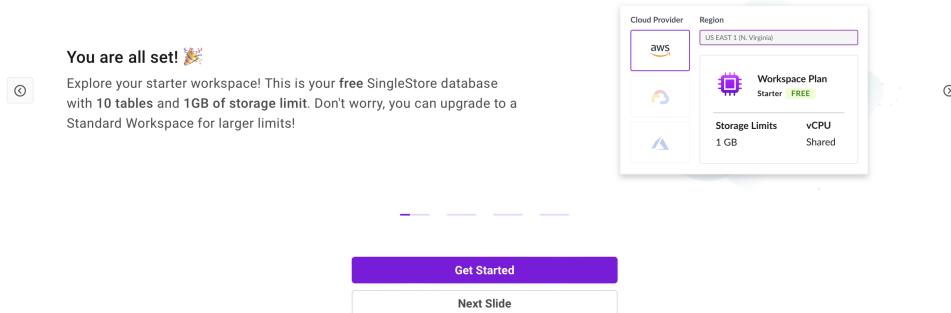


03- Storing Embedding in Vector DB and searching it

1- setup an account in <https://www.singlestore.com/> (to use as your vector DB) . it is free and comes with credit.



create your account. A new workspace and database would have been created for you automatically. As seen, the Database is empty as of now.

Welcome, Inder!

Your New Favorite SingleStore Helper. Go Nuts.

Write and optimize SQL, generate code blocks and build your real-time applications.

Ask SQL.

Suggested for you

- Load files from S3 into Shared Tier
- Hybrid Search
- Try SQL, our AI assistant

Recents

starter-workspace Deployed Starter Workspace

Created less than a minute ago

Help

About SingleStoreDB Cloud

Connect to your workspace

Load your data

Help documentation

Tutorials & Academy

View tutorials

Why your vector database should not be a vector database

How to load 100 billion rows of data ultrafast

Become a SingleStore pro

PREVIEW aws starter-workspace US EAST 1 (N. VIRGINIA) FREE

Overview Access Databases

Introducing our Free-Tier

Get a taste of SingleStore performance for your SQL, JSON and vector data. What will you #build on SingleStore?

Workspace Plan

Storage Limits vCPU Shared

Compute

starter-workspace US East 1 (N. Virginia) Connect db_inder_f8abb

Databases

db_inder_f8abb

starter-workspace > db_inder_f8abb

Tables Views Procedures Functions Aggregates Pipelines

Your database has no tables

Learn how to create a table in our tutorial.

Go to tutorial

2- click on develop and select Open sql editor(top right)

3- Create a table like below. (remove the commented lines while running)

create table if not exists myvectortable(text TEXT, vector BLOB);

4- Add Sample Data in the table

Add the embedding data between the strings

The screenshot shows the SingleStore Data Cloud interface. At the top, there's a toolbar with 'Visual Explain', '+ Load Data', and 'Run'. Below the toolbar is a code editor window displaying the following SQL query:

```

1 -- json_array_pack is used to convert it into blob structure
2 INSERT INTO myvectortable(text,vector) VALUES ("Hello World", JSON_ARRAY_PACK([
3     -0.00772272,
4     0.003593215,
5     -0.007135986,
6     -0.02926657,
7     -0.01384866,
8     0.0109970635,
9     -0.020273846,
10    0.0653201322,
11    -0.00856954,
12    -0.038251483,
13    0.024377837,
14    0.0098056995,
15    -0.027499831,
16    -0.0065689296,
17    0.009138239,
18    0.01548254,
19    0.021892184,
20    -0.009741895,
21    0.011054407,

```

Below the code editor is a message log with 'Save as CSV' and 'Copy to Clipboard' buttons. The status bar indicates '544 ms'. Under the message log, there are tabs for 'Insert ID', 'Affected Rows', and 'Changed Rows'. The 'Affected Rows' tab shows 1 row.

Now you can click on your DB and see the record present in the table.

The screenshot shows the SingleStore Data Cloud interface. On the left, there's a sidebar with 'Homepage', 'Cloud' (selected), and 'Develop'. Under 'Cloud', 'starter-workspace' is selected, and 'db_inder_f8abb' is selected under it. 'myvectortable' is highlighted. The main area shows the table structure with 'text' and 'vector' columns. There are four rows of data:

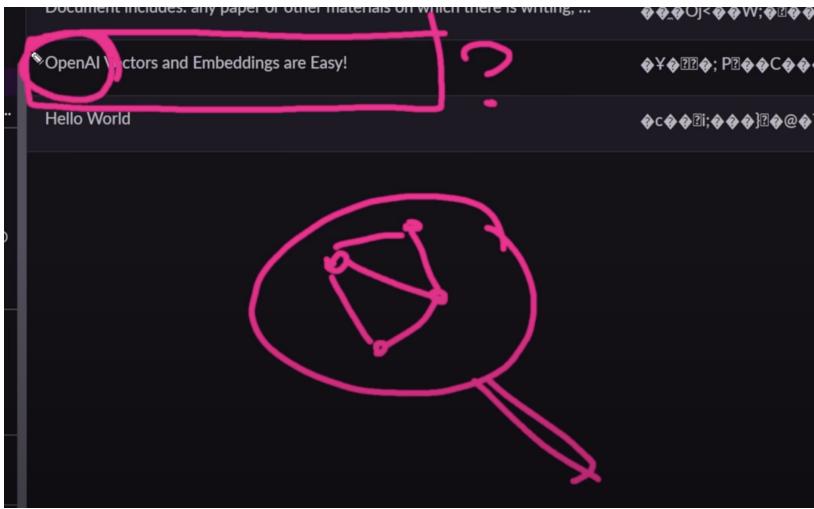
text	vector
Hello World	blob
To fully enable the participants of GrabathonX to start building their prototypes, Grab provided a suite of tool...	blob
Golang is faster than C#	blob

5- Generate more embeddings of different text and add the data in the table

The screenshot shows the SingleStore Data Cloud interface. The sidebar and table structure are identical to the previous screenshot, but now there are five rows of data:

text	vector
Hello World	blob
To fully enable the participants of GrabathonX to start building their prototypes, Grab provided a suite of tool...	blob
Golang is faster than C#	blob
OpenAI and vectors embedding is easy!	blob

How would search work?



Here, suppose we want to search for OpenAI. Then we would first create the embedding of OpenAI. We would then search that embedding with the vector stored in the DB

This would return the list of the embedding with the closest embedding on top.



Example of how to search

Below is the sql query to search:

```
select text, dot_product(vector, JSON_ARRAY_PACK("[]")) as score
from myvectortable
order by score desc
limit 5;
```

Now, i want to search vector, so first I'll create the embedding of vector

```
POST https://api.openai.com/v1/embeddings
```

Params: Authorization, Headers (12), Body (1), Pre-request Script, Tests, Settings
Body: `{ "model": "text-embedding-ada-002", "input": "vector" }`

Body: Cookies (3), Headers (24), Test Results
Pretty, Raw, Preview, Visualize, JSON

```
1 {  
2   "object": "list",  
3   "data": [  
4     {  
5       "object": "embedding",  
6       "index": 0,  
7       "embedding": [  
8         -0.828980179,  
9         -0.61861584,  
10        -0.65981623,  
11        -0.479515234,  
12        -0.000323515,  
13        0.00824362,  
14        -0.006674807,  
15        -0.02346394,  
16        -0.01967364,  
17        -0.01967226,  
18        0.006677576,  
19        0.019372465,  
20        -0.61149465  
]
```

Status: 200 OK Time: 542 ms Size: 33.88 KB Save as example

I'll then run the sql query with the vector embedding .

```
1524   text,          score  
1525   from myvectortable  
1526   order by score desc  
1527   limit 5;
```

Message Logs: select text, d... x | select text, d... x | select text, d... x | 311 ms, Showing 4 rows

text	score
OpenAI and vectors embedding is easy!	0.7857942581176758
Hello World	0.7543723583221436
To fully enable the participants of GrabathonX to start building their prototypes, Grab provided a suite of tools, including the company's in-house LLM operations (LLMOPs) platform. This gave participants access to ...	0.723437488079071
Golang is faster than C#	0.7072648406028748

Try with "vectors" and the score result changes.

```
1519   text,          score  
1520   from myvectortable  
1521   order by score desc  
1522   limit 5;
```

Message Logs: select text, d... x | select text, d... x | select text, d... x | 239 ms, Showing 4 rows

text	score
OpenAI and vectors embedding is easy!	0.8159833550453186
Hello World	0.7634585499763489
To fully enable the participants of GrabathonX to start building their prototypes, Grab provided a suite of tools, including the company's in-house LLM operations (LLMOPs) platform. This gave participants access to ...	0.721002459526062
Golang is faster than C#	0.7107806205749512

Tried with "hello Earth" and see how it changes.

Open AI vector DB / Create Embeddings Copy

POST https://api.openai.com/v1/embeddings

Params Authorization Headers (12) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```

1 {
2   "model": "text-embedding-ada-002",
3   "input": "Hello Earth"
4 }

```

Body Cookies (3) Headers (23) Test Results

Pretty Raw Preview Visualize JSON

Status: 200 OK Time: 488 ms Size: 33.63 KB Save as example

```

1533 0.010368081,
1534 0.002195498,
1535 0.015223853,
1536 0.012031152,
1537 -0.01552287,
1538 -0.023323307,
1539 0.030005682,
1540 -0.01267571,
1541 -0.00426049,
1542 -0.017082958,
1543 -0.020307139
1544
1545 },
1546 ],
1547 "model": "text-embedding-ada-002".

```

1519 0.010368081,
1520 0.011787156,
1521 -0.013481756,
1522 0.0056455666,
1523 -0.0064483616,
1524 0.004751766,
1525 -0.014755827,
1526 -0.019722166,
1527 0.010862109,
1528 0.002195498,
1529 0.015223853,
1530 0.012409195,
1531 -0.01552287,
1532 -0.023323307,
1533 0.030005682,
1534 -0.01267571,
1535 -0.00426049,
1536 -0.017082958,
1537 -0.020307139
1538])") as score
1539 from myvectortable

Message Logs

Save as CSV Copy to Clipboard

247 ms, Showing 4 rows

text	score
Hello World	0.888117253780365
OpenAI and vectors embedding is easy!	0.7400389313697815
To fully enable the participants of GrabathonX to start building their prototypes, Grab provided a suite of tools, including the company's in-house LLM operations (LLMOPs) platform. This gave participants access to ...	0.7210882902145386
Golang is faster than C#	0.7022929191589355