**Process Reproducibility Document**

Trends and descriptive analytics

1. **How many trips were there in each month of each year?**

Company name: Blue bikes

|  |
| --- |
| bluebikes\_2016 |
| bluebikes \_2017 |
| bluebikes \_2018 |
| bluebikes \_2019 |

Data set considered:

SQL QUERY:

WITH

cte AS (SELECT \* FROM bluebikes\_2016

UNION

SELECT \* FROM bluebikes\_2017

UNION

SELECT \* FROM bluebikes\_2018

UNION

SELECT \* FROM bluebikes\_2019

)

SELECT DATE\_PART('year',cte.start\_time) AS year,DATE\_PART('month',cte.start\_time) AS month,COUNT(\*)

FROM cte

GROUP BY DATE\_PART('year',cte.start\_time), DATE\_PART('month',cte.start\_time)

ORDER BY year,month

Data Handling Summary:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Columns | Data type | Description | Action to be taken | Action taken |
| Year | Double precision | Shows the date, year | No action | No action |
| Month | Double precision | Shows the month, year | No action | No action |
| Count | Big int | Number of trips taken | No action | No action |
| Month | Text | This is calculated field | Additional column to be added to display month in text format | additional column added |

1. **Which organisations are showing the most growth in bike rentals?**

Dataset considered:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | bluebikes\_2016 | capitalbikeshare\_2016 | divvybikes\_2016 | santander\_2016 |
| baywheels\_2017 | bluebikes\_2017 | capitalbikeshare\_2017 | divvybikes\_2017 | santander\_2017 |
| baywheels\_2018 | bluebikes\_2018 | capitalbikeshare\_2018 | divvybikes\_2018 | santander\_2018 |
| baywheels\_2019 | bluebikes\_2019 | capitalbikeshare\_2019 | divvybikes\_2019 | santander\_2019 |

SQL QUERY:

WITH

cte\_baywheels AS(

SELECT bike\_id, start\_time FROM baywheels\_2017

UNION

SELECT bike\_id, start\_time FROM baywheels\_2018

UNION

SELECT bike\_id, start\_time FROM baywheels\_2019

),

cte\_bluebikes AS (

SELECT bike\_id, start\_time FROM bluebikes\_2016

UNION

SELECT bike\_id, start\_time FROM bluebikes\_2017

UNION

SELECT bike\_id, start\_time FROM bluebikes\_2018

UNION

SELECT bike\_id, start\_time FROM bluebikes\_2019

),

cte\_capitalbikeshare AS

(SELECT bike\_id,start\_time FROM capitalbikeshare\_2016

UNION

SELECT bike\_id,start\_time FROM capitalbikeshare\_2017

UNION

SELECT bike\_id,start\_time FROM capitalbikeshare\_2018

UNION

SELECT bike\_id,start\_time FROM capitalbikeshare\_2019

),

cte\_divvybikes AS

( SELECT trip\_id AS bike\_id,start\_time FROM divvybikes\_2016

UNION

SELECT trip\_id AS bike\_id,start\_time FROM divvybikes\_2017

UNION

SELECT trip\_id AS bike\_id,start\_time FROM divvybikes\_2018

UNION

SELECT trip\_id AS bike\_id,start\_time FROM divvybikes\_2019

),

cte\_santander AS

(SELECT rental\_id AS bike\_id,start\_date AS start\_time FROM santander\_2016

UNION

SELECT rental\_id AS bike\_id,start\_date AS start\_time FROM santander\_2017

UNION

SELECT rental\_id AS bike\_id,start\_date AS start\_time FROM santander\_2018

UNION

SELECT rental\_id AS bike\_id,start\_date AS start\_time FROM santander\_2019

)

SELECT CONCAT('baywheels','-',DATE\_PART('year',cte\_baywheels.start\_time)) AS company\_name\_year,COUNT(\*) AS count\_trips

FROM cte\_baywheels

GROUP BY DATE\_PART('year',cte\_baywheels.start\_time)

UNION

SELECT CONCAT('bluebikes','-',DATE\_PART('year',cte\_bluebikes.start\_time))AS company\_name\_year,COUNT(\*) AS count\_trips

FROM cte\_bluebikes

GROUP BY DATE\_PART('year',cte\_bluebikes.start\_time)

UNION

SELECT CONCAT('capitalbikeshare','-',DATE\_PART('year',cte\_capitalbikeshare.start\_time))AS company\_name\_year,COUNT(\*)AS count\_trips

FROM cte\_capitalbikeshare

GROUP BY DATE\_PART('year',cte\_capitalbikeshare.start\_time)

UNION

SELECT CONCAT('divvybikes','-',DATE\_PART('year',cte\_divvybikes.start\_time)) AS company\_name\_year,COUNT(\*) AS count\_trips

FROM cte\_divvybikes

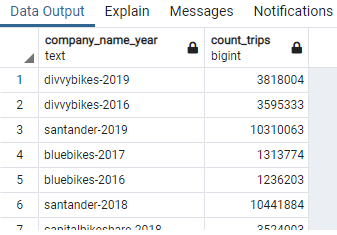
GROUP BY DATE\_PART('year',cte\_divvybikes.start\_time)

UNION

SELECT CONCAT('santander','-',DATE\_PART('year',cte\_santander.start\_time))AS company\_name\_year,COUNT(\*) AS count\_trips

FROM cte\_santander

GROUP BY DATE\_PART('year',cte\_santander.start\_time)



Data Handling Summary:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Columns | Data Type | Description | Action to be taken | Action taken |
| Company\_name\_year | text | Shows Company name and the year | Split the column to keep the company name and year separately in two columns.  Rename the column AS ‘company\_name’ | Column is split to keep the company name and year separately and column is renamed. |
| Count\_trips | bigint | Number of trips taken | No action required | No action |
| Year- calculated field | text | This field is created by splitting the company\_name\_year. | Name the column as ‘year’ | Column is renamed. |

1. **Is there a difference in growth between holiday activity and commuting activity?**

Dataset considered:

|  |  |
| --- | --- |
| bluebikes\_2018 | divvybikes\_2018 |
| bluebikes\_2019 | divvybikes\_2019 |

SQL Query:

WITH

cte\_bluebikes AS

( SELECT bike\_id,to\_char(start\_time,'dy') AS day\_of\_week

FROM bluebikes\_2018

UNION

SELECT bike\_id,to\_char(start\_time,'dy') AS day\_of\_week

FROM bluebikes\_2019 ),

cte\_divvybikes AS

( SELECT trip\_id AS bike\_id,to\_char(start\_time,'dy') AS day\_of\_week

FROM divvybikes\_2018

UNION

SELECT trip\_id AS bike\_id ,to\_char(start\_time,'dy') AS day\_of\_week

FROM divvybikes\_2019

)

SELECT CONCAT('bluebikes','-',cte\_bluebikes.day\_of\_week) AS day\_of\_week,COUNT(cte\_bluebikes.bike\_id)

FROM cte\_bluebikes

GROUP BY cte\_bluebikes.day\_of\_week

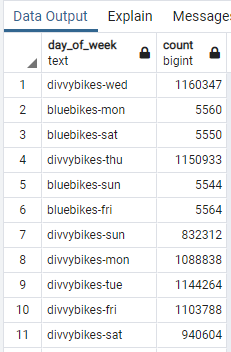
UNION

SELECT CONCAT('divvybikes','-',cte\_divvybikes.day\_of\_week) AS days\_of\_week,COUNT(cte\_divvybikes.bike\_id)

FROM cte\_divvybikes

GROUP BY cte\_divvybikes.day\_of\_week

Screenshot of Output:



Data Handling Summary:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Columns | Data type | Description | Action to be taken | Action taken |
| day\_of\_week | text | Company name is associated with the day of the week. | Split this column to keep company name and day of week separately in two columns. | New column created to keep the company name and day of week in separate columns. |
| count | bigint | Gives a count of the trips taken. | No action required | No action taken |
| company\_name |  | New column created by splitting the day\_of week column | Give a column heading to the new column created. | Column name named as company\_name. |

Geospatial

1. **What was the longest journey? What do we know about it?**

|  |  |
| --- | --- |
| bluebikes\_2019 | bluebikes\_2018 |
| divvybikes\_2019 | divvybikes\_2018 |

Dataset considered:

SQL Query to calculate longest journey in Bluebikes, 2019:

WITH

cte AS (

SELECT bb.bike\_id AS bike\_id ,bb.start\_time AS start\_time,bb.end\_time AS end\_time,(bb.end\_time-bb.start\_time) AS time\_taken,bb.start\_station\_id AS start\_station\_id,

bbs1.latitude AS start\_station\_latitude,bbs1.longtitude AS start\_station\_longitude,bb.end\_station\_id AS end\_station\_id,

bbs2.latitude AS end\_station\_latitude,bbs2.longtitude AS end\_station\_longitude,

calculate\_distance(bbs1.latitude,bbs1.longtitude,bbs2.latitude,bbs2.longtitude,'K') AS calc\_dist,

bb.user\_type AS user\_type,bb.user\_birth\_year AS user\_birth\_year,bb.user\_gender AS user\_gender

FROM bluebikes\_2019 bb

LEFT JOIN bluebikes\_stations bbs1

ON bbs1.id = bb.start\_station\_id

LEFT JOIN bluebikes\_stations bbs2

ON bbs2.id = bb.end\_station\_id

WHERE bbs1.latitude IS NOT NULL AND bbs1.longtitude IS NOT NULL AND

bbs2.latitude IS NOT NULL AND bbs2.longtitude IS NOT NULL

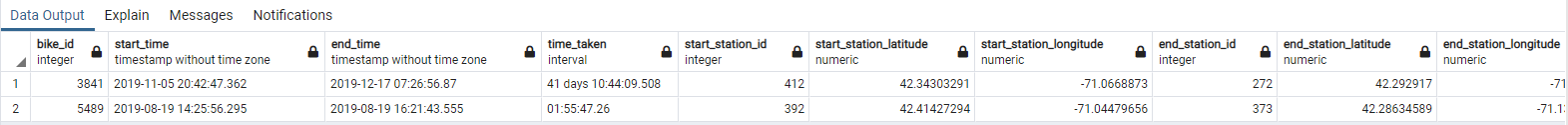
)

SELECT cte.bike\_id,cte.start\_time,cte.end\_time,cte.time\_taken,cte.start\_station\_id,cte.start\_station\_latitude,cte.start\_station\_longitude,cte.end\_station\_id,

cte.end\_station\_latitude,cte.end\_station\_longitude,cte.calc\_dist,cte.user\_type,cte.user\_birth\_year,cte.user\_gender

FROM cte

WHERE cte.calc\_dist = (SELECT MAX(c.calc\_dist) FROM cte c) OR cte.time\_taken =(SELECT MAX(c.time\_taken) FROM cte c)



SQL Query to calculate longest journey in Bluebikes, 2018:

WITH

cte AS (

SELECT bb.bike\_id AS bike\_id ,bb.start\_time AS start\_time,bb.end\_time AS end\_time,(bb.end\_time-bb.start\_time) AS time\_taken,bb.start\_station\_id AS start\_station\_id,

bbs1.latitude AS start\_station\_latitude,bbs1.longtitude AS start\_station\_longitude,bb.end\_station\_id AS end\_station\_id,

bbs2.latitude AS end\_station\_latitude,bbs2.longtitude AS end\_station\_longitude,

calculate\_distance(bbs1.latitude,bbs1.longtitude,bbs2.latitude,bbs2.longtitude,'K') AS calc\_dist,

bb.user\_type AS user\_type,bb.user\_birth\_year AS user\_birth\_year,bb.user\_gender AS user\_gender

FROM bluebikes\_2018 bb

LEFT JOIN bluebikes\_stations bbs1

ON bbs1.id = bb.start\_station\_id

LEFT JOIN bluebikes\_stations bbs2

ON bbs2.id = bb.end\_station\_id

WHERE bbs1.latitude IS NOT NULL AND bbs1.longtitude IS NOT NULL AND

bbs2.latitude IS NOT NULL AND bbs2.longtitude IS NOT NULL

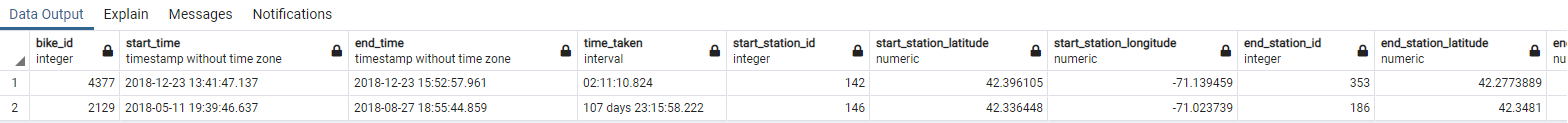
)

SELECT cte.bike\_id,cte.start\_time,cte.end\_time,cte.time\_taken,cte.start\_station\_id,cte.start\_station\_latitude,cte.start\_station\_longitude,cte.end\_station\_id,

cte.end\_station\_latitude,cte.end\_station\_longitude,cte.calc\_dist,cte.user\_type,cte.user\_birth\_year,cte.user\_gender

FROM cte

WHERE cte.calc\_dist = (SELECT MAX(c.calc\_dist) FROM cte c) OR cte.time\_taken =(SELECT MAX(c.time\_taken) FROM cte c)



SQL Query to calculate longest journey in Divvybikes, 2019:

WITH cte2 AS(

SELECT dd.trip\_id AS bike\_id ,dd.start\_time AS start\_time,dd.end\_time AS end\_time,(dd.end\_time-dd.start\_time) AS time\_taken, dd.start\_station\_id AS start\_station\_id,

dds1.latitude AS start\_station\_latitude,dds1.longitude AS start\_station\_longitude,dd.end\_station\_id AS end\_station\_id,

dds2.latitude AS end\_station\_latitude,dds2.longitude AS end\_station\_longitude,

calculate\_distance(dds1.latitude,dds1.longitude,dds2.latitude,dds2.longitude,'K') AS calc\_dist,

dd.user\_type AS user\_type,dd.birthyear AS user\_birth\_year,dd.gender AS user\_gender

FROM divvybikes\_2019 dd

LEFT JOIN divvy\_stations dds1

ON dds1.id = dd.start\_station\_id

LEFT JOIN divvy\_stations dds2

ON dds2.id = dd.end\_station\_id

WHERE dds1.latitude IS NOT NULL AND dds1.longitude IS NOT NULL AND

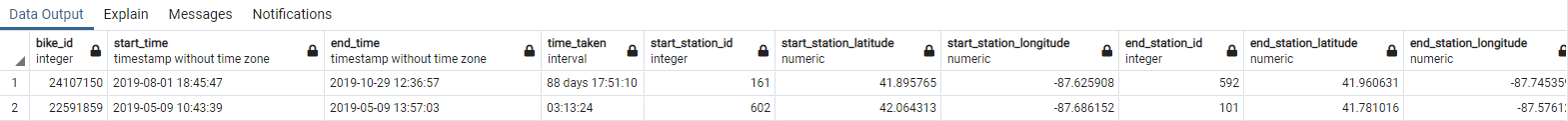
dds2.latitude IS NOT NULL AND dds2.longitude IS NOT NULL )

SELECT cte2.bike\_id,cte2.start\_time,cte2.end\_time,cte2.time\_taken,cte2.start\_station\_id,cte2.start\_station\_latitude,cte2.start\_station\_longitude,cte2.end\_station\_id,

cte2.end\_station\_latitude,cte2.end\_station\_longitude,cte2.calc\_dist,cte2.user\_type,cte2.user\_birth\_year,cte2.user\_gender

FROM cte2

WHERE cte2.calc\_dist = (SELECT MAX(c.calc\_dist) FROM cte2 c) OR cte2.time\_taken =(SELECT MAX(c.time\_taken) FROM cte2 c)



SQL Query to calculate longest journey in Divvybikes, 2018:

WITH cte2 AS(

SELECT dd.trip\_id AS bike\_id ,dd.start\_time AS start\_time,dd.end\_time AS end\_time,(dd.end\_time-dd.start\_time) AS time\_taken, dd.start\_station\_id AS start\_station\_id,

dds1.latitude AS start\_station\_latitude,dds1.longitude AS start\_station\_longitude,dd.end\_station\_id AS end\_station\_id,

dds2.latitude AS end\_station\_latitude,dds2.longitude AS end\_station\_longitude,

calculate\_distance(dds1.latitude,dds1.longitude,dds2.latitude,dds2.longitude,'K') AS calc\_dist,

dd.user\_type AS user\_type,dd.birthyear AS user\_birth\_year,dd.gender AS user\_gender

FROM divvybikes\_2018 dd

LEFT JOIN divvy\_stations dds1

ON dds1.id = dd.start\_station\_id

LEFT JOIN divvy\_stations dds2

ON dds2.id = dd.end\_station\_id

WHERE dds1.latitude IS NOT NULL AND dds1.longitude IS NOT NULL AND

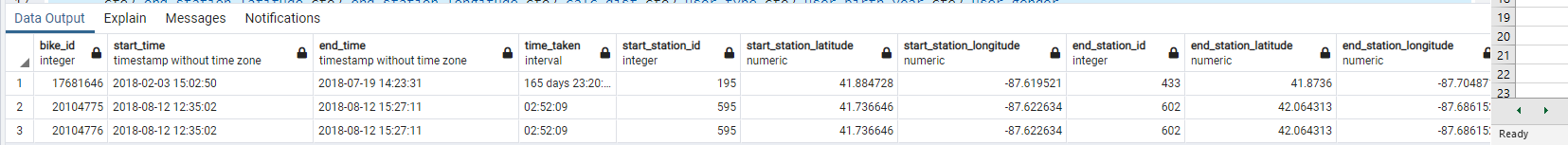
dds2.latitude IS NOT NULL AND dds2.longitude IS NOT NULL )

SELECT cte2.bike\_id,cte2.start\_time,cte2.end\_time,cte2.time\_taken,cte2.start\_station\_id,cte2.start\_station\_latitude,cte2.start\_station\_longitude,cte2.end\_station\_id,

cte2.end\_station\_latitude,cte2.end\_station\_longitude,cte2.calc\_dist,cte2.user\_type,cte2.user\_birth\_year,cte2.user\_gender

FROM cte2

WHERE cte2.calc\_dist = (SELECT MAX(c.calc\_dist) FROM cte2 c) OR cte2.time\_taken =(SELECT MAX(c.time\_taken) FROM cte2 c)



Data Handling Summary:

1. Created new column with column fields company\_name, year, bike\_id, start\_time, end\_time, time\_taken, calc\_dist, user\_type, user\_birthyear, user\_gender.
2. Transported the data values from the current rows to their corresponding column fields created newly.
3. Substituted the values of 0 and 1 in the user\_gender column to “unknown” and “male” respectively. Reference for this info directly taken from Bluebikes website.
4. **How often do bikes need to be relocated?**

|  |
| --- |
| bluebikes\_2019 |
| divvybikes\_2019 |

Dataset considered:

SQL QUERY:

WITH

cte AS (

SELECT bike\_id,start\_station\_id,end\_station\_id,LAG(end\_station\_id,1) OVER( PARTITION BY bike\_id ORDER BY start\_time) AS end\_st\_id\_lag,start\_time,end\_time

FROM bluebikes\_2019

ORDER BY bike\_id,start\_time

),

cte2 AS (

SELECT cte.bike\_id AS bike\_id,COUNT(cte.bike\_id) AS relocated\_count

FROM cte

WHERE cte.start\_station\_id !=cte.end\_st\_id\_lag

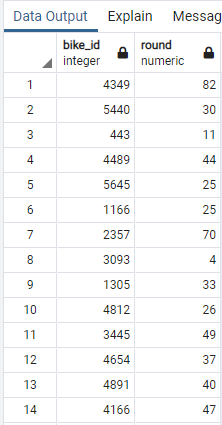
GROUP BY cte.bike\_id )

SELECT DISTINCT cte2.bike\_id AS bike\_id ,ROUND(AVG(cte2.relocated\_count))

FROM cte2

GROUP BY bike\_id

Output from bluebikes\_2019



WITH

cte AS (

SELECT bikeid,start\_station\_id,end\_station\_id,LAG(end\_station\_id,1) OVER( PARTITION BY bikeid ORDER BY start\_time) AS end\_st\_id\_lag,start\_time,end\_time

FROM divvybikes\_2019

ORDER BY bikeid,start\_time

),

cte2 AS (

SELECT cte.bikeid AS bike\_id,COUNT(cte.bikeid) AS relocated\_count

FROM cte

WHERE cte.start\_station\_id !=cte.end\_st\_id\_lag

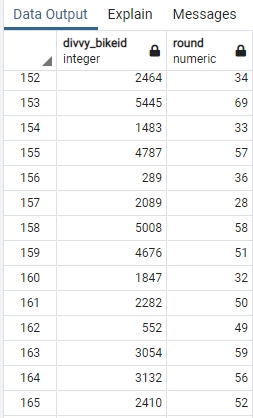
GROUP BY cte.bikeid )

SELECT DISTINCT cte2.bike\_id AS divvy\_bikeid ,ROUND(AVG(cte2.relocated\_count))

FROM cte2

GROUP BY divvy\_bikeid

Output from divvybikes\_2019



Data Handling Summary:

1. Created new columns,company\_name, year, bike\_id, relocated\_count.
2. Transported the values of bike\_id and relocated\_count corresponding to the company\_name and year.

1. **How fast do people cycle?**

|  |
| --- |
| bluebikes\_2019 |
| divvybikes\_2019 |

Dataset considered:

SQL QUERY for bluebikes\_2019 dataset :

WITH

cte AS (

SELECT bike\_id,start\_time,end\_time,EXTRACT( minute FROM (end\_time-start\_time))AS time\_taken,

calculate\_distance(bbs1.latitude,bbs1.longtitude,bbs2.latitude,bbs2.longtitude,'K')AS distance\_travelled,

bbs1.latitude,bbs1.longtitude,bbs2.latitude,bbs2.longtitude

FROM bluebikes\_2019 bb

LEFT JOIN bluebikes\_stations bbs1

ON bbs1.id = bb.start\_station\_id

LEFT JOIN bluebikes\_stations bbs2

ON bbs2.id = bb.end\_station\_id

WHERE bbs1.latitude IS NOT NULL AND bbs1.longtitude IS NOT NULL

AND bbs2.latitude IS NOT NULL AND bbs2.longtitude IS NOT NULL

ORDER BY bb.bike\_id,bb.start\_time

LIMIT 100000

)

SELECT bike\_id,CASE

WHEN cte.time\_taken = 0 THEN 0

ELSE (cte.distance\_travelled/cte.time\_taken)

END AS speed

FROM cte

SQL QUERY for divvybikes\_2019 dataset:

WITH

cte AS (

SELECT trip\_id AS bike\_id,start\_time,end\_time,EXTRACT( minute FROM (end\_time-start\_time))AS time\_taken,

calculate\_distance(dds1.latitude,dds1.longitude,dds2.latitude,dds2.longitude,'K')AS distance\_travelled,

dds1.latitude,dds1.longitude,dds2.latitude,dds2.longitude

FROM divvybikes\_2019 dd

LEFT JOIN divvy\_stations dds1

ON dds1.id = dd.start\_station\_id

LEFT JOIN divvy\_stations dds2

ON dds2.id = dd.end\_station\_id

WHERE dds1.latitude IS NOT NULL AND dds1.longitude IS NOT NULL

AND dds2.latitude IS NOT NULL AND dds2.longitude IS NOT NULL

ORDER BY dd.trip\_id,dd.start\_time

LIMIT 100000

)

SELECT cte.bike\_id,CASE

WHEN cte.time\_taken = 0 THEN 0

ELSE (cte.distance\_travelled/cte.time\_taken)

END AS speed

FROM cte

Data Handling Summary:

1. Created new columns, company\_name, year, bike\_id, speed.
2. Transported the values of bike\_id and speed corresponding to the company\_name and year.