Mohamed Shamir | 17110084

Contents

Database Schema

The Schema for the following problems is as shown below. The integrity constraints are also mentioned in the diagram.

player		
PK	player_id	int
	player_name	varchar(80)
	dob	date
	batting_hand	varchar(80)
	bowling_skill	varchar(80)
	country_name	varchar(80)

wicket_taken		
PK	match_id	int
PK	over_id	int
PK	ball_id	int
	player_out	int
	kind_out	varchar(50)
PK	innings_no	int

batsman_scored		
PK	match_id	int
PK	over_id	int
PK	ball_id	int
	runs_scored	int
PK	innings_no	int

team		
PK	team_id	int
	team_name	varchar(80)

player_match		
PK	match_id	int
	player_id	int
	role	varchar(50)
	team_id	int

match		
PK	match_id	int
	team_1	int
	team_2	int
	match_date	date
	season_id	int
	venue	varchar(50)
	toss_winner	int
	toss_decision	varchar(50)
	win_type	varchar(50)
	win_margin	int
	outcome_type	varchar(50)
	match_winner	int
	man_of_the_match	int

ball_by_ball		
PK	match_id	int
PK	over_id	int
PK	ball_id	int
PK	innings_no	int
	team_batting	int
	team_bowling	int
	striker_batting_position	int
	striker	int
	non_striker	int
	bowler	int

extra_runs		
PK	match_id	int
PK	over_id	int
PK	ball_id	int
	extra_type	varchar(50)
	extra_runs	int
PK	innings_no	int

Creating the database

- Python Scripts are used to generate sql insertion codes.
- Python Script can be accessed using this Link
- All the sql insertion codes are avalible in the insert directory.

```
├── create_table.sql
├── data_insert.ipynb
├── dataset
│   ├── ball_by_ball.csv
│   ├── batsman_scored.csv
│   ├── extra_runs.csv
│   ├── match.csv
│   ├── player.csv
│   ├── player_match.csv
│   ├── team.csv
│   └── wicket_taken.csv
├── images
│   └── schema.png
├── insert
│   ├── ball_by_ball.sql
│   ├── batsman_scored.sql
│   ├── extra_runs.sql
│   ├── match.sql
│   ├── player_match.sql
│   ├── player.sql
│   ├── team.sql
│   └── wicket_taken.sql
├── readme.md
└── run_all.sh
```

- To create the database and insert all the data. Run run_all.sh

```
source run_all.sh
```

Get output in CSV

```
sudo mysql < ./sql_queries/Q1b.sql| sed 's/\t/,/g' > ./results/Q1b.csv
```

Problem 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)

Solution:

```
select player.player_name as bowler_name, last.runs_scored
from
(select FINAL.bowler,min(FINAL.runs_given) runs_scored
from
(select C.bowler,C.match_id,C.innings_no,C.over_id,
SUM(C.runs) runs_given
from
(select B.match_id,B.over_id,B.ball_id,B.innings_no,B.runs,
ball_by_ball.bowler
from
(select A.match_id,A.over_id,A.ball_id,A.innings_no,A.runs
from
(select batsman_scored.match_id,batsman_scored.over_id,
batsman_scored.ball_id,batsman_scored.innings_no,
batsman_scored.runs_scored+ifnull(0,extra_runs.extra_runs) runs
 from batsman_scored left outer join extra_runs
on batsman_scored.match_id=extra_runs.match_id
and batsman_scored.over_id=extra_runs.over_id
and batsman_scored.ball_id=extra_runs.ball_id
and batsman_scored.innings_no=extra_runs.innings_no) as A
inner join match_details on
match_details.match_id=A.match_id) as B
inner join ball_by_ball
on
ball_by_ball.match_id=B.match_id
and ball_by_ball.over_id=B.over_id
and ball_by_ball.innings_no=B.innings_no
and ball_by_ball.ball_id=B.ball_id) as C
group by C.bowler,C.match_id,C.innings_no,C.over_id
order by runs_given asc,C.match_id,C.innings_no,C.over_id) as FINAL
group by FINAL.bowler) as last
inner join player
on
player.player_id=last.bowler order by runs_scored;
```

```
mysql> select player.player_name as bowler_name, last.runs_scored
    -> from
    -> (select FINAL.bowler,min(FINAL.runs_given) runs_scored
    -> from
    -> (select C.bowler,C.match_id,C.innings_no,C.over_id,
    -> SUM(C.runs) runs_given
    -> from
    -> (select B.match_id,B.over_id,B.ball_id,B.innings_no,B.runs,
    -> ball_by_ball.bowler
    -> from
    -> (select A.match_id,A.over_id,A.ball_id,A.innings_no,A.runs
    -> from
    -> (select batsman_scored.match_id,batsman_scored.over_id,
    -> batsman_scored.ball_id,batsman_scored.innings_no,
    -> batsman_scored.runs_scored+ifnull(0,extra_runs.extra_runs) runs
    ->  from batsman_scored left outer join extra_runs
    -> on batsman_scored.match_id=extra_runs.match_id
    -> and batsman_scored.over_id=extra_runs.over_id
    -> and batsman_scored.ball_id=extra_runs.ball_id
    -> and batsman_scored.innings_no=extra_runs.innings_no) as A
    -> inner join match_details on
    -> match_details.match_id=A.match_id) as B
    -> inner join ball_by_ball
    -> on
    -> ball_by_ball.match_id=B.match_id
    -> and ball_by_ball.over_id=B.over_id
    -> and ball_by_ball.innings_no=B.innings_no
    -> and ball_by_ball.ball_id=B.ball_id) as C
    -> group by C.bowler,C.match_id,C.innings_no,C.over_id
    -> order by runs_given asc,C.match_id,C.innings_no,C.over_id) as FINAL
    -> group by FINAL.bowler) as last
    -> inner join player
    -> on
    -> player.player_id=last.bowler order by runs_scored;
+------------------+-------------+
| bowler_name      | runs_scored |
+------------------+-------------+
| SR Watson        |           0 |
| DS Kulkarni      |           0 |
| MM Patel         |           0 |
| WD Parnell       |           0 |
| JP Faulkner      |           0 |
| A Mishra         |           0 |
| AC Thomas        |           0 |
| SP Narine        |           0 |
| SM Pollock       |           0 |
| S Kaul           |           0 |
| Sohail Tanvir    |           0 |
| AR Patel         |           0 |
| KH Pandya        |           0 |
| JW Hastings      |           0 |
```

- SQL Query file for the same. Link
- Result file. Link

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 >

Dont forget to come back fix the zero issue

Solution:

```sql
select player_name as name,count(player_out) as frequency
from player inner join wicket_taken on
player.player_id=wicket_taken.player_out
and wicket_taken.kind_out='caught' group by player_out
order by Frequency DESC,player_name ASC;
```

```
mysql> select player_name,count(player_out) as Frequency from player inner join wicket_taken on player.player_id=wicket_taken.player_out and wicket_taken.kind_out='caugh
t' group by player_out order by Frequency DESC,player_name ASC;
+-------------------+-----------+
| player_name       | Frequency |
+-------------------+-----------+
| SK Raina          |        86 |
| RV Uthappa        |        78 |
| RG Sharma         |        77 |
| V Sehwag          |        69 |
| Yuvraj Singh      |        68 |
| V Kohli           |        66 |
| G Gambhir         |        65 |
| KD Karthik        |        64 |
| YK Pathan         |        63 |
| MS Dhoni          |        60 |
| JH Kallis         |        59 |
| DA Warner         |        55 |
| S Dhawan          |        54 |
| AC Gilchrist      |        53 |
| BB McCullum       |        52 |
| AT Rayudu         |        51 |
| NV Ojha           |        50 |
| AB de Villiers    |        48 |
| M Vijay           |        48 |
| KA Pollard        |        47 |
| PA Patel          |        47 |
| R Dravid          |        47 |
| AM Rahane         |        44 |
| RA Jadeja         |        43 |
| CH Gayle          |        42 |
| DR Smith          |        42 |
| SR Watson         |        42 |
```

- SQL Query file for the same. Link
- Result file. Link

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>

```
select venue as venue_name,SUM(extra_runs.extra_runs) number_of_legbye_runs
from extra_runs inner join match_details on
match_details.match_id=extra_runs.match_id
and extra_runs.extra_type='legbyes' group by venue having
SUM(extra_runs.extra_runs)=(select max(Total_Legbye_Runs ) maximum_legbye
from
(select match_details.venue,SUM(extra_runs.extra_runs) as Total_Legbye_Runs
from extra_runs,match_details where
match_details.match_id=extra_runs.match_id
and extra_runs.extra_type='legbyes' group by match_details.venue ) as K)
order by venue;
```

```
mysql> select venue,SUM(extra_runs.extra_runs) Total_Legbye_Runs
    ->  from extra_runs inner join match_details on match_details.match_id=extra_runs.match_id
    ->  and extra_runs.extra_type='legbyes' group by venue having SUM(extra_runs.extra_runs)=(select max(Total_Legbye_Runs ) maximum_le
gbye from
    ->  (select match_details.venue,SUM(extra_runs.extra_runs) as Total_Legbye_Runs
    ->  from extra_runs,match_details where match_details.match_id=extra_runs.match_id
    ->  and extra_runs.extra_type='legbyes' group by match_details.venue ) as K) order by venue;
+--------------+-------------------+
| venue        | Total_Legbye_Runs |
+--------------+-------------------+
| Eden Gardens |               330 |
+--------------+-------------------+
1 row in set (0.03 sec)
```

- SQL Query file for the same. Link
- Result file. Link

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 >

Solution:

```
select player.player_name,FINAL.average
from
(select
RUNS_GIVEN.bowler,RUNS_GIVEN.runs_given/ifnull(1,WICKET_TAKEN.wickets)
average
from
(select C.bowler, SUM(C.runs) runs_given
from
(select B.match_id,B.over_id,B.ball_id,B.innings_no,B.runs,
ball_by_ball.bowler
from
(select A.If,A.over_id,A.ball_id,A.innings_no,A.runs
from
(select batsman_scored.match_id,batsman_scored.over_id,
batsman_scored.ball_id,batsman_scored.innings_no,
batsman_scored.runs_scored+ifnull(0,extra_runs.extra_runs) runs
```

```
 from batsman_scored left outer join extra_runs
on batsman_scored.match_id=extra_runs.match_id
and batsman_scored.over_id=extra_runs.over_id
and batsman_scored.ball_id=extra_runs.ball_id
and batsman_scored.innings_no=extra_runs.innings_no) as A
inner join match_details on
match_details.season_id=5
and
match_details.match_id=A.match_id) as B
inner join ball_by_ball
on
ball_by_ball.match_id=B.match_id
and ball_by_ball.over_id=B.over_id
and ball_by_ball.innings_no=B.innings_no
and ball_by_ball.ball_id=B.ball_id) as C
group by C.bowler) as RUNS_GIVEN
left outer join
(select B.bowler,COUNT(B.match_id) wickets
from
(select A.match_id,A.over_id,
A.ball_id,A.innings_no,ball_by_ball.bowler
from
(select wicket_taken.match_id,wicket_taken.over_id,
wicket_taken.ball_id,wicket_taken.innings_no
from wicket_taken inner join match_details
on wicket_taken.match_id=match_details.match_id
and match_details.season_id=5) as A
inner join ball_by_ball
on
ball_by_ball.match_id=A.match_id
and ball_by_ball.over_id=A.over_id
and ball_by_ball.innings_no=A.innings_no
and ball_by_ball.ball_id=A.ball_id) as B
group by B.bowler) as WICKET_TAKEN on
WICKET_TAKEN.bowler=RUNS_GIVEN.bowler) as FINAL
inner join player on
FINAL.bowler= player.player_id order by
FINAL.average ASC,player.player_name ASC;
```

```
mysql> select player.player_name,FINAL.bowling_average
    -> from
    -> (select RUNS_GIVEN.bowler,RUNS_GIVEN.runs_given/ifnull(1,WICKET_TAKEN.wickets)
    -> bowling_average
    -> from
    -> (select C.bowler, SUM(C.runs) runs_given
    -> from
    -> (select B.match_id,B.over_id,B.ball_id,B.innings_no,B.runs,
    -> ball_by_ball.bowler
    -> from
    -> (select A.match_id,A.over_id,A.ball_id,A.innings_no,A.runs
    -> from
    -> (select batsman_scored.match_id,batsman_scored.over_id,
    -> batsman_scored.ball_id,batsman_scored.innings_no,
    -> batsman_scored.runs_scored+ifnull(0,extra_runs.extra_runs) runs
    ->  from batsman_scored left outer join extra_runs
    -> on batsman_scored.match_id=extra_runs.match_id
    -> and batsman_scored.over_id=extra_runs.over_id
    -> and batsman_scored.ball_id=extra_runs.ball_id
    -> and batsman_scored.innings_no=extra_runs.innings_no) as A
    -> inner join match_details on
    -> match_details.season_id=5
    -> and
    -> match_details.match_id=A.match_id) as B
    -> inner join ball_by_ball
    -> on
    -> ball_by_ball.match_id=B.match_id
    -> and ball_by_ball.over_id=B.over_id
    -> and ball_by_ball.innings_no=B.innings_no
    -> and ball_by_ball.ball_id=B.ball_id) as C
    -> group by C.bowler) as RUNS_GIVEN
    -> left outer join
    -> (select B.bowler,COUNT(B.match_id) wickets
    -> from
    -> (select A.match_id,A.over_id,
    -> A.ball_id,A.innings_no,ball_by_ball.bowler
    -> from
    -> (select wicket_taken.match_id,wicket_taken.over_id,
    -> wicket_taken.ball_id,wicket_taken.innings_no
    -> from wicket_taken inner join match_details
    -> on wicket_taken.match_id=match_details.match_id
    -> and match_details.season_id=5) as A
    -> inner join ball_by_ball
    -> on
    -> ball_by_ball.match_id=A.match_id
    -> and ball_by_ball.over_id=A.over_id
    -> and ball_by_ball.innings_no=A.innings_no
    -> and ball_by_ball.ball_id=A.ball_id) as B
    -> group by B.bowler) as WICKET_TAKEN on
    -> WICKET_TAKEN.bowler=RUNS_GIVEN.bowler) as FINAL
    -> inner join player on
    -> FINAL.bowler= player.player_id order by
    -> FINAL.bowling_average ASC,player.player_name ASC;
+-----------------+-----------------+
| player_name     | bowling_average |
+-----------------+-----------------+
| SPD Smith       |          5.0000 |
| Y Nagar         |          8.0000 |
| AG Murtaza      |         11.0000 |
| LR Shukla       |         12.0000 |
| AUK Pathan      |         14.0000 |
| AM Nayar        |         16.0000 |
| AN Ahmed        |         16.0000 |
| F du Plessis    |         16.0000 |
| PC Valthaty     |         16.0000 |
| RG Sharma       |         16.0000 |
| RN ten Doeschate|         20.0000 |
| BJ Hodge        |         21.0000 |
| GJ Maxwell      |         21.0000 |
| AL Menaria      |         23.0000 |
| LJ Wright       |         24.0000 |
| CL White        |         25.0000 |
| DJ Harris       |         26.0000 |
| Y Venugopal Rao |         26.0000 |
| NLTC Perera     |         27.0000 |
| R Shukla        |         30.0000 |
```

- SQL Query file for the same. Link
- Result file. Link

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 >

Solution:

```
select C.player_name as batsmen_name,max(C.Total_runs) runs from
(select P.match_id,P.player_name,P.Total_runs from
(select K.match_id,K.player_name,SUM(K.runs_scored) Total_runs
from
(select B.match_id,B.runs_scored,player.player_name from
(select ball_by_ball.match_id,
batsman_scored.runs_scored, ball_by_ball.striker
from ball_by_ball left outer join batsman_scored
on batsman_scored.match_id=ball_by_ball.match_id
and batsman_scored.over_id=ball_by_ball.over_id
and batsman_scored.ball_id=ball_by_ball.ball_id
and batsman_scored.innings_no=ball_by_ball.innings_no) as B
left outer join player on
B.striker=player.player_id) as K
group by K.match_id,K.player_name) as P where P.Total_runs>100) as C
group by C.player_name order by C.player_name ASC;
```

```
mysql> select C.player_name,max(C.Total_runs) Max_runs from
    -> (select P.match_id,P.player_name,P.Total_runs from
    -> (select K.match_id,K.player_name,SUM(K.runs_scored) Total_runs
    -> from
    -> (select B.match_id,B.runs_scored,player.player_name from
    -> (select ball_by_ball.match_id,
    -> batsman_scored.runs_scored, ball_by_ball.striker
    -> from ball_by_ball left outer join batsman_scored
    -> on batsman_scored.match_id=ball_by_ball.match_id
    -> and batsman_scored.over_id=ball_by_ball.over_id
    -> and batsman_scored.ball_id=ball_by_ball.ball_id
    -> and batsman_scored.innings_no=ball_by_ball.innings_no) as B
    -> left outer join player on
    -> B.striker=player.player_id) as K
    -> group by K.match_id,K.player_name) as P where P.Total_runs>100) as C
    -> group by C.player_name order by C.player_name ASC;
+------------------+----------+
| player_name      | Max_runs |
+------------------+----------+
| A Symonds        |      117 |
| AB de Villiers   |      133 |
| AC Gilchrist     |      109 |
| AM Rahane        |      103 |
| BB McCullum      |      158 |
| CH Gayle         |      175 |
| DA Miller        |      101 |
| DA Warner        |      109 |
| DPMD Jayawardene |      110 |
| KP Pietersen     |      103 |
| M Vijay          |      127 |
| MEK Hussey       |      116 |
| MK Pandey        |      114 |
| PC Valthaty      |      120 |
| Q de Kock        |      108 |
| RG Sharma        |      109 |
| SE Marsh         |      115 |
| SPD Smith        |      101 |
| SR Watson        |      104 |
| ST Jayasuriya    |      114 |
| V Kohli          |      113 |
| V Sehwag         |      122 |
| WP Saha          |      115 |
+------------------+----------+
23 rows in set (0.39 sec)
```

- SQL Query file for the same. Link
- Result file. Link

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 >

Solution

```
select player.player_name as batsman_name,D.value
from
(select C.striker,SUM(C.runs_scored)/COUNT(C.match_id) value
from
(select B.match_id,B.striker,SUM(B.runs_scored) runs_scored
from
(select A.match_id,A.striker,batsman_scored.runs_scored
from
(select ball_by_ball.match_id,ball_by_ball.over_id,
ball_by_ball.ball_id, ball_by_ball.innings_no,
ball_by_ball.striker
from ball_by_ball inner join match_details
on ball_by_ball.match_id=match_details.match_id
and match_details.season_id=2) as A
inner join batsman_scored
on
A.match_id=batsman_scored.match_id
and A.over_id=batsman_scored.over_id
and A.ball_id=batsman_scored.ball_id
and A.innings_no = batsman_scored.innings_no) as B
group by B.match_id,B.striker) as C
group by C.striker order by value desc limit 3) as D
inner join
player on
player.player_id=D.striker;
```

```
mysql> select player.player_name as batsman_name,D.value
    -> from
    -> (select C.striker,SUM(C.runs_scored)/COUNT(C.match_id) value
    -> from
    -> (select B.match_id,B.striker,SUM(B.runs_scored) runs_scored
    -> from
    -> (select A.match_id,A.striker,batsman_scored.runs_scored
    -> from
    -> (select ball_by_ball.match_id,ball_by_ball.over_id,
    -> ball_by_ball.ball_id, ball_by_ball.innings_no,
    -> ball_by_ball.striker
    -> from ball_by_ball inner join match_details
    -> on ball_by_ball.match_id=match_details.match_id
    -> and match_details.season_id=2) as A
    -> inner join batsman_scored
    -> on
    -> A.match_id=batsman_scored.match_id
    -> and A.over_id=batsman_scored.over_id
    -> and A.ball_id=batsman_scored.ball_id
    -> and A.innings_no = batsman_scored.innings_no) as B
    -> group by B.match_id,B.striker) as C
    -> group by C.striker order by value desc limit 3) as D
    -> inner join
    -> player on
    -> player.player_id=D.striker;
+---------------+---------+
| batsman_name  | value   |
+---------------+---------+
| ML Hayden     | 47.6667 |
| MK Pandey     | 42.0000 |
| AB de Villiers | 35.7692 |
+---------------+---------+
3 rows in set (0.06 sec)
```

- SQL Query file for the same. Link
- Result file. Link

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 >

Solution:

- To calculate the batting average for all the players

```
select player.player_name as batsman_name,D.value
from
(select C.striker,SUM(C.runs_scored)/COUNT(C.match_id) value
from
(select B.match_id,B.striker,SUM(B.runs_scored) runs_scored
from
(select A.match_id,A.striker,batsman_scored.runs_scored
from
ball_by_ball as A
inner join batsman_scored
on
A.match_id=batsman_scored.match_id
and A.over_id=batsman_scored.over_id
and A.ball_id=batsman_scored.ball_id
and A.innings_no = batsman_scored.innings_no) as B
group by B.match_id,B.striker) as C
group by C.striker order by value desc) as D
inner join
player on
player.player_id=D.striker
order by value;
```

```
mysql> select player.player_name as batsman_name,D.value
    -> from
    -> (select C.striker,SUM(C.runs_scored)/COUNT(C.match_id) value
    -> from
    -> (select B.match_id,B.striker,SUM(B.runs_scored) runs_scored
    -> from
    -> (select A.match_id,A.striker,batsman_scored.runs_scored
    -> from
    -> ball_by_ball as A
    -> inner join batsman_scored
    -> on
    -> A.match_id=batsman_scored.match_id
    -> and A.over_id=batsman_scored.over_id
    -> and A.ball_id=batsman_scored.ball_id
    -> and A.innings_no = batsman_scored.innings_no) as B
    -> group by B.match_id,B.striker) as C
    -> group by C.striker order by value desc) as D
    -> inner join
    -> player on
    -> player.player_id=D.striker
    -> order by value desc;
+----------------------+---------+
| batsman_name         | value   |
+----------------------+---------+
| LMP Simmons          | 42.8182 |
| CH Gayle             | 39.0233 |
| SE Marsh             | 35.8000 |
| N Rana               | 34.6667 |
| ML Hayden            | 34.6452 |
| MEK Hussey           | 34.0862 |
| DA Warner            | 34.0714 |
| MN van Wyk           | 33.4000 |
| V Kohli              | 31.3937 |
| SR Tendulkar         | 29.8590 |
| KH Pandya            | 29.6250 |
| AB de Villiers       | 29.5421 |
| MP Stoinis           | 29.2000 |
| AM Rahane            | 29.0690 |
| PD Collingwood       | 29.0000 |
| SK Raina             | 28.7535 |
```

- Top 3 countries of highest batting average.

```
select E.country_name as
country,sum(E.batting_average)/count(E.country_name) value
from
(select player.player_name as
batsman_name,D.striker,D.batting_average,player.country_name
from
(select C.striker,SUM(C.runs_scored)/COUNT(C.match_id) batting_average
from
(select B.match_id,B.striker,SUM(B.runs_scored) runs_scored
from
(select A.match_id,A.striker,batsman_scored.runs_scored
from
ball_by_ball as A
inner join batsman_scored
on
A.match_id=batsman_scored.match_id
and A.over_id=batsman_scored.over_id
and A.ball_id=batsman_scored.ball_id
and A.innings_no = batsman_scored.innings_no) as B
group by B.match_id,B.striker) as C
group by C.striker order by batting_average desc) as D
inner join
player on
player.player_id=D.striker) as E
group by E.country_name order by value DESC limit 3;
```

```
mysql> select E.country_name,sum(E.batting_average)/count(E.country_name) value
    -> from
    -> (select player.player_name as batsman_name,D.striker,D.batting_average,player.country_name
    -> from
    -> (select C.striker,SUM(C.runs_scored)/COUNT(C.match_id) batting_average
    -> from
    -> (select B.match_id,B.striker,SUM(B.runs_scored) runs_scored
    -> from
    -> (select A.match_id,A.striker,batsman_scored.runs_scored
    -> from
    -> ball_by_ball as A
    -> inner join batsman_scored
    -> on
    -> A.match_id=batsman_scored.match_id
    -> and A.over_id=batsman_scored.over_id
    -> and A.ball_id=batsman_scored.ball_id
    -> and A.innings_no = batsman_scored.innings_no) as B
    -> group by B.match_id,B.striker) as C
    -> group by C.striker order by batting_average desc) as D
    -> inner join
    -> player on
    -> player.player_id=D.striker) as E
    -> group by E.country_name order by value DESC limit 3;
+--------------+-------------+
| country_name | value       |
+--------------+-------------+
| England      | 16.70926429 |
| West Indies  | 16.22694706 |
| Netherlands  | 15.52380000 |
+--------------+-------------+
3 rows in set (0.28 sec)
```

- SQL Query file for the same. Link
- Result file. Link

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

```
create table player_new as SELECT * from player;
```

```
mysql> CREATE TABLE player_new AS SELECT * FROM player;
Query OK, 469 rows affected (0.45 sec)
Records: 469  Duplicates: 0  Warnings: 0

mysql> select * from player_new;
+-----------+---------------------+------------+---------------+-----------------------+--------------+
| player_id | player_name         | dob        | batting_hand  | bowling_skill         | country_name |
+-----------+---------------------+------------+---------------+-----------------------+--------------+
|         1 | SC Ganguly          | 1972-07-08 | Left-hand bat | Right-arm medium      | India        |
|         2 | BB McCullum         | 1981-09-27 | Right-hand bat | Right-arm medium      | New Zealand  |
|         3 | RT Ponting          | 1974-12-19 | Right-hand bat | Right-arm medium      | Australia    |
|         4 | DJ Hussey           | 1977-07-15 | Right-hand bat | Right-arm offbreak    | Australia    |
|         5 | Mohammad Hafeez     | 1980-10-17 | Right-hand bat | Right-arm offbreak    | Pakistan     |
|         6 | R Dravid            | 1973-01-11 | Right-hand bat | Right-arm offbreak    | India        |
|         7 | W Jaffer            | 1978-02-16 | Right-hand bat | Right-arm offbreak    | India        |
|         8 | V Kohli             | 1988-11-05 | Right-hand bat | Right-arm medium      | India        |
|         9 | JH Kallis           | 1975-10-16 | Right-hand bat | Right-arm fast-medium | South Africa |
|        10 | CL White            | 1983-08-18 | Right-hand bat | Legbreak googly       | Australia    |
|        11 | MV Boucher          | 1976-12-03 | Right-hand bat | Right-arm medium      | South Africa |
|        12 | B Akhil             | 1977-10-07 | Right-hand bat | Right-arm medium-fast | India        |
|        13 | AA Noffke           | 1977-04-30 | Right-hand bat | Right-arm fast-medium | Australia    |
|        14 | P Kumar             | 1986-10-02 | Right-hand bat | Right-arm medium      | India        |
|        15 | Z Khan              | 1978-10-07 | Right-hand bat | Left-arm fast-medium  | India        |
|        16 | SB Joshi            | 1970-06-06 | Left-hand bat | Slow left-arm orthodox | India        |
|        17 | PA Patel            | 1985-03-09 | Left-hand bat | Right-arm medium      | India        |
|        18 | ML Hayden           | 1971-10-29 | Left-hand bat | Right-arm medium      | Australia    |
|        19 | MEK Hussey          | 1975-05-27 | Left-hand bat | Right-arm medium      | Australia    |
|        20 | MS Dhoni            | 1981-07-07 | Right-hand bat | Right-arm medium      | India        |
|        21 | SK Raina            | 1986-11-27 | Left-hand bat | Right-arm offbreak    | India        |
|        22 | JDP Oram            | 1978-07-28 | Left-hand bat | Right-arm fast-medium | New Zealand  |
|        23 | S Badrinath         | 1980-08-30 | Right-hand bat | Right-arm offbreak    | India        |
|        24 | K Goel              | 1986-12-24 | Left-hand bat | Right-arm offbreak    | India        |
|        25 | JR Hopes            | 1978-10-24 | Right-hand bat | Right-arm medium      | Australia    |
|        26 | KC Sangakkara       | 1977-10-27 | Left-hand bat | Right-arm offbreak    | Sri Lanka    |
|        27 | Yuvraj Singh        | 1981-12-12 | Left-hand bat | Slow left-arm orthodox | India        |
|        28 | SM Katich           | 1975-08-21 | Left-hand bat | Slow left-arm chinaman | Australia    |
|        29 | IK Pathan           | 1984-10-27 | Left-hand bat | Left-arm medium-fast  | India        |
|        30 | T Kohli             | 1988-12-17 | Right-hand bat | Right-arm medium      | India        |
|        31 | YK Pathan           | 1982-11-17 | Right-hand bat | Right-arm offbreak    | India        |
```

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 >

Solution:

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

Solution:

```
select player.player_name as name,D.runs
from
(select C.striker,max(C.runs) runs
from
```

```
(select B.striker,SUM(B.runs_scored) runs
from
(select A.match_id,A.striker,batsman_scored.runs_scored
from
(select ball_by_ball.match_id,ball_by_ball.over_id,
ball_by_ball.ball_id, ball_by_ball.innings_no,
ball_by_ball.striker
from ball_by_ball inner join match_details
on ball_by_ball.match_id=match_details.match_id
and match_details.season_id=3) as A
inner join batsman_scored
on
A.match_id=batsman_scored.match_id
and A.over_id=batsman_scored.over_id
and A.ball_id=batsman_scored.ball_id
and A.innings_no = batsman_scored.innings_no) as B
group by B.striker) as C
inner join player_match on
C.striker = player_match.player_id
and player_match.role="Captain"
and C.runs>50 group by C.striker) as D
inner join player on
player.player_id = D.striker order by player_name;
```

```
mysql> select player.player_name,D.runs
    -> from
    -> (select C.striker,max(C.runs) runs
    -> from
    -> (select B.striker,SUM(B.runs_scored) runs
    -> from
    -> (select A.match_id,A.striker,batsman_scored.runs_scored
    -> from
    -> (select ball_by_ball.match_id,ball_by_ball.over_id,
    -> ball_by_ball.ball_id, ball_by_ball.innings_no,
    -> ball_by_ball.striker
    -> from ball_by_ball inner join match_details
    -> on ball_by_ball.match_id=match_details.match_id
    -> and match_details.season_id=3) as A
    -> inner join batsman_scored
    -> on
    -> A.match_id=batsman_scored.match_id
    -> and A.over_id=batsman_scored.over_id
    -> and A.ball_id=batsman_scored.ball_id
    -> and A.innings_no = batsman_scored.innings_no) as B
    -> group by B.striker) as C
    -> inner join player_match on
    -> C.striker = player_match.player_id
    -> and player_match.role="Captain"
    -> and C.runs>50 group by C.striker) as D
    -> inner join player on
    -> player.player_id = D.striker order by player_name;
+------------------+------+
| player_name      | runs |
+------------------+------+
| AD Mathews       |  233 |
| BB McCullum      |  114 |
| DA Warner        |  282 |
| DJ Bravo         |   61 |
| DJ Hussey        |   94 |
| DPMD Jayawardene |  436 |
| G Gambhir        |  277 |
| Harbhajan Singh  |  105 |
| JP Duminy        |  157 |
| KC Sangakkara    |  342 |
| KP Pietersen     |  236 |
| LRPL Taylor      |   88 |
| M Vijay          |  458 |
| MS Dhoni         |  287 |
| R Dravid         |  256 |
| RG Sharma        |  404 |
| S Dhawan         |  191 |
| SC Ganguly       |  493 |
| SK Raina         |  505 |
| SR Tendulkar     |  617 |
| SR Watson        |  185 |
| V Kohli          |  307 |
| V Sehwag         |  356 |
| VVS Laxman       |   64 |
| Yuvraj Singh     |  212 |
+------------------+------+
25 rows in set (0.08 sec)
```

- SQL Query file for the same. Link
- Result file. Link

Problem 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)

Solution:

- References privilege is to be obtained on the the relation r2.
- It is needed for the smooth functioning when the database is operated by different users. If the user in the given question, creates a foreign key relation referencing r2. Then, if the admin wants to make some changes in the r1, the foreign key would create problems in altering unless some specific constraints are not set by the relation r1 while defining the foreign key. In order to avoid this, difficulty reference previlege is introduced, here the admin would be knowing who all have created foreign key relations and can eleminate them for the easy operations by other users on the same relation r2.

Problem 3

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

Solution:

- Integrity Constraints ensures the data is inserted, updated or deleted in a certain manner so that it follows certain set of rules. It can be divided into two: Constraints on Single Relations and Constraints on Multiple Relation.
    - The important integrity constraints include Not NULL, Unique ,Check < predicate >. Not NULL ensures the cell will always be non empty while inserting. If the specific cell is empty, it would throw an error. Unique constraint make sures the uniqueness of an item in a column in a database. One could use Check < predictate > to customly decide to follow certain conditions while inserting the data. For eg: if you have semester as an attribute, one could check, insert the semester data if it is present either in one of Fall,Winter,Spring or Summer period.
    - Referential Integrity constraints maintains the data integrity in opertaions involving foreign keys in a database. It is an example of constraint on Multiple Relation. By default foreign key references the primary attributes of the referenced table. If there are more attributes referenced by a foreign key. Then it either need to be specified as primary key or a unique constraint needs to be added to that.
- An authorization constraint gives the database administrator the ability to control the authority of various users to do different operations on the databases. Some form of authorization include, permission to read data,permission to insert new data,permission to update data,permission to delete data and many more. Each of these authorizations is called a previlege.
    - An example can be a database system in IIT Gandhinagar where database admins provide read,write,update,delete permission to the Academic Office and Read access to Students.
    - It can be controlled using the commands **grant** and **revoke**.

Problem 4

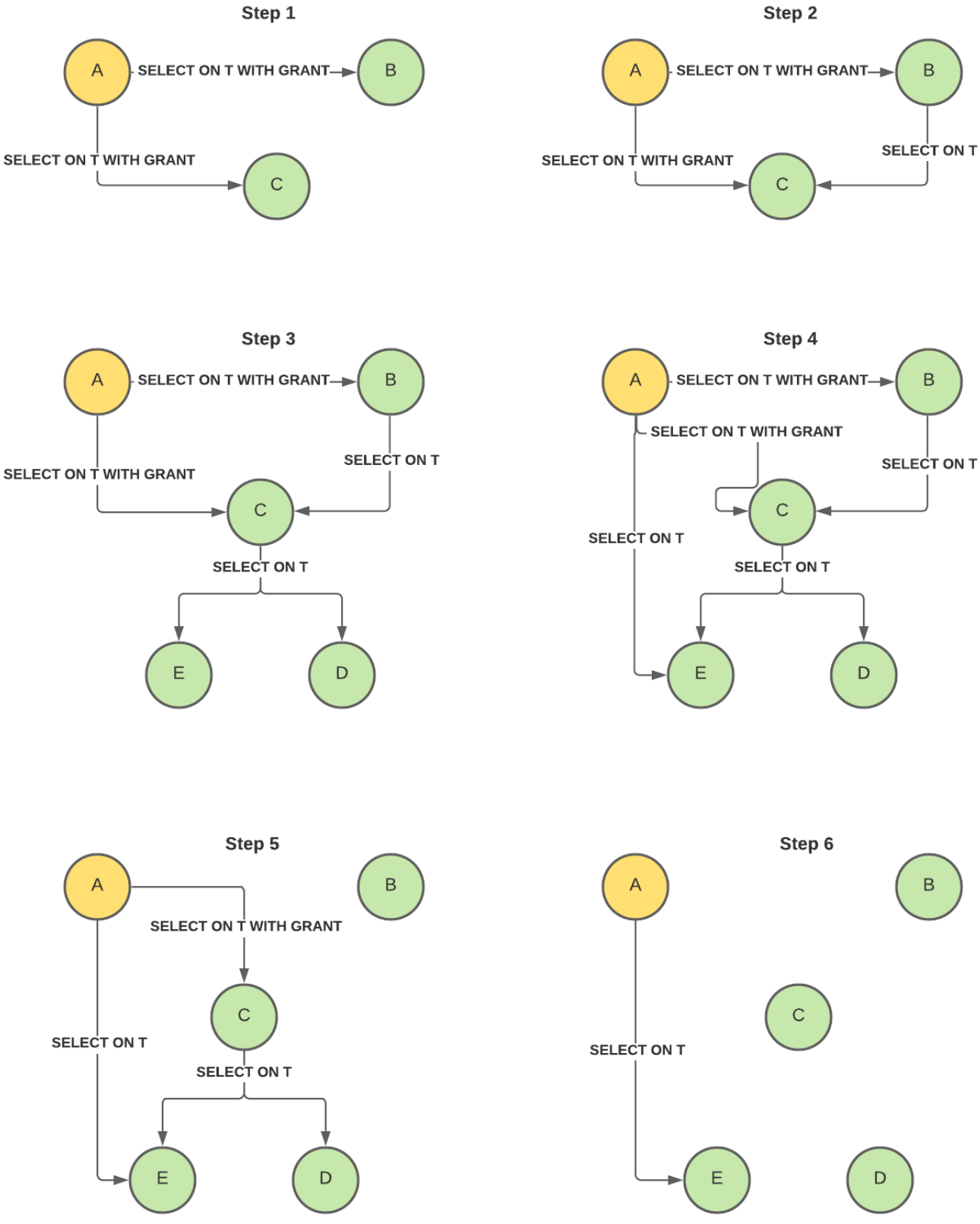
Consider a set of users A, B, C, D, and E. Suppose the user A creates a table T and thus is the owner of T. Now suppose the following set of statements is executed in order:

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

- When does D not have SELECT ON T privilege? Justify your answer. (3 marks)
- What permissions does C have at the end of statement 5? Justify your answer. (2 marks)

Solution:



- **In Step 6**, **D** does not have the SELCT ON T previlege. At first, A had revoked permission on B with "restrict". Since A used restrict, and C (the permission given by B) has a direct connection from A, the perimission to B is revoked. If there C did not have a permission from some one else other than B, then A would not be able to revode the permission of B using restrict. In step 6, A revoked the permission of C using cascade, since it is cascade all the permissions given by C and its children need to be revoked forcefully. E retains its permission through the direct permission given by A while D losses its permission since it did not have any permission from anyone else other than C.
- C had SELECT ON T with Grant Previlege. Since, the permission was given by A directly and has not yet revoked by A.