# **Sport analytics sql codes**

1. **Table for matches**

create table matches (

id int primary key,

city varchar,

date date,

player\_of\_match varchar,

venue varchar,

neutral\_venue int,

team1 varchar,

team2 varchar,

toss\_winner varchar,

toss\_decision varchar,

winner varchar,

result varchar,

result\_margin int,

eliminator varchar,

method varchar,

umpire1 varchar,

umpire2 varchar

);

1. **Table for deliveries**

create table deliveries (

id int,

inning int,

over int,

ball int,

batsman varchar,

non\_striker varchar,

bowler varchar,

batsman\_runs int,

extra\_runs int,

total\_runs int,

is\_wicket int,

dismissal\_kind varchar,

player\_dismissed varchar,

fielder varchar,

extras\_type varchar,

batting\_team varchar,

bowling\_team varchar

);

1. **Import data from csv file ’ipl\_matches.csv’ attached in resources to the table ‘matches’ which was created in q1**

copy matches from 'c:\program files\postgresql\16\data\data\_copy\ipl\_matches.csv' delimiter ',' csv header;

1. **Import data from csv file ’ipl\_ball.csv’ attached in resources to the table ‘deliveries’ which was created in q2**

copy deliveries from 'c:\program files\postgresql\16\data\data\_copy\ipl\_ball.csv' delimiter ',' csv header;

1. **Select the top 20 rows of the *deliveries*table after ordering them by id, inning, over, ball in ascending order.**

select \*

from deliveries

order by id, inning, over, ball asc

limit 20;

1. **Select the top 20 rows of the *matches*table.**

select \*

from matches

limit 20;

1. **Fetch data of all the matches played on 2nd may 2013 from the matches table**

select \*

from matches

where date = '2013-05-02';

1. **Fetch data of all the matches where the result mode is ‘runs’ and margin of victory is more than 100 runs**

select \*

from matches

where result = 'runs' and result\_margin > 100;

&&&&

select \*

from matches

where result = 'runs' and cast(result\_margin as integer) > 100;

1. **Fetch data of all the matches where the final scores of both teams tied and order it in descending order of the date.**

select \*

from matches

where result = 'tie'

order by date desc;

1. **Get the count of cities that have hosted an ipl match.**

select city, count(city)

from matches

group by city;

1. **Create table *deliveries\_v02*with all the columns of the table ‘*deliveries’*and an additional column *ball\_result*containing values *boundary*, *dot*or *other*depending on the *total\_run*(boundary for >= 4, dot for 0 and other for any other number)**

create table deliveries\_v02 as

select \*,

case

when total\_runs >= 4 then 'boundary'

when total\_runs = 0 then 'dot'

else 'other'

end as ball\_result

from deliveries;

1. **Write a query to fetch the total number of boundaries and dot balls from the deliveries\_v02 table**

select ball\_result, count(ball\_result) as total\_count

from deliveries\_v02

where ball\_result in ('boundary','dot')

group by ball\_result;

1. **Write a query to fetch the total number of boundaries scored by each team from the *deliveries\_v02*table and order it in descending order of the number of boundaries scored.**

select batting\_team, count(ball\_result) as total\_boundaries

from deliveries\_v02

where ball\_result = 'boundary'

group by batting\_team

order by total\_boundaries desc;

1. **Write a query to fetch the total number of dot balls bowled by each team and order it in descending order of the total number of dot balls bowled.**

select bowling\_team, count(ball\_result) as total\_dots

from deliveries\_v02

where ball\_result = 'dot'

group by bowling\_team

order by total\_dots desc;

1. **Write a query to fetch the total number of dismissals by dismissal kinds where dismissal kind is not na**

select dismissal\_kind, count(dismissal\_kind) as total\_dismissals

from deliveries\_v02

where dismissal\_kind != 'na'

group by dismissal\_kind;

1. **Write a query to get the top 5 bowlers who conceded maximum extra runs from the *deliveries*table**

select bowler, max(extra\_runs) as max\_extra\_runs

from deliveries

group by bowler

order by max\_extra\_runs desc

limit 5;

1. **Write a query to create a table named *deliveries\_v03*with all the columns of *deliveries\_v02*table and two additional column (named *venue*and *match\_date*) of *venue*and *date*from table *matches***

create table deliveries\_v03 as

select d.\*, m.venue as venue, m.date as match\_date

from deliveries as d full join matches as m

on m.id = d.id;

1. **Write a query to fetch the total runs scored for each venue and order it in the descending order of total runs scored.**

select venue, sum(total\_runs) as total\_runs\_venue

from deliveries\_v03

group by venue

order by total\_runs\_venue desc;

1. **Write a query to fetch the year-wise total runs scored at *eden gardens*and order it in the descending order of total runs scored.**

select extract(year from match\_date) as year, venue, sum(total\_runs) as runs\_at\_eden

from deliveries\_v03

where venue = 'eden gardens'

group by extract(year from match\_date), venue

order by runs\_at\_eden desc;

1. **Get unique team1 names from the *matches*table, you will notice that there are two entries for*rising pune supergiant* one with *rising pune supergiant* and another one with *rising pune supergiants*.  Your task is to create a *matches\_corrected*table with two additional columns *team1\_corr*and *team2\_corr*containing team names with replacing *rising pune supergiants* with *rising pune supergiant*. Now analyse these newly created columns.**

create table matches\_corrected as

select \*,

case when team1 = 'rising pune supergiants' then 'rising pune supergiant' else team1 end as team1\_corr,

case when team2 = 'rising pune supergiants' then 'rising pune supergiant' else team2 end as team2\_corr

from matches;

1. **Create a new table deliveries\_v04 with the first column as ball\_id containing information of match\_id, inning, over and ball separated by ‘-’ (for ex. 335982-1-0-1 match\_id-inning-over-ball) and rest of the columns same as deliveries\_v03**

create table deliveries\_v04 as

select concat(id,'-',inning,'-',over,'-',ball) as ball\_id, \*

from deliveries\_v03;

1. **Compare the total count of rows and total count of distinct ball\_id in deliveries\_v04;**

select count(\*) as total\_rows, count(distinct(ball\_id)) as distinct\_ballid

from deliveries\_v04;

1. **Sql row\_number() function is used to sort and assign row numbers to data rows in the presence of multiple groups. For example, to identify the top 10 rows which have the highest order amount in each region, we can use row\_number to assign row numbers in each group (region) with any particular order (decreasing order of order amount) and then we can use this new column to apply filters. Using this knowledge, solve the following exercise. You can use hints to create an additional column of row number.  
   Create table deliveries\_v05 with all columns of deliveries\_v04 and an additional column for row number partition over ball\_id**

create table deliveries\_v05 as

select \*,

row\_number() over (partition by ball\_id) as r\_num

from deliveries\_v04;

1. **Use the r\_num created in deliveries\_v05 to identify instances where ball\_id is repeating. (hint : select \* from deliveries\_v05 where r\_num=2;)**

select \* from deliveries\_v05 where r\_num > 1;

1. **Use subqueries to fetch data of all the ball\_id which are repeating. (hint: select \* from deliveries\_v05 where ball\_id in (select ball\_id from deliveries\_v05 where r\_num=2);**

select \* from deliveries\_v05

where ball\_id in (select ball\_id from deliveries\_v05 where r\_num > 1);