Data Engineering 2
Big Data Architectures

Traffic Data - NextBike

Team Members

Mr. Kabir Mehta
Mr. Nipun Aggarwal
Mr. Shikhar Deep

Introduction

In today's fast-paced and environmentally conscious world, the demand for sustainable transportation solutions has witnessed a significant rise. Electric bikes, with their low environmental impact and flexibility, have emerged as a popular alternative for urban commuters. As a result, businesses operating in the electric bike industry need to make well-informed decisions based on accurate and up-to-date data to optimise their operations and cater to the evolving demands of their customers.

This report presents a comprehensive overview of a meticulously designed data pipeline that efficiently fetches data from an API, specifically curated to provide information on electric bike availability across various cities. The primary objective of this data pipeline is to empower decision-makers within the organisation with actionable insights derived from the data, enabling them to make informed business decisions.

Application Problem

The current lack of easily accessible and reliable information about the availability and locations of e-bikes presents a significant challenge for urban commuters. As a result, individuals face difficulties in identifying the most convenient and efficient mode of transportation for their specific needs. To bridge this gap and facilitate seamless mobility, there is a need for a user-friendly application that can accurately pinpoint the nearest e-bikes, empowering users to make informed decisions and optimise their travel experience.

Summary of our Approach

Business Requirement Analysis: We began by thoroughly understanding the business requirements and problem statement.

API Integration: We integrated the nextBike API into our data pipeline. Leveraging this API allowed us to fetch real-time data regarding the availability and locations of e-bikes across various cities.

Client-Server Architecture and Data Storage: To efficiently manage and store the fetched data, we implemented a client-server architecture. The server component, powered by Google (GCP - BigQuery), acted as the central repository for storing the collected data. Our First approach was to use mongoDB as a client server but eventually after having some research using Google GCP - BigQuery was much better decision.

Data Understanding and Attribute Analysis: Once the data was fetched and stored, we performed in-depth analysis to understand its underlying attributes and structure. This involved identifying relevant columns which are necessary for our further analysis this included attributes such as, city, and the number of bikes at each station.

Data Preprocessing and Cleaning: To ensure the reliability and consistency of the data, we conducted comprehensive preprocessing and cleaning procedures like removal of un-necessary columns.

Graphical Insights using Google Data Studio: Leveraging the power of Google Data Studio, we generated insightful graphs to facilitate better decision-making. These graphs provided a visual representation of e-bike availability trends, distribution across cities, and other key metrics.

Chapter 1

Introduction

This report outlines a project focused on building a data pipeline to fetch data from the NextBike API using Python and storing it in Google BigQuery. The data stored in BigQuery will serve as a database, which will then be connected to Google Data Studio for analysis and visualisations. In this introduction, we will provide a description of the application domain, identify the problems at the application level, discuss the benefits of the proposed solution, and address the technical challenges with a suggested solution.

Description of Application Domain

The application domain for this project revolves around NextBike, a prominent bike sharing platform operating in various cities worldwide. NextBike allows users to rent bicycles for short periods, offering a convenient and sustainable transportation option. By fetching and analysing data from the NextBike API, valuable insights can be gained regarding user behaviour, rental patterns, and system performance. This project aims to leverage this data to enhance decision-making, operational efficiency, and overall user experience within the bike sharing domain.
Problem on Application Level:

One of the primary challenges at the application level is effectively utilising the abundance of data generated by NextBike. While NextBike collects a wealth of data pertaining to bike rentals, user profiles, and operational statistics, extracting meaningful information from this data can be challenging. Without a streamlined data pipeline, it becomes difficult to derive timely insights, make informed decisions, and optimise the bike sharing system. A lack of comprehensive analysis may hinder efforts to improve user experiences, tailor marketing strategies, and enhance operational efficiency.

Benefits of a solution on Application Level

Implementing a robust data pipeline solution offers numerous benefits on the application level. By fetching data from the NextBike API and storing it in Google BigQuery, the project enables a centralised and scalable database. This facilitates comprehensive analysis, reporting, and visualisation of the NextBike data through integration with Google Data Studio. With access to real-time and historical data, stakeholders can gain actionable insights into user behaviour, demand patterns, and system performance. Such insights empower data-driven decision-making, leading to optimised bike sharing operations, enhanced user experiences, and targeted marketing efforts.

Problem on Technical Level

On the technical level, a key challenge lies in efficiently fetching data from the NextBike API and storing it in Google BigQuery. The NextBike API provides a continuous stream of data, which needs to be reliably and efficiently processed. Handling data ingestion, transformation, and storage in real-time requires a robust and scalable infrastructure. Additionally, ensuring data integrity, security, and performance throughout the pipeline presents technical complexities.

Technical Solution Idea

To address the technical challenges, a suggested solution involves using Python to fetch data from the NextBike API and Google BigQuery as the centralised data storage. Python's extensive ecosystem of libraries, such as Requests, can be leveraged to interact with the NextBike API, fetching data in real-time. The data can then be transformed, cleaned, and prepared for storage using Python's data processing capabilities.
Google BigQuery, a fully managed and scalable data warehouse, serves as an ideal destination for storing the processed data. Its integration with Google Data Studio allows seamless connectivity for data visualisation and analysis. By connecting Google Data Studio to BigQuery, stakeholders can create interactive dashboards, reports, and visualisations, enabling in-depth exploration of the NextBike data.

Chapter 2

Related Work

Study 1

A Data Science Pipeline In Python On Bike Sharing Dataset - George Pipes

Source - https://predictivehacks.com/an-example-of-a-data-science-pipeline-in-python-on-bike-sharing-dataset/

Methodologies to solve the problem

Data Science Pipeline has following phases:

- *Data Collection/Curation* - This phase requires collecting data which contains the hourly count of rental bikes between 2011 and 2012 in Capital bikeshare system with the corresponding weather and seasonal information.

- *Data Management/Representation* - Here we pre process our data and look for duplicates, change data types of attributes if required.

- *Exploratory Data Analysis* -In this phase we see correlation among the attributes and create charts to have insights for example Rental bike by weekday, hour, rental bikes by season and many more.

- *Hypothesis* Testing and Machine Learning -Once we have the basic insights about our data, we try to optimise the supply demand of retail bike by creating ML models.

  Machine Learning model used in this study
    - Linear Regression
    - Random Forest
    - Gradient Boost

- Communication of insights - We found that the number of Bike Rentals depends on the hour and the temperature. Also, it seems that there is an interaction between variables, like hour and day of week, or month and year etc and for that reason, the tree-based models like Gradient Boost and Random Forest performed much better than the linear regression.

Study 2

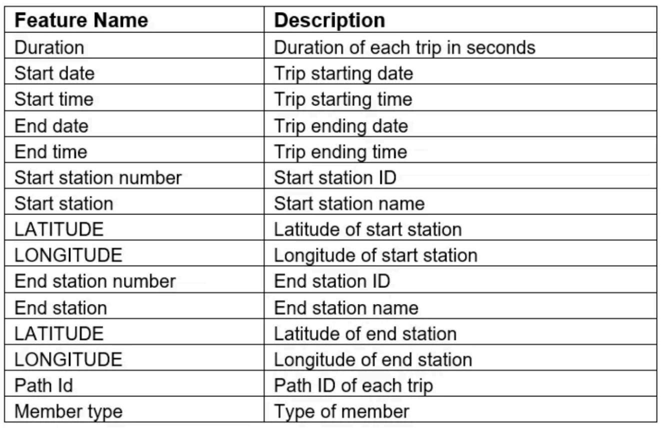
Bike Sharing Demand-Exploratory Data Analysis

Source - https://medium.com/@shawanugya12/bike-sharing-demand-exploratory-data-analysis-2095a1ed4bd8#4064

Problem Statement
How natural and man-made factors are affecting the bike rental demand for Capital Bike share System in Washington DC?

Source of data set used in this study - https://archive.ics.uci.edu/dataset/275/bike+sharing+dataset

Data Attributes



Feature Name	Description
Duration	Duration of each trip in seconds
Start date	Trip starting date
Start time	Trip starting time
End date	Trip ending date
End time	Trip ending time
Start station number	Start station ID
Start station	Start station name
LATITUDE	Latitude of start station
LONGITUDE	Longitude of start station
End station number	End station ID
End station	End station name
LATITUDE	Latitude of end station
LONGITUDE	Longitude of end station
Path Id	Path ID of each trip
Member type	Type of member

Tool used to make visualisation - Tableau Desktop
These are the following use cases that are being solved using above data

1.Variation of count with season
2.Top 10 routes
3.Top busy station and busy hours
4. Density map, where the locations marked blue has the highest concentration of bike rentals.

The research does not use any API to fetch the data. But rather shows the EDA(exploratory Data analysis) techniques that presents us the insights about the bike Sharing industry.

Chapter 3

Data Set

The dataset used for this report is obtained by fetching data through the Nextbike API, a reliable and comprehensive source for information on electric bike availability across various cities. The dataset provides valuable insights into the operational aspects of the Nextbike service, enabling better decision-making and optimising urban mobility.

About NextBike

Nextbike is the European bike sharing market leader and stands for a multimodal and socially just mobility revolution. nextbike by TIER has established bike sharing at 300+ locations in more than 20 countries as an elementary component of liveable cities and regions. Integrated into public transport, nextbike by TIER offers millions of users the right means of transport for their daily needs with classic bicycles, e-bikes and cargo bikes. nextbike by TIER is a brand of TIER mobility SE, the world's leading provider of micro-mobility.

Key Features of the Dataset:
Source - NextBike https://www.nextbike.at/de/niederoesterreich/

Some key information about data

1. The dataset includes real-time information on the availability of electric bikes at different stations across cities. It provides details such as station ID, location coordinates, number of available bikes, and the overall capacity of each station.

2. Geographical Information: The dataset encompasses geographic information, including city names, country codes, and specific location details for each bike station. This information enables spatial analysis and visualisation of bike

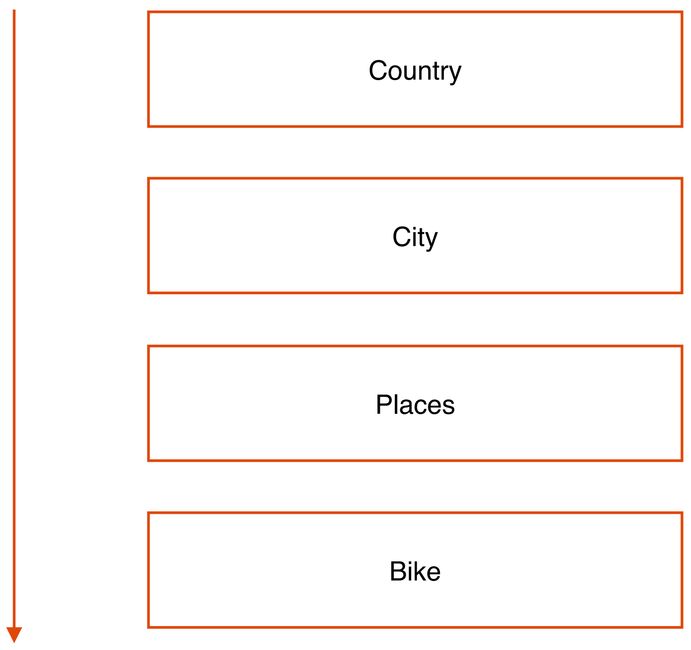
distribution across cities, facilitating insights into demand patterns and hotspot areas.

3. Metadata: Supplementary metadata such as station names, addresses, and station types is also be included in the dataset. This additional information provides contextual details about the stations and contributes to a more comprehensive understanding of the electric bike network.

4. Dynamic Updates: The dataset is designed to be dynamic and regularly updated to reflect the real-time availability of electric bikes. As the Nextbike API continuously receives new data, the dataset captures the latest information, ensuring the timeliness and accuracy of the analysis and insights derived from it.

5. JSON Format: The dataset is structured in JSON (JavaScript Object Notation) format, which allows for easy parsing and extraction of relevant data elements. This format ensures compatibility with various programming languages and facilitates seamless integration with data processing and visualisation tools.

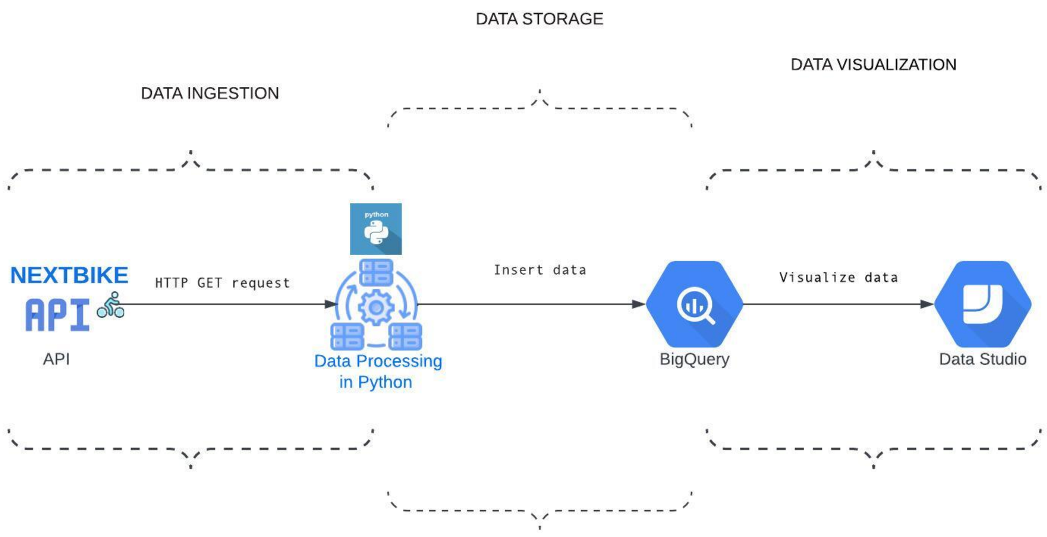
Hierarchy in Dataset
we studied the dataset and this is the hierarchy that we can notice



Chapter 4

Solution

Data Collection/Curation



NextBike provides an API that allows us to access real-time data about their bike-sharing service. By making API calls, we can retrieve information such as bike locations, availability, user data, and more. This data is crucial for our project as it enables us to analyse the bike-sharing patterns, optimise operations, and make data-driven decisions.

To fetch NextBike data, we need to follow a few steps. First, we need to obtain the necessary credentials for accessing the API, in our case we did not require any API key or OAuth tokens. Since is an open API for data usage.

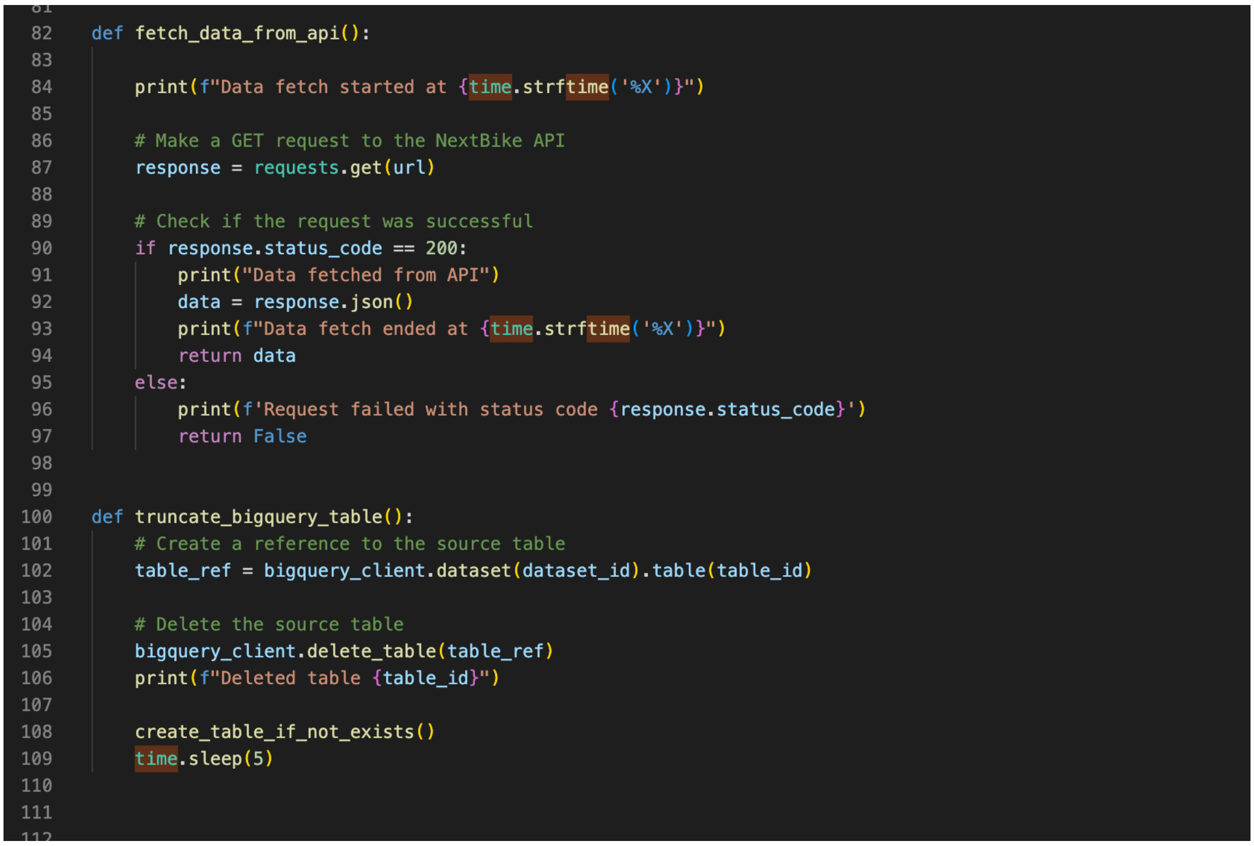
Next, we can utilise Python's requests library to send HTTP requests to the NextBike API endpoints. We need to identify the specific endpoints that provide the desired data, such as bike locations or availability.

Required Steps:

1. Set up the environment: Install necessary libraries such as requests, for data retrieval, processing, and google.cloud.bigquery for loading into BigQuery.

2. The next step is to create a key in Google Cloud Platform which helps us to connect python on our system to the google Big query. This can be done by creating a service account key file in the Google Cloud Console.

3. Fetch data from the NextBike API: We used the requests library in Python to send requests to the NextBike API endpoints and retrieve the desired data. This includes information such as bike locations, availability, user data, or other relevant data points.



```python
def fetch_data_from_api():

    print(f"Data fetch started at {time.strftime('%X')}")

    # Make a GET request to the NextBike API
    response = requests.get(url)

    # Check if the request was successful
    if response.status_code == 200:
        print("Data fetched from API")
        data = response.json()
        print(f"Data fetch ended at {time.strftime('%X')}")
        return data
    else:
        print(f'Request failed with status code {response.status_code}')
        return False


def truncate_bigquery_table():
    # Create a reference to the source table
    table_ref = bigquery_client.dataset(dataset_id).table(table_id)

    # Delete the source table
    bigquery_client.delete_table(table_ref)
    print(f"Deleted table {table_id}")

    create_table_if_not_exists()
    time.sleep(5)
```

Code to fetch data, and update the data

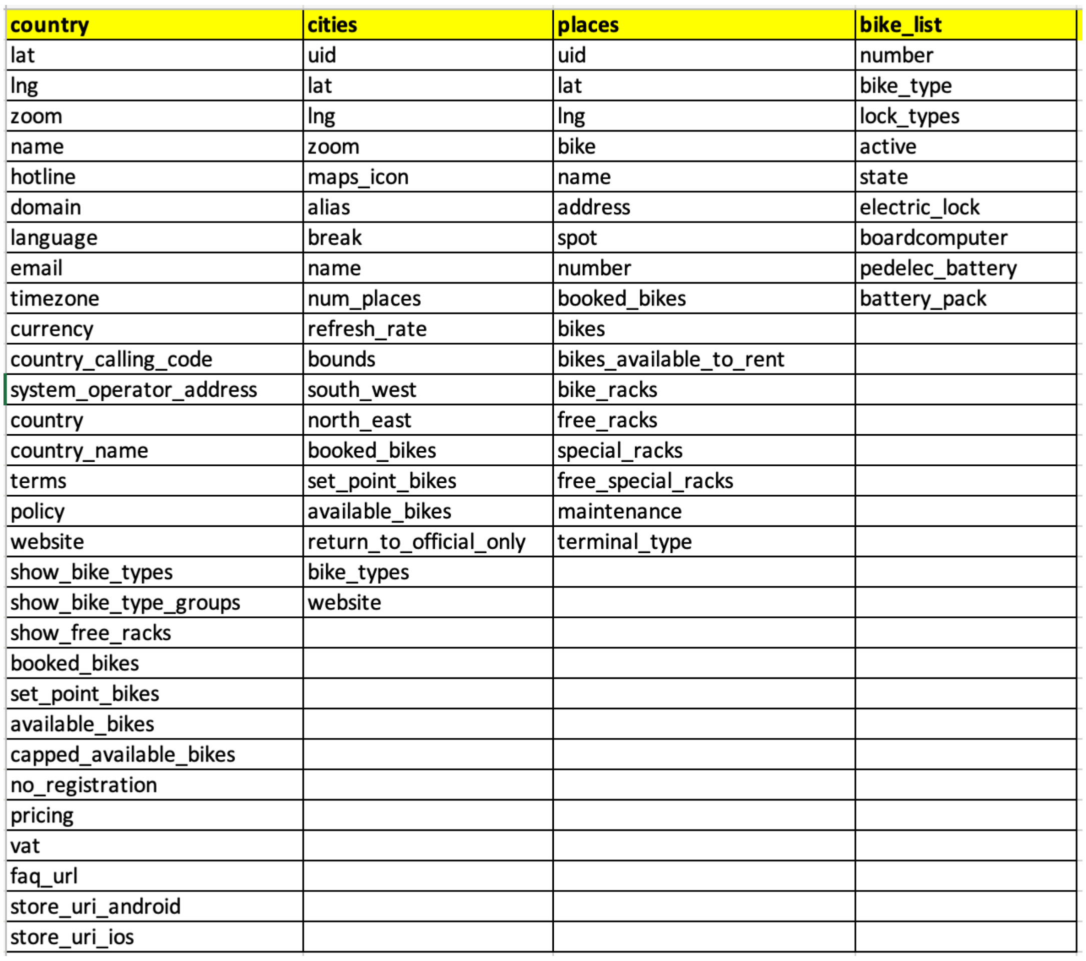
Data Preprocessing

Data preprocessing and cleaning are crucial steps before conducting statistical inference, as they ensure the reliability and accuracy of the results. In the case of the NextBike project, preprocessing data involves removing irrelevant attributes, such as irrelevant demographic information which can reduce noise and improve the efficiency of analysis.

By carefully preprocessing and cleaning the data, we can ensure that the subsequent statistical inferences are based on high-quality and reliable data, leading to more accurate and meaningful insights.

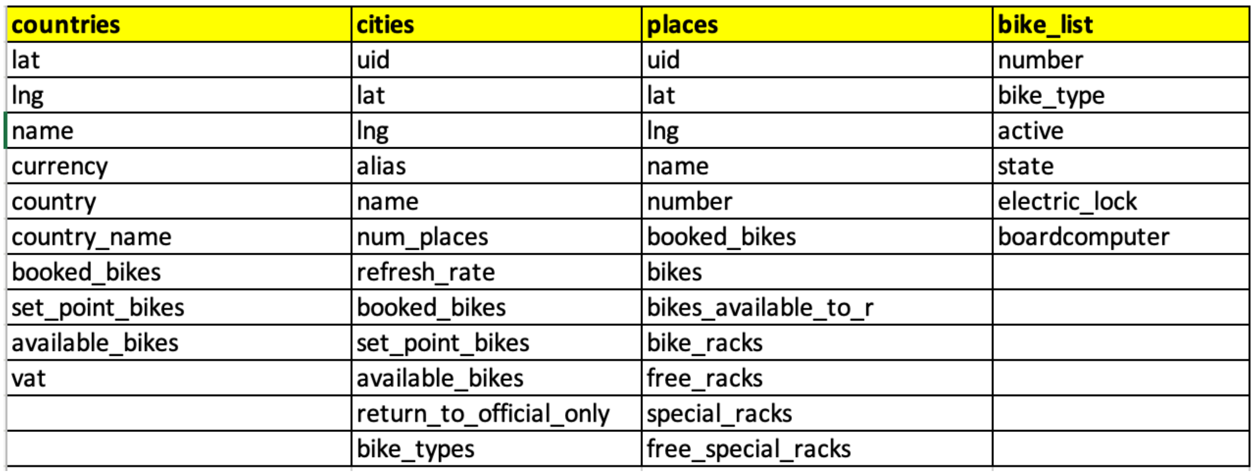
Then we process the fetched data as per our project requirements. Then we perform necessary data transformations, cleaning, or formatting to ensure the data is in a suitable format for further analysis.

Original Data - Schema

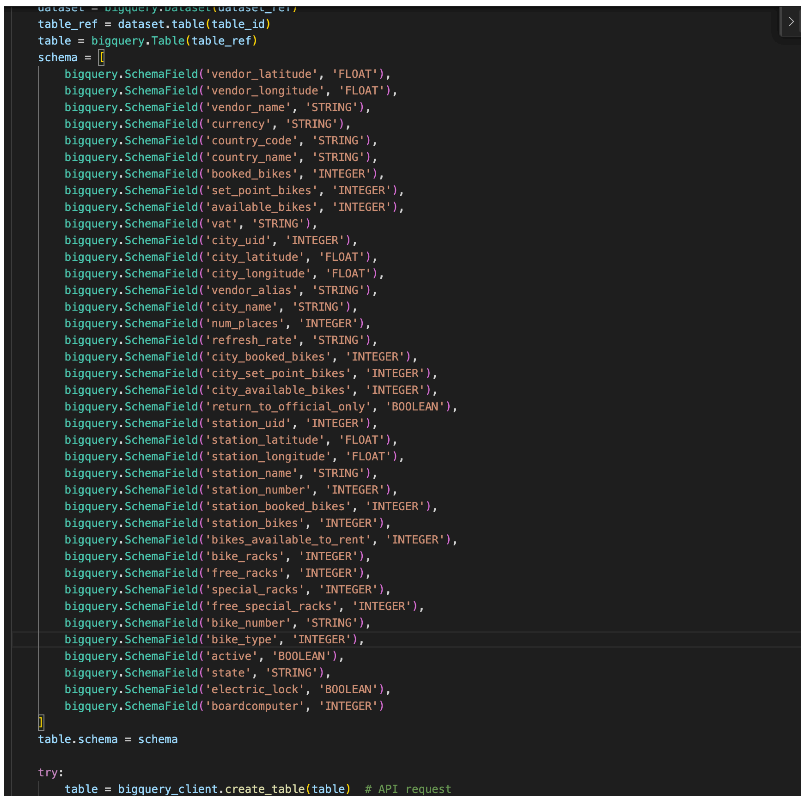


country	cities	places	bike_list
lat	uid	uid	number
lng	lat	lat	bike_type
zoom	lng	lng	lock_types
name	zoom	bike	active
hotline	maps_icon	name	state
domain	alias	address	electric_lock
language	break	spot	boardcomputer
email	name	number	pedelec_battery
timezone	num_places	booked_bikes	battery_pack
currency	refresh_rate	bikes	
country_calling_code	bounds	bikes_available_to_rent	
system_operator_address	south_west	bike_racks	
country	north_east	free_racks	
country_name	booked_bikes	special_racks	
terms	set_point_bikes	free_special_racks	
policy	available_bikes	maintenance	
website	return_to_official_only	terminal_type	
show_bike_types	bike_types		
show_bike_type_groups	website		
show_free_racks			
booked_bikes			
set_point_bikes			
available_bikes			
capped_available_bikes			
no_registration			
pricing			
vat			
faq_url			
store_uri_android			
store_uri_ios			

Data after removing Non - important attributes



countries	cities	places	bike_list
lat	uid	uid	number
lng	lat	lat	bike_type
name	lng	lng	active
currency	alias	name	state
country	name	number	electric_lock
country_name	num_places	booked_bikes	boardcomputer
booked_bikes	refresh_rate	bikes	
set_point_bikes	booked_bikes	bikes_available_to_r	
available_bikes	set_point_bikes	bike_racks	
vat	available_bikes	free_racks	
	return_to_official_only	special_racks	
	bike_types	free_special_racks	



```
dataset = bigquery.Dataset(dataset_ref)
table_ref = dataset.table(table_id)
table = bigquery.Table(table_ref)
schema = [
    bigquery.SchemaField('vendor_latitude', 'FLOAT'),
    bigquery.SchemaField('vendor_longitude', 'FLOAT'),
    bigquery.SchemaField('vendor_name', 'STRING'),
    bigquery.SchemaField('currency', 'STRING'),
    bigquery.SchemaField('country_code', 'STRING'),
    bigquery.SchemaField('country_name', 'STRING'),
    bigquery.SchemaField('booked_bikes', 'INTEGER'),
    bigquery.SchemaField('set_point_bikes', 'INTEGER'),
    bigquery.SchemaField('available_bikes', 'INTEGER'),
    bigquery.SchemaField('vat', 'STRING'),
    bigquery.SchemaField('city_uid', 'INTEGER'),
    bigquery.SchemaField('city_latitude', 'FLOAT'),
    bigquery.SchemaField('city_longitude', 'FLOAT'),
    bigquery.SchemaField('vendor_alias', 'STRING'),
    bigquery.SchemaField('city_name', 'STRING'),
    bigquery.SchemaField('num_places', 'INTEGER'),
    bigquery.SchemaField('refresh_rate', 'STRING'),
    bigquery.SchemaField('city_booked_bikes', 'INTEGER'),
    bigquery.SchemaField('city_set_point_bikes', 'INTEGER'),
    bigquery.SchemaField('city_available_bikes', 'INTEGER'),
    bigquery.SchemaField('return_to_official_only', 'BOOLEAN'),
    bigquery.SchemaField('station_uid', 'INTEGER'),
    bigquery.SchemaField('station_latitude', 'FLOAT'),
    bigquery.SchemaField('station_longitude', 'FLOAT'),
    bigquery.SchemaField('station_name', 'STRING'),
    bigquery.SchemaField('station_number', 'INTEGER'),
    bigquery.SchemaField('station_booked_bikes', 'INTEGER'),
    bigquery.SchemaField('station_bikes', 'INTEGER'),
    bigquery.SchemaField('bikes_available_to_rent', 'INTEGER'),
    bigquery.SchemaField('bike_racks', 'INTEGER'),
    bigquery.SchemaField('free_racks', 'INTEGER'),
    bigquery.SchemaField('special_racks', 'INTEGER'),
    bigquery.SchemaField('free_special_racks', 'INTEGER'),
    bigquery.SchemaField('bike_number', 'STRING'),
    bigquery.SchemaField('bike_type', 'INTEGER'),
    bigquery.SchemaField('active', 'BOOLEAN'),
    bigquery.SchemaField('state', 'STRING'),
    bigquery.SchemaField('electric_lock', 'BOOLEAN'),
    bigquery.SchemaField('boardcomputer', 'INTEGER')
]
table.schema = schema

try:
    table = bigquery_client.create_table(table)  # API request
```

Code to show the schema made in bigquery

Using Big Query and its benefits

Here are the advantages of using Google BigQuery as our database from our perspective:

Scalability: With BigQuery, we can handle massive amounts of NextBike data without worrying about infrastructure limitations. It automatically scales resources based on our needs, allowing us to store and analyse large volumes of data efficiently.

Speed: BigQuery's distributed architecture and columnar storage format enable us to retrieve and analyse data rapidly. It executes complex queries involving aggregations, joins, and filtering quickly, saving us valuable time and providing fast insights.

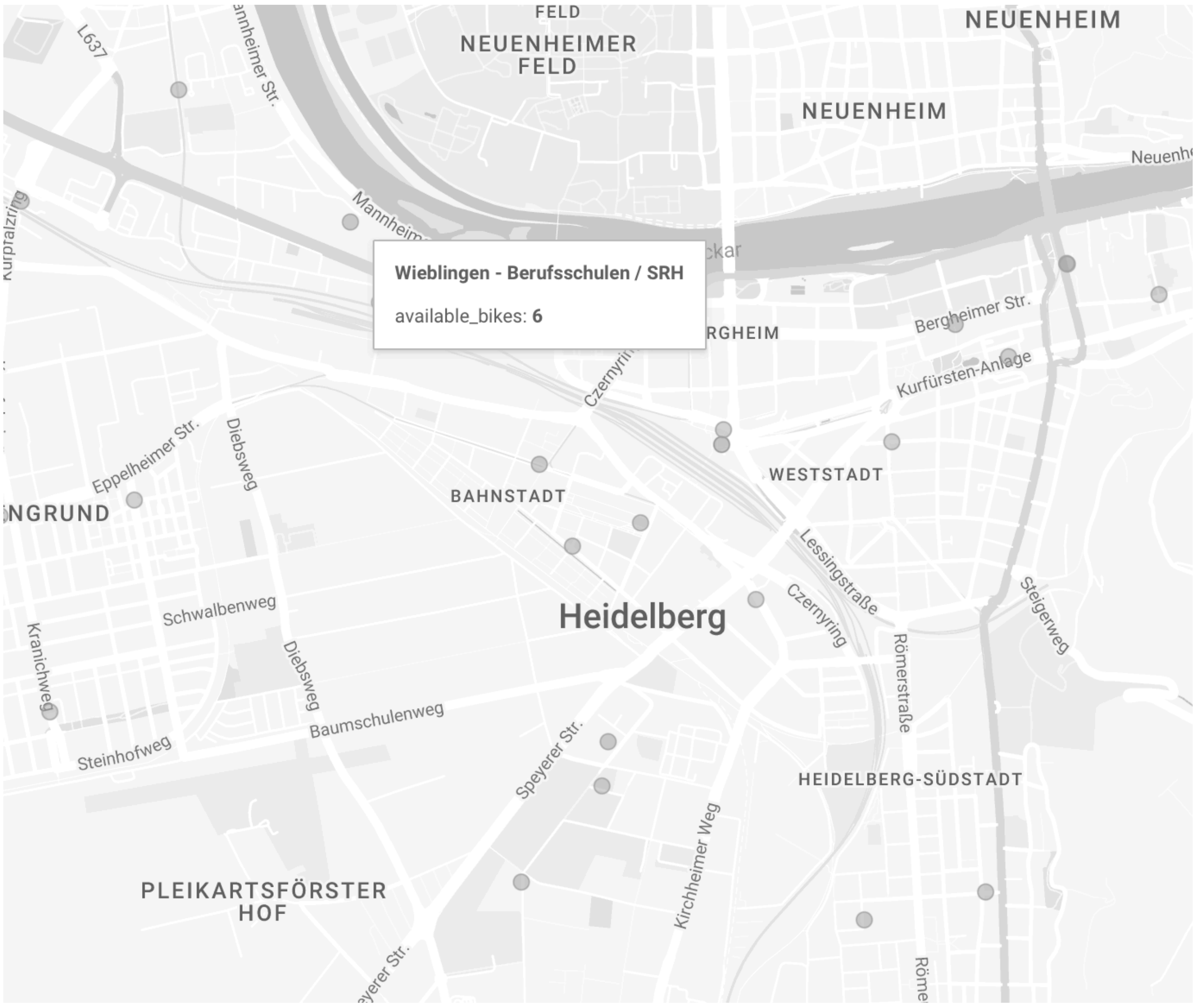
Real-time Analysis: With BigQuery's streaming ingestion, we can continuously load data from the NextBike API in real-time.

Integration with Google Data Studio: By connecting BigQuery with Google Data Studio, we can create interactive and visually appealing dashboards, charts, and reports. This integration enables us to explore and communicate insights effectively, facilitating better understanding and decision-making for stakeholders.

Chapter 5

Insights and inferences

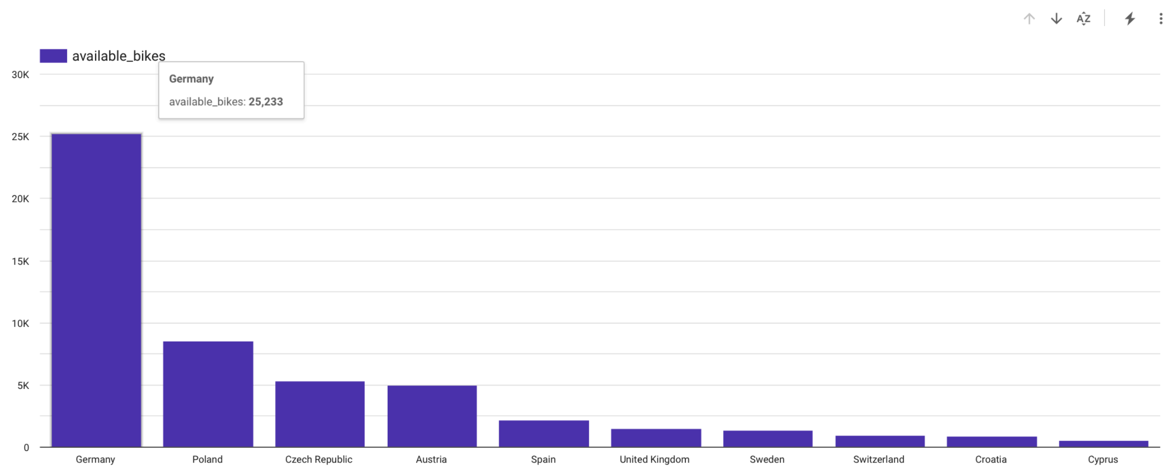
- The given Geo visualisation on data studio gives us the number of free bikes on the location.



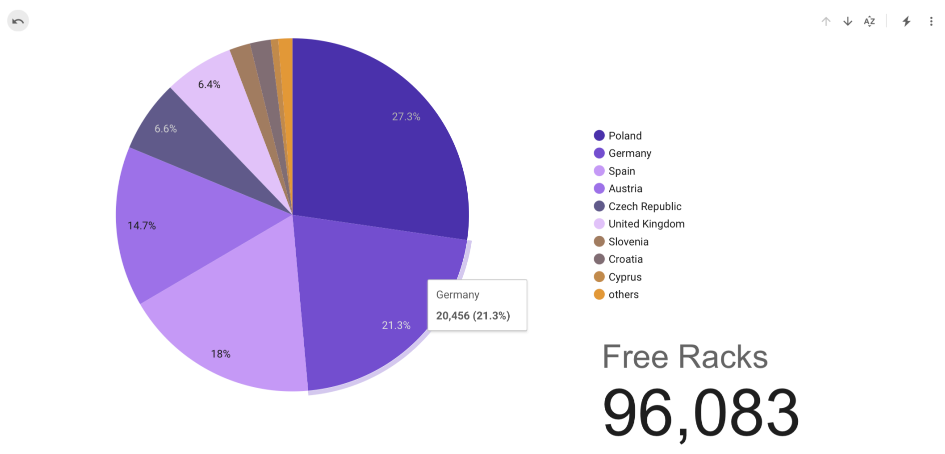
- The given visualisation on data studio gives us the number of available bikes on the location. In this visualisation we have used hierarchy

  Country > City > Station

NextBike



- The given visualisation on data studio gives us the number of free racks and the percentage of racks amongst the countries.



- **Overall in this project we have been successfully able to create a robust pipeline and make all the analytical inferences**

Tools used for all the work done !!

Topic	Tools	Use cases
Documentation	**Microsoft Teams**	**A Microsoft Teams is used to share between users and allow real time editing**
	Microsoft Word	**Used to write Final report**
Diagrams	**lucid.app**	**Used for the data flow diagram**
Development	**Vscode,Python,GoogleBigquery and GoogleData studio**	**Used for Data featching from API to storing Into database and Visualization**

Bibliography

https://predictivehacks.com/an-example-of-a-data-science-pipeline-in-python-on-bike-sharing-dataset/

https://medium.com/@shawanugya12/bike-sharing-demand-exploratory-data-analysis-2095a1ed4bd8#4064

https://archive.ics.uci.edu/dataset/275/bike+sharing+dataset

https://www.nextbike.at/de/niederoesterreich/