

Experiment: 6

Date of Performance:

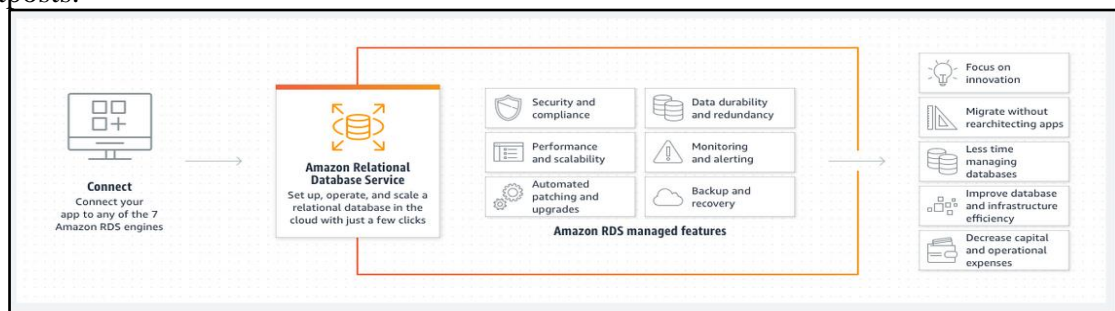
Date of Submission:

Aim: To know the concept of Database as a service running on cloud and to demonstrate the CRUD operations on different SQL and NOSQL databases running on cloud like AWS RDS, Azure SQL/Mongo Lab/Firebase.

Prerequisites: Firebase registration

Theory:

Amazon Relational Database Service (RDS) is a collection of managed services that makes it simple to set up, operate, and scale databases in the cloud. It supports - Amazon Aurora with MySQL compatibility, Amazon Aurora with PostgreSQL compatibility, MySQL, Maria DB, PostgreSQL, Oracle, and SQL Server — and deploy on-premises with Amazon RDS on AWS outposts.



Amazon RDS makes it easy to use replication to enhance availability and reliability for production workloads. Using the Multi-AZ deployment option, developer can run mission-critical workloads with high availability and built-in automated fail-over from your primary database to a synchronously replicated secondary database. Using Read Replicas, developer can scale out beyond the capacity of a single database deployment for read-heavy database workloads.

Benefits

Lower administrative burden

developer can use the AWS Management Console, the Amazon RDS Command Line Interface, or simple API calls to access the capabilities of a production-ready relational database in minutes.

Amazon RDS database instances are pre-configured with parameters and settings appropriate for the engine and class we have selected. developer can launch a database instance and connect your application within minutes. DB Parameter Groups provide granular control and fine-tuning of databases.

Performance

Amazon RDS General Purpose Storage is an SSD-backed storage option delivers a consistent baseline of 3 IOPS per provisioned GB and provides the ability to burst up to 3,000 IOPS above the baseline. This storage type is suitable for a broad range of database workloads.

Scalability

As your storage requirements grow, developer can also provision additional storage. The Amazon Aurora engine will automatically grow the size of your database volume as your database storage needs grow, up to a maximum of 64 TB or a maximum developer define. The MySQL, MariaDB, Oracle, and PostgreSQL engines allow developer to scale up to 64 TB of storage and SQL Server supports up to 16 TB. Storage scaling is on-the-fly with zero downtime.

Availability and durability

Database snapshots are user-initiated backups of your instance stored in Amazon S3 that are kept until developer explicitly delete them. developer can create a new instance from database snapshots whenever developer desire. Although database snapshots serve operationally as full backups, developer are billed only for incremental storage use.

Security

Amazon RDS allows developer to encrypt your databases using keys you manage through AWS Key Management Service (KMS). On a database instance running with Amazon RDS encryption, data stored at rest in the underlying storage is encrypted, as are its automated backups, read replicas, and snapshots.

Cost-effectiveness

There is no up-front commitment with Amazon RDS; developer simply pay a monthly charge for each database instance that developer launch. And, when developer is finished with a database instance, developer can easily delete it. To see more details, visit the Amazon RDS Instance Types page and the Amazon RDS Pricing page. Similar to AWS RDS, Firebase by Google also offers practical and efficient database solution.

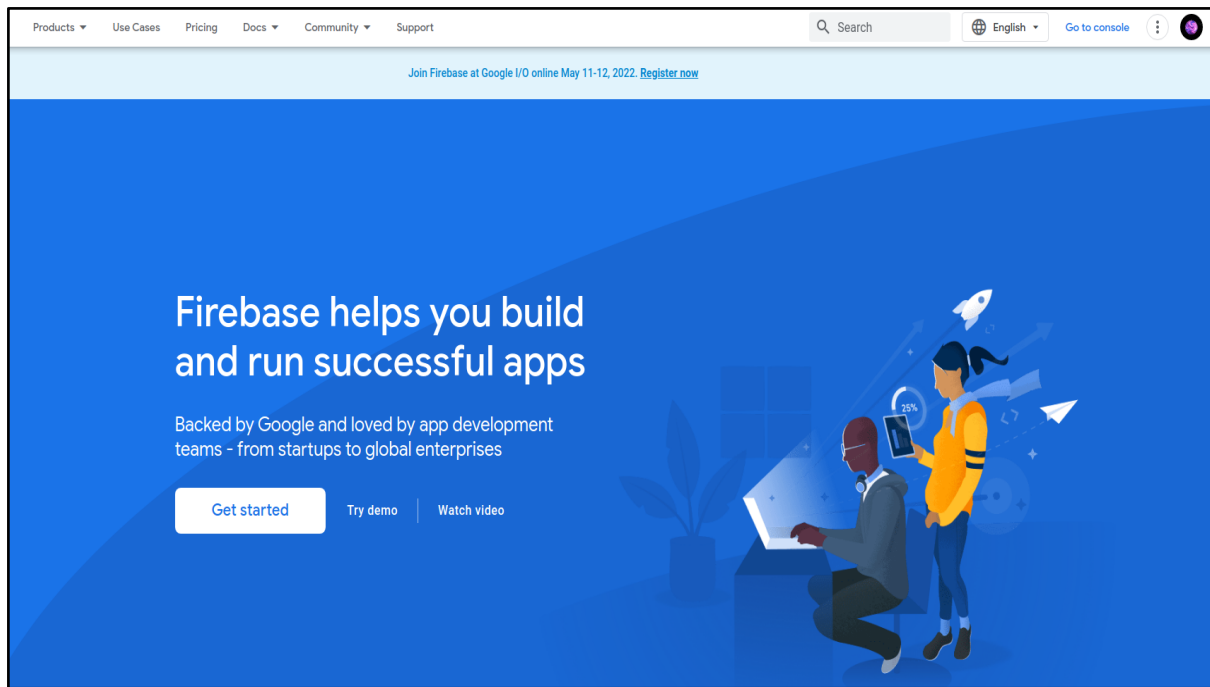
Firebase



Firebase is a mobile platform meant for app development. Firebase offers multiple services such as Realtime database, Firestore, Authentication and Cloud messaging to simplify and automate backend tasks. Cloud Firestore is a NoSQL, document-oriented database. Unlike a SQL database, there are no tables or rows. Instead, developer store data in *documents*, which are organized into *collections*. Each *document* contains a set of key-value pairs. Cloud Firestore is optimized for storing large collections of small documents. All documents must be stored in collections. Documents can contain *sub collections* and nested objects, both of which can include primitive fields like strings or complex objects like lists. Collections and documents are created implicitly in Cloud Firestore. Simply assign data to a document within a collection. If either the collection or document does not exist, Cloud Firestore creates it.

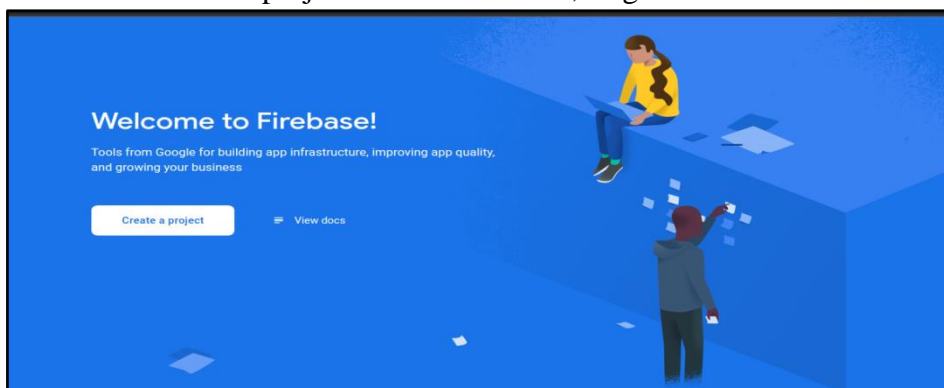
Registering to Firebase

Users register to Firebase using a standard Google account. Once registered, users directly land on the Firebase dashboard where they may create projects.

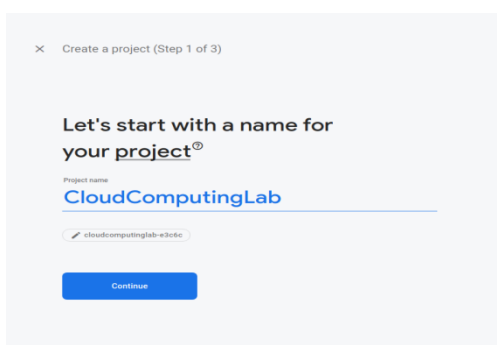


Creating a project

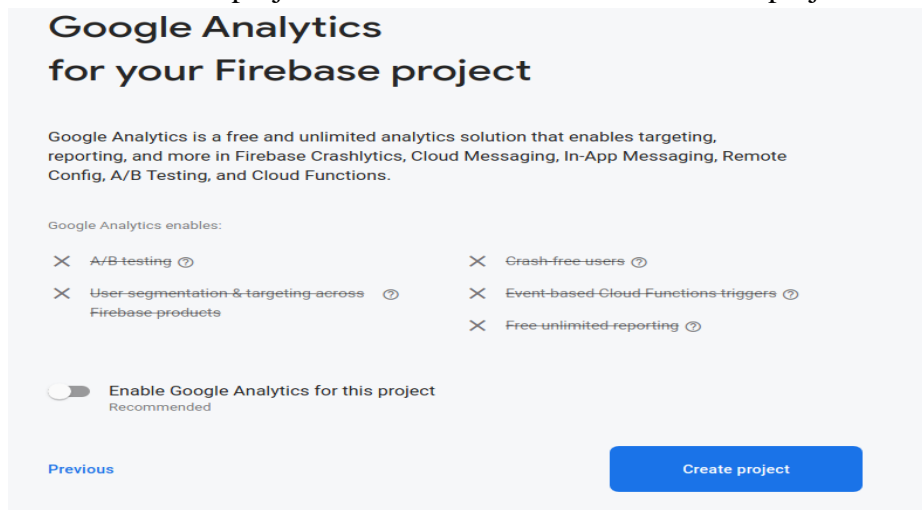
1. To create a project in Firebase, go to the Firebase Console.



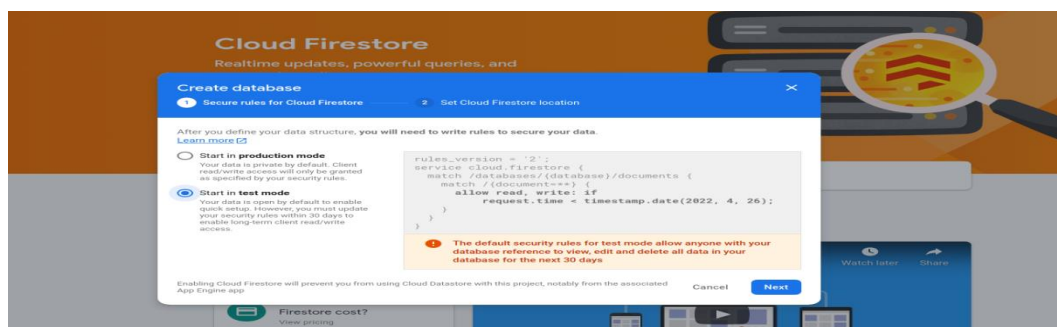
2. Give your project a name.

The image shows the 'Create a project (Step 1 of 3)' form in the Firebase Console. It has a light gray background. At the top, there's a close button (X) and the text 'Create a project (Step 1 of 3)'. The main heading reads 'Let's start with a name for your project®'. Below this, there's a label 'Project name' and a text input field containing 'CloudComputingLab'. Below the input field, there's a small text 'cloudcomputinglab-e3b6e'. At the bottom, there's a blue 'Continue' button.

- For the purposes of this practical, Firebase analytics may be disabled. Disable it and click on Create a project. Wait for Firebase to initialize the project.



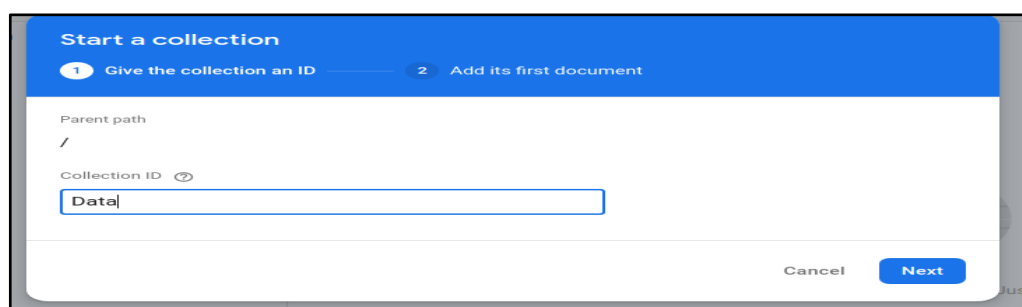
- Once the project is initialized, the standard Firebase project dashboard can be seen. Head over to the Firebase Firestore, under the Build section in the left panel.
- Create a Firestore database. Start the database in **test mode** and keep the default location.



CRUD operations on the database

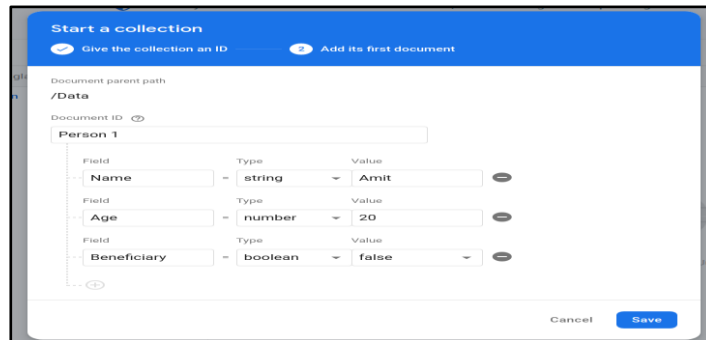
- Create operation

Click on *Start a collection*. Assign a Collection ID of your choice. This will act as a container for all of the documents created within.



Once a Collection is created, Firebase will give a prompt for creating the first document of the Collection. We can create a document ID with an arbitrary name or auto generate the ID. The fields of a document require a name, type and a default value. Fields are created according to the project's requirements.

Once the first document is created, more documents can be created with *Add Document*.



2. Update operation

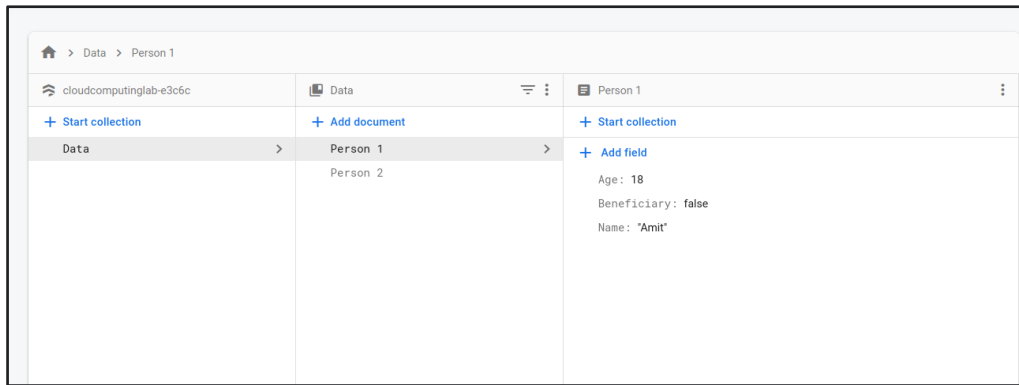
If any of the data needs to be updated, it can be done manually by clicking on the pencil icon (Edit Field) to the right of each field.



The update feature comes with a condition that we can't change the name of a field. But we can change the type or its value. Once we edit a field and click Update, it should update that field in real time.

3. Read operation

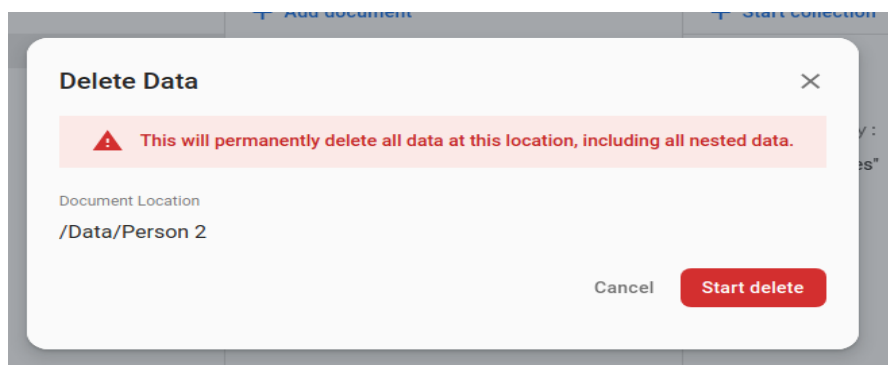
Once the database is integrated into an application, the application can use Firebase SDK's library methods to fetch data from a collection or a document. The application can filter a specific document or a specific field, as needed. If the database needs to be



viewed manually, it can be done by going to the Firestore, where we can get an overview of the data.

4. Delete operation

In Firebase, entire collections, documents or fields can be erased. To delete a document or a collection, click on the kebab menu (3 vertical dots) to the right of the document/collection. Click on “Delete document” (or collection). A prompt will be displayed which asks us to confirm the deletion by typing the entity’s name. This is not needed for a document.



Once deleted, the document/collection is found to be erased from the database.

Output:

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes the Azure logo, a search bar, and the user's profile. The main content area displays the overview of the 'krack (sql-database-server-crud/krack)' SQL database. The left sidebar contains a navigation menu with options like Overview, Activity log, Tags, Diagnose and solve problems, Query editor (preview), Settings, Compute + storage, Connection strings, Properties, Locks, Data management, Replicas, Sync to other databases, Integrations, and Server platform. The main content area is divided into two sections: 'Essentials' and 'Start working with your database'. The 'Essentials' section provides key information about the database, including its resource group, status, location, subscription, and tags. The 'Start working with your database' section offers guidance on how to connect to the database and start working with data, with links to 'Configure access', 'Connect to application', and 'Start developing'.

Microsoft Azure

Home > Microsoft.SQLDatabase.newDatabaseExistingServer_fc4d96171b424d79 | Overview >

krack (sql-database-server-crud/krack) SQL database

Search

Copy Restore Export Set server firewall Delete Connect with... Feedback

Overview

Activity log

Tags

Diagnose and solve problems

Query editor (preview)

Settings

Compute + storage

Connection strings

Properties

Locks

Data management

Replicas

Sync to other databases

Integrations

Azure Synapse Link

Stream analytics (preview)

Add Azure AI Search

Essentials

Resource group (move) : sql-database-server-crud

Status : Online

Location : East US

Subscription (move) : Azure for Students

Subscription ID : 9d479b5e-b083-477c-8246-f93445e3f730

Tags (edit) : Add tags

Server name : sql-database-server-crud.database.windows.net

Connection strings : Show database connection strings

Pricing tier : General Purpose - Serverless: Gen5, 1 vCore

Auto-pause delay : 1 hour

Earliest restore point : 2024-02-28 09:51 UTC

Getting started Monitoring Properties Features Notifications (0) Integrations Tutorials

Start working with your database

Connect to your database and start working with data with a few simple steps. [Learn more](#)

Configure access

Configure network access to your SQL server. [Learn more](#)

Connect to application

Use connection strings to connect to your SQL database from your applications and favorite tools.

Start developing

Work in your database by using tools to add, modify and query data. [Compare tools](#)

Configure

See connection strings

Open Azure Data Studio

Open in Visual Studio

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The left pane displays the 'Object Explorer' with the 'krack' database selected. The right pane shows the 'SQL Query2.sql' file with the following SQL script:

```
SELECT * FROM SalesLT.Address
INSERT INTO SalesLT.Address (AddressLine1, AddressLine2, City, StateProvince, CountryRegion, PostalCode, rowguid, ModifiedDate)
VALUES ('123 Main St', 'Apt 101', 'New York', 'NY', 'USA', '10001', NEWID(), GETDATE());
UPDATE SalesLT.Address
SET AddressLine1 = 'Boston', City = 'Angeles'
WHERE AddressId = 25;
DELETE FROM SalesLT.Address
WHERE City = 'Angeles';
SELECT * FROM SalesLT.Address WHERE AddressLine1 = '123 Main St'
```

The bottom pane shows the 'Results' tab with the following data:

| AddressID | AddressLine1 | AddressLine2 | City | StateProvince | CountryRegion | PostalCode | rowguid | ModifiedDate |
|-----------|--------------|------------------------|-------------|------------------|---------------|------------|--------------------------------------|-------------------------|
| 1 | 456 Elm St | NULL | Los Angeles | Washington | United States | 98011 | 268AF521-7607-4C78-9441-144FD139621A | 2006-07-01 00:00:00.000 |
| 2 | 111 | NULL | Angeles | Washington | United States | 98011 | 96183303-AC42-49C7-9456-F8670789E269 | 2007-04-01 00:00:00.000 |
| 3 | 25 | NULL | Angeles | Texas | United States | 75201 | C80F38D9-4B7D-4854-48D0-14A67494D3C8 | 2006-09-01 00:00:00.000 |
| 4 | 32 | 9228 Via Del Sol | Phoenix | Arizona | United States | 85004 | 12A5EE1F-AC1E-480B-9892-36976169774 | 2006-09-01 00:00:00.000 |
| 5 | 32 | 26910 Indeo Road | Montreal | Quebec | Canada | H1Y 2H5 | 84495F62-3A6B-4E7E-8B05-54F50CC268D2 | 2006-08-01 00:00:00.000 |
| 6 | 195 | 2681 Eagle Peak | Bellevue | Washington | United States | 98004 | 78CCF443-2268-46C2-8472-14C4C14E39C | 2006-09-01 00:00:00.000 |
| 7 | 297 | 7943 Walnut Ave | Perrinton | Washington | United States | 98055 | 53410D44-2778-4B1D-A599-9574662CCE9D | 2006-09-01 00:00:00.000 |
| 8 | 445 | 6388 Lake City Way | Burnaby | British Columbia | Canada | V5A 3A6 | 53572F25-9123-4A6B-A065-102FF35A16EE | 2006-09-01 00:00:00.000 |
| 9 | 446 | 52560 Free Street | Toronto | Ontario | Canada | M4B 1V7 | 801A1D0C-5125-486B-A0A4-CCB026C57CA4 | 2006-09-01 00:00:00.000 |
| 10 | 447 | 22580 Free Street | Toronto | Ontario | Canada | M4B 1V7 | 80CEE379-D8B8-4F34-A450-961083150CBF | 2006-09-01 00:00:00.000 |
| 11 | 448 | 2575 Bloor Street East | Toronto | Ontario | Canada | M4B 1V6 | 2DFD20AD-0928-4F34-A450-961083150CBF | 2007-08-01 00:00:00.000 |
| 12 | 449 | Station E | Chalk River | Ontario | Canada | K0J 1J0 | 8B5A7729-C875-4303-A607-7F9793B4094F | 2005-08-01 00:00:00.000 |
| 13 | 450 | 575 Rue St Amable | Quebec | Quebec | Canada | G1R | 5F3C345A-6475-41D5-B17B-D8B027733B81 | 2006-09-01 00:00:00.000 |
| 14 | 451 | 2512 4th Ave Sw | Calgary | Alberta | Canada | T2P 2G8 | 49644F1E-6F90-48D9-8DB8-90B15F0E7EC | 2006-12-01 00:00:00.000 |
| 15 | 452 | 55 Lakeshore Blvd East | Toronto | Ontario | Canada | M4B 1V6 | A358532F-0E00-49E6-B6B0-DE1099C43506 | 2005-09-01 00:00:00.000 |
| 16 | 453 | 6333 Cote Vertu | Montreal | Quebec | Canada | H1Y 2H7 | 35681F2-4D90-4522-8FA0-1058E64394D8 | 2007-09-01 00:00:00.000 |
| 17 | 454 | 3255 Front Street West | Toronto | Ontario | Canada | M4B 1V6 | EF4DC57D-8B8B-4074-85F0-AC8EF8D0BEC1 | 2007-08-01 00:00:00.000 |
| 18 | 455 | 2550 Signet Drive | Weston | Ontario | Canada | M9V 4W3 | 43D582CF-E95F-4861-B460-2DC7121410A6 | 2006-08-01 00:00:00.000 |

Query executed successfully.

Conclusion:

AWS is the largest cloud provider in the world, delivers multiple services, including virtualized servers, storage, serverless solutions, etc. Firebase is a backend as a service run by Google, delivers an end-to-end app development solution, and it's completely serverless. The right choice will depend upon several factors like the size of the workload, requirements in terms of server-level access, coding flexibility levels, and integration requirements.

Marks & Signature:

| R1 (3 Marks) | R2 (5 Marks) | R3 (4 Marks) | R4 (3 Marks) | Total (15 Marks) | Signature |
|-------------------------------|-------------------------------|-------------------------------|-------------------------------|-----------------------------------|------------------|
| | | | | | |