On-Premises Database Migration to Cloud

1. Overview

This document provides the technical steps to configure a virtual machine (VM) in Google Cloud and migrate a MySQL database from an on-premises environment to the cloud instance. The procedure includes VM setup, MySQL installation, network configuration, and secure database migration.

2. VM Configuration

2.1. VM Specifications

The virtual machine is provisioned with the following specifications:

Zone: `us-west1-a`

Machine Type: `E2-medium` (2 vCPUs, 4 GB RAM)

Storage: 10 GB persistent disk

Firewall Configuration: Allow both HTTP and HTTPS traffic

2.2. Firewall Configuration

First of all, ensured firewall rules are set to allow traffic on **HTTP** and **HTTPS** ports. Allowing traffic on HTTP (port 80) and HTTPS (port 443) in a firewall is essential for several reasons. First, it ensures users can access a website without interruptions. Second, enabling these ports improves user experience. Additionally, using HTTPS protects sensitive data through encryption, safeguarding it from attacks.

Secondly, added a specific rule to allow traffic on port 3306 for MySQL access. Allowing traffic on port 3306 is essential because it is the default port used by MySQL for client-server communication. By configuring the firewall to permit access through this port, authorized clients can connect to the MySQL server running on the cloud VM, enabling essential database operations such as data retrieval, insertion, and updates. This configuration enhances security by limiting access to the necessary service while minimizing exposure to unauthorized connections.

Name	Type	Targets	Filters	Protocols / ports	Action	Priority	Network ↑	Logs
allow-mysql-3306	Ingress	Apply to all	IP ranges:	tcp:3306	Allow	1000	default	Off

3. MySQL Installation and Configuration

3.1.MySQL Installation

As it is a fresh setup of vm, so there was no pre-installed MySQL environment. First, Install MySQL on the VM using the package manager of the selected operating system. For Ubuntu, the following command was used:

sudo apt-get update

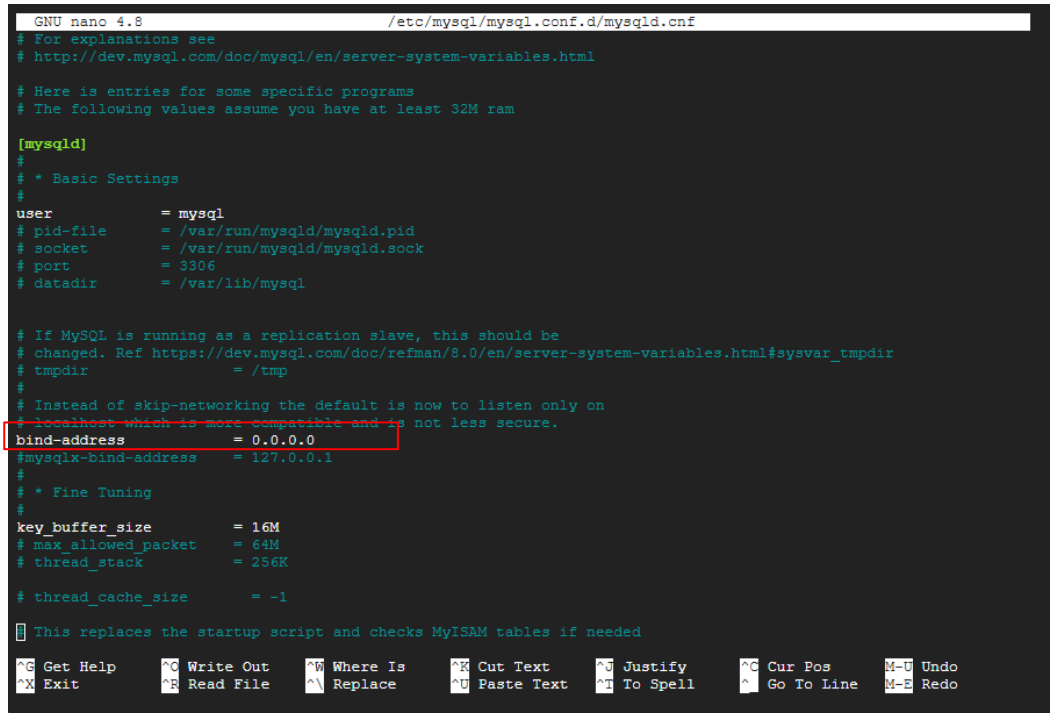
This command will install the MySQL in the on the VM instance.

wget https://downloads.mysql.com/archives/get/p/23/file/mysql-server_8.0.37-1ubuntu24.04_amd64.deb-bundle.tar

3.2.MySQL Configuration

To enable remote access to MySQL, the **bind-address** parameter in the configuration file must be set to **0.0.0.0.** This allows the MySQL server to accept connections from any IP address, rather than just localhost. This change is made in the MySQL configuration file located at **/etc/mysql/mysql.conf.d/mysqd.cnf** , which controls how the MySQL server listens for incoming connections. The following command has been used to edit the file

sudo nano /etc/mysql/mysql.conf.d/mysqd.cnf.



After updating this parameter, the MySQL service must be restarted for the changes to take effect by using the following command

sudo systemctl restart mysql

3.3. STATIC IP CONFIGURATION

Reserve or assign a static IP, ensuring that the IP address remains unchanged even after the VM is restarted. This configuration provides consistent connectivity for the VM, allowing reliable access for applications and services that depend on it.

Name	IP address	Access type	Region	Type ↓	Version	In use by	Subnetwork	VPC Network	Network Tier
migration	35.212.217.246	External	us-west1	Static	IPv4	VM instance migration0n-prmesis (Zone us-west1-a)			Standard

3.4. CLOUD AND ON-PREMISES DATABASE CONNECTIVITY

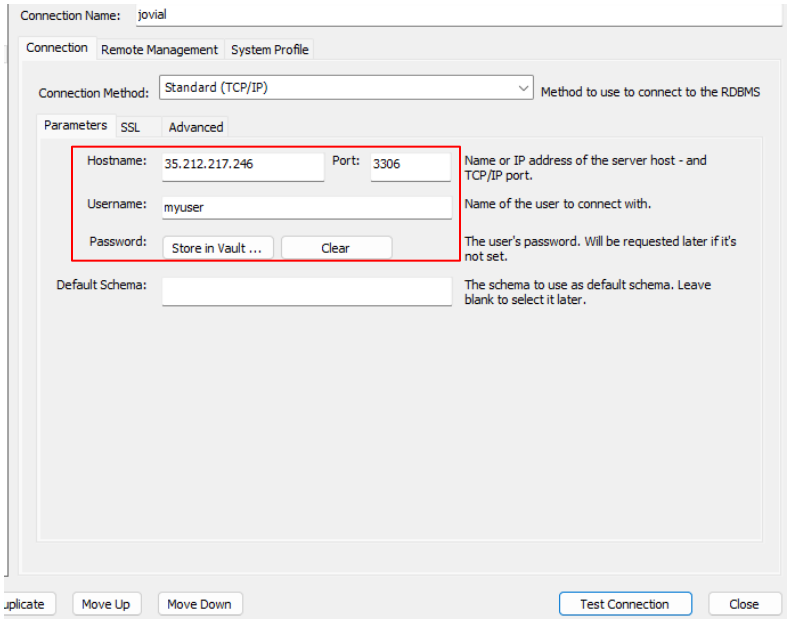
3.4.1. Connection to workbench

Connecting to MySQL Workbench to access a VM in Google Cloud (GCP) is necessary for managing the MySQL database hosted on the VM. MySQL Workbench provides a graphical user interface (GUI) for performing various database operations in on-premises device. Here are the steps were followed:

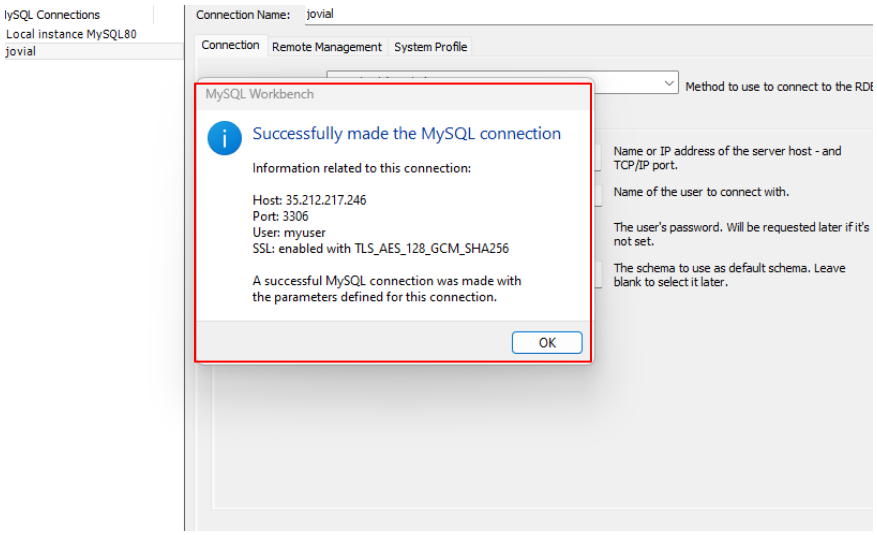
Step 1 : Installed the **Mysql Workbench** on the device:

Step 2 : Opened the **Mysql Workbench** and opened the new connection profile.

Step 3: For MySQL Workbench connectivity to a MySQL server on a cloud VM, the **hostname** and **port** should be the Cloud VM's external IP address and port (typically 3306). The **username** and **password** will be the same as what was set up in the MySQL instance on the VM.

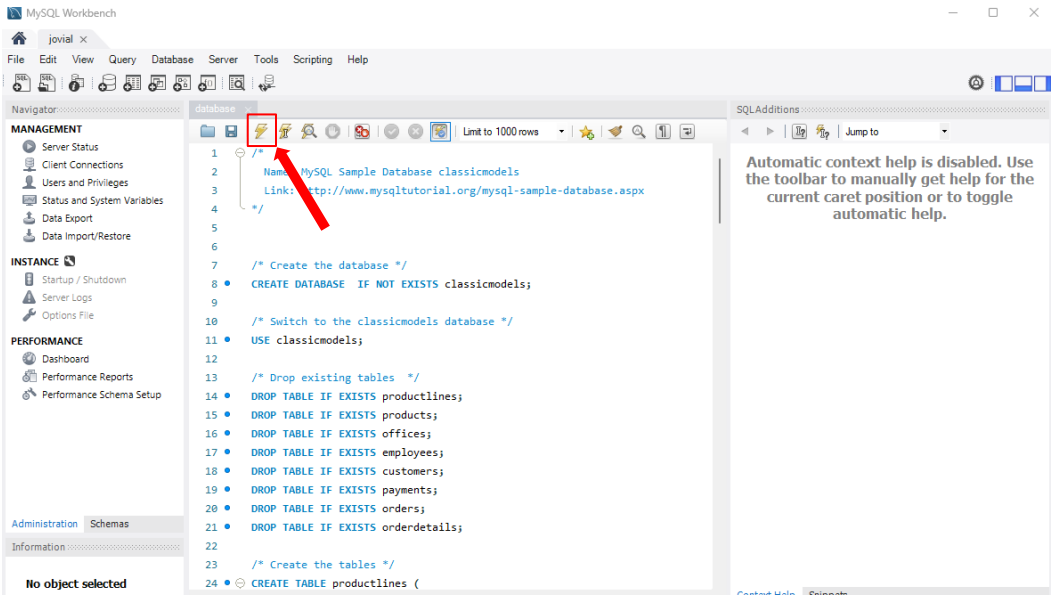


Step 4 : Afterward, the connection was successfully established.



4. DATABASE MIGRATION

First, a new database is created in MySQL Workbench by executing the appropriate SQL query. Once the query is written and executed, MySQL Workbench processes the command and automatically saves the database to the MySQL instance hosted on the Virtual Machine (VM). Here is the execution process by clicked on the execution button;



The data and schema changes are immediately applied to the MySQL server running on the VM, ensuring that the database is persistently stored and accessible for future queries or operations. Below are screenshots that illustrate the process of creating a new database in MySQL Workbench. The first screenshot shows the initial state before the database is created.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.00 sec)

mysql>
```

and the second screenshot displays the confirmation after executing the SQL query, where the newly created database appears in the schema list.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| classicmodels      |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.01 sec)

mysql>
```

Before executing the SQL query to create a new database in MySQL Workbench, it is essential to ensure that both the MySQL server Virtual Machine (VM) and the client VM running MySQL Workbench are powered on.

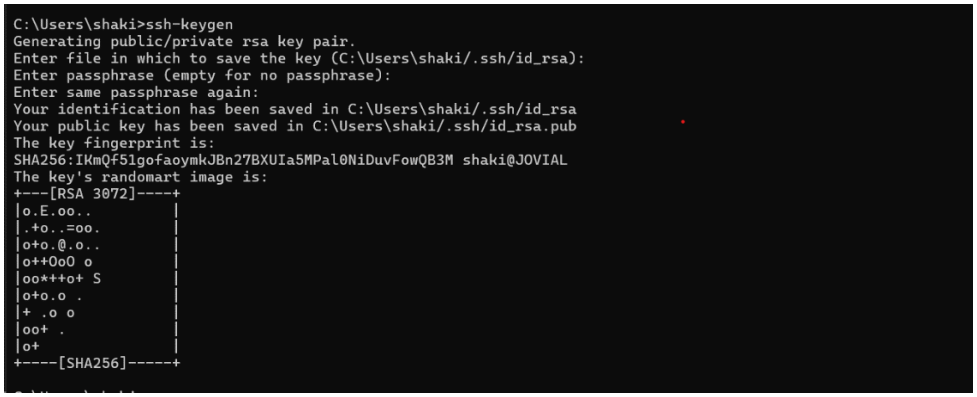
5. DATABASE BACKUP CREATION (ON-PREMISES[MANUAL])

5.1 CONNECTION TO ON-PREMISES DEVICE'S POWERSHELL

Controlling Google Cloud Platform (GCP) SSH from a device's command-line interface (CLI) is essential for direct and secure access to virtual machines (VMs), enabling faster management of resources without browsing the GCP console and clicking the SSH. Additionally, CLI access facilitates remote management, making it possible to perform system updates and application deployments from on-premises device.

Step 1: Opened PowerShell and ran the following command to generate an SSH key pair. This generated a key in that folder

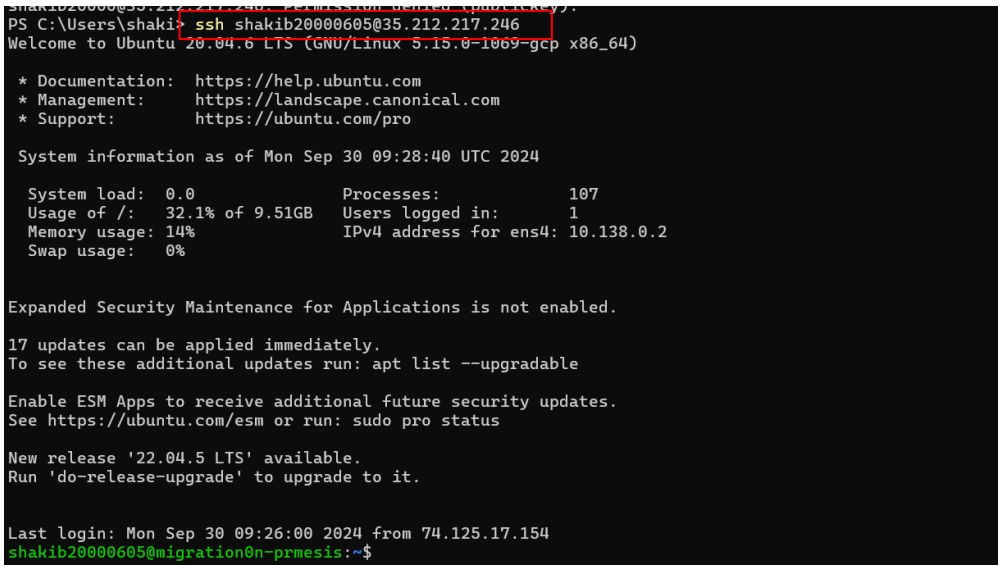
ssh-kygen



Step 2: To connect to the GCP VM, used the following command, replacing with GCP's username and with the external IP address of the VM:

ssh [username]@[ip]

username will be the user's username in the cloud VM and the ip address will be the VM's external IP address



Once executed, the connection will establish, allowing access to the GCP VM shell directly from PowerShell.

5.2 BACKUP EXISTING DATABASES TO THE CLOUD

For database backup, the **mysqldump** command was utilized. MySQL dump is primarily designed to create backups of databases, ensuring that data can be restored in the event of loss or corruption. It facilitates the migration of data between different MySQL servers and allows for selective backups of specific tables or entire databases. Here are the following steps which have been used:

Step 1: Use the **mysqldump** command to create a dump file of the desired database. For example, to back up a database named classicmodels, run. This is the following command:

mysqldump -u [username(database)] -p [database_name] > [path and name of the database]

```
C:\Users\shaki>mysqldump -u root -p classicmodels > C:\Users\shaki\Downloads\mysqlsampledatabase_backup.sql
Enter password: ****

C:\Users\shaki>
```

Step 2: Generated dump file to the cloud environment. This was done using secure copy (SCP). For example, using SCP. SCP(Secure Copy Protocol) provides a secure, efficient, and user-friendly method for transferring files, making it an essential tool for system administrators, developers, and anyone who needs to manage files across different machines securely. Here is the following command

scp [backup database] [username]@[cloud-vm-ip]:[vm path destination]

```
C:\Users\shaki>scp C:\Users\shaki\Downloads\mysqlsampledatabase_backup.sql shakib20000605@35.212.217.246:/home/shakib200
00605
mysqlsampledatabase_backup.sql                                100%  189KB 108.3KB/s   00:01

C:\Users\shaki>
```

Step 3: After transferring, the backup file appears in the cloud storage or VM directory as sql file, ready for import into the MySQL database.

```
shakib20000605@migration0n-prmesis:~$ ls
mysqlsampledatabase_backup.sql  snap
shakib20000605@migration0n-prmesis:~$
```

Step 4: To import the SQL dump file into the MySQL database, the executed following command in the cloud VM shell:

sudo mysql -u [username] -p [database_name] < [destination of the database]

```
shakib20000605@migration0n-prmesis:~$ sudo mysql -u root -p classicmodels_backup < /home/shakib20000605/mysqlsampledatab
ase_backup.sql
Enter password:
```

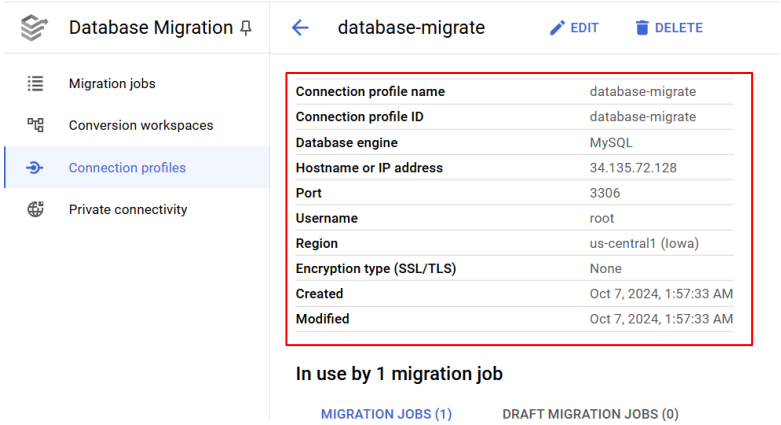
Step 5: The file has been successfully migrated to the database .

```
shakib20000605@migration0n-prmesis:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 28
Server version: 8.0.39-0ubuntu0.20.04.1 (Ubuntu)

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+----------------------+
| Database             |
+----------------------+
| classicmodels        |
| classicmodels_backup |
| information_schema   |
| mysql                |
| performance_schema   |
| sys                  |
+----------------------+
6 rows in set (0.00 sec)

mysql>
```

6 DATABASE BACKUP CREATION (ON-PREMISES [USING CLOUD SQL])

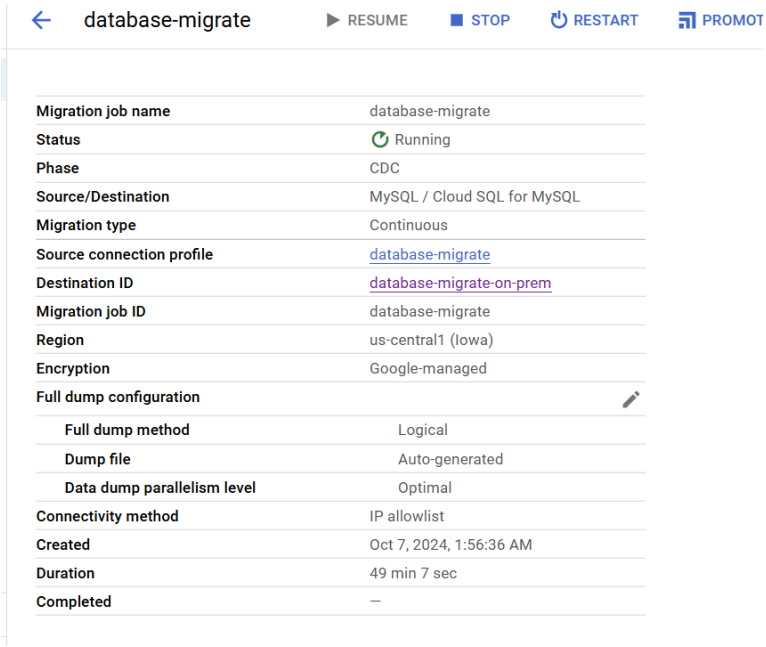
Step 1 : Create Connection Profiles

First, set up connection profiles for both the VM MySQL server and the Cloud SQL destination. These profiles hold the credentials and network info to connect them.



Step 2 : Start Migration Job

Begin a migration job and, during this step, configure a Cloud SQL instance as a read replica to receive the data.



Step 3 : Set Up Cloud SQL Instance

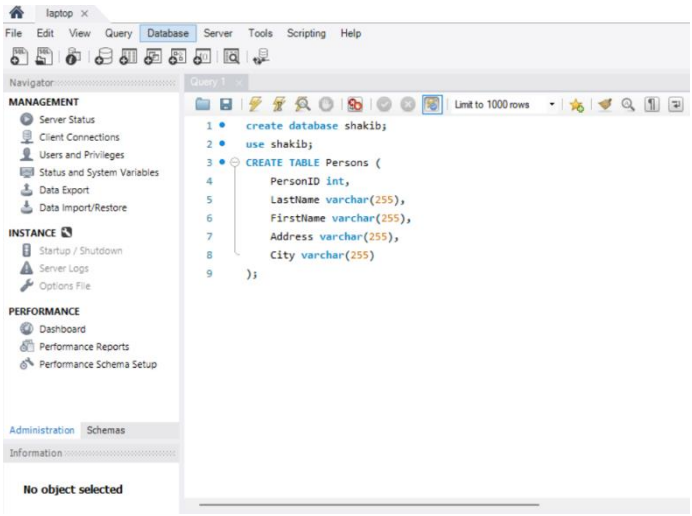
A Cloud SQL instance is created as part of the migration job to serve as the destination.

Step 4: Configure IP for Cloud SQL Access

In the VM's MySQL server, add the outgoing IP of Cloud SQL to its vm's MySQL server settings to enable access between them.

After these steps, the VM MySQL server and Cloud SQL instance can connect, allowing smooth data migration.

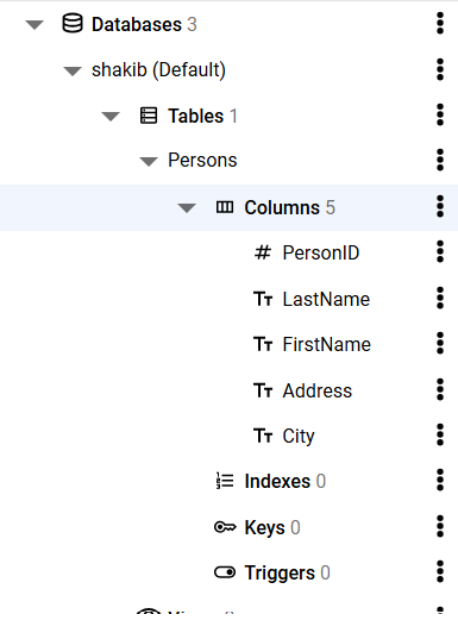
On-Prem workbench:



VM instance:

```
shakib20000605@migration:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 39
Server version: 8.0.37 MySQL Community Server - GPL

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| jovial             |
| mysql              |
| orb                |
| performance_schema |
| shakib             |
| sys                |
+--------------------+
7 rows in set (0.01 sec)

mysql>
```

CloudSql:



7 CONCLUSION

This documentation provides the detailed steps for setting up a VM in Google Cloud, configuring MySQL for remote access, and securely migrating a MySQL database from an on-premises server to the cloud instance. Follow these steps to ensure a successful and secure database migration. It is recommended to set up automated monitoring of the VM and establish a regular backup schedule for the MySQL database in the cloud environment.