**Cloud Spanner:**

* Cloud Spanner is a fully managed, scalable, relational database service delivers high-performance transactions and strong consistency for regional and global (multi-region level) application data.
* Spanner is a managed SQL compliant database. It's not MySQL or postgres so it can't be used as a drop-in replacement for those engines.
* Cloud Spanner is the first scalable, globally distributed, and strongly consistent database service built for the cloud specifically to combine the benefits of relational database structure with non-relational horizontal scale.
* Fully managed, Mission critical application and use when Data volume > 2 TB
* Highly scalable and Petabyte scale
* Just like Cloud SQL, must define schema database i.e., Datatype for each column of each table must be defined.
* Spanner = Cloud SQL + Horizontal Scalable
* Data export, cannot export with gcloud only by using Cloud Console or Cloud Dataflow Job
* Spanner will replicate your data across multiple regions with 99.999% availability, no planned downtime, and enterprise-grade security.
* Is it really achievable? Yes, Spanner is highly available with the right configuration. It

It uses CAP theorem.

* The CAP theorem formulated by Eric Brewer states that in a distributed system that stores data there are 3 fundamental principles:
  + Consistency
    - Consistency means database transactions must only change data according to specific rules.
    - And in the context of Spanner, it refers to the strong consistency

of a right transaction taking effect before that data can be read from anywhere in the system.

* + Availability
    - The system will always be available to serve queries.
  + Partition tolerance.
    - Partition tolerance, means that the distributed system needs to be tolerant of the loss of any partition of its total system.
    - More simply it needs to tolerate failures if a part of it goes offline unexpectedly.



**Features Of Cloud Spanner:**

* Global Scale - Horizontally scalable across rows, regions, and continents, from 1 to hundreds or thousands of nodes.
* Fully Managed - Ease of deployment at every scale and every stage. Synchronous replication and maintenance are automatic and built in.
* Relational Semantics - Everything you would expect from a relational database—schemas, ACID transactions, and SQL queries.
* Multi-Language Support - Client libraries in **C#, Go, Java, Node.js, PHP, Python, and Ruby**.
  + JDBC driver for connectivity with popular third-party tools.
* Transactional Consistency - Purpose-built for external, strong, global transactional consistency.
* Enterprise Grade Security - Data-layer encryption, IAM integration for access and controls, and audit logging.
* Highly Available
* This is different from MySQL and quite familiar to oraclesql.

**Implementation:**

* **Create Spanner Instance**
* **Create database hospital\_db**
* **Create 2 table**
  + **Patient**
    - **Patient id**
    - **Patient name**
  + **Appointment**
    - **Appointment id**
    - **Patient id**
    - **Physician**

1. **Enable Cloud Spanner API**
2. **Create Patient table**

CREATE TABLE Patient (

patient\_id INT64,

patient\_name STRING(100)

) PRIMARY KEY(patient\_id );

CREATE TABLE Patient (

  patient\_id INT64,

  patient\_name STRING(100),

) PRIMARY KEY(patient\_id);

1. **Create Appointment table**

CREATE TABLE Appointment (

appointment\_id int64,

patient\_id INT64,

physician STRING(100)

) PRIMARY KEY(patient\_id, appointment\_id ),

INTERLEAVE IN PARENT PATIENT ON DELETE CASCADE;

CREATE TABLE Appointment (

  appointment\_id INT64,

  patient\_id INT64,

  physician STRING(100),

) PRIMARY KEY(patient\_id, appointment\_id),

  INTERLEAVE IN PARENT Patient ON DELETE CASCADE;

1. **Insert values into table**

INSERT Patient (patient\_id, patient\_name)  
VALUES (1, 'Marc'),(2, 'Richards'), (3, ‘swati’), (4, ‘dell’), (5, ‘joey’)

INSERT INTO

  Patient (patient\_id,

    patient\_name)

VALUES

  (1,"Siva"),(2, "khaula"), (3,"vijay");

INSERT Appointment(appointment\_id,patient\_id, physician)

VALUES (100,1,’Phil’),(101,3,’Marc’),(102,5,’vijay’),(104,2,’ajith’)

INSERT INTO

  Appointment (appointment\_id,

    patient\_id,

    physician)

VALUES (100,1,'neethu'),(101,3,'vignesh'),(103, 1,'neethu');

SELECT

  \*

FROM

  Appointment;

WHERE

  patient\_id=<patient\_id>

  AND appointment\_id=<appointment\_id>;

**Resources:**

* <https://cloud.google.com/spanner/docs>
* <https://cloud.google.com/spanner/docs/concepts>
* <https://codelabs.developers.google.com/codelabs/cloud-spanner-first-db#0>
* <https://github.com/googleapis/python-spanner/tree/main/samples/samples>
* <https://github.com/search?q=org%3Agoogleapis+spanner>
* <https://cloud.google.com/spanner/docs/sql-best-practices>
* <https://en.wikipedia.org/wiki/CAP_theorem>
* <https://en.wikipedia.org/wiki/Quorum_(distributed_computing)>
* <https://cloud.google.com/spanner/docs/reference/standard-sql/dml-syntax?_ga=2.158034623.-46277215.1646062875>