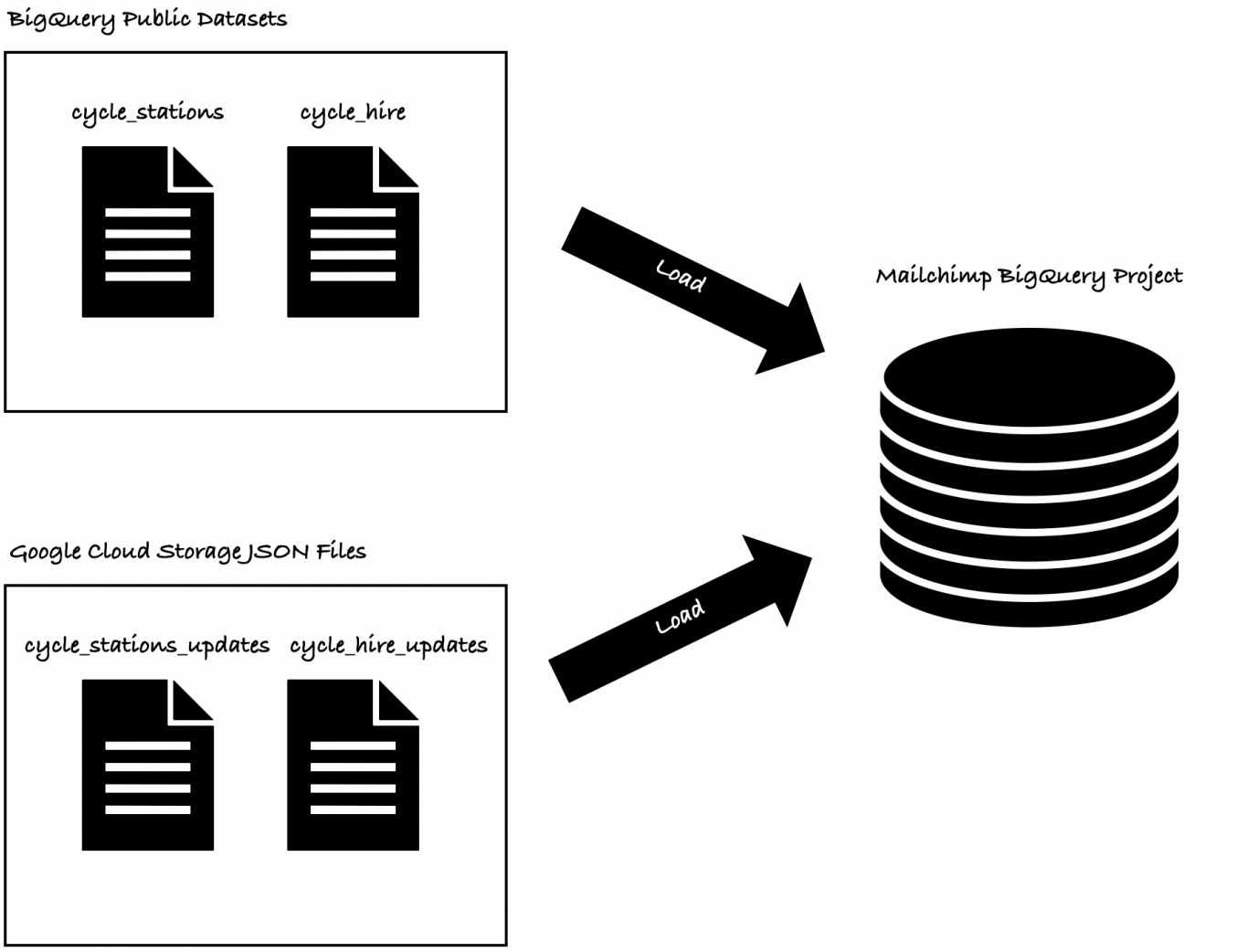
Step 1

Load data into BigQuery Project. Bring data from multiple sources into one place.



As shown in the above diagram, we have two data sources:
1. BigQuery Public Dataset: London Bicycles data
2. Google Cloud Storage JSON files that has the records that needs to be updated in the London Bicycles data.

Steps to copy public dataset into BigQuery project:
i. Created a new project folder inside BigQuery named: "Mailchimp BigQuery Project"
ii. Go to the Public dataset link that has been provided: London Bicycle data set
iii. Click on copy dataset
iv. Create new dataset inside "Mailchimp BigQuery Project"
v. Specify the source dataset (ii) and destination dataset that has been created in the step (iv).
vi. Specify the correct Google Cloud location. In our case, it is EU, as the dataset resides in the EU region of google cloud.
vii. Click on copy and let the job run for few minutes. After few minutes, dataset is ready to use in your project folder.

Steps to copy public dataset into BigQuery project:

i. Click on the three dots besides the London bicycle dataset inside your project, create new table, specify source as google cloud storage.
ii. Specify the correct bucket name and file name.
iii. Specify the destination dataset (Can create this on the spot).
iv. Specify Schema details or you can click on auto detect schema.
v. Click on create table and wait for few seconds to let the BigQuery convert the JSON file into tabular format. Once the processing is done, you have the data ready to use inside your project in the table format.

Step 2

Transform data. Check data quality issues. Ensure the accuracy and quality of data. Update data.

a. Initial Exploration

i) Quick counts check and making sure the keys are unique.

```
-- 1. Quick counts check and making sure the keys are unique.
SELECT count(*), count(distinct id) FROM `mailchimp-bigquery-project-1.eu_london_bicycles.cycle_stations`; -- 788 records
SELECT count(*), count(distinct id) FROM `mailchimp-bigquery-project-1.eu_london_bicycles.cycle_stations_update`; -- 791 records

SELECT count(*), count(distinct rental_id) FROM `mailchimp-bigquery-project-1.eu_london_bicycles.cycle_hire`; -- ~ 24.4M records
SELECT count(*), count(distinct rental_id) FROM `mailchimp-bigquery-project-1.eu_london_bicycles.cycle_hire_update`; -- 1.2M records
```

ii) Make sure that the records in update tables are present in the base tables.

```
-- 2. making sure that the records in update tables are present in the base tables.
select count(*), count(distinct s.id)
from `mailchimp-bigquery-project-1.eu_london_bicycles.cycle_stations` s
inner join `mailchimp-bigquery-project-1.eu_london_bicycles.cycle_stations_update` su on cast(s.id as string) = cast(su.id as string); -- 783 recor

select count(*), count(distinct h.rental_id)
from `mailchimp-bigquery-project-1.eu_london_bicycles.cycle_hire` h
inner join `mailchimp-bigquery-project-1.eu_london_bicycles.cycle_hire_update` hu on cast(h.rental_id as string) = hu.rental_id; -- 1.2M records (e
```

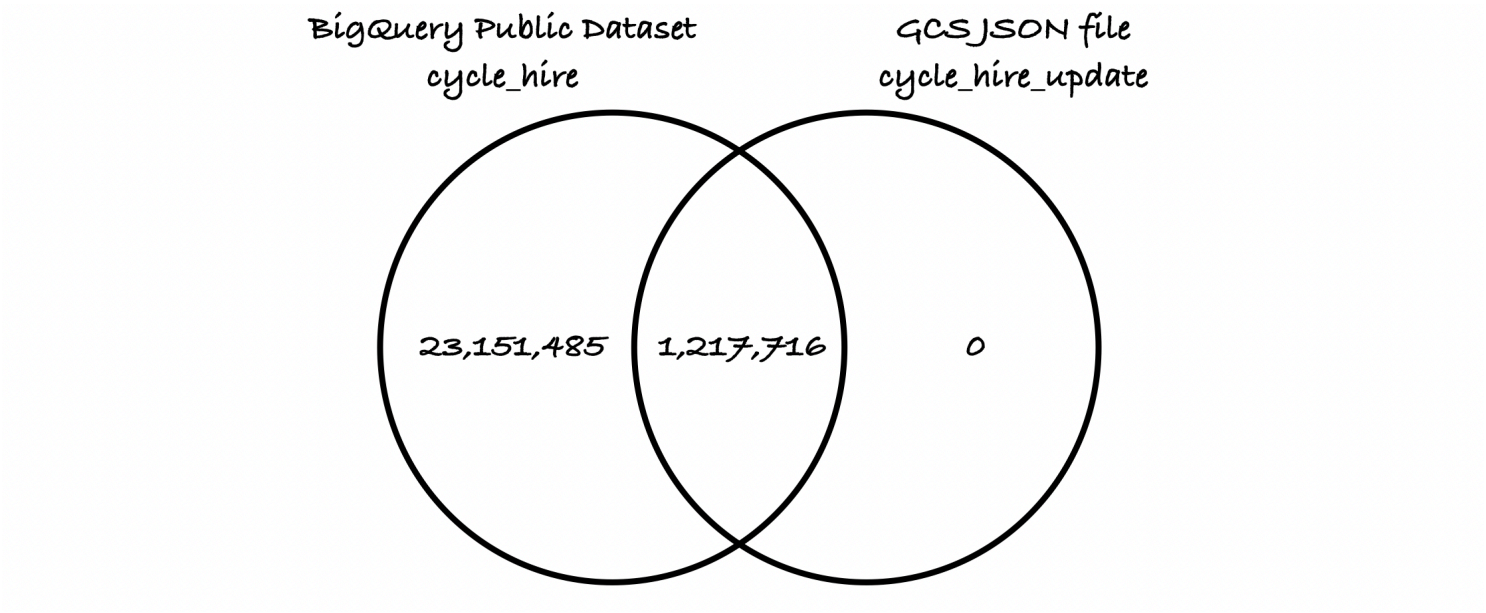
iii) In Stations table, find records that are not matching. Run comparisons. Result: 8 records missing in the base station table

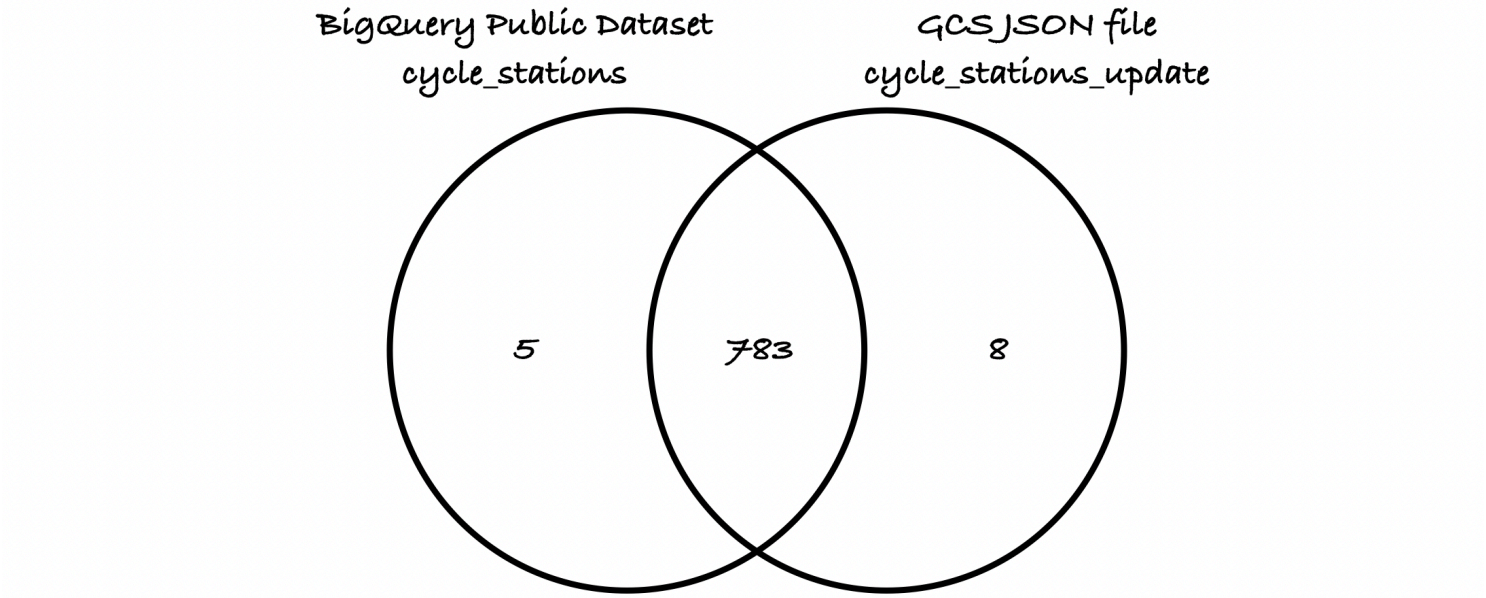
```
-- 3. In Stations table, find records that are not matching.
-- how many records do not need update?
select count(distinct id) from eu_london_bicycles.cycle_stations
where id not in (select id from eu_london_bicycles.cycle_stations_update);

-- how many records do need update?
select count(distinct id) from eu_london_bicycles.cycle_stations
where id in (select id from eu_london_bicycles.cycle_stations_update);

-- how many records are not present in the base table?
select count(distinct id) from eu_london_bicycles.cycle_stations_update
where id not in (select id from eu_london_bicycles.cycle_stations);
-- In case of a real life project, I would have investigated more on these records on - why they are missing?
```

iv) In hire table, find records that are not matching. Run comparisons.

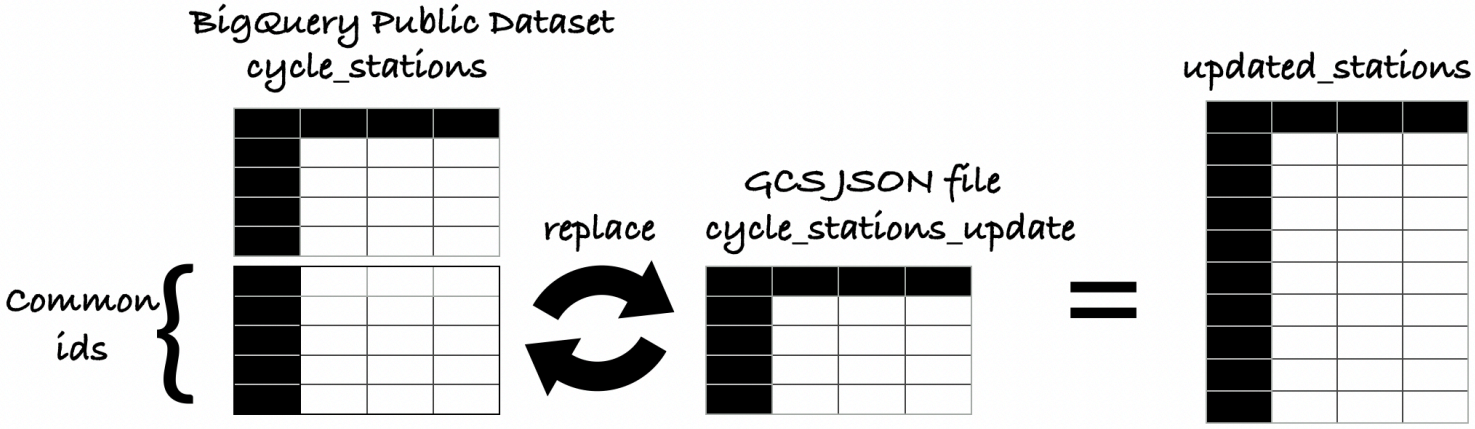
```
-- 4. In hire table, find records that are not matching.
-- how many records do not need update?
select count(distinct rental_id) from eu_london_bicycles.cycle_hire
where rental_id not in (select rental_id from eu_london_bicycles.cycle_hire_update);

-- how many records do need update?
select count(distinct rental_id) from eu_london_bicycles.cycle_hire
where rental_id in (select rental_id from eu_london_bicycles.cycle_hire_update);

-- how many records are not present in the base table?
select count(distinct rental_id) from eu_london_bicycles.cycle_hire_update
where rental_id not in (select rental_id from eu_london_bicycles.cycle_hire);
```

Based on above counts:





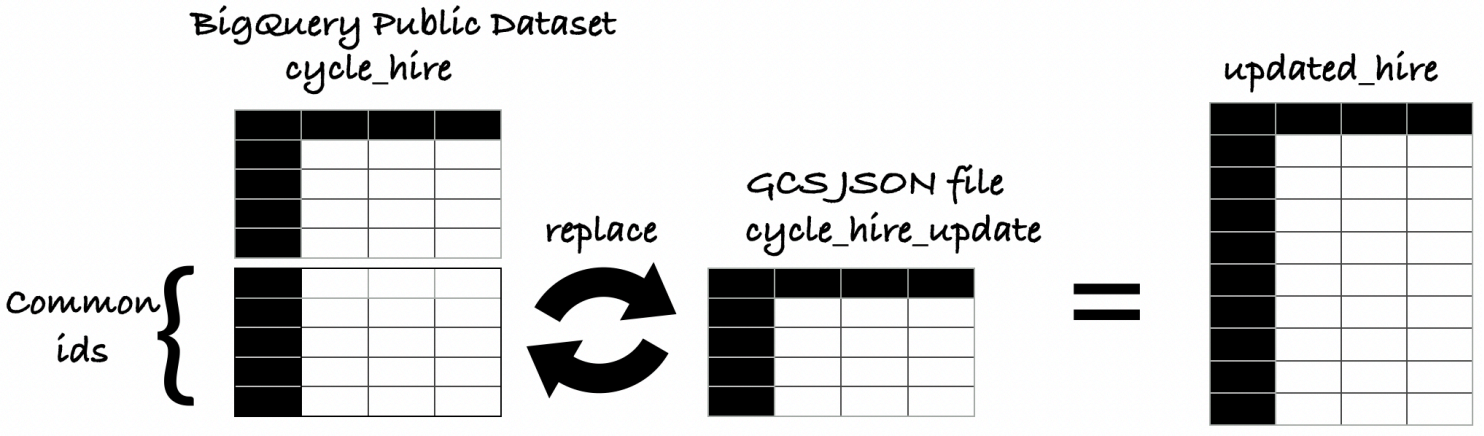
b. Update the cycle_stations table with GCD data i.e. cycle_stations_update table.



As shown in the above diagram, we will replace the records with common ids between cycle_stations and cycle_stations_update with cycle_stations_update records. Once records are replaced, the new table is named as updated_stations.

c. Update the cycle_hire table with GCD data i.e. cycle_hire_update table.

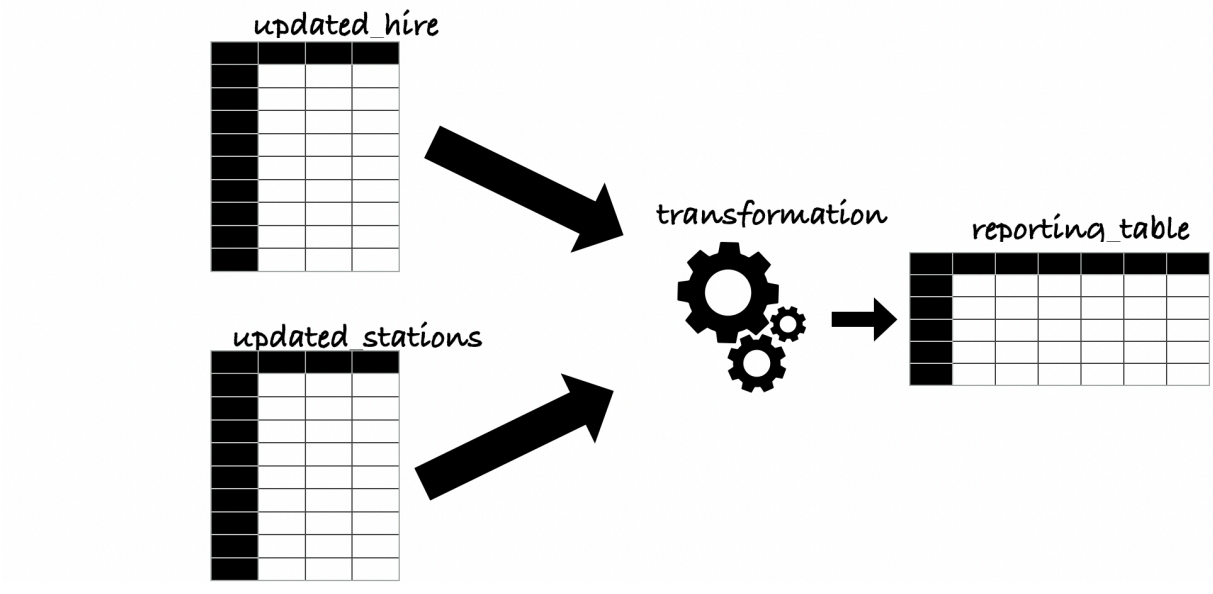
We will follow the exact same method for hire data as well. i.e. replace the records with common ids between cycle_hire and cycle_hire_update with cycle_hire_update records. Once records are replaced, the new table is named as updated_hire. Illustrated in below diagram for better understanding:



Step 3

Merge multiple sources to create a reporting table.

i) We need to create a date dimension for this table, as the granularity of this table should be at the station and date level.
ii) What should be the minimum and maximum dates of the reporting table? > Ideally it should be till current_date(), but just for the simplicity of this project, I am ending at the max end date in hire table (which is '*2017-06-14*'). Otherwise, we would have a lot of unnecessary records starting from 2017 till 2022 today.
iii) All the columns (metrics) that I have created for the reporting table is showing in the reference table (Metadata for reporting table) at the end of this documentation. Please refer that. *[1]*



Step 4

Data Validation to make sure the accuracy and reliability of the output data model.

i) Validate each metrics created in the reporting table against the raw data by taking few examples.
ii) All the Validation examples and steps are documented in the SQL file.

References:

1. Please refer to the Metadata Catalog for transformed table: i.e. our reporting table.

Column Name	Data type	Description
dim_date	string	This column contains every day starting from "2015-01-04" till "2017-06-14". Reason for this range: min_start_date of any bike ride is - "2015-01-04" max_end_date of any bike ride is - "2017-06-14"
station_id	string	This contains every station_id that exist in cycle_stations table in BigQuery.
no_of_rides_started	int	Total Number of rides that have started from the station that corresponds to the specified station_id in column station_id and started on the date specified in column dim_date.
no_of_rides_ended	int	Total Number of rides that have ended at the station that corresponds to the specified station_id in column station_id and ended on the date specified in column dim_date.
no_of_bikes_started	int	Total Number of bikes that have started from the station that corresponds to the specified station_id in column station_id and started on the date specified in column dim_date.
no_of_bikes_ended	int	Total Number of bikes that have ended at the station that corresponds to the specified station_id in column station_id and ended on the date specified in column dim_date.
common_station_rode_to	int	The most common station that has been rode to by the bikers that start from the station that corresponds to the specified station_id in column station_id and the ride has been started on the date specified in column dim_date.
common_station_rode_from	int	The most common station that has been rode from by the bikers that end at the station that corresponds to the specified station_id in column station_id and the ride has been ended on the date specified in column dim_date.

Column Name	Data type	Description
total_time_spent_on_rides	int	Total number of time spent on riding bikes all across the London cycles at all the time.
avg_time_spent_on_rides	float	Average number of time spent on riding bikes all across the London cycles at all the time.
median_time_spent_on_rides	int	Median number of time spent on riding bikes all across the London cycles at all the time.
time_spent_on_rides_by_station	int	Time spent on riding bikes that have started from the station specified in the column station_id.
avg_time_spent_on_rides_by_station	float	Average time spent on riding bikes that have started from the station specified in the column station_id.
median_time_spent_on_rides_by_station	int	Median time spent on riding bikes that have started from the station specified in the column station_id.
time_spent_on_rides_by_station_by_day	int	Time spent on riding bikes that have started from the station specified in the column station_id and have started on the date specified in the column dim_date.
avg_time_spent_on_rides_station_by_day	float	Average time spent on riding bikes that have started from the station specified in the column station_id and have started on the date specified in the column dim_date.
median_time_spent_on_rides_station_by_day	int	Median time spent on riding bikes that have started from the station specified in the column station_id and have started on the date specified in the column dim_date.