Home Credit Default Analysis using Google Big Query

Ananya Varsha (40197013) Saghana Mahesh Sarma (40198979)

Concordia University Concordia University

Montreal, Canada Montreal, Canada

[ananyavarsha28@gmail.com](mailto:ananyavarsha28@gmail.com) [maheshsaghana0@gmail.com](mailto:maheshsaghana0@gmail.com)

Sivakumaran Janardhanan (40155790) Vishanth Surresh (40181942)

Concordia University Concordia University

Montreal, Canada Montreal, Canada

[sivamjsk96@gmail.com](mailto:sivamjsk96@gmail.com) [vishanthsurresh@gmail.com](mailto:vishanthsurresh@gmail.com)

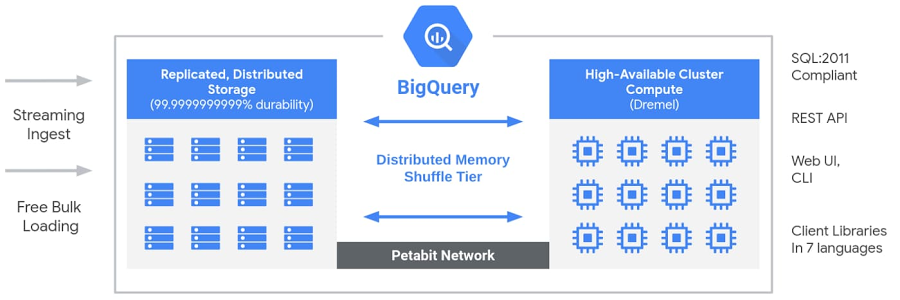
ABSTRACT

Many people face lot of struggles to get loan due to insufficient or non-existent credit histories. This unfortunately leads people to believe untrustworthy lenders. Thus, opening an opportunity to analyze the Home Credit Default data using Big-Data systems. This project, therefore, aims to analyze this Credit Default data using Google Big Query platform. In this report, we majorly discuss about how distributed system aspects are implemented in Google Big Query, how we used Big Query to analyze the data and the comparisons between single system database and Google Big Query.

1 INTRODUCTION

2 Architecture

Big Query is a cloud-based data platform that allows users to have complete control over their data. The warehouse enables large-scale economic queries and data volumes that are comparable to those seen on Google to be carried out at Google-like speeds. Business insights are strengthened by taking use of low prices and Google's world-class scalability and security infrastructure. Big-Query is a petabyte-scale data warehouse solution for big-data analysis that is one of the fastest on the market. With no have to manage the infrastructure, customers can specialize in building meaningful insights using familiar SQL without the necessity for a database administrator. it is also economical because you merely procure the processing and storage you utilize [1].



The server-less architecture of Big Query separates storage and computation, allowing for independent scaling as needed. This structure provides customers with both great flexibility and cost control because they don't have to keep running expensive computing resources all the time. On a very high level, big query has two service components – storage service and query service. These are connected by Goggle's Jupiter's high-speed network infrastructure. The storage component manages data with high durability. When a user submits a query, Big Query initiates computing unit on demand, reading data from distributed storage and executing the query. All data is compressed and stored in a columnar storage format called the Colossus File System. This provides durability and availability. Storage service supports bulk data ingestion and streaming ingestion. It works with large amounts of data as well as real-time data streams. The query component uses compute slots to process the SQL queries and after computation returns the result. Big Query uses slots which are basically virtual CPUs that are allocated based on the query size.

If compute isn't being employed, we will power it off and just keep the storage nodes running. It significantly reduces the prices incurred because the customers don’t need to keep the expensive compute resources running all the time. thanks to this, both storage and compute may be scaled differently, without affecting or slowing down one another. If we are streaming in or bulk loading the info into Big Query, while parallelly executing heavy queries at the identical time, both of the operations can run together with no dependency on one another. That’s the major reason why Big Query is more economical and scalable.

Big Query uses four internal components to complete and serve your query results. The primary component, Dremel generates execution trees from SQL queries. The tree's leaves are referred to as slots, and that they are answerable for reading data from storage and doing any necessary calculations. The tree's branches are 'mixers,' which are accountable for aggregation. Within the second component Colossus, all of the information is compressed and stored in a very columnar storage structure. Colossus is answerable for replication, recovery (in the event that drives fail), and distributed administration (so there's no single point of failure). Colossus allows customers to store several of petabytes of knowledge in an exceedingly single location. The third component, Jupiter network allows compute and storage to speak with each other. The Shuffle may be a common nickname for the Network. The fourth component, BORG is to blame for coordinating in order that all of the engine's sections can operate together effortlessly. Borg controls the mixers and slots, further as allocating hardware resources.

2.2 Tables

Curabitur vitae nulla dapibus, ornare dolor in, efficitur enim. Cras fermentum facilisis elit vitae egestas. Mauris porta, neque non rutrum efficitur, odio odio faucibus tortor, vitae imperdiet metus quam vitae eros. Proin porta dictum accumsan Table 1.

Duis cursus maximus facilisis. Integer euismod, purus et condi-mentum suscipit, augue turpis euismod libero, ac porttitor tellus neque eu enim. Nam vulputate est non tellus efficitur pharetra. Ae-nean molestie tristique venenatis. Nam congue pulvinar vehicula. Duis lacinia mollis purus, ac aliquet arcu dignissim ac Table 2.

Table 2: Frequency of Special Characters.

|  |  |  |
| --- | --- | --- |
| Non-English or Math | Frequency | Comments |
| ∅ | 1 in 1000 | For Swedish names |
| 𝜋 | 1 in 5 | Common in math |
| $ | 4 in 5 | Used in business |
| Ψ | 1 in 40 000 | Unexplained usage |

Nulla sit amet enim tortor. Ut non felis lectus. Aenean quis felis faucibus, efficitur magna vitae. Curabitur ut mauris vel augue tem-por suscipit eget eget lacus. Sed pulvinar lobortis dictum. Aliquam dapibus a velit.

2.3 Listings and Styles

Aenean malesuada fringilla felis, vel hendrerit enim feugiat et. Proin dictum ante nec tortor bibendum viverra. Curabitur non nibh ut mauris egestas ultrices consequat non odio.

* Duis lacinia mollis purus, ac aliquet arcu dignissim ac. Viva-mus accumsan sollicitudin dui, sed porta sem consequat.
* Curabitur ut mauris vel augue tempor suscipit eget eget lacus. Sed pulvinar lobortis dictum. Aliquam dapibus a velit.
* Curabitur vitae nulla dapibus, ornare dolor in, efficitur enim.

Ut sagittis, massa nec rhoncus dignissim, urna ipsum vestibulum odio, ac dapibus massa lorem a dui. Nulla sit amet enim tortor. Ut non felis lectus. Aenean quis felis faucibus, efficitur magna vitae.

1. Duis lacinia mollis purus, ac aliquet arcu dignissim ac. Viva-mus accumsan sollicitudin dui, sed porta sem consequat.
2. Curabitur ut mauris vel augue tempor suscipit eget eget lacus. Sed pulvinar lobortis dictum. Aliquam dapibus a velit.
3. Curabitur vitae nulla dapibus, ornare dolor in, efficitur enim.

Cras fermentum facilisis elit vitae egestas. Mauris porta, neque non rutrum efficitur, odio odio faucibus tortor, vitae imperdiet metus quam vitae eros. Proin porta dictum accumsan. Aliquam dapibus a velit. Curabitur vitae nulla dapibus, ornare dolor in, efficitur enim. Ut maximus mi id arcu ultricies feugiat. Phasellus facilisis purus ac ipsum varius bibendum.

2.3 Math and Equations

Curabitur vitae nulla dapibus, ornare dolor in, efficitur enim. Cras fermentum facilisis elit vitae egestas. Nam vulputate est non tel-lus efficitur pharetra. Vestibulum ligula est, varius in suscipit vel, porttitor id massa. Cras facilisis suscipit orci, ac tincidunt erat.

 (1)

3 CITATIONS

Some examples of references. A paginated journal article [2], an enumerated journal article [7], a reference to an entire issue [6], a monograph (whole book) [15], a monograph/whole book in a series (see 2a in spec. document) [13], a divisible-book such as an anthology or compilation [10] followed by the same example, how-ever we only output the series if the volume number is given [9](so Editor00a’s series should NOT be present since it has no vol. no.), a chapter in a divisible book [17], a chapter in a divisible book in a series [8], a multi-volume work as book [14], an article in a proceedings (of a conference, symposium, workshop for example)(paginated proceedings article) [3], a proceedings article with all possible elements [16], an example of an enumerated proceedings article [11], an informally published work [12], a doctoral disserta-tion [5], a master’s thesis [4], an finally two online documents or world wide web resources [1, 18].

ACKNOWLEDGMENTS

This work was supported by the [...] Research Fund of [...] (Number [...]). Additional funding was provided by [...] and [...]. We also thank [...] for contributing [...].

REFERENCES

[1] https://cloud.google.com/blog/products/data-analytics/new-blog-series-bigquery-explained-overview

[2] Sten Andler. 1979. Predicate path expressions. In *Proceedings of the 6th. ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages (POPL '79)*. ACM Press, New York, NY, 226-236. DOI:https://doi.org/10.1145/567752.567774

[3] Ian Editor (Ed.). 2007. *The title of book one* (1st. ed.). The name of the series one, Vol. 9. University of Chicago Press, Chicago. DOI:https://doi.org/10.1007/3-540-09237-4.

[4] David Kosiur. 2001. *Understanding Policy-Based Networking* (2nd. ed.). Wiley, New York, NY..

Conference Name:ACM Woodstock conference

Conference Short Name:WOODSTOCK’18

Conference Location:El Paso, Texas USA

ISBN:978-1-4503-0000-0/18/06

Year:2018

Date:June

Copyright Year:2018

Copyright Statement:rightsretained

DOI:10.1145/1234567890

RRH: F. Surname et al.

Price:$15.00