International Journal of Science and Research (IJSR)
ISSN: 2319-7064
ResearchGate Impact Factor (2018): 0.28 | SJIF (2018): 7.426

Enhancing Big Data Interoperability: Automating Schema Expansion from Parquet to BigQuery

Preyaa Atri

Email: *preyaa.atri91[at]gmail.com*

Abstract: ***In the realm of data engineering, efficient data migration and transformation are pivotal. The Parquet Schema Expansion Migrator for BigQuery is a Python library designed to streamline the process of migrating column data from Parquet files to Google BigQuery tables, while expanding the BigQuery table schema to accommodate columns present in the Parquet data but missing from the BigQuery schema. This paper explores the problem of schema evolution in data warehouses, introduces the library as a solution, discusses its uses and impact, and outlines future enhancements and recommendations for robust data type management.***

Keywords: BigQuery, Parquet, Schema Migration, Data Engineering, Cloud Storage, Data Transformation

1. Introduction

As Big Data grows in volume, velocity, and variety, data warehouses like Google BigQuery must efficiently adapt to schema evolution without extensive downtime or manual intervention (Dawelbeit & McCrindle, 2016). Parquet files, often used for storing data due to their efficiency and compression, may have schemas that evolve separately from those in BigQuery. This mismatch poses a challenge for data engineers who need to ensure data integrity and accessibility during schema evolution. The Parquet Schema Expansion Migrator library was developed to address this challenge. Additionally, the development of the Parquet Schema Expansion Migrator library aligns with the need for scalable and distributed systems, as discussed by (Sharma et al., 2018), to effectively process and store data in parallel, ensuring seamless schema evolution. Additionally, the work by Boss & Broussard (2017) underscores the challenges of archiving and preserving digital applications, emphasizing the importance of adapting to new technologies and methodologies to prevent data loss.

2. Problem Statement

Schema evolution is a natural part of the data lifecycle, especially in big data environments. BigQuery administrators often face challenges when Parquet files evolve, adding new columns that are not present in the BigQuery table schema, resulting in potential data loss or the need for cumbersome manual schema updates.

3. Proposed Solution

The Parquet Schema Expansion Migrator mitigates these challenges by providing a Python - based solution that transfers column data from Parquet files to BigQuery tables, automatically expanding the BigQuery schema with any missing columns found in the Parquet files.

Parquet Schema Expansion Migrator library offers a suite of functionalities:

- **Seamless Data Transfer:** The library facilitates the transfer of column data from Parquet files to BigQuery tables.
- **Automatic Schema Expansion:** It automatically detects missing columns in the BigQuery table schema compared to the Parquet data. Subsequently, it expands the BigQuery schema by adding these missing columns.
- **Data Manipulation with pandas:** The library leverages pandas DataFrames, a popular Python library for data analysis and manipulation, to efficiently handle data during the migration process.
- **Google Cloud Storage (GCS) Integration:** The library supports interaction with GCS, enabling it to retrieve Parquet files stored in the cloud.

Library Documentation & Usage

The library provides a primary function, Parquet_Schema_Expansion_Migrator_for_BigQuery, that takes several arguments:

- bucket_name: Name of the GCS bucket containing the Parquet file.
- parquet_file_path: Path to the specific Parquet file within the bucket.
- project_id: Google Cloud Project (GCP) project ID where the BigQuery dataset resides.
- dataset_id: ID of the BigQuery dataset containing the target table.
- table_id: ID of the BigQuery table where the data will be migrated.

By providing these arguments, users can initiate the data transfer process with automatic schema expansion if necessary.

Installation

The library can be installed using one of the below mentioned pip commands:

```
pip install Parquet_Schema_Expansion_Migrator_for_BigQuery
```

Example

The following code snippet demonstrates how to use the Parquet_Schema_Expansion_Migrator_for_BigQuery function to migrate data from a Parquet file to a BigQuery table:

```
Python
from Parquet_Schema_Expansion_Migrator_for_BigQuery import
Parquet_Schema_Expansion_Migrator_for_BigQuery
# Replace placeholders with your actual values
bucket_name = "your_bucket_name"
parquet_file_path = "path/to/your/file. parquet"
project_id = "your_project_id"
dataset_id = "your_dataset_id"
table_id = "your_table_id"
Parquet_Schema_Expansion_Migrator_for_BigQuery (bucket_name, parquet_file_path, project_id, dataset_id, table_id)
```

In this example, the script imports the Parquet_Schema_Expansion_Migrator_for_BigQuery function from the library. Then, it defines variables for the GCS bucket name, Parquet file path, GCP project ID, BigQuery dataset ID, and BigQuery table ID. Finally, it calls the Parquet_Schema_Expansion_Migrator_for_BigQuery function, providing the defined variables as arguments. This initiates the data transfer process from the Parquet file to the BigQuery table, with automatic schema expansion if required.

4. Dependencies and Considerations

The efficacy of the Parquet Schema Expansion Migrator is deeply intertwined with its foundational components – a selection of powerful libraries that are central to its operation. The library leverages pandas, known for its prowess in data manipulation (McKinney, 2010); pyarrow, which facilitates the handling of Parquet files; and the google - cloud - storage and google - cloud - bigquery libraries, essential for engaging with Google Cloud's storage and database services. These building blocks are critical, not just as isolated tools, but as part of an integrated whole that ensures a fluid and effective migration process.

To fully realize the library's capabilities, these dependencies must not only be present but also kept up to date. While the current iteration of the library delivers robust functionality, users should anticipate the need to craft or incorporate additional functions, such as **alter_schema** and **ExecuteBqQuery**. These functions are envisioned to extend the library's schema manipulation abilities, though they remain beyond its current feature set.

Moreover, the library's current approach to schema expansion—defaulting to STRING data types for new columns—presents an opportunity for refinement. Advancing to a more discerning method that can intelligently ascertain the correct data types or accept user - defined parameters would significantly enhance its adaptability and precision.

5. Benefits and Applications

The Parquet Schema Expansion Migrator for BigQuery streamlines the integration of evolving Parquet files into BigQuery's analytical environment. Its key advantages include automating the otherwise manual and error - prone process of schema updates, which simplifies migration workflows. The library capitalizes on the powerful pandas library, renowned for its data manipulation capabilities, reducing development time for data engineers. It also ensures data consistency across BigQuery tables by automatically aligning them with the latest Parquet file schemas.

In practice, the library finds its utility in various scenarios, including data warehousing, where it aids in transferring historical Parquet - stored data to BigQuery for advanced analytics. It also fits seamlessly into ETL pipelines, ensuring Parquet data is processed and loaded efficiently into BigQuery. For cloud - centric architectures, it facilitates the migration of data from cloud storage directly into BigQuery, underscoring its importance in a cloud - first data strategy.

6. Conclusion & Future Scope

In conclusion, the Parquet Schema Expansion Migrator for BigQuery emerges as a pivotal tool in data migration, significantly enhancing the efficiency of data transfers from Parquet to BigQuery. This contribution is not only valuable for its immediate impact on data consistency and development efficiency in data engineering practices but also holds the potential to be a cornerstone in the advancement of AI - based models. The library's ability to automate schema expansion and ensure data consistency is not merely a convenience but a necessity in the era of big data, where AI algorithms thrive on vast and well - structured datasets.

The future scope of this library extends beyond the realm of data engineering. By seamlessly integrating evolving Parquet data into BigQuery, it lays the groundwork for building robust AI algorithms. The automated schema adaptation ensures that AI models are trained on the most up - to - date and comprehensive data, leading to more accurate and reliable predictions. The library's potential for customization through user - defined data type mappings further empowers AI practitioners to tailor data preprocessing to the specific needs of their models. As AI continues to permeate various industries, the Parquet Schema Expansion Migrator for BigQuery stands as an essential tool, bridging the gap between raw data and intelligent algorithms, and ultimately contributing to the progress of AI - driven innovation.

References

[1] O. Dawelbeit and R. McCrindle, "Efficient dictionary compression for processing rdf big data using google bigquery", 2016 IEEE Global Communications Conference (GLOBECOM), 2016. https: //doi. org/10.1109/glocom.2016.7841775
[2] K. Sharma, U. Marjit, & U. Biswas, "Efficiently processing and storing library linked data using apache spark and parquet", Information Technology and Libraries, vol.37, no.3, p.29 - 49, 2018. https: //doi. org/10.6017/ital. v37i3.10177
[3] K. Boss and M. Broussard, "Challenges of archiving and preserving born - digital news applications", IFLA Journal, vol.43, no.2, p.150 - 157, 2017. https: //doi. org/10.1177/0340035216686355
[4] McKinney, Wes. (2010). Data Structures for Statistical Computing in Python.56 - 61.10.25080/Majora - 92bf1922 - 00a.

[5] Apache Parquet, "File Format Documentation, " [Online]. Available: https: //parquet. apache. org/docs/file - format/
[6] Pandas development team [Online]. pandas. DataFrame. to_gbq. Available: https: //pandas. pydata. org/docs/reference/api/pandas. DataFrame. to_gbq. html
[7] Google Cloud Platform. [Online]. BigQuery Documentation. Available: https: //cloud. google. com/bigquery/docs