IPL

Rahul Bansal (2016CS10344), Vinayak Rastogi (2016CS10345), Pushpam Anand (2016CS10347)

Due date: March 1, 2020, 11:55pm IST

1 Project Description

The project is about a website that is made for people who are cricket enthusiasts and people looking to debate each other on who's a better player or a team. The website displays ball by ball (granularity of the data) player statistics and gives it in the form the user wants, be it searching for specific players, or teams, or simply team statistics, balling statistics, batting statistics, the top players in each aspect, and a lot more. We have a robust database design with the appropriate constraints and triggers, so that users can add data as well.

2 Data Sources and Statistics

We downloaded our dataset from https://www.kaggle.com/raghu07/ipl-data-till-2017. The data was available in .csv format. So we directly downloaded it and analyzed it using MS Excel. The date was structured but it was not clean.

We obtained our data from https://www.kaggle.com/raghu07/ipl-data-till-2017. This contains the data of all IPL seasons from 2008 to 2017 on a ball by ball granularity. The data was availabe on the website in csv format. Minor pre-processing steps were taken such as removing the characters that were not UTF encoded, but not much because the datasets present on kaggle are mostly clean and ready to be used.

Following are the tables for the statistics.

Entity	**Attributes**
team	team id, team name
player	player id, player name, dob, batting hand, bowling skill, country name
match	match id, team1, team2, match date, season year, venue id, toss winner, match winner, toss name, win type, outcome type, manofmatch, win margin
ball	match id, over id, ball id, innings no, striker batting position, extra type, runs scored, extra runs, out type, striker, non striker, bowler, player out, fielder

Table 1: List of Entities and Attributes

Table	**No. of Tuples**	**Data-set Size**	**Time to load**
team	13	306B	5.445 ms
player	497	32KB	6.789 ms
match	637	83.1KB	10.994 ms
ball	150451	8.89MB	2086.679 ms

Table 2: Data Statistics

3 Functionality and working

3.1 User's View of the System

- **Home**:
  This is the main page of the website which does not conatin any SQL queries. It just contains some information about IPL. Disclaimer: The information is copy pasted from wikipedia

- **Teams**:
  The page just displays all the teams in table format which have played in IPL since its inception. It also includes the teams which were banned from playing or have not played in the last season or are no longer playing in the current season.

- **Players**:
  The page displays all the players in table format which have ever played in any IPL season since its inception. The webpage contains a filter which the user can use to select players with certain characteristics. The filter is based on the country of origin of the player, the hand which the player uses to bat and the hand which the player uses to bowl. The user can select multiple countries at a given time to select players from multiple countries. User can also search a player by name and get his statistics.

- **Stats**:
  This page gives the top batsman and bowler by various options. For batsman one can sort by runs,highest score, strike rate, average,most fours,most sixes,most fifties and most centuries,fastest 50 and fastest 100. Bowlers can be sorted by most wickets, maiden, dot balls, average and economy rate.

- **Player Stats**:
  User can search a player by his name and get his batting and bowling statistics.

- **Add Player**:
  This page provides the user an option to add a new player to the database. For adding a player, the user must enter the same details of a player as the table in the database contains except the id. The id again is automatically generated with the help of PHP.

- **Interesting Facts**:
  Displaying top 10 players according to various different sorting criteria such as most runs, most 100s, most 50s, most maiden overs, most wickets, most dot balls etc.

- **Points Table**
  Year-wise distribution of points across teams.

3.2 Special Functionality

- **Views**:
  We have created many views such as batting stats views, balling stats views, maiden overs views within the database. And whenever that particular data is required the views are not made in real time and exist in the database so that the retrieval does not take much time.

  All the views created and used are described in the list of queries subsection mentioned below.

- **Triggers**:
  We have defined 2 triggers, to implement certain constraints which can not be defined other- wise, such that whenever any new data is added into the player table it gets updated in all the other tables where its relevant,

  1. batting stats which has various entries such as total runs, matches, sixes, fours, etc table is updated in sync when ever the player table gets updated:

```
CREATE OR REPLACE FUNCTION test_trigger()
  RETURNS trigger AS
$$
```

```
BEGIN
INSERT INTO batting_stats(striker,player_name, batting_hand, total_runs, top_score, hundreds
, fiftys ,matches, average ,total_balls , total_six , total_four  , strike_rate )
VALUES(NEW.player_id,NEW.player_name,NEW.batting_hand,0,0,0,0,0,0,0,0,0,0);
RETURN NEW;
END;
$$
LANGUAGE 'plpgsql';
CREATE TRIGGER ins_bs
  AFTER INSERT
  ON player
  FOR EACH ROW
  EXECUTE PROCEDURE test_trigger();
```

2. Bowling stats which has various entries such as total wickets, matches, dots, runs, etc table is updated in sync when ever the player table gets updated:

```
CREATE OR REPLACE FUNCTION test_trigger()
RETURNS trigger AS
$$
BEGIN
INSERT INTO batting_stats(striker,player_name, batting_hand, total_runs, top_score,
hundreds  , fiftys ,matches, average ,total_balls , total_six , total_four  , strike_rate )
VALUES(NEW.player_id,NEW.player_name,NEW.batting_hand,0,0,0,0,0,0,0,0,0,0);
RETURN NEW;
END;
$$
LANGUAGE 'plpgsql';
CREATE TRIGGER ins_bs
AFTER INSERT
ON player
FOR EACH ROW
EXECUTE PROCEDURE test_trigger();
```

- **Foreign keys**:
  Foreign keys have been defined in each of the tables for appropriate inter-linking between different tables.

3.3 List of Queries

- **Creation of Tables**:

```
CREATE TABLE player(PLAYER_SK int ,Player_Id int ,Player_Name text,DOB text ,Batting_hand text ,
Bowling_skill text ,Country_Name text,PRIMARY KEY (Player_Id));
COPY player FROM '/Users/rahulbansal/Desktop/8th_Sem/Database/ipl/ipl-data-till-2017/Player.csv'
delimiter ',' csv header;

CREATE TABLE match(Match_SK int ,match_id int ,Team1 text ,Team2 text,match_date text,Season_Year
int ,Venue_Name text ,
City_Name text ,Country_Name text,Toss_Winner text ,match_winner text,Toss_Name text,Win_Type
text, Outcome_Type text ,ManOfMach text ,Win_Margin text ,Country_id int,PRIMARY KEY (match_id));
COPY match FROM '/Users/rahulbansal/Desktop/8th_Sem/Database/ipl/ipl-data-till-2017/Match.csv'
delimiter ',' csv header;

CREATE TABLE ball(match_id int,over_id int,ball_id int,innings_no int,team_batting text,
team_bowling text,striker_batting_position int,extra_type text,runs_scored int,extra_runs int,
wides int,legbyes int,byes int,noballs int,penalty int,bowler_extras int,out_type text,caught int,
bowled int,run_out int,lbw int,retired_hurt int,stumped int,caught_bowled int,hit_wicket int,
obstructingfield int,bowler_wicket int,match_date text,season int,striker int,non_striker int,
```

```
bowler int,player_out int,fielders int,striker_match_sk int,strikersk int,nonstriker_match_sk int,
nonstriker_sk int,fielder_match_sk int,fielder_sk int,bowler_match_sk int,bowler_sk int,
playerout_match_sk int,battingteam_sk int,bowlingteam_sk int,keeper_catch int,player_out_sk int,
matchdatesk text,FOREIGN KEY (match_id) REFERENCES match(match_id),FOREIGN KEY (striker)
REFERENCES player(player_id), FOREIGN KEY (non_striker) REFERENCES player(player_id),
FOREIGN KEY (bowler) REFERENCES player(player_id));
COPY ball FROM '/Users/rahulbansal/Desktop/8th_Sem/Database/ipl/ipl-data-till-2017/Balls.csv'
delimiter ',' csv header;

CREATE TABLE teams(Team_SK int, Team_Id int, Team_Name text);
COPY teams FROM '/Users/rahulbansal/Desktop/8th_Sem/Database/ipl/ipl-data-till-2017/Team.csv'
delimiter ',' csv header;
```

- **Teams**:

```
select * from teams;
```

- **Players**:

```
select Player_Name,DOB,batting_hand,bowling_skill,country_name from player;
```

- **Stats(batting)**:

```
create view Batting_stats as select playerfs.striker, player.Player_Name,player.
Batting_hand, playerfs.total_runs, per_striker_match_stats.top_score,
per_striker_match_stats.Hundreds, per_striker_match_stats.Fiftys, per_striker_match_stats.matches,
ROUND(cast(per_striker_match_stats.Average as numeric),2) as Average, playerfs.total_balls,
playerfs.total_six, playerfs.total_four, ROUND(cast(playerfs.strike_rate as numeric),2) as
Strike_rate from playerfs,per_striker_match_stats,player
where playerfs.striker = per_striker_match_stats.striker and playerfs.striker =
player.Player_Id order by striker;
```

- **Stats(batting - fastest 50)**:

```
create view ball_cum_runs as
select match_id,innings_no,striker,over_id,ball_id,count(ball_id) filter(where extra_type not in
('wides')) over was num_balls,count(ball_id) filter(where runs_scored=6) over w as num_6s,
count(ball_id) filter(where runs_scored=4) over w as num_4s,
sum(runs_scored) over w as cumm_runs
from ball where innings_no<=2
window w as (partition by match_id,striker order by over_id,ball_id);

create view ball_cum_runs1 as
select match_id,innings_no,player.player_name,min(num_balls) as balls_faced,max(num_6s)
as num_6s,max(num_4s) as num_4s,max(cumm_runs) as runs
from ball_cum_runs join player on ball_cum_runs.striker=player.player_id
where cumm_runs>=50
group by match_id,striker,player.player_name,innings_no
order by Balls_Faced
limit 100;

create view team_match as
select match_id,innings_no,team_batting,cast(team_bowling as int)
from ball
```

```
group by (match_id,innings_no,team_batting,team_bowling) order by match_id;

create view ball_cum_runs2 as
select ball_cum_runs1.match_id,player_name,team_bowling as team_id,match_date,balls_faced,num_6s,
num_4s,runs
from ball_cum_runs1
join team_match on ball_cum_runs1.match_id=team_match.match_id and ball_cum_runs1.innings_no=
team_match.innings_no
join match on
ball_cum_runs1.match_id=match.match_id;

select player_name,team_name as Against,
match_date,balls_faced,num_6s,num_4s,runs from
ball_cum_runs2 join teams on
ball_cum_runs2.team_id=teams.team_id
order by
balls_faced,runs desc,num_6s desc,num_4s desc
limit 10;
```

- **Stats(batting - fastest 100)**:

```
create view ball_cumm_runs as
select match_id,innings_no,striker,over_id,ball_id,count(ball_id)
filter(where extra_type not in ('wides')) over w
as num_balls,count(ball_id) filter(where runs_scored=6) over w as num_6s,
count(ball_id) filter(where runs_scored=4) over w as num_4s,
sum(runs_scored) over w as cumm_runs
from ball where innings_no<=2
window w as
(partition by match_id,striker order by over_id,ball_id);

create view ball_cumm_runs1 as
select match_id,innings_no,player.player_name,min(num_balls) as balls_faced,max(num_6s)
as num_6s,max(num_4s) as num_4s,max(cumm_runs) as runs
from ball_cumm_runs join player on
ball_cumm_runs.striker=player.player_id
where cumm_runs>=100
group by match_id,striker,player.player_name,innings_no
order by Balls_Faced
limit 100;

create view ball_cumm_runs2 as
select ball_cumm_runs1.match_id,player_name,team_bowling as team_id,match_date,balls_faced,
num_6s,num_4s,runs
from ball_cumm_runs1 join team_match on
ball_cumm_runs1.match_id=team_match.match_id and ball_cumm_runs1.innings_no=team_match.innings_no
join match on
ball_cumm_runs1.match_id=match.match_id;

select player_name,team_name as Against,match_date,balls_faced,num_6s,num_4s,runs
from ball_cumm_runs2 join teams on ball_cumm_runs2.team_id=teams.team_id
order by balls_faced,runs desc,num_6s desc,num_4s desc limit 10;
```

- **Stats(bowling)**:

```
create view Bowling_stats as select bowler,
```

```
player.Player_Name, sum(runs_scored) as Runs, sum(bowler_wicket) as Wickets_taken,
count(ball_id) as total_balls,
count(ball_id) filter(where runs_scored=6) as total_six,
count(ball_id) filter(where runs_scored=4) as total_four,
count(match_id) as Matches,
ROUND(cast((sum(runs_scored)/cast(count(ball_id) as decimal)) as numeric),2)*6 as Economy
from ball,player
where player.Player_Id = ball.bowler
group by (bowler, player.Player_Name)
order by bowler;
```

- **Player Stats**:

```
create view playerscore as select striker,sum(runs_scored) as
total_runs
from ball
group by striker;

create view playerfs as
select striker,count(ball_id) as total_balls, count(ball_id)
filter(where runs_scored=6) as total_six,
count(ball_id) filter(where runs_scored=4) as total_four,sum(runs_scored) as total_runs,
(sum(runs_scored)/cast(count(ball_id) as decimal))*100 as strike_rate
from ball
group by striker
order by striker asc;
```

- **Match Stats**:

```
create view match_stats as
select match_id, striker, sum(runs_scored) as runs
from ball
group by (match_id,striker)
order by striker;

create view per_striker_match_stats as select striker, count(match_id)
filter(where runs>=100) as Hundreds,
count(match_id)
filter(where runs>=50 and runs<100) as Fiftys, count(match_id) as matches, max(runs) as top_score,
AVG(runs) as Average
from match_stats
group by striker
order by striker;

create view num_matches_1 as select Team1,Season_Year, count(*)
from match
group by (Team1,Season_Year)
order by Season_Year;

create view num_matches_2 as
select Team2,Season_Year, count(*)
from match
group by (Team2,Season_Year)
order by Season_Year;

create view num_matches as
```

```
select Team1, num_matches_1.Season_Year, num_matches_1.count+num_matches_2.count as
matches_played from num_matches_1,num_matches_2
where num_matches_2.Team2 = Team1 and num_matches_2.Season_Year = num_matches_1.Season_Year
order by Team1;

create view matches_won as select match_winner,Season_Year,count(*)
from match
group by (match_winner,Season_Year)
order by Season_Year;

create view matches_stats as
select Team1, matches_won.Season_Year, matches_played, matches_won.count as matches_won,
matches_played - matches_won.count as
matches_lost, (matches_won.count)*2 as points from matches_won,num_matches
where Team1 = matches_won.match_winner and matches_won.Season_Year = num_matches.Season_Year
order by matches_won.Season_Year;
```

- **Over Stats**:

```
create view overs_runs as
select bowler, player_name, over_id,match_id,innings_no, sum(runs_scored) as
runs_given from ball,player
where bowler = player.player_id
group by (bowler, player_name, over_id,match_id,innings_no)
order by bowler;

create view balls_dot as
select bowler, player_name, count(ball_id)
from ball,player
where bowler = player.player_id and runs_scored=0
group by (bowler,player_name)
order by bowler;

create view maiden_overs as
select player_name, count(*) as nmaiden
from (
select distinct bowler, over_id, count(*)
from ball
where runs_scored = 0
and ( ( extra_runs = 0 and (extra_type = 'wides' or extra_type = 'noballs'
or
extra_type = 'No Extras')) or
( extra_type = 'legbyes' or extra_type = 'byes' or extra_type = 'penalty') )
group by match_id, bowler, over_id) as nmaiden1, player
where nmaiden1.count = 6 and bowler = player_id
group by player_name
order by nmaiden desc;
```

- **Add Player**:

```
INSERT INTO player (PLAYER_SK,Player_Id ,Player_Name,DOB ,Batting_hand  ,Bowling_skill ,
Country_Name) VALUES ('" + (sk-0) + "', '" + (pid-0) + "', '" + pname + "', '" +
pdob + "', '" + phand + "','" + pskill + "', '" +
pcountry + "')";
```

3.4 Query Times

Query	**Running Time(ms)**
Creating Player Table	22.04 ms
Create match table	25.8 ms
Create ball table	9180 ms
Create team table	7.8 ms
Runs Given in 1 Over	15.2 ms
Batting Stats	10.5 ms
Bowling stats	3.6 ms
Fastest 50	12.8 ms
Fastest 100	10.5 ms
Dot Balls	2.2 ms
Maiden Overs	4.3 ms
Points Table	14.2 ms

4 ER Diagram