CS 432: Databases

Assignment 3: SQL and Database Design

Name: Shruti Katpara
Roll No: 18110084

1. Write SQL queries for the following questions. Questions ‘a’ to ‘g’ carry 2 marks each. Questions ‘h’-’j’ carry 1 mark each. (17 marks+5 marks)
   Note: Required output attribute(s) are given next to each query, also export each output in Q1X.csv, where X is a,b...j.
   Any deviation from the given format would result in zero marks.

   a. For all the matches_id(entire IPL), find the minimum runs scored in any over and the bowler who bowled that over. Sort by increasing match_id, followed by increasing innings_no, then finally by increasing over_ids. Output: <bowler_name><runs_scored>
   Note: Runs scored in an over is the sum of the batsmen_scored+ extra_runs(wides and "no_balls" only. It should not be match specific)

```
create view onea as
select ball_by_ball.match_id, ball_by_ball.innings_no, ball_by_ball.over_id,
ball_by_ball.bowler, player.player_name , (sum(coalesce(extra_runs.extra_runs,
0)) + sum(coalesce(runs_scored,0))) as total_runs
from ball_by_ball
left join extra_runs on ( (extra_runs.extra_type = "noballs" or
extra_runs.extra_type="wides") and ball_by_ball.match_id = extra_runs.match_id
and ball_by_ball.over_id = extra_runs.over_id and ball_by_ball.ball_id =
extra_runs.ball_id and ball_by_ball.innings_no = extra_runs.innings_no)
left join batsman_scored on (ball_by_ball.match_id = batsman_scored.match_id
and ball_by_ball.over_id = batsman_scored.over_id and ball_by_ball.ball_id =
batsman_scored.ball_id and ball_by_ball.innings_no =
batsman_scored.innings_no)
left join player on (ball_by_ball.bowler = player.player_id)
group by ball_by_ball.match_id, ball_by_ball.innings_no, ball_by_ball.over_id
order by ball_by_ball.match_id, ball_by_ball.innings_no, ball_by_ball.over_id
;

select over_id, player_name, min(total_runs) as min_runs
from onea
group by over_id
```

```
order by match_id, innings_no, over_id
;
drop view onea;
```

b. Find the names of all the batsmen(players) and the frequency of their “caught” out in increasing order of the number of “caught”. If a tie occurs, sort names alphabetically. Hint: Frequency can be 0 too. <names><frequency>

```
select player.player_name, count(temp.kind_out) as frequency
from player
left outer join (select * from wicket_taken where wicket_taken.kind_out = "caught") as temp
on temp.player_out = player.player_id
group by player.player_id
order by frequency, player.player_name
;
```

c. List the stadium(s) where the maximum number of “legbyes” (runs) is taken. If ties occur, show alphabetical order. <venue_name><number_of_legbye_runs>

```
create view onec as
select match_match.venue, count(*) as number_of_legbye_runs
from match_match
natural join extra_runs
where extra_type = "legbyes"
group by venue
order by number_of_legbye_runs desc, venue
;

select *
from onec
where number_of_legbye_runs in
(select max(number_of_legbye_runs) from onec)
order by venue
;

drop view onec;
```

d. Find the bowler(s)(players) who has the best average(no. of runs given/wickets taken) in edition 5. If a tie occurs, sort names alphabetically. <bowler_name><average>

```
create view oned as
select  player.player_name,
case
when ( sum( coalesce( batsman_scored.runs_scored, 0 ) ) / count(
wicket_taken.player_out ) ) is not null
then ( sum( coalesce( batsman_scored.runs_scored, 0 ) ) / count(
wicket_taken.player_out ) )
when ( sum( coalesce( batsman_scored.runs_scored, 0 ) ) / count(
wicket_taken.player_out ) ) is null
then 1000000000000
end as average
from ball_by_ball
left join batsman_scored on (ball_by_ball.match_id = batsman_scored.match_id
and ball_by_ball.over_id = batsman_scored.over_id and ball_by_ball.ball_id =
batsman_scored.ball_id and ball_by_ball.innings_no =
batsman_scored.innings_no)
left join wicket_taken on (ball_by_ball.match_id = wicket_taken.match_id and
ball_by_ball.over_id = wicket_taken.over_id and ball_by_ball.ball_id =
wicket_taken.ball_id and ball_by_ball.innings_no = wicket_taken.innings_no)
left join player on (ball_by_ball.bowler = player.player_id)
left join match_match on (ball_by_ball.match_id = match_match.match_id)
where match_match.season_id = 5
group by match_match.season_id, ball_by_ball.bowler
order by average, player.player_name
;

select *
from oned
where average in
(select min(average) from oned)
order by player_name
;

drop view oned;
```

e. Find out the names of all batsmen(players) who scored more than 100 runs in a match and, their runs scored. Sort names alphabetically. (if multiple entries of the same player, show the one with the highest runs).<batsmen_name><runs>

```
create view onee as
select match_id, striker, player_id, player_name, sum(runs_scored) as total_run
from ball_by_ball
```

```
natural join batsman_scored
inner join player on player.player_id = ball_by_ball.striker
group by match_id, striker
having sum(runs_scored) > 100
order by player_name asc
;

select player_name, max(total_run) as total_runs
from onee
group by player_name
;

drop view onee;
```

f. Find out the top 3 batsmen(players) whose [number of runs scored/number of matches played] is the best in edition 2. Sort alphabetically. <batsman_name><value>

```
select player.player_name, sum( runs_scored ) / count( distinct(ball_by_ball.match_id)) as average
from ball_by_ball
left join batsman_scored on (ball_by_ball.match_id = batsman_scored.match_id and ball_by_ball.over_id = batsman_scored.over_id and ball_by_ball.ball_id = batsman_scored.ball_id and ball_by_ball.innings_no = batsman_scored.innings_no)
left join player on (ball_by_ball.striker = player.player_id)
left join match_match on (ball_by_ball.match_id = match_match.match_id)
where match_match.season_id = 2
group by match_match.season_id, ball_by_ball.striker
order by average desc, player.player_name
limit 3
;
```

g. Find out the batting average(as calculated in the above question (f)) of all players. Then only show the list of the top 3 countries with the highest country batting average(∑batting average/Total number of players in that country)<country><value>

```
create view oneg as
select player.player_name as batsman, player.country_name, sum( coalesce(runs_scored,0) ) / count( distinct( ball_by_ball.match_id)) as average
```

```
from ball_by_ball
left join batsman_scored on (ball_by_ball.match_id = batsman_scored.match_id
and ball_by_ball.over_id = batsman_scored.over_id and ball_by_ball.ball_id =
batsman_scored.ball_id and ball_by_ball.innings_no =
batsman_scored.innings_no)
left join player on (ball_by_ball.striker = player.player_id)
group by striker
;

select country_name,sum(average)/count( distinct(batsman)) as country_avg
from oneg
group by country_name
order by country_avg desc
limit 3
;

drop view oneg;
```

h. Write down a simple query to make a copy of the player table(with data).

```
CREATE TABLE player_copy LIKE player;
INSERT INTO player_copy SELECT * FROM player;

select * from player_copy;

drop table player_copy;
```

i. Using view, create a table say “indian_players” which contains information about the total runs scored by all the Indian players till now and sort them alphabetically.<name><runs>

```
create view onei as
select player.player_name, player.country_name , sum( coalesce(runs,0)) as
total_runs
from player
left join
(select ball_by_ball.striker as id, batsman_scored.runs_scored as runs
from ball_by_ball, batsman_scored
where ball_by_ball.match_id = batsman_scored.match_id and
ball_by_ball.over_id = batsman_scored.over_id and ball_by_ball.ball_id =
batsman_scored.ball_id and ball_by_ball.innings_no =
batsman_scored.innings_no
) as temp on temp.id = player.player_id and player.country_name = "India"
group by player.player_id
;

create view indian_players as
select player_name, total_runs
from onei
where country_name = "India"
order by player_name
;

select * from indian_players;

drop view onei;
drop view indian_players;
```

j. List all captains who scored more than 50 runs in edition 3. Sort names alphabetically <name><runs>

```
select player.player_name, sum(batsman_scored.runs_scored) as total_runs
from ball_by_ball
inner join match_match on ( match_match.season_id=3 and
match_match.match_id = ball_by_ball.match_id)
inner join player_match on ( player_match.role = "Captain" and
player_match.match_id = ball_by_ball.match_id and player_match.player_id =
ball_by_ball.striker)
inner join batsman_scored on (ball_by_ball.match_id =
batsman_scored.match_id and ball_by_ball.over_id = batsman_scored.over_id
and ball_by_ball.ball_id = batsman_scored.ball_id and ball_by_ball.innings_no =
batsman_scored.innings_no)
inner join player on player.player_id = ball_by_ball.striker
group by ball_by_ball.striker
having total_runs > 50
;

select player.player_name, sum(batsman_scored.runs_scored) as total_runs
from batsman_scored
inner join ball_by_ball on (ball_by_ball.match_id = batsman_scored.match_id
and ball_by_ball.over_id = batsman_scored.over_id and ball_by_ball.ball_id =
batsman_scored.ball_id and ball_by_ball.innings_no =
batsman_scored.innings_no)
```

```
inner join match_match on ( match_match.season_id=3 and
match_match.match_id = batsman_scored.match_id)
inner join player_match on ( player_match.role = "Captain" and
player_match.match_id = batsman_scored.match_id and player_match.player_id
= ball_by_ball.striker)
inner join player on player.player_id = ball_by_ball.striker
group by ball_by_ball.striker
having total_runs > 50
;
```

2. Suppose a user creates a new relation r1 with a foreign key referencing another relation r2. What authorization privilege does the user need on r2? Why should this not simply be allowed without any such authorization? (max 500 words) (4 marks)

   Creating a foreign key constraint requires at least one of the SELECT, INSERT, UPDATE, DELETE, or REFERENCES privileges on the parent table.

   If the user do not haveany of privileges on r2 then it will not be able to create foreign key constraint on relation r1. Because the job of foreign key is ensure the data consistency. And it cannot be ensured without having any privileges on the parent relation.

3. Explain the difference between integrity constraints and authorization constraints. (explain them with examples) (max 500 words) (4 marks)

Integrity constraint	Authorization constraint
It ensures the data consistency among the relations when data is modified by any authorised users	It ensures data security and handles privileges of the relations and views among different users.
Ex: referential integrity constraint, entity integrity constraint, default value constraint, etc	Ex: read, write, update,delete privilege to different relations, etc
Suppose a primary key P is defined in some relation A and same primary key is referred as foreign key in relation B. When the data entry of primary key P is deleted by the owner then due to the integrity constraint, all the data entries having primary key P referred as foreign key in other relations will be deleted.	For a relation A, the owner can restrict some users from updating some columns or the whole relation having privileged data to ensure data security.

	The owner who created a relation has all the access for that relation

4. Consider a set of users A, B, C, D, and E. Suppose the user A creates a table T and thus is the owner of T. Now suppose the following set of statements is executed in order:

   1. User A: grant select on T to B, C with grant option
   2. User B: grant select on T to C
   3. User C: grant select on T to D, E
   4. User A: grant select on T to E
   5. User A: revoke select on T from B restrict
   6. User A: revoke select on T from C cascade

   Let SELECT ON T be called P
   A has all the privileges, as it is the owner of the table
   After stat 1: B,C will also have privilege P granted by A
   After stat 2: C will have privilege P granted by B
   After stat 3: D,E will also have privilege P granted by C
   After stat 4: E will have privilege P granted by A
   After stat 5: B's privilege P will be revoked by A. And as it is restricted revoke, there will not be a cascading effect.
   After stat 6: C's privilege P will be revoked by A. And as it is cascading revoke, D's privilege will also be revoked but not E's because user A has granted privilege to E.

   - When does D not have SELECT ON T privilege? Justify your answer. (3 marks)
     According to above explanation of actions, D will not have privilege after 6th statement
   - What permissions does C have at the end of statement 5? Justify your answer. (2 marks)

     C will have permission of P given by both B and A
     Because after stat 5 only B's privilege was restricted.