# RailwayOps

Vishal Tripathi, Shubhang Kalkar, Harshvardhan Singh Gahlaut,
Mukul Mahajan, Saba Tehrani

***Abstract*—The Indian railway system, a lifeline for millions, faces challenges related to operational efficiency and user-friendliness in its management. This project introduces an innovative solution by integrating SQL and Python to improve the user experience within the Indian Railways.**

## I. Introduction

THe need for descriptive and quantitative research on the growth of railroads in Europe led to the creation of the Geographic Information System (GIS). The intention was to draw attention to how crucial the spatial element is to this infrastructure study. In contemporary situations, Geographic Information Systems (GIS) are becoming acknowledged as valuable instruments for assessing socio-economic data and creating descriptive maps.

Our research is a thorough investigation of one of the biggest and most complex railway networks in the world, with a focus on the expansive and dynamic area of Indian Railways. Our project, which involves an intricate network of data and operational details, aims to unravel the complexities present in the Indian Railways system. We explore the core of this vast railway network using data taken straight from the official Indian Railway website in an effort to gain valuable insights that advance our knowledge of its functioning, effectiveness, and possible areas for improvement. Our focus is not only on the sheer volume of data, but also on its transformative potential, which helps us make wise decisions and improve the effectiveness of this vital link in India's transportation system. [1]

It aims to decipher the complexity present in the Indian Railways system, encompassing a wealth of data and operational subtleties. Utilizing a dataset obtained straight from the official IR website, we explore the core of this vast railway network with the goal of obtaining significant insights that advance our knowledge of its functioning, effectiveness, and possible areas for improvement. Our focus is on the transformative potential of the data, which not only allows us to make educated decisions but also improves the efficiency of this vital transportation infrastructure in India.

## II. Flow of our project

We start with an unnormalized CSV file of data. To effectively store this raw data, we make use of Amazon S3's capabilities as our data lake. A scalable and affordable option for managing massive amounts of raw data is offered by Amazon S3. This first step makes sure that our data is safely kept and readily available for further processing.

We move the denormalized data from Amazon S3 to our data warehouse, Snowflake, in order to get valuable insights. The strong design of Snowflake enables us to handle and organize the data effectively. In this stage, three normalized tables are created by transforming the unnormalized data using a transformation procedure. In addition to ensuring data integrity, this standardization lays the groundwork for more productive research.

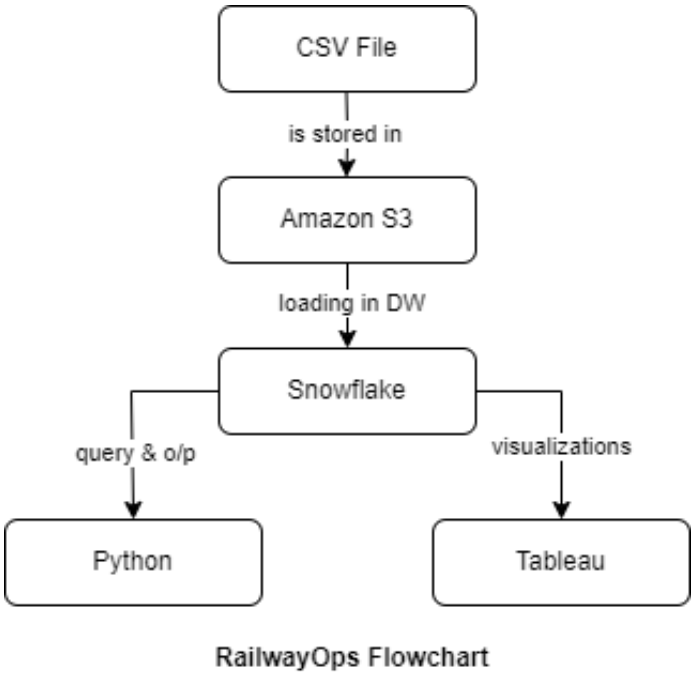


Fig. 1. RailwayOps Flowchart

Python is used to communicate with Snowflake and analyze data on the three normalized tables. It is a strong programming language. We can easily establish a connection to Snowflake using Python, run queries, and get the precise data we require for our investigation. Python is a great tool for doing in-depth data processing and extracting insights because of its large ecosystem of libraries and versatility.

The final step of our data pipeline involves connecting Tableau to Snowflake to visualize the processed data. Tableau serves as the visualization layer, allowing us to create dynamic and interactive visual representations of our findings. By connecting Tableau with Snowflake, we ensure that our visualizations are based on the most up-to-date and relevant data. This enhances the interpretability of our results and facilitates effective communication of insights.

## III. Datasets & Sources

The dataset for our project was obtained straight from the Indian Railways' official website, which forms the basis of our data analysis. With 186,115 rows and 12 columns, this dataset has a significant amount of data that gives an overview of the information. We added seven more columns to the dataset in order to increase its dimensions and make it more relevant to our investigation. These additional columns contain the source and destination locations' latitude and longitude coordinates in addition to the matching state in which these geographic coordinates are located. We were able to extract and include location data into our dataset using Google extension GeoCode, which is connected with Google Sheets. We mapped the station code to the corresponding state that it is located in with the help of Wikipedia. Thus, we converted the original structured dataset into an unstructured, unnormalized CSV file, setting the foundation for the next steps in our pipeline of data processing.

**1. Dataset from the Official Indian Railways Website**

https://data.gov.in/resource/indian-railways-time-table-trains-available-reservation-01112017

**2. Used to map stations with their States**

https://en.wikipedia.org/wiki/List_of_railway_stations_in_India

Each column in our carefully curated dataset is essential in comprehending the complexities of the enormous railway system. Important details including names of the source and destination stations, train numbers, names of the trains, intermediate stops, arrival and departure times, and distances are all included in the dataset. These components serve as the foundation for deriving insightful information on the dynamics and motions of trains throughout the network. But when these data are transformed into eye-catching visuals that provide a comprehensive picture of the railway ecosystem, their entire potential becomes apparent.

With the help of our dataset, we are able to identify intriguing trends like the geographical distribution of specific trains. We may learn a great deal about the scope of the railroad network and the variety of areas it connects by following the states that each train travels through. The dataset also makes it easier to identify busy train stops, which helps determine which states have the busiest stations. We understood the significance of latitude and longitude coordinates in order to visualize this geographical environment. We incorporated Google's GeoCode Extension to accomplish this, getting exact position information for each station. In addition to offering precise spatial data, these coordinates are the foundation for mapping visualizations that eloquently show the geographic distribution of train lines and station operations. [2]

With strict database architecture, we were able to properly transform the dataset into three normalized tables that complied with the Third Normal Form (3NF). This category contains the following three tables: the "train station", "train info", and the "stations info" In addition to improving data integrity, this normalization technique set the stage for effective and optimal database management. In our project, we leveraged the power of multiple technologies to enable thorough analysis of railway data. Strong database administration, querying, and analysis are possible with Snowflake. By organizing the data in 3NF, we ensure that our database is effective and suitable for complex searches. Furthermore, we are creating interactive dashboards and plots with Tableau, a data visualization tool, so that users may obtain insightful information.

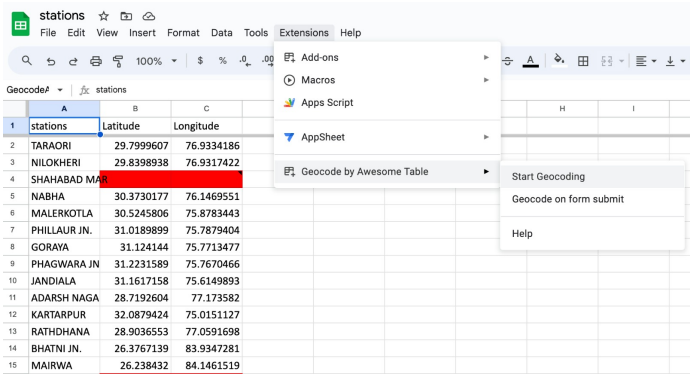


Fig. 2. GeoCode For Latitude & Longitude

Additionally, we made use of Python's abilities to establish a seamless connection between our analytical tools and our Snowflake database. We were able to do pertinent operations on the data thanks to this interface, which gave us the ability to execute advanced analytics and decision-making within the framework of Indian railway operations. With these methods and technologies in place, we have been able to deliver data-driven modifications to railway operations and extract important insights from the Indian railway dataset.

## IV. Data Modeling

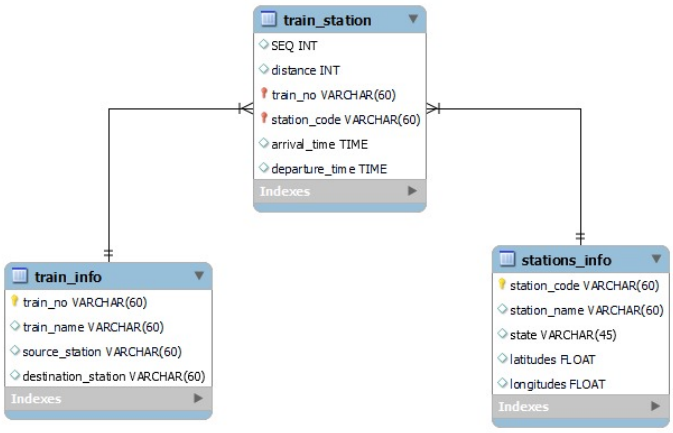


Fig. 3. ER Model

**Normalization:** We minimized redundancy and guaranteed data integrity constraints by normalizing the tables to at least the third normal form (3NF). By using a systematic strategy, redundant data was removed and overall database performance was improved.

**Indexes:** Creating indexes on foreign key columns—like the Train No. and Station Code of TrainStation—improved database access patterns and sped up data retrieval, which greatly improved query performance.

**Primary & Foreign Keys:** Referential integrity was maintained by making sure that primary and foreign key relationships were clearly specified and upheld. This prevented database discrepancies by ensuring that relationships between tables were appropriately recorded. [3]

**Unique Constraints:** An extra degree of data integrity was introduced by taking unique constraints into account, as demonstrated, by making sure each Train No in the TrainInfo table is unique. This strengthened the uniqueness of key identifiers and protected against unintentional duplication.

**Default Constraints:** By carefully choosing default values and constraints, data was brought into compliance with business rules, which decreased the possibility of inconsistent or incorrect entries. This proactive strategy assisted in preserving the consistency and quality of the data across the database.

**Data Types:** The efficiency of storage and query execution were maximized by choosing the right data types for every column. This careful thought process guaranteed that the database handled data in a way consistent with its purpose and nature, and that it makes efficient use of its resources. [3]

## V. AWS S3 - Data Lake

In the AWS (Amazon Web Services) ecosystem, Amazon S3 (Simple Storage Service) is a flexible and extensively used option for creating data lakes. S3 is a data lake storage solution that enables us to store enormous volumes of unstructured and raw data at scale. This covers a wide range of data kinds, including text, pictures, videos, and more. Because of its design, which guarantees scalability, durability, and accessibility, S3 is a fundamental part of building data lakes that meet the changing demands of analytics and data storage.

### *A.* Creating S3 Bucket

The first step in creating an S3 bucket is to access the S3 service via the AWS Management Console. You can then start the process of making a new bucket from there. Selecting the AWS region in which the data will be kept, giving the bucket a globally unique name, and setting up extra parameters like versioning, logging, and access control are important steps in this process. Your data is centrally stored in the bucket that is formed once these parameters are set. [4]

### *B.* Folder Upload

It is simple to create folders to organize data inside the S3 bucket once it has been created. By selecting the ”Create folder” option, you can add a folder to the bucket and navigate to the desired place. Folders help organize and classify data

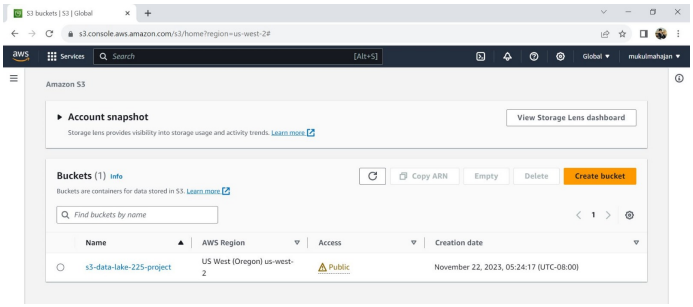


Fig. 4. New S3 Bucket in US West Oregon Region

inside the bucket, which makes data administration more effective. [4]

Once the folder were created, we used the ”Upload” feature to pick the folder and upload the CSV file of our project. This stage laid the foundation for further data processing and analysis operations by ensuring that the project data is arranged logically and easily accessible within the S3 data lake. To summarize, the utilization of Amazon S3 as a data lake offered a sturdy framework for the storage and administration of various datasets, and the establishment of buckets and folders expedited the arrangement and retrieval of project-specific data from the S3 environment.

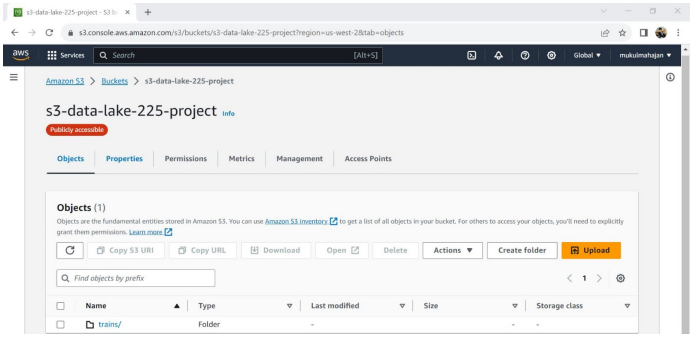


Fig. 5. New Folder named Trains created

### *C.* About S3 Bucket Policy

A bucket policy in AWS S3 is a collection of guidelines established by the bucket owner to manage rights for accessing the objects (files) kept in an S3 bucket. This policy is attached to the S3 bucket and written in JSON (JavaScript Object Notation). A key component of controlling security and access for objects stored in S3 is bucket policies, which let you define who may access your data and what actions they can take.

## VI. ETL using SnowFlake

Executing Extract, Transform, Load (ETL) procedures was part of the procedure. Data was transferred into the data warehouse through the ETL procedure after being extracted from outside sources. After the raw data were transformed, a STAR Schema with one fact table and two dimension

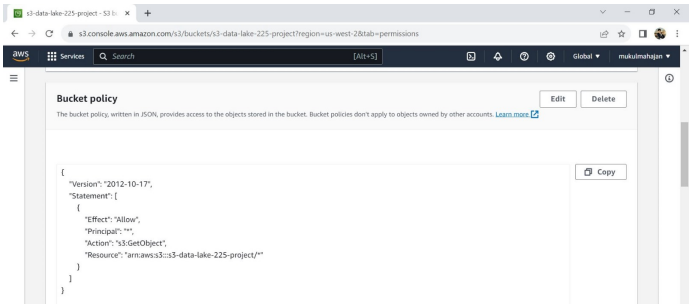


Fig. 6. S3 Bucket Policy

tables was produced. The converted data was then put into a consumption zone schema, giving analysts an organized setting in which to run queries and produce useful insights. Within the particular project, Snowflake, a data warehousing solution, was essential. Three schemas were developed for the ”DB225PROJECT” database, which made the converted data easier to retrieve and arrange.

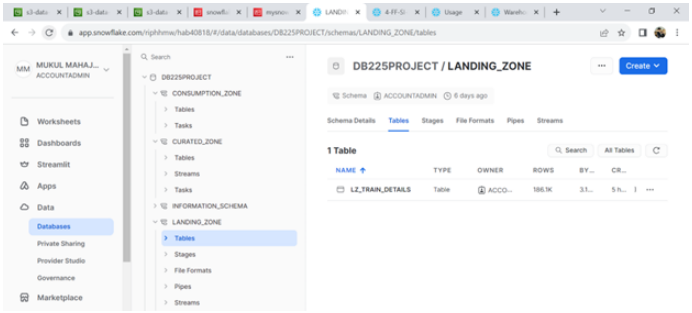


Fig. 7. Dashboard - Landing Zone

Using Snowflake as the data warehousing platform gave the project more power because it offered a flexible and scalable environment for data management and querying. By ensuring that unstructured data was transformed into a structured STAR Schema, the ETL procedure enhanced the effectiveness of ensuing analytical queries. Information retrieval and storage were made even easier with the establishment of the ”DB225PROJECT” database, which has separate schemas. This method not only followed industry standards for data warehousing, but it also set up analysts to gain valuable insights from the carefully selected data in the consumption zone schema. The project’s goals were mostly met because to Snowflake’s ability to manage multi-cluster processing and effective querying, which offered a reliable infrastructure for data transformation, storage, and analysis. [5]

**1. LANDING_ZONE Schema:** Information falls on a transient table within this schema from the external stage AWS S3 data lake.

**2. CURATED_ZONE Schema:** This schema optimizes memory and query execution time by moving the table from the landing zone schema to a new transient table inside the curated zone schema. It also curates data types of the table’s attributes.

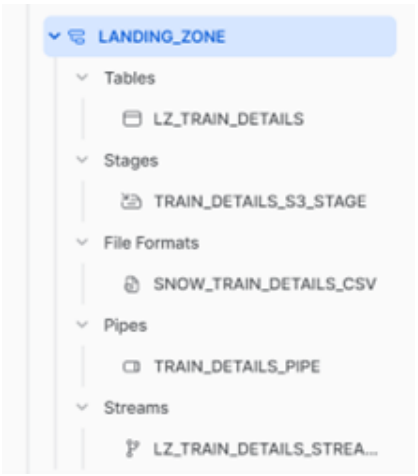


Fig. 8. Landing Zone

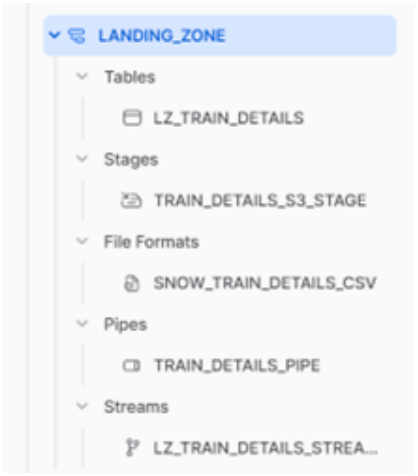


Fig. 9. Curated Zone

**3. CONSUMPTION ZONE:** The table from the curated zone schema is divided into three distinct permanent tables inside this schema: There are two dimension tables (train_info, stations_info) and one fact table (trainstation).

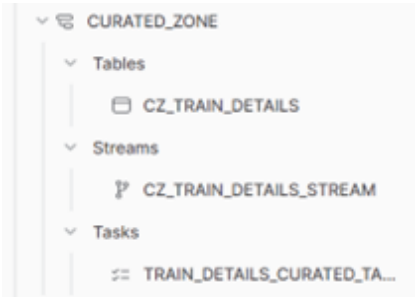


Fig. 10. Consumption Zone

Snowflake tracks the whole data flow that results in excellent data governance by monitoring all the data sources originating from both internal and external stages.

## VII. SnowFlake - Data Warehouse

Snowflake is a cutting-edge, cloud-based data warehousing software that transforms how businesses handle and examine their data. Snowflake, positioned as a Data Cloud platform, offers a fully-managed, scalable solution for real-time data processing, storing, and querying of massive volumes. The architecture of Snowflake is one of its primary features; it divides computation and storage resources so that users can expand each one separately. Because businesses can dynamically modify their resources based on their specific demands, whether handling large data loads during peak hours or reducing resources during periods of reduced demand, this unique design provides optimal performance and cost efficiency. In addition, Snowflake's design facilitates deployments across several clouds, regions, and clusters, offering redundancy and flexibility to satisfy even the most stringent organizational needs.

We set up the "Compute WH" data warehouse in a small size, optimizing its design to improve scalability and performance. A scaling policy that mandates the use of a minimum of one cluster and permits the system to dynamically scale up to a maximum of two clusters depending on workload demands is included in the implementation. This strategy makes sure the data warehouse stays effective and adaptable to changing user needs and query complexity. We optimize query execution times by using a standard scaling approach, taking into account that various queries could need varying amounts of processing power. [5]

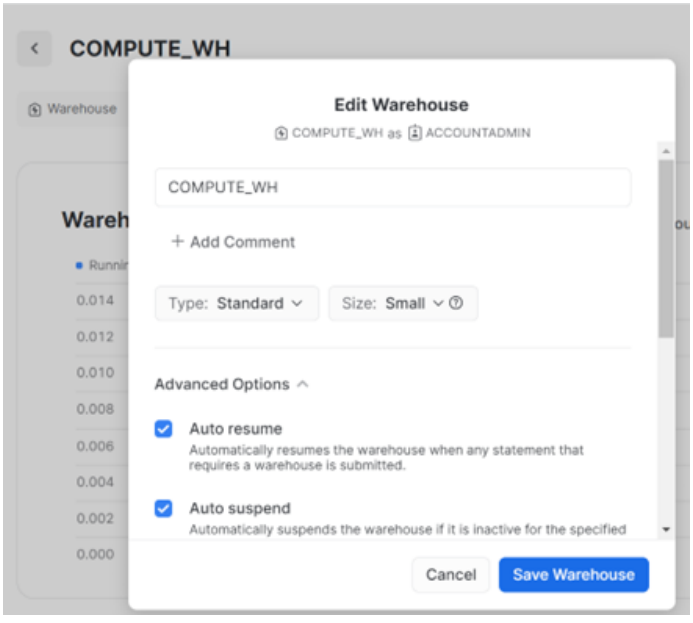


Fig. 11. New Data Warehouse Created

This data warehouse system is remarkable for its automatic load balancing capabilities. The system automatically starts load balancing and uses another cluster to divide the computational burden when a query execution requires more processing power than the present cluster can provide. This load balancing and dynamic scaling approach greatly improves the overall performance of query executions. It guarantees that the data warehouse can effectively manage varying workloads, offering users and applications a timely and reliable analytical environment.

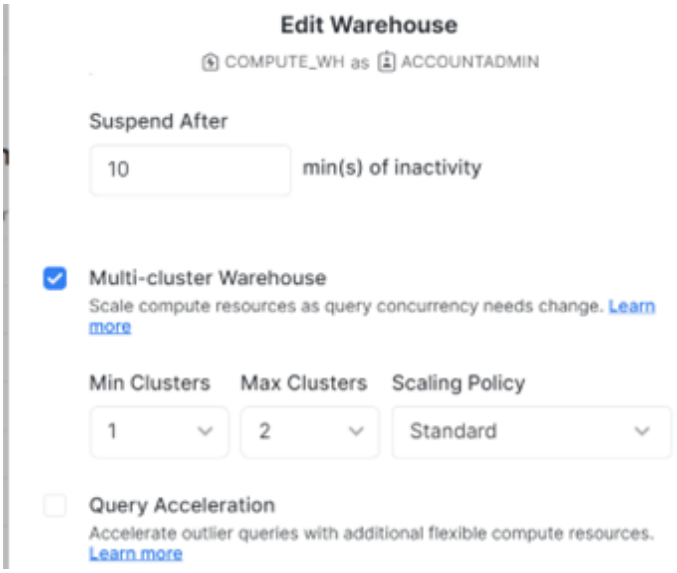


Fig. 12. Clusters in Warehouse

Performance is improved by the multi-cluster data warehouse architecture, which also fits in with an economical and adaptable usage model. This solution provides a virtual data warehouse that can be tailored to meet unique business needs by enabling on-premise implementation. In addition, the pay-as-you-go pricing model makes sure that businesses only pay for the computer resources they actually use, matching costs to actual use. SnowFlake's clever mix of cost-effectiveness, load balancing, and scalability makes it an agile and dynamic option for companies looking to get the best query performance possible from their analytical projects.

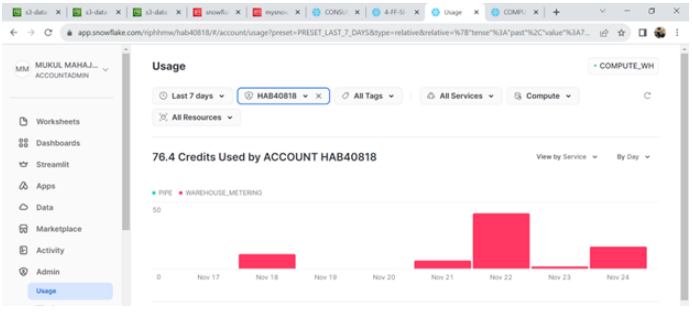


Fig. 13. Payment according to usage

## VIII. Technical Difficulties Faced

It was important for us to include state information for each station because the original dataset lacked the data needed for analytical queries and visualizations.For this, we manually gathered Station-state mappings using a Wikipedia page. VLOOKUP in Excel was then used to process and include this data into the dataset.

It was quite difficult to retrieve latitudes and longitudes for about 8100 distinct sites. Based on station names, latitudes and longitudes were retrieved using a Google API found in Google Sheets. This automated method made getting geographic coordinates easier.

There were some issues connecting Amazon S3 to Snowflake; in particular, there was a 403 error that needed to be fixed in the bucket policy. To resolve this issue, the data was successfully saved in Amazon S3 and connected to Snowflake for additional processing after the access problem was fixed and the bucket policy was updated.

It was difficult to load and normalize the dataset in Snowflake without using direct insert instructions after connecting it to S3. We tried to include copying data from the unnormalized table to normalized tables based on unique primary keys and utilizing the AS command.

It was difficult to install and configure the Snowflake driver in Tableau, and it was later found that the driver was incompatible with macOS. After resolving installation concerns, the compatibility issue was fixed by moving to Windows, which made it possible for Snowflake and Tableau to successfully communicate.

There were difficulties in configuring Snowflake with Python and creating analytical queries, which called for careful evaluation of analytics and visualizations. Using the Snowflake dataset for perceptive analysis, analytical queries and visualizations were created upon a successful connection.

## IX. Lessons Learned

**1. Excel Functions & Data Environment:** A thorough investigation of Excel tools like VLOOKUP, COUNTA, and UNIQUE as well as strategies like PIVOT tables resulted from the requirement to improve the dataset. This was a valuable learning opportunity that demonstrated the significance of data manipulation abilities throughout the preparation phase.

**2. Wikipedia Data Extractions:** An inventive method of data enrichment was shown in the procedure used to obtain station-state mappings from a Wikipedia page. Using data that has been manually gathered from a publically accessible source highlights the value of creativity and flexibility when working with a variety of datasets.

**3. Geocoding with Google API in Google Sheets:** The team discovered a special feature of Google Sheets — the use of extensions — after overcoming the difficulty of acquiring latitudes and longitudes for a large number of unique stations. Acquiring the ability to use the Google API for geocoding offered a workable way to enrich geographic data.

**4. Data Lake and Amazon S3 Integration:** The team first learned about data lakes when they decided to store data in Amazon S3 and combine it with Snowflake. Connecting, setting up pipelines, and automating the loading of data from S3 to Snowflake improved knowledge of cloud-based analytics and scalable data storage.

**5. Snowflake Database Management:** The intricate workings of Snowflake were revealed by connecting it to Amazon S3. A thorough understanding of cloud-based relational databases was obtained by learning how to load CSV files, transform them into normalized tables, and handle several schemas, processes, and streams in Snowflake.

**6. Tableau and External Data Source Integration:** Discovering Tableau's integration with Snowflake broadened the scope of possibilities beyond conventional data sources such as CSV and MySQL. For thorough data analysis, it is crucial to comprehend how to link external sources for visualization. This highlights the significance of interoperability.

**7. Cross-Platform Compatibility and Troubleshooting:** The difficulties in getting Tableau to work with Snowflake on macOS and the need to migrate to Windows in order to resolve the issue highlighted how crucial cross-platform compatibility is. The emphasis of this learning experience was on the importance of problem-solving abilities and adaptability in order to ensure smooth integration between various contexts and tools.

**8. Python Integration with Snowflake:** The last step was setting up a Python connection to Snowflake and utilizing SnowSQL to write analytical queries. This illustrated how adaptable Snowflake is as a database system and showed how integrating programming languages may lead to more sophisticated analytics.

## X. Pair Programming

Our team enthusiastically implemented pair programming during the project, which promoted cooperation and knowledge exchange. This is how we put pair programming into practice and benefited from it:

**1. Collaborative Task Execution:** We deliberately tried to collaborate when working on coding assignments. For example, when we implemented the feature that needed more columns in our dataset, team members worked in pairs to determine how to best integrate the new data fields and make the necessary schema changes.

**2. Shared Code Ownership:** One of the core values in our team was shared code ownership. No one "owned" a particular portion of the software; instead, duties were handled collaboratively. This fostered a sense of togetherness and accountability while also guaranteeing group responsibility.

**3. Frequent Code Reviews:** An essential component of our pair programming method was code reviews. Every piece of code was reviewed collaboratively by the team, who contributed insights, offered helpful criticism, and made sure our code followed best practices overall. This continuous feedback cycle improved our codebase's quality.

**4. Problem Solving:** We purposefully held collaborative problem-solving meetings when faced with challenging problems or glitches. During these meetings, team members with varying specialties and strengths were paired, which allowed us to address problems as a group and come up with creative solutions.

## XI. Visualizations

Tableau proved to be a crucial tool in our research, which examined the vast dataset of Indian Railways, since it allowed us to convert unstructured data into insightful visual representations. By utilizing the smooth interaction between Tableau and Snowflake, our data warehousing technology, we were able to create visually engaging stories that revealed complex patterns and insights related to railway operations. We were able to develop dynamic and interactive dashboards that displayed the geographical relationships, station operations, and geographic distribution of rail routes thanks to Tableau's versatile interface. Tableau has shown to be a priceless tool in democratizing access to analytical insights by enabling end users to dynamically explore the data through the wise use of filters, parameters, and computed fields.

Here are some of our Visualizations:

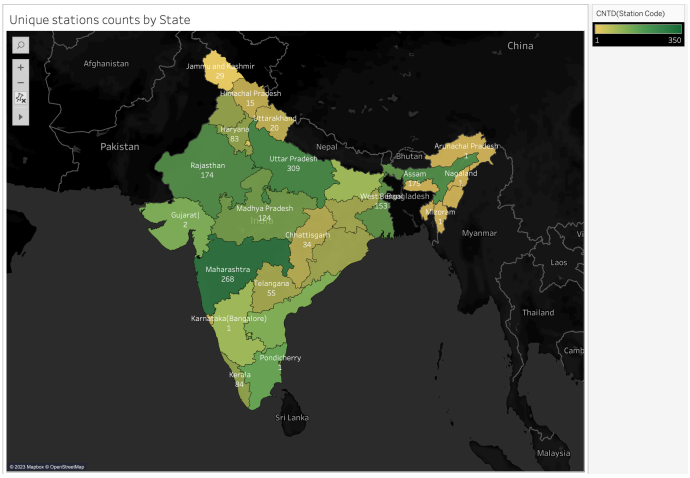


Fig. 14. Unique Stations Count by States

**1. Unique Stations Count by States**

```
SELECT state, count(DISTINCT(station_code))
FROM station_info
WHERE state != 'N/A' AND state != '0'
GROUP BY state
ORDER BY count(DISTINCT(station_code)) DESC;
```

This figure shows the number of distinct stations in each of the Indian states, with Uttar Pradesh leading the way with 309 stations, and Maharashtra following closely after at 268. The legend uses a color scheme to show changes in density. This succinct graphic effectively conveys the distribution and density of railroad stations among the states.

**2. Train Route by Station**

```
SELECT ts.train_no, t.train_name, ts.SEQ, ts.station_code,
s.station_name, ts.arrival_time, ts.departure_time
FROM train_station ts
JOIN train_info t ON t.train_no = ts.train_no
JOIN stations_info ON s.station_code = ts.station_code
WHERE ts.train_no = '19571'
```

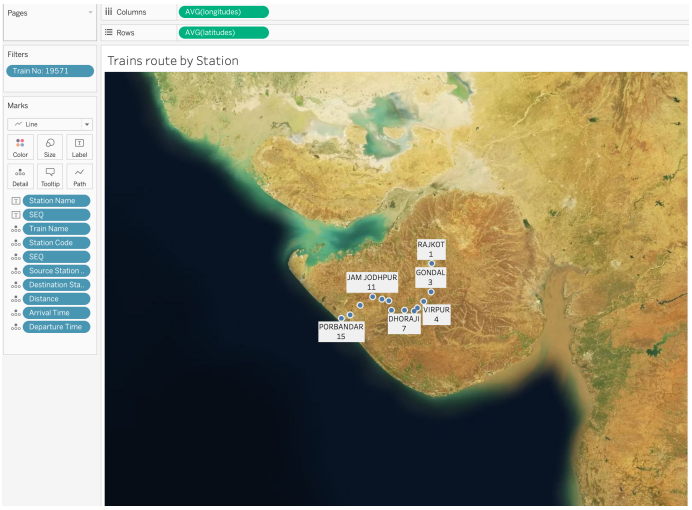


Fig. 15. Train Route by Station

```
ORDER BY ts.SEQ;
```

This train route visualization for the 19571 Rajkot Porbandar Express allows viewers to easily follow the journey through 14 stations to reach their destination. The order in which these stations are visited can be clearly understood by examining the sequence numbers shown in the figure. For both passengers and stakeholders, this representation helps with travel planning, comprehending the route structure, and assessing the status of the journey.

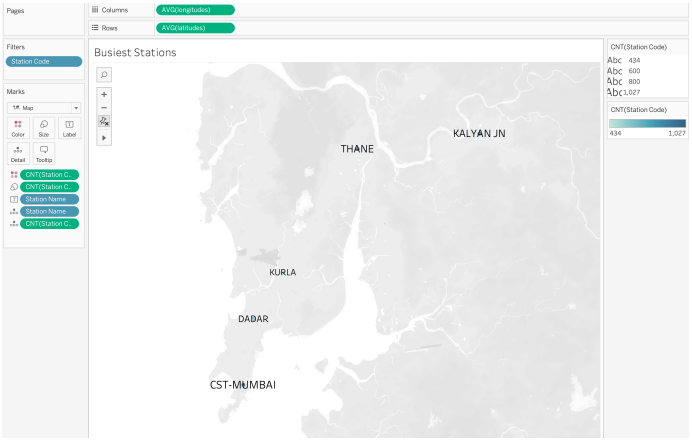


Fig. 16. Busiest Stations

**3.**

```
SELECT station_code, COUNT(station_code),
COUNT(DISTINCT(train_no))
FROM train_station
GROUP BY station_code
ORDER BY COUNT(station_code) DESC;
```

**4. Maximum Distance covered by Trains (Top 10)**

```
SELECT train_no, distance
```

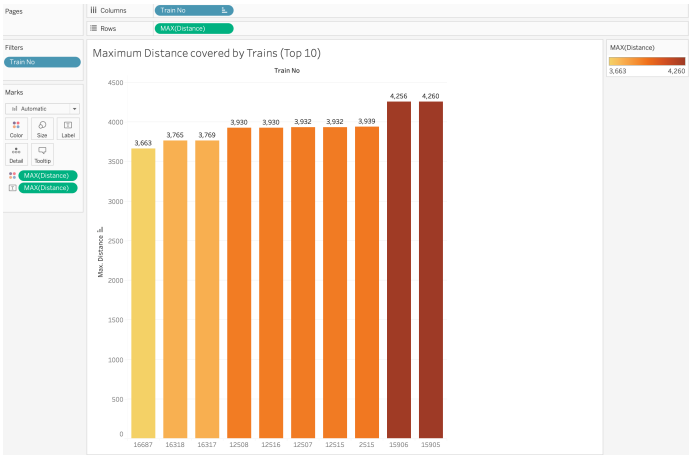


Fig. 17. Maximum Distance covered by Trains (Top 10)

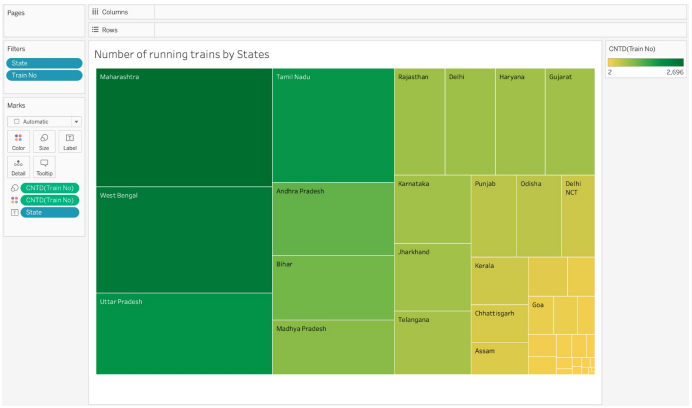


Fig. 19. Number of running Trains by States

```
FROM train_station
WHERE (train_no, SEQ) in (
SELECT trin_no, FLOOR(MAX(SEQ+0), 0)
FROM Train_station
GROUP BY train_no)
ORDER BY distance DESC LIMIT 10;
```

**5.**

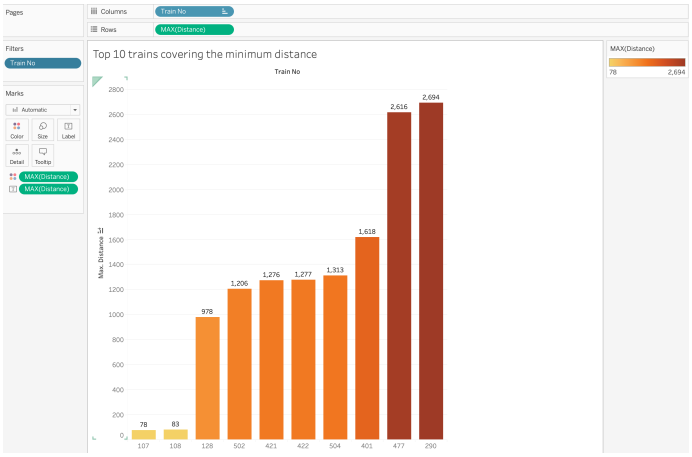


Fig. 18. Minimum Distance covered by Trains (Top 10)

**6.**

## XII. **Conclusion**

In Conclusion, our study of Indian Railways, which we conducted using Snowflake's strong architecture as our preferred data warehousing solution, has greatly improved our understanding and analytical skills in the broad field of railway operations. The methodical ETL procedure made it easier to extract, transform, and load raw data from a variety of sources when it was applied to Snowflake's DB225PROJECT database. We used the capability of Snowflake as the data warehouse for processing, Python for scripting and analysis, Tableau for visualizations, and Amazon S3 as a data lake to carefully navigate through the complexities of the Indian Railways. Our dedication to data integrity and query performance is demonstrated by the development of a STAR Schema and the thorough examination of normalization, indexing, and restrictions.

We have significantly enhanced our comprehension and analytical abilities in the wide field of railway operations through our research of Indian Railways, which we carried out with Snowflake's robust architecture as our chosen data warehousing solution. Applying the systematic ETL procedure to Snowflake's DB225PROJECT database simplified the process of extracting, transforming, and loading raw data from several sources. To carefully navigate through the complexity of the Indian Railways, we made use of Tableau for visualizations, Python for scripting and analysis, Snowflake as the processing data warehouse, and Amazon S3 as a data lake. The creation of a STAR Schema and the careful analysis of normalization, indexing, and constraints show our commitment to data integrity and query performance.

## REFERENCES

(1) Wolski, A.; Hofhauser, B. In *21st International Conference on Data Engineering Workshops (ICDEW'05)*, 2005, pp 1210–1210.

(2) Patel, R.; Joshi, R.; Professor, A. Envision of I-RS (I-Railway System) -based on Cloud Computing. **2015**, *4*.

(3) Zhongjie, Y. In *2022 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*, 2022, pp 864–872.

(4) Gupta, A.; Dhanda, N.; Gupta, K. K. In *2023 11th International Conference on Emerging Trends in Engineering Technology - Signal and Information Processing (ICETET - SIP)*, 2023, pp 1–6.

(5) Gershkovich, S.; Graziano, K., 2023.

## APPENDIX

**1. GitHub Link of Project:**https://github.com/Vishal3041/Railway_Ops

**2. Project Elevator Pitch Video:** https://drive.google.com/drive/folders/1jjUhzVCDH2QJ0bDvYKRGiG8NUnryxawA?usp=drive_link

| Sr. No. | Criteria | How it is met |
|---|---|---|
| 1. | Presentation Skills - Includes time management | Made a PPT and will be presenting in class |
| 2. | Code Walkthrough | Connected SnowFlake with Python (In Class) |
| 3. | Discussion / QnA | In Class |
| 4. | Demo | In Class |
| 5. | Version Control - Use of Git / GitHub or equivalent; must be publicly accessible | Created RailwayOps repository on GitHub using Git which has the whole project |
| 6. | Significance to the real world | Used in Analysis and real time tracking of trains which will help customers plan their trip |
| 7. | Lessons learned Included in the report and presentation? How substantial and unique are they? | Excel Functions & Data Environment, Geocoding with Google API in Google Sheets, Data Lake and Amazon S3 Integration, Snowflake Database Management |
| 8. | Innovation | Created Visualizations with Tableau for train analysis which will help customers plan their trip |
| 9. | Teamwork | Added with Sprint Planning MoM |
| 10. | Technical difficulty | Latitude-Longitude Retrieval, Connecting Amazon S3 with SnowFlake |
| 11. | Practiced pair programming? | Collaborative Task Execution, Shared Code Ownership, Frequent Code Reviews |
| 12. | Practiced agile / scrum (1-week sprints)? | Submitted additional file for this |
| 13. | Used Grammarly / other tools for language? | Used Grammarly for spell check and correct vocabulary |
| 14. | Slides | Made a PPT for this project |
| 15. | Report - Format, completeness, language, plagiarism, whether turnItIn could process it (no unnecessary screenshots), etc | Made a report in Overleaf and made use of Grammarly |
| 16. | Used unique tools | Made this report in Overleaf (LaTeX file uploaded with project submission) |
| 17. | Performed substantial analysis using database techniques | Took help from Research Papers and blogs |
| 18. | Used a new database or data warehouse tool not covered in the HW or class | Used Snowflake as our primary Data Warehouse and |
| 19. | Used appropriate data modeling techniques | Used Normalizing, Indexing, Data Types & constraints methods and made ER Diagram |
| 20. | Used ETL tool | Performed Extract, Transform and Load operations in SnowFlake |
| 21. | Demonstrated how Analytics support business decisions | Created Visualizations using SQL Queries to analyze different scenarios |
| 22. | Used RDBMS | SnowSQL |
| 23. | Used Datawarehouse | Used SnowFlake Database |
| 24. | Includes DB Connectivity / API calls | Connected Python & SnowFlake and wrote queries for visualizations in SnowSQL |
| 25. | Used NOSQL | Not Used |