Automating Cloud SQL Backups with Cloud Scheduler, Pub/Sub, and Cloud Functions.

Automating backups for Cloud SQL databases is crucial for ensuring data safety and compliance. This writeup outlines a workflow using Cloud Scheduler, Pub/Sub, and Cloud Functions to schedule manual backups.

Workflow Overview

Cloud Scheduler:

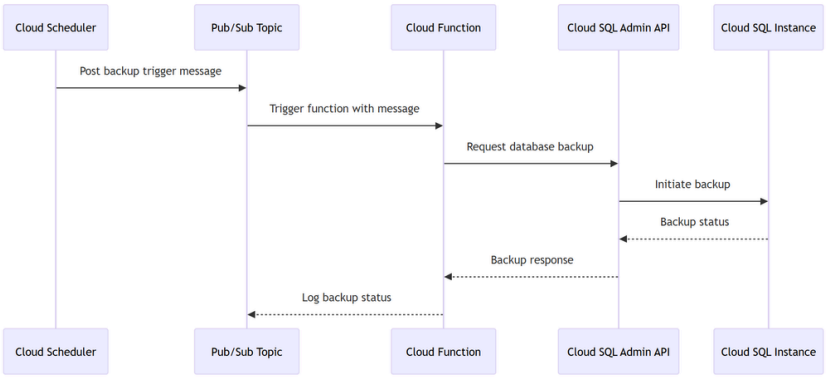
Create a Cloud Scheduler job that posts a backup trigger message to a Pub/Sub topic at a scheduled time. The message includes details about the Cloud SQL instance and project ID.

Pub/Sub:

The Pub/Sub topic receives the message from Cloud Scheduler and triggers a Cloud Function.

Cloud Function:

The Cloud Function uses the Cloud SQL Admin API to initiate a backup of the specified Cloud SQL instance. It logs the backup status to ensure monitoring and debugging.



Here's a comprehensive list of tasks to set up a POC workflow for automating Cloud SQL backups, including a sample application setup and end-to-end validation.

Tasks to Set Up the Workflow

1. **Create a Pub/Sub Topic:** Create a new Pub/Sub topic named backup-trigger. This topic will receive messages from Cloud Scheduler to trigger backups.
2. **Set Up Cloud Scheduler:** Create a Cloud Scheduler job that posts a message to the backup-trigger Pub/Sub topic at a specified interval (e.g., daily).
3. **Create a Cloud Function:** Develop a Cloud Function that listens to the backup-trigger Pub/Sub topic.
   This function will request a database backup using the Cloud SQL Admin API.
4. **Create a Sample Web Application:** Develop a simple web application (e.g., using Flask or Django) hosted on Cloud Run. This app will interact with a Cloud SQL database.
5. **Set Up Cloud SQL Instance:** Create a Cloud SQL instance with MySQL or PostgreSQL.
6. **Populate Sample Data:** Insert sample data into the Cloud SQL database. This data will be used for testing the backup and restore process.
7. **End-to-End Validation**
   a. **Restore Backup to a New Instance:** This ensures that backups are valid and can be restored successfully.
   b. **Verify Data Integrity:** Run data integrity checks to ensure data consistency between the original and restored instances.
   c. **Test Application Functionality:** Test the sample application against the restored database to ensure functionality.