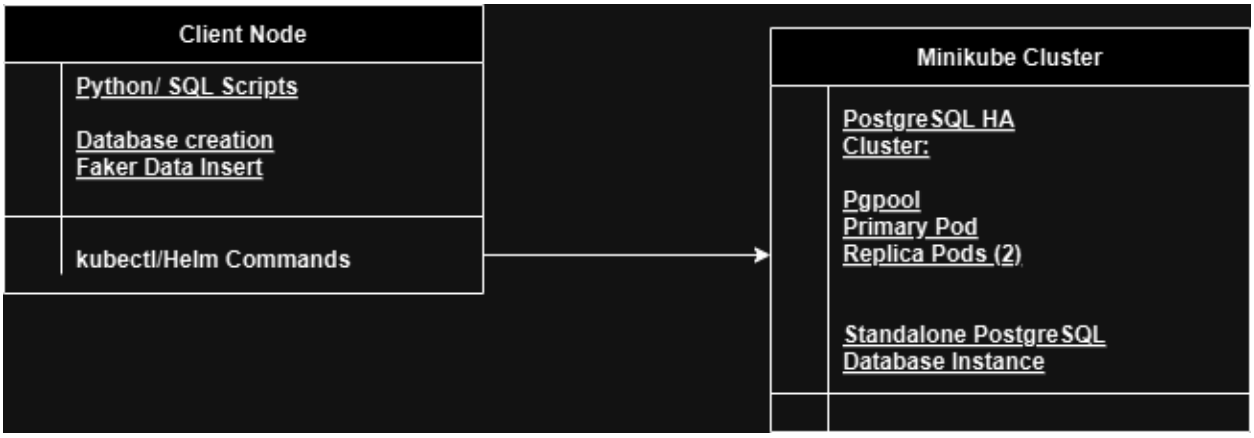

Architectural Diagram

The architecture comprises:

1. **Minikube Kubernetes Cluster**: Hosting a PostgreSQL HA cluster and a standalone PostgreSQL instance.
2. **PostgreSQL HA Cluster**:
    - Three PostgreSQL Pods (primary and two replicas).
    - Pgpool load balancer for distributing traffic.
3. **Standalone PostgreSQL**: Used for asynchronous replication.
4. **Client Machine**:
    - Python and SQL scripts for database creation and populating tables using Faker.
    - Interaction with the cluster using kubectl and Helm.

Diagram Description

- **Components**:
    - **Client Node**:
        - Python script connects to the PostgreSQL HA Cluster via Pgpool or directly to the standalone PostgreSQL.
    - **Minikube Cluster**:
        - PostgreSQL HA Cluster:
            - One primary Pod.
            - Two replica Pods.
            - Pgpool service.
        - Standalone PostgreSQL instance.
    - **Network Communication**:
        - Asynchronous replication setup between HA cluster and standalone PostgreSQL.
        - External traffic routed through Pgpool.



Architectural Diagram

1. Overview

This solution deploys a PostgreSQL HA cluster with load balancing, a standalone PostgreSQL database for replication, and scripts to manage database creation and data population.

---

2. Prerequisites

- Minikube installed and running.
- Helm installed on the local machine.
- Python installed with psycopg2 and Faker libraries.
- GitHub account for storing scripts.

---

3. Deployment Steps

A. Minikube Kubernetes Cluster

1. Start Minikube with required resources:
2. minikube start --cpus=2 --memory=2048 --disk-size=20g

B. Deploy PostgreSQL HA Cluster

1. Add Bitnami Helm repository:
2. helm repo add bitnami https://charts.bitnami.com/bitnami
3. Deploy the PostgreSQL HA cluster:
4. helm install postgresql-cluster bitnami/postgresql-ha

C. Deploy Standalone PostgreSQL Database

1. Deploy the standalone PostgreSQL instance:

2. helm install postgresql-standalone bitnami/postgresql

D. Set Up Asynchronous Replication

1. Configure the HA cluster primary Pod for replication:
    - Update postgresql.conf and pg_hba.conf.
2. Run pg_basebackup on the standalone instance.

E. Script to Create Database and Populate Data

1. Create the database with two related tables:
    - SQL script for database setup.
2. Insert 100,000 records using Faker:
    - Python script for data population.

F. Test the Setup

1. Validate replication by checking data synchronization between HA cluster and standalone PostgreSQL.

G. Push to GitHub

1. Push all scripts and configurations to a GitHub repository.

---

4. Technologies Used

- **Minikube**: Local Kubernetes cluster for resource-constrained environments.
- **Helm**: Tool for managing Kubernetes applications.
- **PostgreSQL**: Database system used for HA and standalone setups.
- **Python**: Language for scripting database operations.
- **Faker**: Library for generating fake data.