**Sakina Banu- GDE-day 11 Homework.**

**1. SQL vs NoSQL Databases in Google Cloud Platform (GCP)**

SQL databases (like Cloud SQL) and NoSQL databases (like Firestore) are both used in Google Cloud, but they work differently.

**Comparison Table:**

| **Feature** | **SQL (Cloud SQL)** | **NoSQL (Firestore)** |
| --- | --- | --- |
| **Data Model** | Tables with rows and columns | Documents in key-value or JSON format |
| **Schema** | Fixed/predefined structure | Flexible schema that adapts easily |
| **Scalability** | Vertical (more power to one server) | Horizontal (add more servers) |
| **Use Cases** | Banking, inventory, ERP systems | Chat apps, real-time dashboards |

**Advantages of SQL:**

* Good for structured data.
* Supports complex queries.
* Strong data consistency.

**Disadvantages of SQL:**

* Less flexible.
* Harder to scale.

**Advantages of NoSQL:**

* Scales easily.
* Flexible data models.
* Works well with real-time data.

**Disadvantages of NoSQL:**

* Less strict consistency.
* Limited support for complex queries.

**2. Data Consistency and Integrity Differences**

**SQL (Cloud SQL)** uses ACID principles, which means:

* Data is always correct and consistent.
* Useful for things like bank transactions.

**NoSQL (Firestore)** uses BASE principles:

* Fast and available.
* Sometimes data takes a few seconds to update everywhere.

**Example:**

* Use SQL for financial apps.
* Use NoSQL for social media apps where speed matters more than perfect accuracy.

**3. Vertical vs Horizontal Scaling in GCP**

* **Vertical Scaling:** Add more power (CPU, RAM) to a single machine.
  + Example: Cloud SQL.
* **Horizontal Scaling:** Add more servers to handle more users.
  + Example: Firestore, Cloud Spanner.

**4. Pros and Cons of Scaling Types**

| **Scaling Type** | **Pros** | **Cons** |
| --- | --- | --- |
| Vertical Scaling | Easy to apply | Costly, limited growth, downtime |
| Horizontal Scaling | More scalable, avoids single failure | Needs redesign, more complex setup |

**5. Scaling Cloud SQL in GCP**

**Scenario:** Cloud SQL is slow or out of resources.

**Option 1: Vertical Scaling**

* Upgrade the machine to a higher type.
* *Impact:* Fast solution but may be expensive.

**Option 2: Add Read Replicas**

* Copies read-only data to another server.
* *Impact:* Improves speed for users; good for high traffic apps.

**6. Creating PostgreSQL in GCP – Step-by-Step**

**Steps:**

1. Go to GCP Console → SQL section.
2. Click **Create Instance** → Choose **PostgreSQL**.
3. Set instance ID, password, and region.
4. Pick machine type and storage.
5. Create instance → Go to **Users** and **Databases** to add users and DBs.

**7. PostgreSQL Configuration Options**

| **Option** | **Why It’s Important** |
| --- | --- |
| **Region** | Lower delay if it's close to your users. |
| **Machine Type** | Impacts speed and performance. |
| **Storage** | SSD is fast but expensive; HDD is cheaper. |
| **Backups** | Important for restoring if anything goes wrong. |
| **High Availability** | Keeps the app running if one server fails. |