Conestoga College

Cloud Development and Operations

Sem 2

Database Automation

Assignment - 1

Name: Durlabh Tilavat

Professor: Rich Hildred

Submission Date: 2025/05/16

Question 1: Understanding Database Automation (10 Points)

1.1) Explanation of Database Automation:

Database automation is the use of technology (scripts, tools, or platforms) to automate routine database processes that would otherwise require manual involvement. Backups, updates, data migrations, monitoring, and performance optimization are among the duties performed. Automation is critical in modern data management because firms deal with large datasets that demand real-time processing and availability.

Automation improves efficiency, data integrity, and security while minimizing human mistake. Automated systems are used by financial businesses to run end-of-day reports, backup critical data, and enforce access rules. Healthcare systems automate backups and maintain compliance with data preservation standards outlined in laws such as HIPAA.

1.2) Benefits of Automating Database Tasks:

Reduced Errors: Automation eliminates the possibility of manual errors. For example, LinkedIn employs automation to manage schema changes across its distributed databases without losing data.

Improved Reliability: Automated failover and monitoring systems maintain availability. To ensure uptime, Amazon RDS automatically handles backups and patches.

Faster Deployments: DevOps pipelines provide faster deployments, including database modifications. Etsy leverages Flyway and Liquibase to deploy schema updates with application code.

Cost efficiency: eliminates the need for big teams to accomplish activities manually. Companies such as Dropbox employ automated provisioning and maintenance scripts to save infrastructure overhead.

Question 2: Scripting for Database Automation (10 Points)

2.1) backup_script.py - Python Script for MySQL Backup

Explanation:
This script uses Python’s subprocess module to run a mysqldump command. It ensures that the filename is unique using the current timestamp. It checks the return code to verify successful execution and handles errors gracefully.

```
import subprocess

import time

import os


# Define backup parameters

user = "root"

password = "your_password"

database = "your_database_name"

backup_dir = "backups"


# Ensure backup directory exists

os.makedirs(backup_dir, exist_ok=True)


# Create a unique filename with timestamp

timestamp = time.strftime("%Y%m%d-%H%M%S")

filename = f"{database}_backup_{timestamp}.sql"

filepath = os.path.join(backup_dir, filename)


# Run the backup using mysqldump

try:

    result = subprocess.run(

        ["mysqldump", "-u", user, f"-p{password}", database],

        stdout=open(filepath, "w"),

        stderr=subprocess.PIPE

    )

    if result.returncode == 0:

        print(f"Backup successful: {filepath}")

    else:
        print(f"Backup failed. Error: {result.stderr.decode()}")
except Exception as e:
    print(f"An exception occurred: {e}")
```

2.2) deploy_changes_script.py - Python Script for Deploying Changes

Explanation:
This script connects to a MySQL database using mysql.connector, creates a new table (employees) if it doesn’t already exist, commits the changes, and handles errors using try-except blocks.

```
import mysql.connector
from mysql.connector import Error


try:
    # Connect to the database
    connection = mysql.connector.connect(
        host='localhost',
        user='root',
        password='your_password',
        database='your_database_name'
    )

    if connection.is_connected():
        cursor = connection.cursor()

        # SQL command to add a new table
        create_table_query = """
        CREATE TABLE IF NOT EXISTS employees (
            id INT AUTO_INCREMENT PRIMARY KEY,
            name VARCHAR(100),
            department VARCHAR(50),
            hire_date DATE
        );
        """
        cursor.execute(create_table_query)
        connection.commit()
        print("Table 'employees' has been created or already exists.")

except Error as e:
    print(f"Error: {e}")

finally:
    if connection.is_connected():
        cursor.close()
        connection.close()
        print("Database connection closed.")
```

Reference

MySQL :: MySQL Documentation. (n.d.). https://dev.mysql.com/doc/

Meta. (2025, May 15). *Engineering at Meta*. Engineering at Meta. https://engineering.fb.com/