

SQL for Data Analytics

(Postgresql)

Powered by Internshala



IPL Database

Using different SQL functions to Analyze ball-by- ball data for all the IPL cricket matches played between 2008 and 2020.

By:
Rahul Routu
Computer Science Dept.
Baba Institute of Technology & Sciences

1)Acknowledgement:

I take this opportunity to convey my sincere thanks & gratitude to all those who have directly or indirectly helped and contributed towards the completion of this project.

First & foremost, I would like to thank “Internshala” for giving me the opportunity in doing this project. During this training & project I got understand the Data Analysis Subject which I was learned. This is the step where I have understood the importance of doing projects. It has been a great experience with Internshala.

***R.RAHUL
CSE, BITS Vizag.***

2)Table of Contents:

S.No	Topic	Pg. No
1.	Acknowledgement	3
2.	Table of Contents	4
3.	About Training & Company	5
4.	Introduction	6
5.	Technology used in the Project	7
6.	Technical Details of Project 6.1: Project Code 6.2: Project Snapshot	8-22
7.	Flow Chart	23
8.	Applications	23
9.	Conclusion	24
10.	References	24

3)About the Training:

I have completed my training from “Internshala” in Data Analytics for sql. And many of Indian and foreign teacher uploaded their content. Internshala is an internship and online training platform, based in Gurgaon, India.

The Founder is Sarvesh Agrawal, an IIT Madras alumnus, in 2011, the website helps students to find internships & job opportunities with organisations in India. The platform, which was founded in 2010, started out as a WordPress blog that aggregated internships across India and articles on education, technology and skill gap. Internshala launched its online trainings in 2014. As of 2023, the platform had 3.5 million students and 80,000 companies.

I have attached my Training Certificate which Internshala has provided me after the completion of course.



Introduction:

In database analytics is the technology that allows data processing to be conducted within the database by building analytic logic into the database itself. Doing so eliminates the time and effort required to transform data and move it back and forth between a database and a separate analytics application.

PostgreSQL is an advanced object-oriented database management system that supports an extended subset of the SQL standard, including transactions, foreign keys, subqueries, triggers, user-defined types and functions.



4)Technology used in the Project:

In database analytics is the technology that allows data processing to be conducted within the database by building analytic logic into the database itself. Doing so eliminates the time and effort required to transform data and move it back and forth between a database and a separate analytics application.

PostgreSQL is an advanced object-oriented database management system that supports an extended subset of the SQL standard, including transactions, foreign keys, subqueries, triggers, user-defined types and functions.

PostgreSQL has earned a strong reputation for its proven architecture, reliability, data integrity, robust feature set, extensibility and the dedication of the open-source community behind the software to consistently deliver performant & innovative solutions.

PostgreSQL runs on all major operating systems has been ACID compliant since 2001, & has powerful add-ons such as the popular PostGIS geospatial database extender. It is no surprise that PostgreSQL has become the open-source relational database of choice for many people & organisations.

Moreover, it is free to download, supports various Operating Systems, highly secure & reliable, compatible with several data types & supports Multiversion concurrency control (MVCC)

6) Technical Details of the Project:

Sports Analysis using PostgreSQL

In this project, we have to perform the job of a sports Analyst. We have given two datasets related IPL(Indian Premier League) cricket matches. One dataset contains ball-by-ball & the other contains match-wise data. You have to import the datasets into an SQL database & perform the tasks to find important insights from the datasets.

Given Data:

- This below CSV file contains match-wise data & has data of 816 IPL matches. This table has 17 columns & below is the description of the columns in the table.

Column title	Description
id	Unqueie Match ID as per ESPNCricinfo
city	City in which stadium is located
date	Date on which match is held
player_of_match	Player awarded with best performance
venue	Stadium name
neutral_venue	Is the venue neutral i.e. is not homeground to the playing teams
team1	Team 1
team2	Team 2
toss_winner	Team who won the toss
toss_decision	Decision of the toss winner
winner	Match wiining team
result	Result based on victory by runs or by wickets
result_margin	Margin of wickets or runs
eliminator	Was a superover bowled or not
method	Was DL (duckworth lewis) method applied
umpire1	First umpire
umpire2	Second umpire

- This CSV file contains ball-by-ball data & it has info. Of all the 193468 balls between the years 2008 & 2020. It has 17 columns & below is the 17 columns.

Column title	Description
id	Unique Match ID as per ESPNCricinfo
inning	Inning Number
over	Over Number in an inning
ball	Ball Number in an over
batsman	Batsman on strike
non_striker	Batsman at non-striker end
bowler	Bowler
batsman_runs	Runs off bat
extra_runs	Extra runs
total_runs	Total Runs scored
is_wicket	Is the delivery a wicket?
dismissal_kind	Type of dismissal
player_dismissed	Player who got dismissed
fielder	Fielder involved in the dismissal
extras_type	Type of extras
batting_team	Team to which batsman belongs
bowling_team	Team to which bowler belongs

6.1) Project Code:

Queries for the following tasks:

1) Create a table named 'matches' with appropriate data types for columns

id	city	date	player_of_match	venue			
	neutral_venue		team1	team2	toss_winner		
	toss_decision	winner	result	result_margin			
	eliminator	method	umpire1	umpire2.			

➤ *CREATE table matches(
id int primary key,
city char(50),
date date ,
player_of_match char(100),
venue char(100),
neutral_venue int,
team_1 char(50),
team_2 char(50),
toss_winner char(50),
toss_decision char(10),
winner char(50),
result char(10),*

```
result_margin int,  
eliminator char(5),  
method varchar(5),  
umpire_1 char(50),  
umpire_2 char(50));
```

➤ *SELECT *From matches;*

2) Create a table named 'deliveries' with appropriate data types for columns

```
id   inning   over   ball   batsman   non_striker   bowler  
batsman_runs   extra_runs   total_runs  
is_wicketdismissal_kind   player_dismissed   fielder  
extras_type   batting_team   bowling_team
```

➤ *CREATE table deliveries(*

```
id int,  
inning int,  
over int,  
ball int,  
batsman char(100),  
non_striker char(100),  
bowler char(100),  
batsman_runs int,  
extra_runs int,
```

total_runs int,
is_wicket int,
dismissal_kind char(50),
player_dismissed char(100),
fielder char(100),
extras_type char(25),
batting_team char(50),
bowling_team char(50));

➤ *SELECT *From deliveries;*

3) Import data from csv file 'IPL_matches.csv' attached in resources to the table 'matches' which was created in Q1

➤ *"C:\Users\DELL\Downloads\IPL_matches.csv"*

4) Import data from csv file 'IPL_Ball.csv' attached in resources to the table 'deliveries' which was created in Q2

➤ *"C:\Users\DELL\Downloads\IPL_Ball.csv"*

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

➤ *SELECT id,over,ball*
From deliveries

ORDER BY id ASC

limit 20;

6) Select the top 20 rows of the matches table.

➤ *SELECT *From matches*

ORDER BY id ASC

limit 20;

7) Fetch data of all the matches played on 2nd May 2013 from the matches table.

➤ *SELECT *From matches*

WHERE date='02/05/2013';

8) 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';

9) 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;

10) Get the count of cities that have hosted an IPL match.

➤ *SELECT COUNT(DISTINCT city) as "Total_Venues"*
From matches;

11) 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)

(Hint 1 : CASE WHEN statement is used to get condition based results)

(Hint 2: To convert the output data of select statement into a table, you can use a subquery. Create table table_name as [entire select statement].

➤ *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;
*SELECT *FROM deliveries_v02;*

12) Write a query to fetch the total number of boundaries and dot balls from the deliveries_v02 table.

- *SELECT *From deliveries_v02;
SELECT ball_result, Count (*) From deliveries_v02 Where
ball_result <>'1,2,3' Group by ball_result;*

13) 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 by each team.

- *SELECT *From deliveries_v02;
SELECT batting_team,Count(*) From deliveries_v02
Where ball_result = 'boundary' Group by batting_team
Order by Count;*

14) 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 *From deliveries_v02;
SELECT bowling_team, Count(*) From deliveries_v02
Where ball_result = 'dot' Group by bowling_team Order
by Count;*

15) Write a query to fetch the total number of dismissals by dismissal kinds where dismissal kind is not NA

➤ *SELECT *From deliveries;*
SELECT dismissal_kind, Count() From deliveries*
Where dismissal_kind<>'NA' Group by dismissal_kind
ORDER by Count DESC;

16) Write a query to get the top 5 bowlers who conceded maximum extra runs from the deliveries table

➤ *SELECT Distinct "bowler" as Bowler, count*
(extra_runs) as "Max_extra_runs" from deliveries
group by bowler
Order by "Max_extra_runs" desc
Limit 5;

17) 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 a.*, b.venue,
b.match_date from deliveries_v02 as a
left join (select max(venue) as venue, max(date) as
match_date, id from matches
Group by id) as b on a.id = b.id;*

*SELECT *From deliveries_v03;*

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

➤ *SELECT *From deliveries_v03;*
*SELECT venue, sum(total_runs) as runs From
deliveries_v03 group by venue order by runs desc;*

19) 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 IPL_year,
sum(total_runs) as runs from deliveries_v03
where venue = 'Eden Gardens' group by IPL_year order
by runs desc;*

20) 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.

➤ *SELECT Distinct team_1 from matches;*

*CREATE table matches_corrected as select *,
replace(team_1, 'Rising Pune Supergiants', 'Rising
Pune Supergiant') as team1_corr, replace(team_2,
'Rising Pune Supergiants', 'Rising Pune Supergiant') as
team2_corr From matches;*

SELECT Distinct team1_corr from matches_corrected;

21) 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,'-',
over,'-',ball) as ball_id, *from deliveries_v03 ;
SELECT *From deliveries_v04;*

22) Compare the total count of rows and total count of distinct ball_id in deliveries_v04;

- *SELECT * from deliveries_v04 Limit 20;*
SELECT Count(distinct ball_id) From deliveries_v04;
SELECT Count() from deliveries_v04;*

23) Create table deliveries_05 with all columns of deliveries_04 and an additional column for row number.

- *CREATE table deliveries_v05 as SELECT *, row_number() over (partition by ball_id) as r_num From deliveries_v04;*

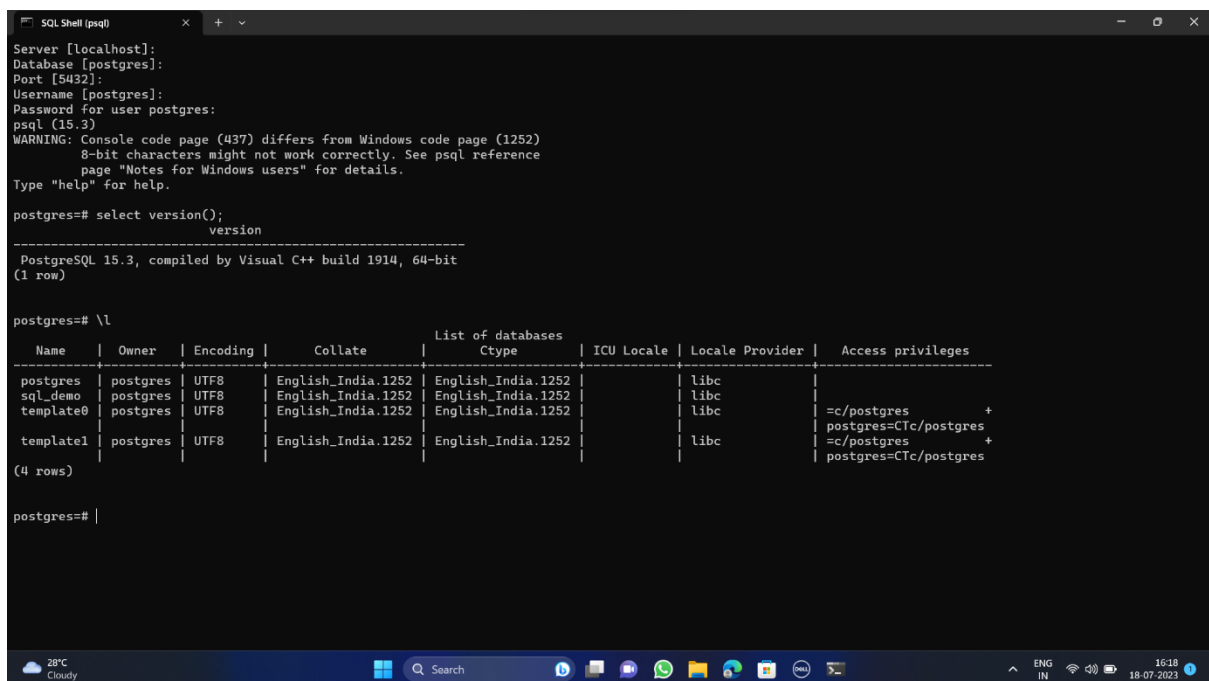
24) Use the r_num created in deliveries_v05 to identify instances where ball_id is repeating. (HINT : select * from deliveries_v05 WHERE r_num=2;)

- *SELECT *From deliveries_v05 WHERE r_num=2;*

25) 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=2);*

6.2) Project Snapshots:



```
SQL Shell (psql)
Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]:
Password for user postgres:
psql (15.3)
WARNING: Console code page (437) differs from Windows code page (1252)
8-bit characters might not work correctly. See psql reference
page "Notes for Windows users" for details.
Type "help" for help.

postgres=# select version();
          version
-----
PostgreSQL 15.3, compiled by Visual C++ build 1914, 64-bit
(1 row)

postgres=# \l

```

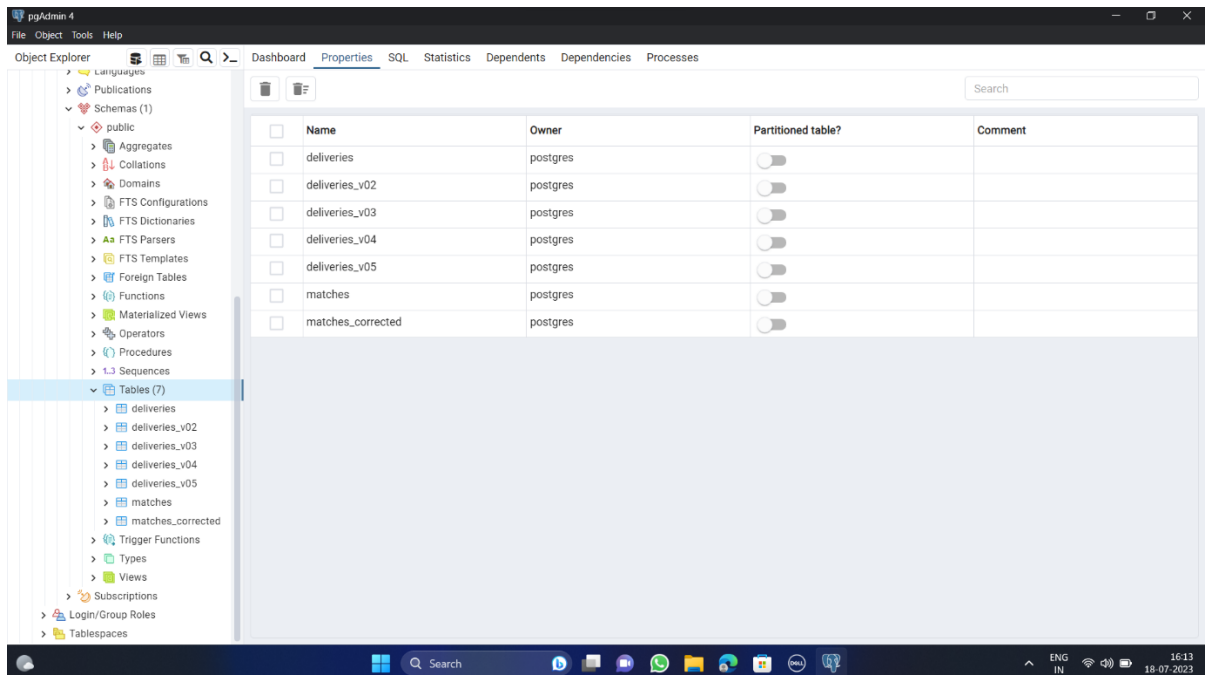
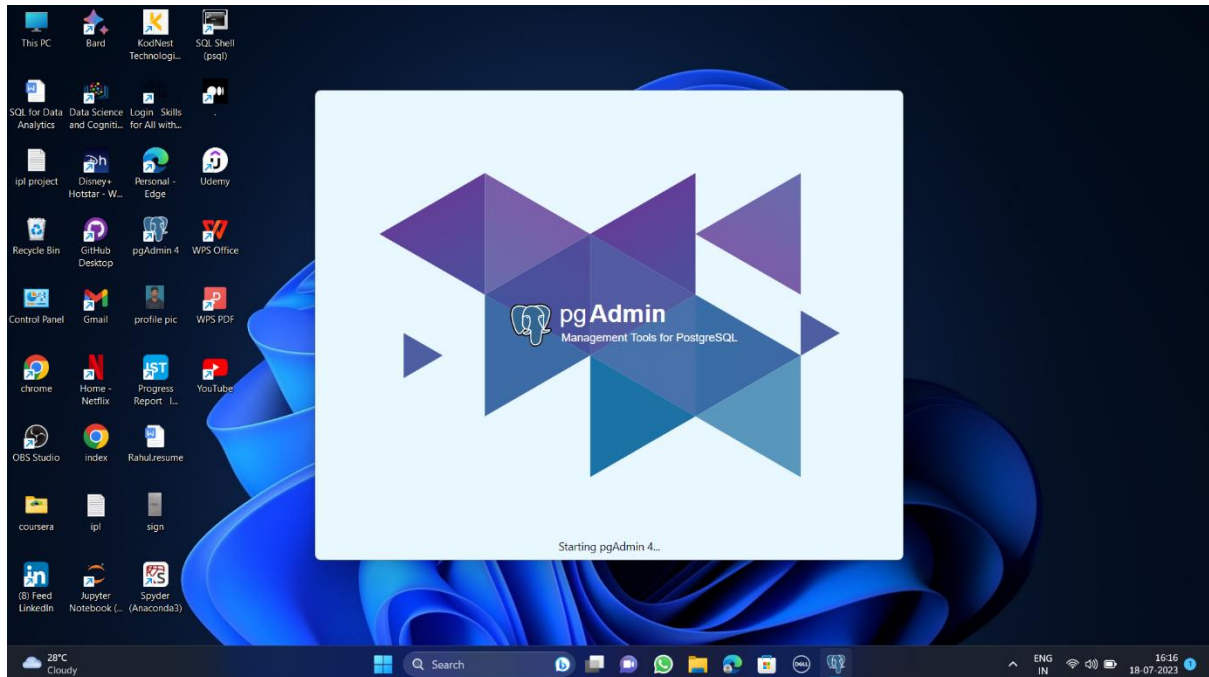
Name	Owner	Encoding	Collate	List of databases Ctype	ICU Locale	Locale Provider	Access privileges
postgres	postgres	UTF8	English_India.1252	English_India.1252		libc	
sql_demo	postgres	UTF8	English_India.1252	English_India.1252		libc	
template0	postgres	UTF8	English_India.1252	English_India.1252		libc	=c/postgres + postgres=Ctc/postgres
template1	postgres	UTF8	English_India.1252	English_India.1252		libc	=c/postgres + postgres=Ctc/postgres

(4 rows)

```
postgres=#
```

SQL Shell (psql)

PgAdmin 4



pgAdmin 4

File Object Tools Help

Object Explorer

- Publications
- Schemas (1)
 - public
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Operators
 - Procedures
 - Sequences
 - Tables (7)
 - deliveries
 - deliveries_v02
 - deliveries_v03
 - deliveries_v04
 - deliveries_v05
 - matches
 - matches_corrected
 - Trigger Functions
 - Types
 - Views
 - Subscriptions
 - Login/Group Roles
 - Tablespaces

Dashboard Properties SQL Statistics Dependents Dependencies Processes sql_demo/post... sql_demo/post... sql_demo/post... sql_demo/post

Query Query History

1 SELECT * From matches;

2

Data Output Messages Notifications

	id integer	city character	date date	player_of_match character	venue character	neutral integer
1	335982	Bangalore	2008-04-18	BB McCullum	M Chinnaswamy Stadium	...
2	335983	Chandigarh	2008-04-19	MEK Hussey	Punjab Cricket Association Stadium, Mohali	...
3	335984	Delhi	2008-04-19	MF Maharoo	Feroz Shah Kotla	...
4	335985	Mumbai	2008-04-20	MV Boucher	Wankhede Stadium	...
5	335986	Kolkata	2008-04-20	DJ Hussey	Eden Gardens	...
6	335987	Jaipur	2008-04-21	SR Watson	Sawai Mansingh Stadium	...
7	335988	Hyderabad	2008-04-22	V Sehwag	Rajiv Gandhi International Stadium, Uppal	...
8	335989	Chennai	2008-04-23	ML Hayden	MA Chidambaram Stadium, Chepauk	...
9	335990	Hyderabad	2008-04-24	YK Pathan	Rajiv Gandhi International Stadium, Uppal	...
10	335991	Chandigarh	2008-04-25	KC Sangakkara	Punjab Cricket Association Stadium, Mohali	...
11	335992	Bangalore	2008-04-26	SR Watson	M Chinnaswamy Stadium	...
12	335993	Chennai	2008-04-26	JDP Oram	MA Chidambaram Stadium, Chepauk	...
13	335994	Mumbai	2008-04-27	AC Gilchrist	Dr DY Patil Sports Academy	...
14	335995	Chandigarh	2008-04-27	SM Katich	Punjab Cricket Association Stadium, Mohali	...
15	335996	Bangalore	2008-04-28	MS Dhoni	M Chinnaswamy Stadium	...
16	335997	Kolkata	2008-04-29	ST Jayasuriya	Eden Gardens	...
17	335998	Delhi	2008-04-30	GD McGrath	Feroz Shah Kotla	...

Total rows: 816 of 816 Query complete 00:00:00.282 Ln 2, Col 1

pgAdmin 4

File Object Tools Help

Object Explorer

- Publications
- Schemas (1)
 - public
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Operators
 - Procedures
 - Sequences
 - Tables (7)
 - deliveries
 - deliveries_v02
 - deliveries_v03
 - deliveries_v04
 - deliveries_v05
 - matches
 - matches_corrected
 - Trigger Functions
 - Types
 - Views
 - Subscriptions
 - Login/Group Roles
 - Tablespaces

Dashboard Properties SQL Statistics Dependents Dependencies Processes sql_demo/post... sql_demo/post... sql_demo/post... sql_demo/post

Query Query History

1 SELECT * From deliveries;

2

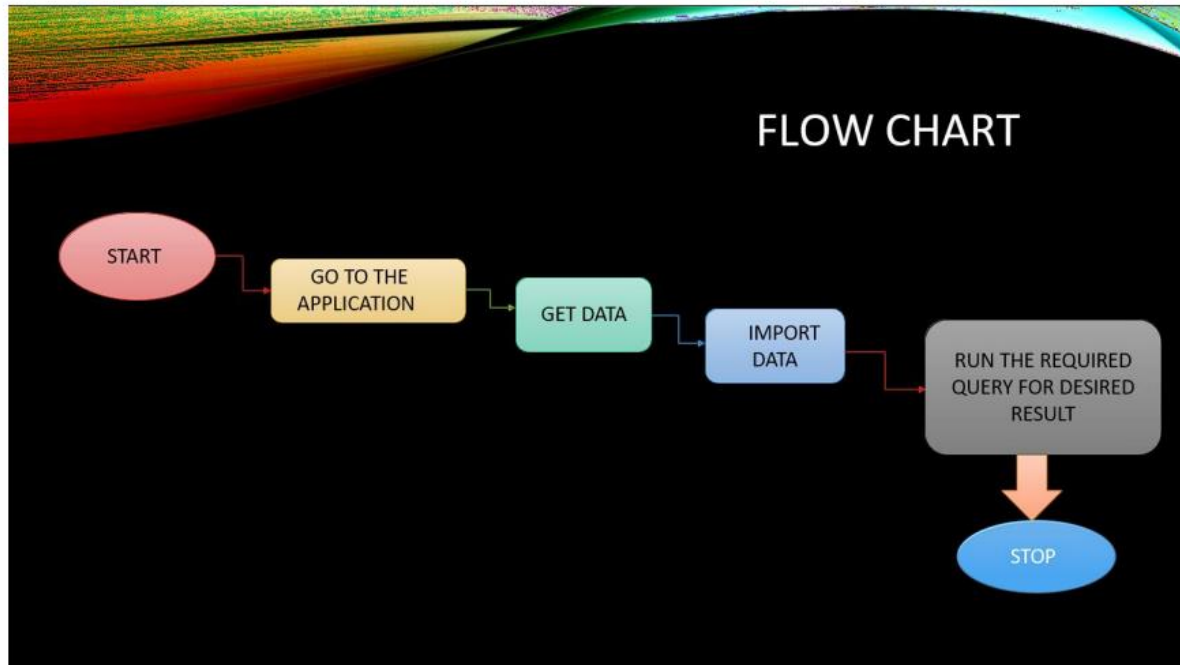
Data Output Messages Notifications

	id integer	inning integer	over integer	ball integer	batsman character varying (100)	non_striker character varying (100)	bowler character varying (100)	batsman_runs integer	extra_runs integer	total_runs integer	is_wicket integer
1	335982	1	6	5	RT Ponting	BB McCullum	AA Noffke	1	0	1	0
2	335982	1	6	6	BB McCullum	RT Ponting	AA Noffke	1	0	1	0
3	335982	1	7	1	BB McCullum	RT Ponting	Z Khan	0	0	0	0
4	335982	1	7	2	BB McCullum	RT Ponting	Z Khan	1	0	1	0
5	335982	1	7	3	RT Ponting	BB McCullum	Z Khan	1	0	1	0
6	335982	1	7	4	BB McCullum	RT Ponting	Z Khan	1	0	1	0
7	335982	1	7	5	RT Ponting	BB McCullum	Z Khan	1	0	1	0
8	335982	1	7	6	BB McCullum	RT Ponting	Z Khan	1	0	1	0
9	335982	1	8	1	BB McCullum	RT Ponting	JH Kallis	0	0	0	0
10	335982	1	8	2	BB McCullum	RT Ponting	JH Kallis	0	0	0	0
11	335982	1	8	3	BB McCullum	RT Ponting	JH Kallis	0	0	0	0
12	335982	1	8	4	BB McCullum	RT Ponting	JH Kallis	1	0	1	0
13	335982	1	8	5	RT Ponting	BB McCullum	JH Kallis	1	0	1	0
14	335982	1	8	6	BB McCullum	RT Ponting	JH Kallis	2	0	2	0
15	335982	1	9	1	RT Ponting	BB McCullum	SB Joshi	1	0	1	0
16	335982	1	9	2	BB McCullum	RT Ponting	SB Joshi	1	0	1	0
17	335982	1	9	3	RT Ponting	BB McCullum	BB McCullum	1	0	1	0

Total rows: 1000 of 193468 Query complete 00:00:00.617 Ln 1, Col 1

Successfully run. Total query runtime: 617 msec. 193468 rows affected.

7) Flow Chart:



8) Applications:

- It is a form of Business Intelligence which helps in solving specific problems & challenges.
- For instance, through Data Analytics, an e-commerce company can track customer behaviour & use the insights to improve the entire experience.
- It is mainly used or managing large & complex databases.

9) Conclusion:

Now I've analyzed our data, the last step is to draw your conclusions. Conclusions summarize whether the experiment or survey results support or contradict the original hypothesis. We should include key facts from our background research to help explain the results.

10) References:

- <https://www.google.com/>
- <https://www.postgresql.org/docs>
- <https://www.simplilearn.com/>

Thank You