A Review of Google Cloud's Database Offerings

Nilesh Jagnik
Mountain View, USA
nileshjagnik@gmail.com

***Abstract*—Choosing the right database service is critical for optimal performance, scalability and reliability of an application. A well-functioning database provides a strong backbone for your application. For applications using Google Cloud, there are several database options. Some of these options are relational databases where data is stored in tables and columns, and relationships between different tables and columns can be specified in the database schema. Others are simpler, unstructured storage services. In this paper, we review all the different database options in Google Cloud. We discuss the benefits and suitability of each one in different scenarios.**

Keywords—databases, relational databases, cloud, NoSQL databases

I. INTRODUCTION

Most software systems store data for internal processing and external serving. This is especially true for backend servers that process, manage, store and serve data. For this reason, it is extremely important to adopt a good data storage option. Even though it is possible to use simple text or spreadsheets to store data, it is often better to use a more sophisticated option - databases. Databases are organized collections of structured data. These databases are usually managed by software called Database Management Systems (DBMS).

In the past, databases were required to be self-hosted. But due to rise in popularity of Cloud platforms, similar to servers and application engines, it is more convenient to use database management systems provided by Cloud platforms. There are many popular Cloud platforms that exist today, but in this paper, we take a look at different hosted database options in Google Cloud. It is easy to use Google Cloud databases if you are already using other Google Cloud solutions.

Google Cloud has several database options to suit various needs. These vary from relational databases like Cloud SQL, AlloyDB and Spanner to non-relational databases like Firestore and Bigtable. There are also other options to consider like Memorystore for in-memory storage and bare metal options for Oracle workloads.

Picking the right database option is important for optimal performance and scalability of your application. The right choice often depends on various factors determined by the use case. In this paper, we review all the different database options offered by Google Cloud and go over the strengths and weaknesses of each one compared to the others.

II. RELATIONAL DATABASES

Data items stored in a relational database have predefined relationships between them. Typically, data is organized in tables, columns and rows. This structure allows easily expressing the relationship between data. In a relational database, it is possible to define relationships between different data structures with the help of a schema definition.

A. Database Schema

In a relational database, data is stored in tables. Each row in a table contains attributes related to a single data entry and has unique identified associated with it, called the primary key. This primary key is constructed from the values of a predefined set of columns called primary key columns. Each row of a table can be used to create a relationship with another table. Such rows are called

foreign keys and they contain references to a primary key of another table.

B. Structured Query Language (SQL)

It is possible to write and read data from the database with the help of a Structured Query Language (SQL). SQL utilizes the relationships defined in the database schema to efficiently read and write data to the database.

C. Transactions

While reading and writing data, it is often desirable that data is not read in an inconsistent state, e.g., in the middle of a partially complete write. A transaction refers to all data changes in a write as a single unit.

III. BENEFITS OF RELATIONAL DATABASES

There are several qualities of relational databases which make them ideal for applications which care about preserving data relationships between data, e.g., in financial and retail sector.

A. Flexibility

It is easy to add or modify tables in a backward compatible way. We don't need to delete pre-existing data for modifications. Note that only certain types of modification are backward compatible.

B. ACID Properties

ACID refers to a collection of four related properties which are satisfied by relational databases:

1) Atomicity: All transactions in a relational database either fully succeed or are completely rolled back. Transactions should never partially update data.

2) Consistency: After a transaction is complete, data should be in a correct state as defined in the schema definition.

3) Isolation: Transactions should not contend with one another. The effect of a transaction should be invisible to others until is fully committed.

4) Durability: Once a transaction is applied, its effects are permanent.

C. Ease of Use

The support for SQL makes it easy to write and read data, even for users who are non-technical.

IV. GOOGLE CLOUD RELATIONAL DATABASES

A. Cloud SQL

Cloud SQL provides fully managed database service supporting popular database engines like MySQL, PostgreSQL and SQL Server. It is a cost-effective option which allows easy migration for databases already setup using aforementioned database engines. It provides high availability with very low downtimes. It has built-in security features such as data encryption and network security controls. It also provides support for automated backups.

Although Cloud SQL is a good choice for small to medium scale applications needing a managed, vertically-scalable and secure relational database, it is not suited for large-scale, globally distributed applications. Its performance is not as good as other options for high throughput applications.

B. AlloyDB

AlloyDB is a fully managed, PostgreSQL-compatible database service optimized for high performance workloads with advanced features for large datasets, high transaction frequency and complex queries. AlloyDB offers machine learning based database management, resource optimization and query optimization. Other features of AlloyDB include high availability with non-disruptive maintenance, data backups and disaster recovery, security and access control and encryption.

Although AlloyDB only supports PostgreSQL, its performance for high throughput applications is much better compared to Cloud SQL. This comes with the tradeoff of higher cost. AlloyDB is a strong choice for applications needing high performance and scalability with minimal management overhead. AlloyDB is ideal for applications that have user facing and responsive interactions.

C. Cloud Spanner

Cloud Spanner is a fully managed, enterprise grade, globally distributed and strongly consistent database service. Spanner can distribute data across multiple regions and zones, providing low-latency access. It offers automatic horizontal scalability and high availability. Spanner also offers strong consistency – meaning that for external readers it appears as if transactions were committed one after another even

though under the hood, Spanner distributes them across multiple servers. Even with these guarantees, Spanner offers high performance, although not as high as AlloyDB. Spanner supports PostgreSQL and GoogleSQL as query languages.

Cloud Spanner is ideal for mission-critical applications which require high availability, along with multi-region data consistency and scalability.

D. *Bare Metal*

Google Cloud also provides the ability to onboard specialized workflows that use Oracle databases. This provides an easy way to migrate pre-existing workflows to Google Cloud. Note that bare metal databases are not fully managed, unlike the other relational database options.

V. NON RELATIONAL DATABASES

Non-relational databases, also known as NoSQL databases, store complex unstructured data in a non-tabular format. NoSQL databases use simple and flexible schema and leave it up to the reader of data to interpret additional structure that may be present in data. These databases are often used to organize large quantities of unstructured data.

Common types of NoSQL databases are document stores, key-value stores, column-oriented databases, graph databases and in-memory databases. NoSQL databases can vary a lot in functionality and the exact nature of NoSQL databases depends on the use case and type.

VI. NOSQL VS RELATIONAL DATABASES

A. *Flexible Schema*

Unlike relational databases, NoSQL database schemas are flexible, making this ideal of unstructured or semi-structured data storage. This also increases adaptability and makes it easy to add new types of data and modify the existing schema.

B. *Scalability*

Scaling NoSQL databases is easier compared to relational databases since the database does not need to do work to maintain relationships between data.

C. *Performance*

Compared to relational databases, NoSQL databases have faster query time due to only allowing simpler lookups. It is easier to add duplication due to reduced relationship maintenance overhead.

D. *Huge Data Storage*

It is possible to store large amount of unstructured data easily. This is usually more difficult in relational databases.

E. *Consistency*

NoSQL datastores aim for eventual consistency, meaning data will be in a consistent state after some delay. Many NoSQL databases do not have consistency guarantees like relational databases.

F. *Transactions*

NoSQL databases usually do not guarantee transaction atomicity and isolation. These databases are not suited to transactional workloads.

VII. GOOGLE CLOUD NOSQL OPTIONS

A. *Firestore*

Firestore is a fully managed, serverless, scalable document database which has automatic scaling and high availability. Firestore is unique in that it offers ACID transactions. Firestore also has an offline mode which is great for on-device and real-time applications. Firestore is a good choice for general purpose data storage.

B. *Cloud Bigtable*

Cloud Bigtable is a NoSQL database that supports very low latency and high throughput applications. It can support reads and writes at a very high rate, sub-millisecond latency and can scale up to a large amount of data, enabling storage of petabytes of data. Cloud Bigtable is a good choice for MapReduce workloads.

C. *Memorystore*

Memorystore is a fully managed, in-memory data store that supports Redis and Memcached. Memorystore has features like high availability, failover, patching, and monitoring. It supports scaling up to 250 nodes and terabytes of keyspace. It offers 60x higher throughput in comparison to Redis and has microsecond latencies. Memorystore is great for caching and other short lived data storage.

CONCLUSION

With the many options for data storage provided by Google Cloud, it can be difficult to choose the right one. The first choice to make is between relational and non-relational database. This is often an easy choice to make based on the requirements of the application. However, within each category, there are several options available. To choose the right one, it is important to study the needs of your application. Each offering has a niche in which it outperforms others. For the best user experience, developers should identify which niche their application belongs to.

REFERENCES

[1] Priyanka Vergadia, "Your Google Cloud database options, explained (Aug 2021)," https://cloud.google.com/blog/topics/developers-practitioners/your-google-cloud-database-options-explained

[2] "What Is a Database? (Nov 2020)," https://www.oracle.com/database/what-is-database

[3] "What is a Relational Database (RDBMS)? (Jan 2021)," https://www.oracle.com/database/what-is-a-relational-database

[4] "What is a relational database? (Jan 2021)," https://cloud.google.com/learn/what-is-a-relational-database

[5] "What is NoSQL? (Feb 2021)," https://www.oracle.com/database/nosql/what-is-nosql

[6] "What is a NoSQL database? (Feb 2021)," https://cloud.google.com/discover/what-is-nosql

[7] Yifat Perry, "Google Cloud Database Services Explained: How to Choose the Right Service for Your Workloads (Feb 2021)," https://cloud.google.com/discover/what-is-nosql