**Trigger that ensures that we are not inserting two same teams in the matches table

```
create or replace function check_for_match()
returns trigger as
$BODY$
declare
        last_date date;
begin
        if NEW.team_1 = NEW.team_2 then
        raise exception 'You have entered same teams please enter diffrent teams';
    return null;
  end if;
  select max(date) into last_date from matches;
  if NEW.date <= last_date then
        raise exception 'Please enter valid date';
    return null;
  end if;
  return NEW;
end;
$BODY$ language plpgsql;

create trigger matches
before insert or update on matches
        for each row execute procedure check_for_match();
```

```
insert into matches values (8,'check','Lords','2019-7-25','IND','IND',true);
```

**We made the below trigger to check that the player that we are inserting into playing_11 table is in consistent with the matches that we have entered in matches table

```
create or replace function check_for_player()
returns trigger as
$BODY$
declare
        team1 varchar(3);
   team2 varchar(3);
   team varchar(3);
begin
        select team_1,team_2 into team1,team2 from matches where match_no = NEW.match_no;
   select team_id into team from players where player_name = NEW.player_name;
   if team != team1 and team != team2 then
        raise exception '% is not playing match %',NEW.player_name,NEW.match_no;
      return NULL;
   end if;

   return NEW;
end;
$BODY$ language plpgsql;

create trigger playing_11
```

```
before insert or update on playing_11
        for each row execute procedure check_for_player();
```

**Trigger is ensuring that whatever data we are inserting in the ball_by_ball_batting table is valid according to the playing_11 of that match.

```
create or replace function check_for_batsman()
returns trigger as
$BODY$
declare
        team1 varchar(3);
   team2 varchar(3);
   team varchar(3);
   cnt integer;
   i varchar(20);
begin
        select team_1,team_2 into team1,team2 from matches where match_no = NEW.match_no;
   select team_id into team from players where player_name = NEW.player_name;
   if team != team1 and team != team2 then
        raise exception '% is not playing match %',NEW.player_name,NEW.match_no;
      return NULL;
   end if;
   cnt:=0;
```

```
   for i in select player_name from playing_11 where match_no=NEW.match_no and player_name =
NEW.player_name
   loop
         cnt := cnt + 1;
   end loop;
   if cnt=0 then
         raise exception '% is not in playing 11',NEW.player_name;
      return null;
   end if;
   if NEW.over_no > 9 then
         raise exception 'Please enter valid over number';
      return null;
   end if;

   return NEW;
end;
$BODY$ language plpgsql;

create trigger ball_by_ball_batting
before insert or update on ball_by_ball_batting
         for each row execute procedure check_for_batsman();

insert into ball_by_ball_batting values (1,1,10,0,'l',2,'M S Dhoni');
```

**Trigger to ensure that data of ball_by_ball_bowling is valid according to playing_11 table and ball_by_ball_batting table.

```
create or replace function check_for_bowler()
returns trigger as
$BODY$
declare
        team1 varchar(3);
   team2 varchar(3);
   team varchar(3);
   opp_team varchar(3);
   inn integer;
   i varchar(20);
   cnt integer;
begin
        select team_1,team_2 into team1,team2 from matches where match_no = NEW.match_no;
   select team_id into team from players where player_name = NEW.player_name;
   if team != team1 and team != team2 then
        raise exception '% is not playing match %',NEW.player_name,NEW.match_no;
      return NULL;
   end if;
   cnt:=0;
   for i in select player_name from playing_11 where match_no=NEW.match_no and player_name = NEW.player_name
   loop
        cnt := cnt + 1;
```

```
   end loop;
   if cnt=0 then
         raise exception '% is not in playing 11',NEW.player_name;
      return null;
   end if;
   if NEW.over_no > 9 then
         raise exception 'Please enter valid over number';
      return null;
   end if;
         select team_id into opp_team from players natural join ball_by_ball_batting where inning =
NEW.inning and match_no=NEW.match_no;
   if team = opp_team then
         raise exception 'You have entered wrong innings';
      return NULL;
   end if;
   return NEW;
end;
$BODY$ language plpgsql;

create trigger ball_by_ball_bowling
before insert or update on ball_by_ball_bowling
         for each row execute procedure check_for_bowler();

insert into ball_by_ball_bowling values (1,1,0,'Jasprit Bumrah');
```