Stored Procedure

```
Set search_path to cricket;

Create type player_strike_rate as (match_id smallint, player_id smallint, strike_rate
numeric(5,2));

Create type player_economy_rate as (match_id smallint, player_id smallint,
economy_rate numeric(5,2));

Create type player_average as (player_id smallint, total_matches smallint, total_runs
smallint, average numeric(5,2));

create type player_bowling_average as (player_id smallint, runs_given smallint,
wickets_taken smallint, average numeric(5,2));

Create type player_rank as (player_id smallint, player_points smallint, player_rank
smallint);

create type bowler_rank as (player_id smallint, player_points smallint, player_rank
smallint);
```

1.

```
create or replace function Strike_Rate()
returns setof player_strike_rate as $body$
```

```
declare
	strikerate player_strike_rate;
	stats "statistics"%rowtype;
	game "match"%rowtype;
	cricketer "player"%rowtype;

begin
	for game in select * from "match"
	loop
		for stats in select * from "statistics" where "Match_ID"=game."ID"
		loop
			for cricketer in select * from "player" where "ID"=stats."Player_ID"
			loop
				strikerate.match_id = game."ID";
				strikerate.player_id = cricketer."ID";
				if stats."Balls" is not null then
					if stats."Balls" = '0' then
						strikerate.strike_rate := '0.00';
					else
						strikerate.strike_rate := stats."Runs_scored" *
100.00 / stats."Balls";
					end if;
				else
					strikerate.strike_rate := null;
				end if;
			end loop;
			return next strikerate;
		end loop;
	end loop;
	return;
end $body$ language 'plpgsql';
```

2.

```
create or replace function Economy_Rate()
returns setof player_economy_rate as $body$

declare
	economyrate player_economy_rate;
	stats "statistics"%rowtype;
	game "match"%rowtype;
	cricketer "player"%rowtype;
	complete_overs numeric(3,1);
	partial_overs numeric(3,1);
	total_balls smallint;

begin
	for game in select * from "match"
	loop
		for stats in select * from "statistics" where "Match_ID"=game."ID"
		loop
			for cricketer in select * from "player" where "ID"=stats."Player_ID"
			loop
				economyrate.match_id = game."ID";
				economyrate.player_id = cricketer."ID";
				if stats."Overs" is not null then
					complete_overs := trunc(stats."Overs");
					partial_overs := (stats."Overs" - complete_overs) * 10;
					total_balls := (complete_overs * 6) + partial_overs;
					economyrate.economy_rate := stats."Runs_given" *
6.00 / total_balls;
				else
					economyrate.economy_rate := NULL;
				end if;
			end loop;
			return next economyrate;
		end loop;
	end loop;
	return;
end $body$ language 'plpgsql';
```

3.

```
create or replace function average_runs()
returns setof player_average as $body$

declare
        matches smallint;
        runs smallint;
        cricketer "player"%rowtype;
        game "match"%rowtype;
        stats "statistics"%rowtype;
        my_average player_average;

begin
        for cricketer in select * from "player"
        loop
                runs := 0;
                matches := 0;
                for stats in select * from "statistics"
                where "Player_ID" = cricketer."ID"
                loop
                        for game in select * from "match"
                        where "ID" = stats."Match_ID"
                        loop
                                if stats."Runs_scored" is not null then
                                        runs = runs + stats."Runs_scored";
                                        matches = matches + 1;
                                end if;
                        end loop;
                end loop;
                my_average.player_id = cricketer."ID";
                my_average.total_matches = matches;
                my_average.total_runs = runs;
                if matches = 0 then
                        my_average.average = null;
                else
                        my_average.average = runs * 1.00 / matches;
                end if;
                return next my_average;
        end loop;
        return;
end $body$ language 'plpgsql';
```

4.

```
create or replace function batsman_ranking()
returns setof player_rank as $body$

declare
        my_rank player_rank;
        ranks smallint;
        my_avg player_average;
        temps smallint;

begin
        temps := 1;
        for my_avg in select * from average_runs()
        order by average desc
        loop
                if temps = 16 then
                        return;
                else
                        if my_avg.average is not null then
                                my_rank.player_id = my_avg.player_id;
                                my_rank.player_points = my_avg.average;
                                my_rank.player_rank = temps;
                                temps = temps + 1;
                                return next my_rank;
                        end if;
                end if;
        end loop;
        return;
end $body$ language 'plpgsql';
```

5.

```
create or replace function bowling_average()
returns setof player_bowling_average as $body$

declare
        wickets smallint;
        runs smallint;
        cricketer "player"%rowtype;
        game "match"%rowtype;
        stats "statistics"%rowtype;
        my_average player_bowling_average;

begin
        for cricketer in select * from "player"
        loop
                runs := 0;
                wickets := 0;
                for stats in select * from "statistics"
                where "Player_ID" = cricketer."ID"
                loop
                        for game in select * from "match"
                        where "ID" = stats."Match_ID"
                        loop
                                if stats."Runs_given" is not null then
                                        runs = runs + stats."Runs_given";
                                        wickets = wickets + stats."Wickets";
                                end if;
                        end loop;
                end loop;
                my_average.player_id = cricketer."ID";
                my_average.wickets_taken = wickets;
                my_average.runs_given = runs;
                if wickets = 0 then
                        my_average.average = runs;
                else
                        my_average.average = runs * 1.00 / wickets;
                end if;
                return next my_average;
        end loop;
        return;
end $body$ language 'plpgsql';
```

6.

```
create or replace function bowler_ranking()
returns setof bowler_rank as $body$

declare
        my_rank bowler_rank;
        ranks smallint;
        my_avg player_average;
        temps smallint;

begin
        temps := 1;
        for my_avg in select * from bowling_average()
        order by average
        loop
                if temps = 16 then
                        return;
                else
                        if my_avg.average > 0 then
                                my_rank.player_id = my_avg.player_id;
                                my_rank.player_points = 1000 - my_avg.average;
                                my_rank.player_rank = temps;
                                temps = temps + 1;
                                return next my_rank;
                        end if;
                end if;
        end loop;
        return;
end $body$ language 'plpgsql';
```