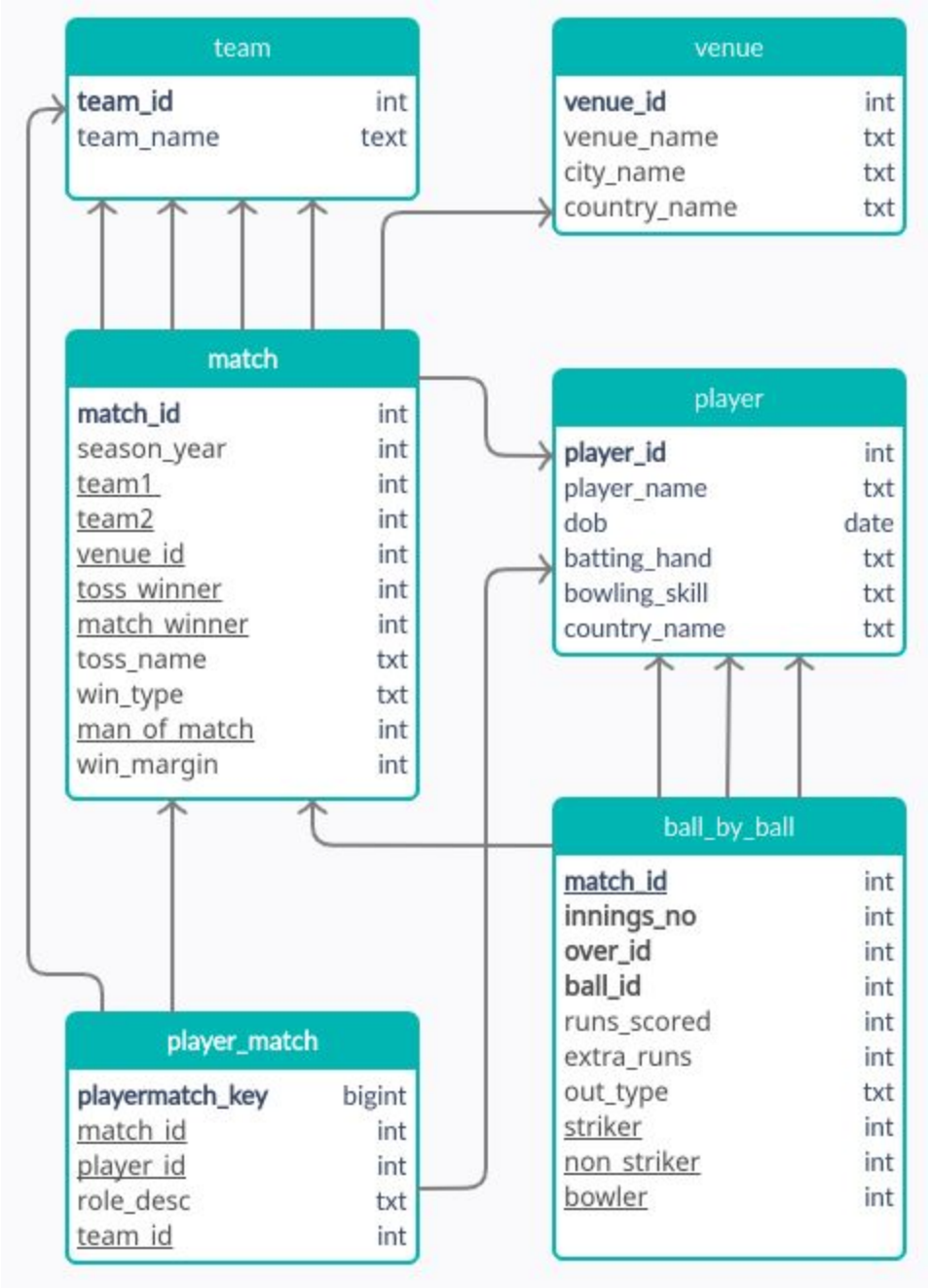
387 - Lab 3 - Advanced SQL

INLAB:

1. Write DDL for the schema shown below. Your DDL file should contain both the table definitions as well as the constraints specified at the end of the schema.
2. With the given set of CSV files that contain data for this schema, write a Python program to generate a SQL script file to upload data into Postgres.

Schema:



Constraints:

- Primary Key (bold) and Foreign Key (underlined) constraints present in the given schema
- **out_type** can only take the values 'caught', 'caught and bowled' , 'bowled' , 'stumped' , 'retired hurt', 'keeper catch', 'lbw', 'run out', ‘hit wicket’ or NULL
- **role_desc** can only take the values 'Player', 'Keeper', 'CaptainKeeper' or 'Captain'
- **toss_name** can only take the values 'field' or 'bat'
- **win_type** can only take the values 'wickets', 'runs' or NULL
- **runs_scored** in ball_by_ball should be between 0 and 6
- **innings_no** in ball_by_ball should be either 1 or 2 only.

OUTLAB - Write SQL statements for the following queries:

Note:
In every question(except q.6) don't include extra_runs in the runs scored by a batsman.
In the wickets taken by a bowler don't include the wickets with out_type as 'run out', 'hit wicket' and 'retired hurt'.

1. Find, for each match venue, the average number of runs scored per match (total of both teams) in the stadium. You can get the runs scored from the ball_by_ball table. Output <venue_name, avg_runs> , in descending order of average runs per match.

   Note : Compute avg_run up to 3 decimal places only.

2. Find players who faced the maximum number of balls per match on average; a batsman faced a ball if there is an entry in ball_by_ball with that player as the striker. Limit your answer to the top 10 by using sparse rank (you may get more than 10 in case of ties).

   Output <player_id, player_name, avg cnt>

3. Find players who are the most frequent six hitters, that is, players who hit a 6 in the highest fraction of balls that they face. Output the player id, player name, the number of times the player has got 6 runs in a ball, the number of balls faced, and the fraction of 6s. Output <player_id, player_name, numsixes, numballs, frac>

   (Note 1: The striker attribute in the ball_by_ball relation is the player who scored the runs.)

   (Note 2: Int divided by int gives an int, so make sure to multiply by 1.0 before division.)

4. Find top 3(exactly 3) batsmen’ and top 3(exactly 3) bowlers’ player_ids who got highest no of runs and highest no of wickets respectively in each season. Output (season_year, batsman, runs, bowler, wickets). Here batsman & bowler are player_ids of the players. Incase of ties output the player with lesser player_id first. Order by season_year (earlier year comes first)

and rank(batsman and bowler with more no of runs and wickets in a particular season comes first). There will be (no_of_seasons*3) rows.

5. Find the ids of players who got the highest no of partnership runs for each match. There can be multiple rows for a single match. Output <match_id, player1_id, player2_id, runs1, runs2, pship_runs>(player1's contribution i.e. runs1 >= player2's contribution i.e. runs2), in descending order of pship_runs (incase of ties compare match_id in ascending order).

   Note: extra_runs shouldn't be counted

6. For all the matches with win type as 'wickets', find the over ids in which the runs scored are less than 6 runs. Output <match_id, innings_no, over_id>.

   Note : Runs scored in an over also include the extra_runs.

7. List top 5 batsmen(exactly 5) by number of sixes hit in the season 2013? Break ties alphabetically. Output <player_name>.

8. List 5 bowlers(exactly 5) by lowest strike rate(average number of balls bowled per wicket taken) in the season 2013? Break ties alphabetically. Output <player_name>.

9. For each country(with at least one player bowled out) find out the number of its players who were bowled out in any match. Output <country_name, count>. Here the country is the home country of the player

10. List the names of right- handed players who have scored at least a century in any match played in 'Pune'. Order the output alphabetically on player_name. Output <player_name>.

11. Find the win percentage for all the teams that have won at least one match(across all seasons). Order the result alphabetically on team names. Output <team_name, win_percentage>.

    Win percentage of a team can be calculated as = (number of matches won by the team / total number of the matches played by the team) * 100

    Note : Calculate percentage upto 3 decimal places.