COL362/632: Introduction to DBMS

Assignment 1

Due: 03 February 2023 11:59PM (**hard deadline, will not be extended**)

Modified on: Saturday 28th January, 2023
Modified on: Monday 30th January, 2023
Modified on: Tuesday 31st January, 2023

General Instructions

1. Please complete the assignment *individually*. No collaborations with other entities – person(s), AI agents, websites, discussion fora etc.– is allowed.
2. You are expected to use PostgreSQL 8.4.22 for this assignment.
3. There are a total of 30 queries in this assignment. Answer all queries.
4. **Marking scheme:** Queries carry different weightage based on the perceived difficulty level of the query. These weights are not revealed to you (intentionally). The weightage will be released after evaluation.
5. Command to run .sql files: \i /path/to/file.sql
6. You will submit only one file: $< entrynumber > .sql$. The format of the file should be as follows. One line should identify the query number (note the two hyphens before and after the query number), followed by the actual, syntactically correct SQL query. Leave a blank line after each query.
   --1--
   SQL QUERY - 1
   --2--
   SQL QUERY - 2
   --3--
   SQL QUERY - 3

Also note that all the documents that are linked in this file are accessible only within IITD using your OneDrive account.

Plagiarism Rules

If caught, you will receive 0 for the assignment, and a **penalty** of -10% in the course.

1 Dataset

We will be using a large dataset of baseball statistics for this assignment. This dataset is a compilation of pitching, hitting, and fielding statistics for US Major League Baseball from 1871 through 2014. The data has been cleaned to an extent and organized for the purpose of this assignment.

1.1 Description

The design follows these general principles. Each player is assigned a unique number (playerID). All of the information relating to that player is tagged with his playerID. The playerIDs are linked to names and birthdates in the People table[1].

The database is comprised of the following main tables:

Table Name	**Description**
People	Player names, DOB, and biographical info
Batting	batting statistics
Pitching	pitching statistics
Fielding	fielding statistics

These main tables are supplemented by the following tables:

Table Name	**Description**
AllStarFull	All-Star appearances
HallofFame	Hall of Fame voting data
Managers	managerial statistics
Teams	yearly stats and standings
TeamFranchises	franchise information
Salaries	player salary data
SeriesPost	post-season series information
AwardsManagers	awards won by managers
AwardsPlayers	awards won by players
AwardsShareManagers	award voting for manager awards
AwardsSharePlayers	award voting for player awards
Appearances	details on the positions a player appeared at
Schools	list of colleges that players attended
CollegePlaying	list of players and the colleges they attended

Detailed description of the dataset with tables and their attributes can be found here (link accessible through IITD OneDrive).

[1] In this document we will use 'table' and 'relation' interchangeably.

1.2 Schema

Schema diagram for the dataset can be found here (link accessible through IITD OneDrive).

1.3 Instructions

1. Download the cleaned up data from this link. It is a zip file that contains CSV files for each of the tables described above. It also contains two sql files to be used for data loading.
2. In order to load this data into you database, you need to first define the structure for these data tables. Run the file database_structure.sql using \i /path/to/database_structure.sql
3. Once the table structures are defined, you can load the data to your database by running \i /path/to/import_database.sql. **Note that you first need to change import_database.sql by modifying the path of csv files for the corresponding tables**.

1.4 Instructions for Queries

1. Do not output duplicates for any query.
2. Output format mentioned with each query contains the exact column names of expected output.
3. Each year counts as one season. [seasonid = yearid]
4. For any person, managerid is same as playerid.
5. Consider a person to be alive if their death date is missing i.e. all of deathday, deathmonth and deathyear are missing.
6. Runscore of a player is runscore $= 2 \times$ Doubles $+ 3 \times$ Triples $+ 4 \times$ Home Run
7. Batting average for a season is defined as the number of hits divided by the number of At-bats. Career batting average is the average of batting average across all seasons played by a player in their entire career.
8. When asked to output playername, it should be in the format "firstname lastname". Case should be same as given in namefirst, namelast columns in people table. Note that the separator is space. Whenever any one of namefirst and namelast is null, output the attribute which is not null and if both are null output empty string.
9. When asked to output date of birth or date of death, it should be in the format "YYYY-MM-DD". If any component of DOB is null, output empty string.
10. When asked to output birth address, it should be in the format "birthcity birthstate birthcountry". All letters should be lowercase. Note that the separator is space. If any component of birth address is null, output birthaddr as empty string.
11. When asked to output school address, it should be in the format "schoolcity schoolstate". All letters should be lowercase. Note that the separator is space. If any component of school address is null, output school address as empty string.

12. For a team, winning percentage for a season is defined as number of team wins (W) divided by number of games played by team (G) in that season times 100. [winning_percentage = (W/G)*100]

13. Career Saves for a pitcher is defined as the total number of saves (SV) by the picther across all seasons.

14. **Graph 1:** The concept of graphical analysis can be applied to this dataset. Consider the tables pitching and allstarfull. Now, we can make a graph such that all the players who appear in these tables are nodes of this graph. The edges are defined such that there exists an edge between two nodes (players) if they have played in the same team in the same season, the weight of the edge is the number of seasons played together in the same team. The graph thus formed is undirected and weighted. If players A and B played 5 seasons together in team X and 2 seasons together in team Y, the graph will have one edge between A and B with weight 7. For allstarfull table, consider only the tuples where the player actually played in the game i.e. GP = 1

15. **Graph 2:** Consider the table seriespost. We can define a graph such that all the teams present in seriespost table are the nodes of this graph. The edges are defined as follows: there exists an edge from team A to team B if there is an entry in seriespost table such that winner team = team A and loser team = team B. There can be multiple entries where winner is team A and loser is team B, however, in the graph there will be only one directed edge from A to B. The graph thus formed is directed and unweighted.

16. A path is a sequence of **non-repeated** nodes connected through edges. In case of directed graph, a path is a sequence of nodes in which there is a (directed) edge pointing from each node in the sequence to its successor in the sequence.

17. For directed graph, path length from A to B is defined as the number of edges that occur when going from A to B.

18. Two paths are considered to be different if they have atleast one uncommon node.

19. Path length for Graph 1 is defined as the sum of weights of all edges on the path. Path length for Graph 2 is defined as the number of edges on the path.

20. A cycle is path such that its starting and ending nodes are the same.

21. **NULL handling:** In this database, there are many NULL values. Use the following instructions for handling NULL values in your queries unless otherwise mentioned:

    - If a string value is given as null, consider it as empty string
    - If an integer value is given as null, consider it as 0
    - If a boolean value is given as null, consider it as false

22. Conider teamid as the sole identifier for teams, and use it to distinguish teams from one another. Note that there are teams whose names have changed over the years i.e. there are multiple names associated with the same teamid. Irrespective of the name, as long as the teamid is same, it is still considered to be the same team. When asked to output teamname for such a team, output the latest name.

2 Queries

1. Show top 10 batsmen with highest total Caught stealing count in their entire lifetime.
   - Output format: playerid, firstname, lastname, total_caught_stealing
   - Order by: 1. total_caught_stealing (descending order), 2. firstname (ascending order), 3. lastname (ascending order), 4. playerid (ascending order)
2. Show details of top 10 batsmen with highest runscore calculated for their entire career (runscore for player's entire career is the sum of runscores across all games he played).
   - Output format: playerid, firstname, runscore
   - Order by: 1. runscore (descending order), 2. firstname (descending order), 3. playerid (ascending order)
3. For each player show the name of the player and total points received by them from the year 2000 and later.
   - Output format: playerid, playername, total_points
   - Order by: 1. total_points (descending order), 2. playerid (ascending order)
4. Output the top 10 players with the highest career batting average who have played at least 10 seasons. Ignore the entries where any of hits or at-bats is null or where at-bats is zero. Ignore these tuples while calculating career batting average and also while counting seasons.
   - Output format: playerid, firstname, lastname, career_batting_average
   - Order by: 1. career_batting_average (descending order) 2. playerid (ascending order) 3. firstname (ascending order) 4. lastname (ascending order)
5. Output players with the number of seasons they have played in decreasing order. Note that a player might have played multiple roles out of batter, pitcher and fielder in the same season, it still counts as one season.
   - Output format: playerid, firstname, lastname, date_of_birth, num_seasons
   - Order by: 1. num_seasons (descending order) 2. playerid (ascending order) 3. firstname (ascending order) 4. lastname (ascending order) 5. date_of_birth (ascending order)
6. For each team, consider only the seasons where it has had a division win (DivWin), now output the maximum number of wins (W) the team has had in a single such season.
   - Output format: teamid, teamname, franchisename, num_wins
   - Order by: 1. num_wins (descending order) 2. teamid (ascending order) 3. teamname (ascending order) 4. franchisename (ascending order)
7. Output top 5 teams with the highest winning percentage in a season and at least 20 wins across all seasons. Also output the season where team has highest winning percentage. Don't output the same team twice i.e. output the top 5 different teams.
   - Output format: teamid, teamname, seasonid, winning_percentage

- Order by: 1. winning_percentage (descending order) 2. teamid (ascending order) 3. teamname (ascending order) 4. seasonid (ascending order)

8. For each team, output the highest-paid player for every season. If there are multiple highest paid players for a team, output all of them.

   - Output format: teamid, teamname, seasonid, playerid, player_firstname, player_lastname, salary
   - Order by: 1. teamid (ascending order) 2. teamname (ascending order) 3. seasonid (ascending order) 4. playerid (ascending order) 5. player_firstname (ascending order) 6. player_lastname (ascending order) 7. salary (descending order)

9. Which group (out of 'pitcher', 'batsman', return the appropriate string) of players have the highest average salary. (For players common in batting and pitching table consider them in both). Ignore the players whose salaries are not given. Note that here 2 interpretations are possible and both are accepted,

   - First one is if a player played as batsman in that (team, year, league) i.e has an entry in batting table then consider this entry for batting and add the corresponding salary of the player for batting avg salary.
   - Second one is you can consider a player as batsman if he has ever batted, and add all his salaries across all years/leagues/teams for batting avg salary.

   - Output format: player_category, avg_salary
   - Order by: The returned output will have just 1 row

10. For every player, find the number of batchmates (i.e went to at least 1 same college in the same year)

    - Output format: playerid, playername, number_of_batchmates
    - Order by: 1. number_of_batchmates (descending order), 2. playerid (ascending order)

11. Top 5 teams that have won the most (WSWin) world series titles; the team must have played at least 110 games in the season it won the world series title i.e. when counting the total number of WS titles won by a team, consider only the seasons where the team has played at least 110 games.

    - Output format: teamid, teamname, total_WS_wins
    - Order by: 1. total_WS_wins (descending order) 2. teamid (ascending order) 3. teamname (ascending order)

12. Top 10 pitchers with the highest career saves; the pitchers must have played at least 15 seasons in their career (as a pitcher). Also output the number of seasons played by these pitchers.

    - Output format: playerid, firstname, lastname, career_saves, num_seasons
    - Order by: 1. career_saves (descending order) 2. num_seasons (descending order) 3. playerid (ascending order) 4. firstname (ascending order) 5. lastname (ascending order)

13. Find all pitchers who have pitched for at least 5 different teams in their career. Also, output the earliest two different teams they have pitched for. If the pitcher pitched for more than two teams in the earliest year, use the attribute "stint" to find the earliest two teams among them.

- Output format: playerid, firstname, lastname, birth_address, first_teamname, second_teamname
- Order by: 1. playerid (ascending order) 2. firstname (ascending order) 3. lastname (ascending order) 4. birth_address (ascending order) 5. first_teamname (ascending order) 6. second_teamname (ascending order)

14. Insert the details of the players and awards given below to awardsplayers table. Use the data given to take care of any key constraints that might occur. Use the updated tables for queries 14 to 30.Empty cell for league id implies empty string, for the column "tie" it implies NULL value and empty string for league id as it can't be NULL.

award id	player id	first name	last name	league id	year id	tie
Best Baseman	dunphil02	Phil	Dunphy		2014	True
Best Baseman	tuckcam01	Cameron	Tucker		2014	True
ALCS MVP	scottm02	Michael	Scott	AA	2015	False
Triple Crown	waltjoe	Joe	Walt		2016	
Gold Glove	adamswi01	Willie	Adams		2017	False
ALCS MVP	yostne01	Ned	Yost		2017	

For each award, output the player who has won it the maximum number of times. In case two players have won the award max number of times, output the alphabetically first one based on playerid.

- Output format: awardid, playerid, firstname, lastname, num_wins
- Order by: 1. awardid (ascending order) 2. num_wins (descending order)

15. For each team in every season between 2000 and 2010 (including both 2000 and 2010), output the first manager that managed the team (managerial order = 0,1) for that particular season. In case, more than one managers co-managed the team first, output all of them.

- Output format: teamid, teamname, seasonid, managerid, managerfirstname, managerlastname
- Order by: 1. teamid (ascending order) 2. teamname (ascending order) 3. seasonid (descending order) 4. managerid (ascending order) 5. managerfirstname (ascending order) 6. managerlastname (ascending order)

16. Find the college's names (last college if they attended multiple) of top 10 players having the highest number of awards. If the player did not attend any college then put empty string as college's name. Also, note that we are asking for highest number of awards and NOT highest number of distinct awards.

- Output format: playerid, colleges_name, total_awards
- Order by: 1. total_awards (descending order), 2. colleges_name (ascending order), 3. playerid (ascending order)

17. Output people who have won awards as both players and managers. Also, output the first award won as player and the first award won as manager by such people. In case of tie, output alphabetically first award.

- Output format: playerid, firstname, lastname, playerawardid, playerawardyear, managerawardid, managerawardyear

- Order by: 1. playerid (ascending order) 2. firstname (ascending order) 3. lastname (ascending order)

18. Output the people who have been honored in at least two different categories in the hall of fame and have also played as an all-star player at least once in their career. Also output the first season such person played as an all-star player. A player is honored if the entry is present in halloffame table (ignore induct column). Similarly, if an entry is present in allstarfull, the player is considered an all star player. However, since it is asked that the player must have played as an all star player, put in the constraint GP = 1 for allstarfull table. num_honored_categories means number of different categories the player has been honoured in, in halloffame.

    - Output format: playerid, firstname, lastname, num_honored_categories, seasonid
    - Order by: 1. num_honored_categories (descending order) 2. playerid (ascending order) 3. firstname (ascending order) 4. lastname (ascending order) 5. seasonid (ascending order)

19. Output the players who have played at least two positions out of first baseman, second baseman, and third baseman sometime in their career, along with the total number of games (G_all) they have played in their career. Also, output the number of games played by these players as first baseman (G_1b), second baseman (G_2b) and third baseman (G_3b) across their career.

    - Output format: playerid, firstname, lastname, G_all, G_1b, G_2b, G_3b
    - Order by: 1. G_all (descending order) 2. playerid (ascending order) 3. firstname (ascending order) 4. lastname (ascending order) 5. G_1b (descending order) 6. G_2b (descending order) 7. G_3b (descending order)

20. Output top 5 schools with the highest number of players. Also output the players who have studied in these schools.

    - Output format: schoolid, schoolname, schooladdr, playerid, firstname, lastname
    - Order by: 1. schoolid (ascending order) 2. schoolname (ascending order) 3. schooladdr (ascending order) 4. playerid (ascending order) 5. firstname (ascending order) 6. lastname (ascending order)

21. Output players born in the same city and state who have either both batted or both pitched in the same team at least once in their career. Note that players need not have played for that team in the same season. Also, output another column stating whether these players batted for the same team (batted) or pitched for the same team (pitched) or did both for the same team (both) and name the column "role". Ignore the entries where birthcity or birthstate is null. You have to output every such pair of players, say players A and B satisfy the constraint, then you must output two entries: one with A as player1, B with player2 and second with B as player1, A as player2.

    - Output format: player1_id, player2_id, birthcity, birthstate, role
    - Order by: 1. birthcity (ascending order) 2. birthstate (ascending order) 3. player1_id (ascending order) 4. player2_id (ascending order)

22. For each award in every season, output the players who have received more than (>=) average points for that award in that season.

- Output format: awardid, seasonid, playerid, playerpoints, averagepoints
- Order by: 1. awardid (ascending order) 2. seasonid (ascending order) 3. playerpoints (descending order) 4. playerid (ascending order)

23. Output people who have never received any awards (neither as player nor as manager). Also, output another column (datatype: bool) stating whether or not the person is alive: True for alive, False otherwise. Name the column "alive".

    - Output format: playerid, playername, alive
    - Order by: 1. playerid (ascending order) 2. playername (ascending order)

24. Using Graph 1, find whether there exists a path of length three or more between webbbr01 and clemero02. Output a boolean value: True for yes and False for no.

    - Output format: pathexists

25. Using Graph 1, find the shortest path between garcifr02 and leagubr01. Output the path length. Output 0 if no such path exists.

    - Output format: pathlength

26. Using Graph 2, count the number of paths from team ARI to DET.

    - Output format: count

27. Using Graph 2, find the teams that are at most three edges away from team HOU. In other words, find all the teams such that there exists a path from HOU to the team which is of length 3 or less. Do not output HOU itself. In case there is a team that occurs twice in the output, keep the entry with larger num_hops.

    - Output format: teamid, num_hops
    - Order by: 1. teamid (ascending order)

28. Using graph 2, find the longest path from WS1 to any other team. Output the team at which this longest path ends. Also output the path length. In case of tie, output all end teams in alphabetical order.

    - Output format: teamid, teamname, pathlength
    - Order by: 1. teamid (ascending order) 2. teamname (ascending order)

29. Consider seriespost table, find the winner teams that have ties greater than losses in at least one season, lets call these teams set A. Using graph 2, find the shortest path from Set A teams to team NYA. You have to output the shortest path from each team in set A to NYA.

    - Output format: teamid, pathlength
    - Order by: 1. teamid (ascending order) 2. pathlength (ascending order)

30. Using Graph 2, find the longest cycle containing DET. Output the length of this cycle and the number of such cycles present in the graph. Output cyclelength as 0 if no such cycle exists.

    - Output format: cyclelength, numcycles