**Setup**

In the Google Cloud Console, on the project selector page, select or create a Google Cloud project. Make sure that billing is enabled for your Cloud project. Learn how to check if billing is enabled on a project. You will use Cloud Shell, a command-line environment running in Google Cloud that comes preloaded with bq. From the Cloud Console, click Activate Cloud Shell on the top right corner. Enable necessary APIs for this implementation if you haven't already: Cloud Spanner API. To do this, navigate to the Cloud Shell Terminal and enter the following command: gcloud services enable spanner.googleapis.com

**Create Spanner Instance, Database and Table**

Create an instance named "spanner-vertex" and a database named "spanner-vertex-embeddings". Create a table using the

DDL: **CREATE TABLE patents\_data ( id string(25), type string(25), number string(20), country string(2), date string(20), abstract string(300000), title string(100000),kind string(5), num\_claims numeric, filename string(100), withdrawn numeric, ) PRIMARY KEY (id);**

## Prepare & Load Patent Data

For building the Patent Search App, we will use the Patent Published dataset in BigQuery

## Create Remote Model for Gemini 1.0 Pro

We will convert the patent abstracts into a consolidated summary consisting of a title and keywords. For this we will use the Gemini 1.0 Pro model from Vertex AI remotely from Spanner. Run the following DDL from Spanned Studio

Editor: **CREATE MODEL gemini\_pro\_model INPUT( prompt STRING(MAX), ) OUTPUT( content STRING(MAX), ) REMOTE OPTIONS ( endpoint = '//aiplatform.googleapis.com/projects/<<YOUR\_PROJECT\_ID>>/locations/us-central1/publishers/google/models/gemini-pro', default\_batch\_size = 1 );**

## Create Remote Model for Text Embeddings

We will convert the Gemini 1.0 Pro model's response (that is the consolidated summary consisting of a title and keywords) into embeddings for performing the match. For this we will use the Text Embedding Gecko 003 model from Vertex AI

remotely from Spanner. Run the following DDL from Spanned Studio

Editor: **CREATE MODEL text\_embeddings INPUT(content STRING(MAX)) OUTPUT( embeddings STRUCT< statistics STRUCT<truncated BOOL, token\_count FLOAT64>, values ARRAY> ) REMOTE OPTIONS ( endpoint = '//aiplatform.googleapis.com/projects/<<YOUR\_PROJECT\_ID>>/locations/us-central1/publishers/google/models/textembedding-gecko@003');**

**project id 🡪your project id created in googlecloud**

## Generate Embeddings for the Generated Insights

In this step, we will generate embeddings for the generated insights. Let's run the following DDL and DML in the Spanner Studio

Editor: --**DDL CREATE TABLE patents\_data\_embeddings (id string(100), patents\_embeddings ARRAY) PRIMARY KEY (id);**

--**DML INSERT INTO patents\_data\_embeddings (id, patents\_embeddings) SELECT id, embeddings.values as patents\_embeddings FROM ML.PREDICT(MODEL text\_embeddings, (SELECT id, gemini\_response as content FROM patents\_data\_gemini)); Check the results using the following query: select title, abstract, b.patents\_embeddings from patents\_data a inner join patents\_data\_embeddings b on a.id = b.id;**

## Similarity Vector Search

Now that the embeddings are created for the generative insights, let's create embeddings for the search text and prepare it for Vector Search. This is the search text that will be entered into the search application (by the user).  Please note we haven't built the web app here, you can build a simple search app to visualize this use case with the data layer we are setting up here by following the web app example in this repo.  For this implementation, we will be using the K-Nearest Neighbors Similarity Search capability. Remember you can also directly integrate with Vertex AI Vector Search from Spanner. Read about it here. To create the embeddings for the search text / topic and run the Vector Search in Spanner, run the following query in the Spanner Studio

Editor: **SELECT a.id, a.title, a.abstract, 'A new Natural Language Processing related Machine Learning Model' search\_text, COSINE\_DISTANCE(c.patents\_embeddings, (SELECT embeddings.values FROM ML.PREDICT( MODEL text\_embeddings, (SELECT 'A new Natural Language Processing related Machine Learning Model' as content)))) as distance FROM patents\_data a inner join patents\_data\_gemini b on a.id = b.id inner join patents\_data\_embeddings c on a.id = c.id ORDER BY distance LIMIT 10;**

If you notice, in the above query, I have searched for the text 'A new Natural Language Processing related Machine Learning Model' in the patents data to find the 10 closest matches using the COSINE\_DISTANCE method.

As you can observe in your results, the matches are pretty close to the search text. That's it! It is that simple to perform Similarity Vector Search using Embeddings model on Spanner data.