than or equal to the specified value. It is worth noting that sorting of the results must be done outside of the function.

***f\_filter\_apg***

This function takes arguments:

@filter\_value as decimal

It returns the players in the database with average assists per game values greater than or equal to the specified value. It is worth noting that sorting of the results must be done outside of the function.

***f\_filter\_rpg***

This function takes arguments:

@filter\_value as decimal

It returns the players in the database with average rebounds per game values greater than or equal to the specified value. Sorting of the results must be done outside of the function.

***f\_filter\_major\_stats***

This function takes arguments:

@filter\_value\_points as decimal(4,1),

@filter\_value\_rebounds as decimal(4,1),

@filter\_value\_assists as decimal(4,1),

It returns the players in the database with average points per game, assists per game, and rebounds per game values greater than or equal to the specified value. Sorting of the results must be done outside of the function.

**Stored Procedures:** Listed below are the stored procedures included in our project.

***p\_insert\_team\_colors***

This procedure takes:

@team\_city as varchar

@team\_state as char

@team\_name as varchar

@color\_primary as varchar

@color\_secondaryas varchar

This procedure takes multiple arguments and inserts data into the colors, teams, and team\_colors tables. It makes it easier to insert data into two tables with one command. Also, the foreign key assignments are taken care of in the procedure.

***p\_insert\_team\_drafts***

This procedure takes:

@team\_name as varchar

@draft\_pick as int

@draft\_year as char

@player\_firstname as varchar

@player\_lastname as varchar

This procedure takes multiple arguments and inserts data into both the drafts and team drafts tables. It makes it easier to insert data into two tables with one command. Also, the foreign key assignments are taken care of in the procedure.

***p\_insert\_players***

This procedure takes:

@player\_firstname as varchar

@player\_lastname as varchar

@player\_dob as data

@player\_age as int

@player\_jerseynumber as int

@team\_name as varchar

@player\_position as varchar

This procedure takes multiple arguments and adds a player to the players table. It also handles connections in the players position table. This procedure provides a shortcut to adding data to these tables.

***p\_insert\_player\_statistics***

The procedure takes the following arguments:

@player\_firstname as varchar

@player\_lastname as varchar

@team\_name as varchar

@player\_year as char

@player\_statistic\_gp as int

@player\_statistic\_mpg as decimal

@player\_statistic\_ppg as decimal

@player\_statistic\_apg as decimal

@player\_statistic\_rpg as decimal

@player\_statistic\_bpg as decimal

@player\_statistic\_spg as decimal

@player\_statistic\_tpg as decimal

@player\_statistic\_fg\_made\_pg as decimal

@player\_statistic\_fg\_taken\_pg as decimal

@player\_statistic\_ft\_made\_pg as decimal

@player\_statistic\_ft\_taken\_pg as decimal

@player\_statistic\_3pt\_made\_pg as decimal

@player\_statistic\_3pt\_taken\_pg as decimal

@player\_statistic\_fouls\_taken\_pg as decimal

This procedure takes these arguments and fills them into the player statistic table. It also takes the name of the name and team of the player and assigns the correct player to their statistics. This procedure makes it easier to insert data into both of the tables.

***p\_insert\_team\_statistics***

The procedure takes the following arguments:

@team\_statistic\_year as char

@team\_statistic\_wins as int

@team\_statistic\_losses as int

@team\_statistic\_ppg as decimal

@team\_statistic\_apg as decimal

@team\_statistic\_rpg as decimal

@team\_statistic\_bpg as decimal

@team\_statistic\_spg as decimal

@team\_statistic\_tpg as decimal

@team\_statistic\_fg\_made\_pg as decimal

@team\_statistic\_fg\_taken\_pg as decimal

@team\_statistic\_ft\_made\_pg as decimal

@team\_statistic\_ft\_taken\_pg as decimal

@team\_statistic\_3pt\_made\_pg as decimal

@team\_statistic\_3pt\_taken\_pg as decimal

@team\_statistic\_fouls\_taken\_pg as decimal

@team\_name as varchar

This procedure takes these arguments and fills them into the team statistic table. It also takes the name of the team and team of the player and assigns the correct team to their statistics. This procedure makes it easier to insert data into both of the tables.

***p\_insert\_salaries***

The procedure takes the following arguments:

@salary\_contract\_length as int

@salary\_contract\_amount as money

@player\_firstname as varchar

@player\_lastname as varchar

@team\_name as varchar

This procedure takes these arguments and fills them into the team statistic table. It also takes the name of the team and team of the player and assigns the correct salary to their players. This procedure makes it easier to insert data into both of the tables.

**Triggers:** Listed below are the triggers that have been implemented in our project.

***t\_active\_roster\_limit***

This trigger checks to make sure that a team does not have more than fifteen players on the active roster during a particular season. If a team does attempt to add a sixteenth player, a customized warning is sent that says “Team has reached maximum roster size. Drop or trade a player to add to active roster.”

***t\_salary\_cap***

This trigger checks to make sure that a certain team does not exceed the NBA salary cap limit of one-hundred and fifty million dollars. If a team attend to sign a player that will exceed the salary cap, an error message will display that says “Team does not have enough cap space to add this player.”