

Generating SQL for SQLite using Ollama, ChromaDB

This notebook runs through the process of using the `vanna` Python package to generate SQL using AI (RAG + LLMs) including connecting to a database and training. If you're not ready to train on your own database, you can still try it using a sample [SQLite database](#).

Which LLM do you want to use?

- [OpenAI via Vanna.AI \(Recommended\)](#)
Use Vanna.AI for free to generate your queries
- [OpenAI](#)
Use OpenAI with your own API key
- [Azure OpenAI](#)
If you have OpenAI models deployed on Azure
- [\[Selected\] Ollama](#)
Use Ollama locally for free. Requires additional setup.
- [Mistral via Mistral API](#)
If you have a Mistral API key
- [Other LLM](#)
If you have a different LLM model

Where do you want to store the 'training' data?

- [Vanna Hosted Vector DB \(Recommended\)](#)
Use Vanna.AI's hosted vector database (pgvector) for free. This is usable across machines with no additional setup.
- [\[Selected\] ChromaDB](#)
Use ChromaDB's open-source vector database for free locally. No additional setup is necessary -- all database files will be created and stored locally.
- [Marqo](#)
Use Marqo locally for free. Requires additional setup. Or use their hosted option.
- [Other VectorDB](#)

Use any other vector database. Requires additional setup.

Setup

!pwd!pip install vanna!pip install 'vanna[chromadb]'!pip install ollama!pip show vanna # 0.5.5, 0.2.1!pip show ollama # 0.2.0

```
In [1]: import warnings
import re

warnings.filterwarnings('ignore', category=DeprecationWarning, message='^Number of requested results')
# warnings.filterwarnings('ignore', category=DeprecationWarning, message=re.escape(r'^Some regex pattern')),

import os

import re
from time import time

from vanna.ollama import Ollama
from vanna.chromadb.chromadb_vector import ChromaDB_VectorStore
```

```
In [2]: class MyVanna(ChromaDB_VectorStore, Ollama):
    def __init__(self, config=None):
        ChromaDB_VectorStore.__init__(self, config=config)
        Ollama.__init__(self, config=config)
```

```
In [3]: file_db = "~/Downloads/chinook.sqlite"
model_name = 'codegemma'
```

```
In [4]: config = {
    'model': model_name, # 'mistral' # "starcoder2"
}
vn = MyVanna(config=config)
```

```
In [5]: hostname = os.uname().nodename
print("Hostname:", hostname)
```

Hostname: ducklover1

```
In [6]: file_db = os.path.abspath(os.path.expanduser(file_db))
        vn.connect_to_sqlite(file_db)
```

```
In [7]: vn.run_sql_is_set
```

```
Out[7]: True
```

```
In [8]: def remove_collections(collection_name=None, ACCEPTED_TYPES = ["sql", "ddl", "documentation"]):
        if not collection_name:
            collections = ACCEPTED_TYPES
        elif isinstance(collection_name, str):
            collections = [collection_name]
        elif isinstance(collection_name, list):
            collections = collection_name
        else:
            print(f"\t{collection_name} is unknown: Skipped")
            return

        for c in collections:
            if not c in ACCEPTED_TYPES:
                print(f"\t{c} is unknown: Skipped")
                continue

            # print(f"vn.remove_collection('{c}')"")
            vn.remove_collection(c)
```

```
In [9]: def strip_brackets(ddl):
        """
        This function removes square brackets from table and column names in a DDL script.

        Args:
            ddl (str): The DDL script containing square brackets.

        Returns:
            str: The DDL script with square brackets removed.
        """
        # Use regular expressions to match and replace square brackets
        pattern = r"\[([^\]]+)\]" # Match any character except ] within square brackets
        return re.sub(pattern, r"\1", ddl)
```

```
In [10]: if False:
         remove_collections()
```

Training

SQLite sample database

You only need to train once. Do not train again unless you want to add more training data.

```
In [11]: # show training data
         training_data = vn.get_training_data()
         training_data
```

Out[11]:

	id	question	content	training_data_type
0	01c4a964-460b-5e1c-af1e-622c8210b835-sql	\n Hint: album quantity is found in invo...	SELECT i.CustomerId, COUNT(ii.InvoiceLineId) A...	sql
1	0658ba3d-98ff-51f4-9006-a24f87045858-sql	How many customers are there	SELECT COUNT(*) FROM "customers"	sql
2	127fd4bd-b9af-539d-9313-1d0234d073b7-sql	\n There are 3 tables: artists, albums and...	SELECT a.Name, COUNT(t.TrackId) AS TotalTracks...	sql
3	3013d1b4-feb2-519d-bfb9-114500436e3d-sql	\n Find the customer with the most invo...	SELECT c.CustomerId, COUNT(i.InvoiceId) AS Tot...	sql
4	32b99e7b-31ab-55d8-8431-fb010fa7af85-sql	\n Find the top 5 customers who spent th...	SELECT c.CustomerId, SUM(i.Total) AS TotalSpen...	sql
5	49e67df3-a604-51f8-ad01-b8f5a2043eac-sql	\n Get the total number of invoices for e...	SELECT c.CustomerId, COUNT(i.InvoiceId) AS Tot...	sql
6	584873f8-1904-50f1-8f80-7ccf08059264-sql	\n List all customers from Canada and th...	SELECT c.Email, c.Country\nFROM "customers" c\...	sql
7	6bed484b-9a80-57f4-ad89-5f775b5df252-sql	\n Get the average invoice total for each...	SELECT c.CustomerId, AVG(i.Total) AS AverageIn...	sql
8	6f22268c-5062-5f11-ba2d-8555f06b409d-sql	\n Find all tracks with a name containing...	SELECT * \nFROM "tracks" \nWHERE LOWER(Name) L...	sql
9	70b4f686-c71b-5ee8-9458-6bbc776349bf-sql	\n Find all invoices since 2010 and the t...	SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmo...	sql
10	9a396a33-ecea-51a8-bd05-28f58a86eb86-sql	\n Hint: album quantity is found in invo...	SELECT c.CustomerId, COUNT(ii.TrackId) AS Tota...	sql
11	9a9c970b-b94c-5f22-b54c-b86921a38b65-sql	\n Identify artists who have albums with...	SELECT a.ArtistId, a.Name AS ArtistName\nFROM ...	sql
12	a7185c88-7417-5b75-a52e-4eae5f9deca-sql	\n List all albums and their correspondin...	SELECT a.Title, a.ArtistId, ar.Name AS ArtistN...	sql
13	aea89953-21b2-55d1-9dda-431ee6033c3d-sql	\n List all invoices with a total exceedi...	SELECT * \nFROM "invoices" \nWHERE Total > 10.00	sql
14	d1d70c18-f5d9-5970-a32c-914deeca1087-sql	\n Find the customer who bought the most...	SELECT c.CustomerId, COUNT(ii.TrackId) AS Tota...	sql
15	d8a2f948-dffa-5524-a5f9-174cc1a8da73-sql	Can you list all tables in the SQLite database...	SELECT name FROM sqlite_master WHERE type='table'	sql
16	d8a37163-5ce5-58cd-a316-	what are the top 5 countries	SELECT c.Country, COUNT(*) AS	sql

	id	question	content	training_data_type
	ea5598d44d27-sql	that customers co...	TotalCustomers\n...	
17	dd282d7c-a4ef-5e3a-87e0-cb45fac50808-sql	\n Find the total number of invoices per ...	SELECT i.BillingCountry, COUNT(*) AS TotalInvo...	sql
18	e7c4b3aa-664f-5f87-8b25-449a4482f3fd-sql	\n Get all playlists containing at least...	SELECT pt.PlaylistId, p.Name AS PlaylistName, ...	sql
19	f33f8cb6-1b12-5ea7-8d9a-aef8166b9970-sql	\n Find the top 5 most expensive tracks (...)	SELECT t.TrackId, t.Name, t.UnitPrice\nFROM "t...	sql
20	f626b681-4d8f-563a-beee-1ea759baaa82-sql	\n List all genres and the number of trac...	SELECT g.Name, COUNT(t.GenreId) AS TotalTracks...	sql
21	fd25ebba-4066-5a0f-8613-7b1c2ace0339-sql	\n List all employees and their reporting...	SELECT e.FirstName, e.LastName, mt.FirstName A...	sql
0	039f9d54-59f7-5f29-8c04-14dbc3e95671-ddl	None	CREATE TABLE "artists"\n(\n ArtistId IN...	ddl
1	0db84e3d-ef41-563c-803e-21c1b985dc19-ddl	None	CREATE TABLE "invoices"\n(\n InvoiceId ...	ddl
2	10cba811-ddba-5042-9e90-d764dfcd1629-ddl	None	CREATE INDEX IFK_InvoiceCustomerId ON "invoice...	ddl
3	2c711317-b93d-5f60-a728-cb1c6fcbc040-ddl	None	CREATE INDEX IFK_CustomerSupportRepId ON "cust...	ddl
4	37319c81-65f7-50ee-956b-795de244bee5-ddl	None	CREATE TABLE sqlite_stat1(tbl,idx,stat)	ddl
5	40bd77cd-e1de-5872-8693-624117ff413c-ddl	None	CREATE INDEX IFK_InvoiceLineInvoiceId ON "invo...	ddl
6	41130543-7164-562a-90a7-0fd0a409c154-ddl	None	CREATE TABLE "albums"\n(\n AlbumId INTE...	ddl
7	458debc8-8082-5450-a17a-66028bd55ace-ddl	None	CREATE TABLE "playlists"\n(\n PlaylistI...	ddl
8	4815f3fd-925b-53ce-9dfa-0e4285d5abd3-ddl	None	CREATE TABLE "invoice_items"\n(\n Invoi...	ddl
9	48d484e9-984c-58ff-b391-75521c69d486-ddl	None	CREATE INDEX IFK_PlaylistTrackTrackId ON "play...	ddl
10	551e1120-a6ee-554f-8b8a-ccf4f22d3636-ddl	None	CREATE INDEX IFK_AlbumArtistId ON "albums" (Ar...	ddl

	id	question	content	training_data_type
11	5ff4911e-45c1-5a59-9566-243a9b6a3320-ddl	None	CREATE TABLE "employees"\r\n(\r\nEmployeeI...	ddl
12	65df0648-bf05-5f75-9365-c21f54b2302d-ddl	None	CREATE TABLE "media_types"\r\n(\r\nMediaTy...	ddl
13	6b585176-e66d-5b23-8d86-ca8a80e3af3d-ddl	None	CREATE INDEX IFK_EmployeeReportsTo ON "employe...	ddl
14	868758b8-e018-55e7-8cc3-75c0e6d211c8-ddl	None	CREATE INDEX IFK_TrackAlbumId ON "tracks" (Alb...	ddl
15	9ea4613d-c1be-5a77-ada9-c54ee3f0cab7-ddl	None	CREATE INDEX IFK_TrackMediaTypeId ON "tracks" ...	ddl
16	a9c9a852-608d-5ef2-aede-26ba098d83d1-ddl	None	CREATE INDEX IFK_TrackGenreId ON "tracks" (Gen...	ddl
17	b42cc9e1-9219-5a42-9a06-de906f76239e-ddl	None	CREATE TABLE "tracks"\r\n(\r\n TrackId INTE...	ddl
18	c387b9d2-5ff4-5a07-8364-f5dab45bb2a9-ddl	None	CREATE TABLE "genres"\r\n(\r\n GenreId INTE...	ddl
19	d654f328-dc36-549e-84c3-06ee0db7e0f7-ddl	None	CREATE TABLE "playlist_track"\r\n(\r\n Play...	ddl
20	d93f0d68-023d-5afb-8121-ba346699d318-ddl	None	CREATE TABLE "customers"\r\n(\r\n CustomerI...	ddl
21	e5879308-329e-543f-a693-0c14e2f9972e-ddl	None	CREATE INDEX IFK_InvoiceLineTrackId ON "invoic...	ddl
22	ea84418b-1a28-59b4-a1f4-2fb674208adc-ddl	None	CREATE TABLE sqlite_sequence(name,seq)	ddl
0	9d2550eb-8e22-54cd-9fad-9e1be65ab03a-doc	None	In the SQLite database invoice means order	documentation

```
In [12]: df_ddl = vn.run_sql("SELECT type, sql FROM sqlite_master WHERE sql is not null")
```

```
In [13]: df_ddl
```

Out[13]:

	type	sql
0	table	CREATE TABLE "albums"\r\n(\r\n [AlbumId] IN...
1	table	CREATE TABLE sqlite_sequence(name,seq)
2	table	CREATE TABLE "artists"\r\n(\r\n [ArtistId] ...
3	table	CREATE TABLE "customers"\r\n(\r\n [Customer...
4	table	CREATE TABLE "employees"\r\n(\r\n [Employee...
5	table	CREATE TABLE "genres"\r\n(\r\n [GenreId] IN...
6	table	CREATE TABLE "invoices"\r\n(\r\n [InvoiceId...
7	table	CREATE TABLE "invoice_items"\r\n(\r\n [Invo...
8	table	CREATE TABLE "media_types"\r\n(\r\n [MediaT...
9	table	CREATE TABLE "playlists"\r\n(\r\n [Playlist...
10	table	CREATE TABLE "playlist_track"\r\n(\r\n [Pla...
11	table	CREATE TABLE "tracks"\r\n(\r\n [TrackId] IN...
12	index	CREATE INDEX [IFK_AlbumArtistId] ON "albums" (...
13	index	CREATE INDEX [IFK_CustomerSupportRepId] ON "cu...
14	index	CREATE INDEX [IFK_EmployeeReportsTo] ON "emplo...
15	index	CREATE INDEX [IFK_InvoiceCustomerId] ON "invoi...
16	index	CREATE INDEX [IFK_InvoiceLineInvoiceId] ON "in...
17	index	CREATE INDEX [IFK_InvoiceLineTrackId] ON "invo...
18	index	CREATE INDEX [IFK_PlaylistTrackTrackId] ON "pl...
19	index	CREATE INDEX [IFK_TrackAlbumId] ON "tracks" ([...
20	index	CREATE INDEX [IFK_TrackGenreId] ON "tracks" ([...
21	index	CREATE INDEX [IFK_TrackMediaTypeId] ON "tracks...
22	table	CREATE TABLE sqlite_stat1(tbl,idx,stat)

In [14]:

```
if False:
    for ddl in df_ddl['sql'].to_list():
```



```
ddl = strip_brackets(ddl)
vn.train(ddl=ddl)
```

```
In [15]: if False:
         # Sometimes you may want to add documentation about your business terminology or definitions.
         vn.train(documentation="In the SQLite database invoice means order")
```

Asking the AI

Whenever you ask a new question, it will find the 10 most relevant pieces of training data and use it as part of the LLM prompt to generate the SQL.

```
In [16]: ts_start = time()

        SELECT name FROM sqlite_master WHERE type = 'table';
```

```
In [17]: vn.ask(question="Can you list all tables in the SQLite database catalog?")
```

```
Add of existing embedding ID: d8a2f948-dffa-5524-a5f9-174cc1a8da73-sql
Add of existing embedding ID: 0658ba3d-98ff-51f4-9006-a24f87045858-sql
Add of existing embedding ID: 127fd4bd-b9af-539d-9313-1d0234d073b7-sql
Add of existing embedding ID: 32b99e7b-31ab-55d8-8431-fb010fa7af85-sql
Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1
```

```
[{'role': 'system', 'content': 'You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables \nCREATE TABLE sqlite_stat1(tbl,idx,stat)\n\nCREATE TABLE sqlite_sequence(name,seq)\n\nCREATE TABLE "playlists"\n(\n    PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\n\nCREATE TABLE "genres"\n(\n    GenreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\n\nCREATE TABLE "tracks"\n(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES "albums" (AlbumId) \n    \n    FOREIGN KEY (GenreId) REFERENCES "genres" (GenreId) \n    \n    FOREIGN KEY (MediaTypeId) REFERENCES "media_types" (MediaTypeId) \n    \n    \n)\n\nCREATE TABLE "media_type_s"\n(\n    MediaTypeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\n\nCREATE TABLE "artists"\n(\n    ArtistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\n\nCREATE TABLE "invoice_items"\n(\n    InvoiceLineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    InvoiceId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    Quantity INTEGER NOT NULL,\n    FOREIGN KEY (InvoiceId) REFERENCES "invoices" (InvoiceId) \n    \n    FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \n    \n    \n)\n\nCREATE TABLE "playlist_track"\n(\n    PlaylistId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId) \n    \n    FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \n    \n    \n)\n\nCREATE TABLE "albums"\n(\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Title NVARCHAR(160) NOT NULL,\n    ArtistId INTEGER NOT NULL,\n    FOREIGN KEY (ArtistId) REFERENCES "artists" (ArtistId) \n    \n    \n)\n\n===Additional Context \n\nIn the SQLite database invoice means order\n\n===Response Guidelines \n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question. \n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before. \n'}], {'role': 'user', 'content': 'Can you list all tables in the SQLite database catalog?'}, {'role': 'assistant', 'content': "SELECT name FROM sqlite_master WHERE type='table'"}, {'role': 'user', 'content': ' \n    There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId, \n    Can you find the top 10 most popular artists based on the number of tracks\n'}, {'role': 'assistant', 'content': 'SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nJOIN "tracks" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10'}, {'role': 'user', 'content': ' \n    List all albums and their corresponding artist names\n'}, {'role': 'assistant', 'content': 'SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM "albums" a\nJOIN "artists" ar ON a.ArtistId = ar.ArtistId'}, {'role': 'user', 'content': ' \n    List all customers from Canada and their email addresses:\n'}, {'role': 'assistant', 'content': 'SELECT c.Email, c.Country\nFR
```

```

OM "customers" c\nWHERE c.Country = \'Canada\'}, {'role': 'user', 'content': ' \n      Find the customer w
ho bought the most albums in total quantity (across all invoices): \n'}, {'role': 'assistant', 'content':
'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.Customer
Id = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY T
otalAlbums DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n      List all genres and the number of tracks i
n each genre:\n'}, {'role': 'assistant', 'content': 'SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM
"genres" g\nJOIN "tracks" t ON g.GenreId = t.GenreId\nGROUP BY g.Name'}, {'role': 'user', 'content': " \n
List all employees and their reporting manager's name (if any):\n"}, {'role': 'assistant', 'content': 'SELE
CT e.FirstName, e.LastName, mt.FirstName AS ManagerFirstName, mt.LastName AS ManagerLastName\nFROM "employe
es" e\nLEFT JOIN "employees" mt ON e.ReportsTo = mt.EmployeeId'}, {'role': 'user', 'content': ' \n      Hin
t: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most album
s in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUN
T(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "i
nvoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT
5'}, {'role': 'user', 'content': ' \n      Find the top 5 most expensive tracks (based on unit price):\n'},
{'role': 'assistant', 'content': 'SELECT t.TrackId, t.Name, t.UnitPrice\nFROM "tracks" t\nORDER BY t.UnitPr
ice DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Get all playlists containing at least 10 tracks
and the total duration of those tracks:\n'}, {'role': 'assistant', 'content': 'SELECT pt.PlaylistId, p.Name
AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM "playlist_track" pt\nJOIN "playlists" p ON pt.P
laylistId = p.PlaylistId\nJOIN "tracks" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING
COUNT(pt.TrackId) >= 10'}, {'role': 'user', 'content': 'Can you list all tables in the SQLite database cata
log?'}]

```

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```

[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the
question. Your response should ONLY be based on the given context and follow the response guidelines and fo
rmat instructions. \n===Tables \nCREATE TABLE sqlite_stat1(tbl,idx,stat)\n\nCREATE TABLE sqlite_sequence(na
me,seq)\n\nCREATE TABLE \"playlists\"\n\n(\n    PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    \n    Name NVARCHAR(120)\n)\n\nCREATE TABLE \"genres\"\n\n(\n    GenreId INTEGER PRIMARY KEY AUTOINCREM
ENT NOT NULL,\n    Name NVARCHAR(120)\n)\n\nCREATE TABLE \"tracks\"\n\n(\n    TrackId INTEGER PRIMARY
KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId
INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT
NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENC
ES \"albums\" (AlbumId) \n\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFER
ENCES \"genres\" (GenreId) \n\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId)
REFERENCES \"media_types\" (MediaTypeId) \n\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE
TABLE \"media_types\"\n\n(\n    MediaTypeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARC
HAR(120)\n)\n\nCREATE TABLE \"artists\"\n\n(\n    ArtistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NUL
L,\n    Name NVARCHAR(120)\n)\n\nCREATE TABLE \"invoice_items\"\n\n(\n    InvoiceLineId INTEGER PRIMA

```

```

RY KEY AUTOINCREMENT NOT NULL,\r\n    InvoiceId INTEGER NOT NULL,\r\n    TrackId INTEGER NOT NULL,\r\n
UnitPrice NUMERIC(10,2) NOT NULL,\r\n    Quantity INTEGER NOT NULL,\r\n    FOREIGN KEY (InvoiceId) REFERE
NCES \"invoices\" (InvoiceId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n    FOREIGN KEY (TrackId)
REFERENCES \"tracks\" (TrackId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"play
list_track\"(\r\n    PlaylistId INTEGER NOT NULL,\r\n    TrackId INTEGER NOT NULL,\r\n    CONSTRAINT
PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\r\n    FOREIGN KEY (PlaylistId) REFERENCES \"playlists
\" (PlaylistId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n    FOREIGN KEY (TrackId) REFERENCES
\"tracks\" (TrackId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"albums\"(\r\n
    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Title NVARCHAR(160) NOT NULL,\r\n    Arti
stId INTEGER NOT NULL,\r\n    FOREIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistId) \r\n\t\t\tON DELETE N
O ACTION ON UPDATE NO ACTION\r\n)\n\n\n===Additional Context\n\nIn the SQLite database invoice means order
\n\n===Response Guidelines\n1. If the provided context is sufficient, please generate a valid SQL query wi
thout any explanations for the question.\n2. If the provided context is almost sufficient but requires kno
wledge of a specific string in a particular column, please generate an intermediate SQL query to find the d
istinct strings in that column. Prepend the query with a comment saying intermediate_sql\n3. If the provid
ed context is insufficient, please explain why it can't be generated.\n4. Please use the most relevant tab
le(s).\n5. If the question has been asked and answered before, please repeat the answer exactly as it was
given before.\n\"}, {\"role\": \"user\", \"content\": \"Can you list all tables in the SQLite database catalog?\"},
{\"role\": \"assistant\", \"content\": \"SELECT name FROM sqlite_master WHERE type='table'\"}, {\"role\": \"user\", \"co
ntent\": \"\n    There are 3 tables: artists, albums and tracks, where albums and artists are linked by Arti
stId, albums and tracks are linked by AlbumId,\n    Can you find the top 10 most popular artists based on t
he number of tracks\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\n
FROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nJOIN \"tracks\" t ON al.AlbumId = t.Alb
umId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10\"}, {\"role\": \"user\", \"content\": \"\n    List all
albums and their corresponding artist names\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT a.Title, a.Arti
stId, ar.Name AS ArtistName\nFROM \"albums\" a\nJOIN \"artists\" ar ON a.ArtistId = ar.ArtistId\"}, {\"role\":
\"user\", \"content\": \"\n    List all customers from Canada and their email addresses:\n\"}, {\"role\": \"assis
tant\", \"content\": \"SELECT c.Email, c.Country\nFROM \"customers\" c\nWHERE c.Country = 'Canada'\"}, {\"role\":
\"user\", \"content\": \"\n    Find the customer who bought the most albums in total quantity (across all inv
oices):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM
\"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.Invoice
Id = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1\"}, {\"role\": \"user\", \"content\":
\"\n    List all genres and the number of tracks in each genre:\n\"}, {\"role\": \"assistant\", \"content\": \"SEL
ECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM \"genres\" g\nJOIN \"tracks\" t ON g.GenreId = t.GenreId
\nGROUP BY g.Name\"}, {\"role\": \"user\", \"content\": \"\n    List all employees and their reporting manager's
name (if any):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT e.FirstName, e.LastName, mt.FirstName AS Manag
erFirstName, mt.LastName AS ManagerLastName\nFROM \"employees\" e\nLEFT JOIN \"employees\" mt ON e.ReportsTo
= mt.EmployeeId\"}, {\"role\": \"user\", \"content\": \"\n    Hint: album quantity is found in invoice_items,
\n    Find the top 5 customers who bought the most albums in total quantity (across all invoice
s):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"cu
stomers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId =
ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \"

```

```
{
  "role": "assistant",
  "content": "SELECT t.TrackId, t.Name, t.UnitPrice\nFROM \"tracks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5",
  "role": "user",
  "content": "\n\nGet all playlists containing at least 10 tracks and the total duration of those tracks:\n",
  "role": "assistant",
  "content": "SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM \"playlist_track\" pt\nJOIN \"playlists\" p ON pt.PlaylistId = p.PlaylistId\nJOIN \"tracks\" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10",
  "role": "user",
  "content": "Can you list all tables in the SQLite database catalog?"
}
```

Add of existing embedding ID: d8a2f948-dffa-5524-a5f9-174cc1a8da73-sql

Insert of existing embedding ID: d8a2f948-dffa-5524-a5f9-174cc1a8da73-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:23:55.953309228Z', 'message': {'role': 'assistant', 'content': "SELECT name FROM sqlite_master WHERE type='table'"}, 'done_reason': 'stop', 'done': True, 'total_duration': 61378011525, 'load_duration': 2077984799, 'prompt_eval_count': 1658, 'prompt_eval_duration': 56210673000, 'eval_count': 12, 'eval_duration': 2455748000}
```

```
SELECT name FROM sqlite_master WHERE type='table'
```

```
SELECT name FROM sqlite_master WHERE type='table'
```

```

      name
0      albums
1  sqlite_sequence
2      artists
3      customers
4      employees
5      genres
6      invoices
7  invoice_items
8      media_types
9      playlists
10 playlist_track
11      tracks
12  sqlite_stat1
```

Ollama parameters:

```
model=codegemma:latest,
```

```
options={},
```

```
keep_alive=None
```

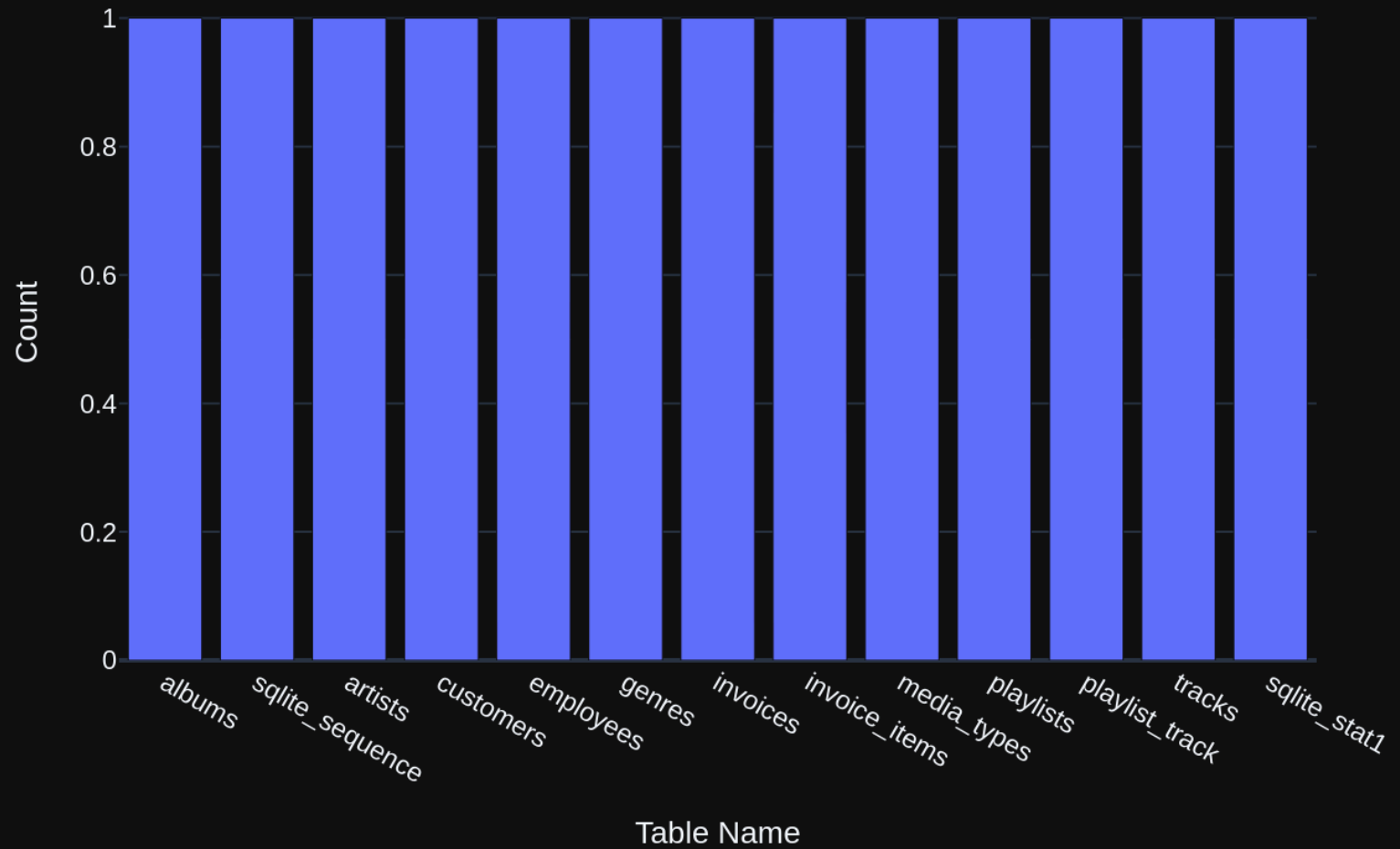
Prompt Content:

```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: 'Can you list all tables in the SQLite database catalog?'\n\nThe DataFrame was produced using this query: SELECT name FROM sqlite_master WHERE type='table'\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n name      object\ndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:24:15.807720729Z', 'message': {'role': 'assistant', 'content': "\n\npython\nimport plotly.express as px\n\nfig = px.bar(df, x='name', title='Tables in SQLite Database Catalog')\nfig.update_layout(\n    xaxis_title='Table Name',\n    yaxis_title='Count')\nfig.show()\n\n"}, 'done_reason': 'stop', 'done': True, 'total_duration': 19826653274, 'load_duration': 667808, 'prompt_eval_count': 149, 'prompt_eval_duration': 5641634000, 'eval_count': 64, 'eval_duration': 14094628000}
```

Tables in SQLite Database Catalog



```

Out[17]: ("SELECT name FROM sqlite_master WHERE type='table'",
          name
0         albums
1  sqlite_sequence
2         artists
3         customers
4         employees
5         genres
6         invoices
7  invoice_items
8         media_types
9         playlists
10  playlist_track
11         tracks
12  sqlite_stat1,
Figure({
  'data': [{'alignmentgroup': 'True',
            'hovertemplate': 'name={x}<br>count={y}<extra></extra>',
            'legendgroup': '',
            'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
            'name': '',
            'offsetgroup': '',
            'orientation': 'v',
            'showlegend': False,
            'textposition': 'auto',
            'type': 'bar',
            'x': array(['albums', 'sqlite_sequence', 'artists', 'customers', 'employees',
                       'genres', 'invoices', 'invoice_items', 'media_types', 'playlists',
                       'playlist_track', 'tracks', 'sqlite_stat1'], dtype=object),
            'xaxis': 'x',
            'y': array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]),
            'yaxis': 'y'}],
  'layout': {'barmode': 'relative',
             'legend': {'tracegroupgap': 0},
             'template': '...',
             'title': {'text': 'Tables in SQLite Database Catalog'},
             'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Table Name'}},
             'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'Count'}}})
)))

```

```
In [18]: vn.ask(question="which table stores customer's orders")
```


Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

```
items detail table is unnecessary\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(i.Total)
```

```

AS TotalSpent\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId
\nORDER BY TotalSpent DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Find the customer with the mos
t invoices \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices
\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY Tot
alInvoices DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n      Find the customer who bought the most alb
ums in total quantity (across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, C
OUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN
"invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT
1'}, {'role': 'user', 'content': ' \n      Hint: album quantity is found in invoice_items, \n      \n      Find
the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assi
stant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoic
es" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.Cus
tomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Hint: album quantity is
found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity
(across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId)
AS TotalAlbums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.Custome
rId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Get the total number of in
voices for each customer\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS
TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': 'what are the top 5 countries that customers come from?'}, {'role': 'assistant', 'content': 'SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM "customers" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      List all customers from Canada and their email addresses:\n'}, {'role': 'assistant', 'content': 'SELECT c.Email, c.Country\nFROM "customers" c\nWHERE c.Country = \'Canada\'}}, {'role': 'user', 'content': ' \n      Get the average invoice total for each customer:\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': ' \n      Find the total number of invoices per country:\n'}, {'role': 'assistant', 'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "invoices" i\nGROUP BY i.BillingCountry'}, {'role': 'user', 'content': "which table stores customer's orders"}]

```

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```

[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the
question. Your response should ONLY be based on the given context and follow the response guidelines and fo
rmat instructions. \n===Tables\nCREATE TABLE \"invoices\"\n(\n    InvoiceId INTEGER PRIMARY KEY AUTOIN
CREMENT NOT NULL,\n    CustomerId INTEGER NOT NULL,\n    InvoiceDate DATETIME NOT NULL,\n    Billin
gAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCou
ntry NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2) NOT NULL,\n    FORE
IGN KEY (CustomerId) REFERENCES \"customers\" (CustomerId) \n)\n\nCREATE TABLE \"invoice_items\"\n(\n    InvoiceLineId INTEGER PRIMARY KEY AUTOIN
CREMENT NOT NUL

```

```

L,\r\n    InvoiceId INTEGER NOT NULL,\r\n    TrackId INTEGER NOT NULL,\r\n    UnitPrice NUMERIC(10,2) NO
T NULL,\r\n    Quantity INTEGER NOT NULL,\r\n    FOREIGN KEY (InvoiceId) REFERENCES \"invoices\" (InvoiceI
d) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (Tra
ckId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"customers\"(\r\n    Custom
erId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    FirstName NVARCHAR(40) NOT NULL,\r\n    LastName N
VARCHAR(20) NOT NULL,\r\n    Company NVARCHAR(80),\r\n    Address NVARCHAR(70),\r\n    City NVARCHAR(4
0),\r\n    State NVARCHAR(40),\r\n    Country NVARCHAR(40),\r\n    PostalCode NVARCHAR(10),\r\n    Phone NV
ARCHAR(24),\r\n    Fax NVARCHAR(24),\r\n    Email NVARCHAR(60) NOT NULL,\r\n    SupportRepId INTEGER,\r\n
FOREIGN KEY (SupportRepId) REFERENCES \"employees\" (EmployeeId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO A
CTION\r\n)\n\nCREATE TABLE \"employees\"(\r\n    EmployeeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NUL
L,\r\n    LastName NVARCHAR(20) NOT NULL,\r\n    FirstName NVARCHAR(20) NOT NULL,\r\n    Title NVARCHAR(3
0),\r\n    ReportsTo INTEGER,\r\n    BirthDate DATETIME,\r\n    HireDate DATETIME,\r\n    Address NVARCHAR
(70),\r\n    City NVARCHAR(40),\r\n    State NVARCHAR(40),\r\n    Country NVARCHAR(40),\r\n    PostalCode N
VARCHAR(10),\r\n    Phone NVARCHAR(24),\r\n    Fax NVARCHAR(24),\r\n    Email NVARCHAR(60),\r\n    FOREIGN
KEY (ReportsTo) REFERENCES \"employees\" (EmployeeId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)
\n\nCREATE TABLE sqlite_sequence(name,seq)\n\nCREATE TABLE \"playlists\"(\r\n    PlaylistId INTEGER PRI
MARY KEY AUTOINCREMENT NOT NULL,\r\n    Name NVARCHAR(120)\r\n)\n\nCREATE TABLE sqlite_stat1(tbl,idx,stat)
\n\nCREATE TABLE \"albums\"(\r\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Title NV
ARCHAR(160) NOT NULL,\r\n    ArtistId INTEGER NOT NULL,\r\n    FOREIGN KEY (ArtistId) REFERENCES \"artist
s\" (ArtistId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"playlist_track\"(\r\n
    PlaylistId INTEGER NOT NULL,\r\n    TrackId INTEGER NOT NULL,\r\n    CONSTRAINT PK_PlaylistTrack
PRIMARY KEY (PlaylistId, TrackId),\r\n    FOREIGN KEY (PlaylistId) REFERENCES \"playlists\" (PlaylistId)
\r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackI
d) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"media_types\"(\r\n    MediaTy
peId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Name NVARCHAR(120)\r\n)\n\n\n===Additional Context
\n\nIn the SQLite database invoice means order\n\n===Response Guidelines\n1. If the provided context is su
fficient, please generate a valid SQL query without any explanations for the question.\n2. If the provided
context is almost sufficient but requires knowledge of a specific string in a particular column, please gen
erate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comme
nt saying intermediate_sql\n3. If the provided context is insufficient, please explain why it can't be gen
erated.\n4. Please use the most relevant table(s).\n5. If the question has been asked and answered befor
e, please repeat the answer exactly as it was given before.\n\"}, {\"role\": \"user\", \"content\": \" \n
Find the top 5 customers who spent the most money overall, \n \n Hint: order total can be found on inv
oices table, calculation using invoice_items detail table is unnecessary\n\"}, {\"role\": \"assistant\", \"conte
nt\": \"SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.Custo
merId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5\"}, {\"role\": \"user\", \"conten
t\": \" \n Find the customer with the most invoices\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.Cus
tomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId =
i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1\"}, {\"role\": \"user\", \"content\": \"
\n Find the customer who bought the most albums in total quantity (across all invoices): \n\"}, {\"role\":
\"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN
\"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGRO

```

```
UP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1"}], {"role": "user", "content": " \n    Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": " \n    Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": " \n    Get the total number of invoices for each customer\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}, {"role": "user", "content": "what are the top 5 countries that customers come from?"}, {"role": "assistant", "content": "SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM \"customers\" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5"}, {"role": "user", "content": " \n    List all customers from Canada and their email addresses:\n"}, {"role": "assistant", "content": "SELECT c.Email, c.Country\nFROM \"customers\" c\nWHERE c.Country = 'Canada'"}, {"role": "user", "content": " \n    Get the average invoice total for each customer:\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}, {"role": "user", "content": " \n    Find the total number of invoices per country:\n"}, {"role": "assistant", "content": "SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM \"invoices\" i\nGROUP BY i.BillingCountry"}, {"role": "user", "content": "which table stores customer's orders"}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:25:36.27719413Z', 'message': {'role': 'assistant', 'content': 'The "invoices" table stores customer orders. Each row in the table represents a single invoice.'}, 'done_reason': 'stop', 'done': True, 'total_duration': 80053449238, 'load_duration': 674455, 'prompt_eval_count': 1846, 'prompt_eval_duration': 74525417000, 'eval_count': 20, 'eval_duration': 4892958000}
The "invoices" table stores customer orders. Each row in the table represents a single invoice.
The "invoices" table stores customer orders. Each row in the table represents a single invoice.
Couldn't run sql: Execution failed on sql 'The "invoices" table stores customer orders. Each row in the table represents a single invoice.': near "The": syntax error
```

In [19]: `vn.ask(question="How many customers are there")`

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

```
erall, \n      \n      Hint: order total can be found on invoices table, calculation using invoice items data
```

```

il table is unnecessary \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Find the customer who bought the most albums in total quantity (across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n Get the total number of invoices for each customer\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': 'what are the top 5 countries that customers come from?'}, {'role': 'assistant', 'content': 'SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM "customers" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Hint: album quantity is found in invoice_items, \n \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Hint: album quantity is found in invoice_items, \n \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Get the average invoice total for each customer:\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': ' \n Find the total number of invoices per country:\n'}, {'role': 'assistant', 'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "invoices" i\nGROUP BY i.BillingCountry'}, {'role': 'user', 'content': 'How many customers are there'}]

```

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```

[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables\nCREATE TABLE \"invoices\"\n(\n    InvoiceId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    CustomerId INTEGER NOT NULL,\n    InvoiceDate DATETIME NOT NULL,\n    BillingAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (CustomerId) REFERENCES \"customers\" (CustomerId)\n)\nCREATE INDEX IFK_CustomerSupportRepId ON \"customers\" (SupportRepId)\nCREATE TABLE \"customers\"\n(\n    CustomerId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    FirstName NVARCHAR(40) NOT NULL,\n    LastName NVARCHAR(20) NOT NULL,\n    Company NVARCHAR(80),\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60) NOT NULL,\n    Support

```

24/199


```

ty is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": "      \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": "      \n      Get the average invoice total for each customer:\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}, {"role": "user", "content": "      \n      Find the total number of invoices per country:\n"}, {"role": "assistant", "content": "SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM \"invoices\" i\nGROUP BY i.BillingCountry"}, {"role": "user", "content": "How many customers are there"}]

```

Add of existing embedding ID: 0658ba3d-98ff-51f4-9006-a24f87045858-sql

Insert of existing embedding ID: 0658ba3d-98ff-51f4-9006-a24f87045858-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:26:50.267923127Z', 'message': {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM "customers"', 'done_reason': 'stop', 'done': True, 'total_duration': 73945077797, 'load_duration': 958339, 'prompt_eval_count': 1681, 'prompt_eval_duration': 71552952000, 'eval_count': 8, 'eval_duration': 1764248000}
```

```
SELECT COUNT(*) FROM "customers"
```

```
SELECT COUNT(*) FROM "customers"
```

```
    COUNT(*)
```

```
0          59
```

Ollama parameters:

```
model=codegemma:latest,
```

```
options={},
```

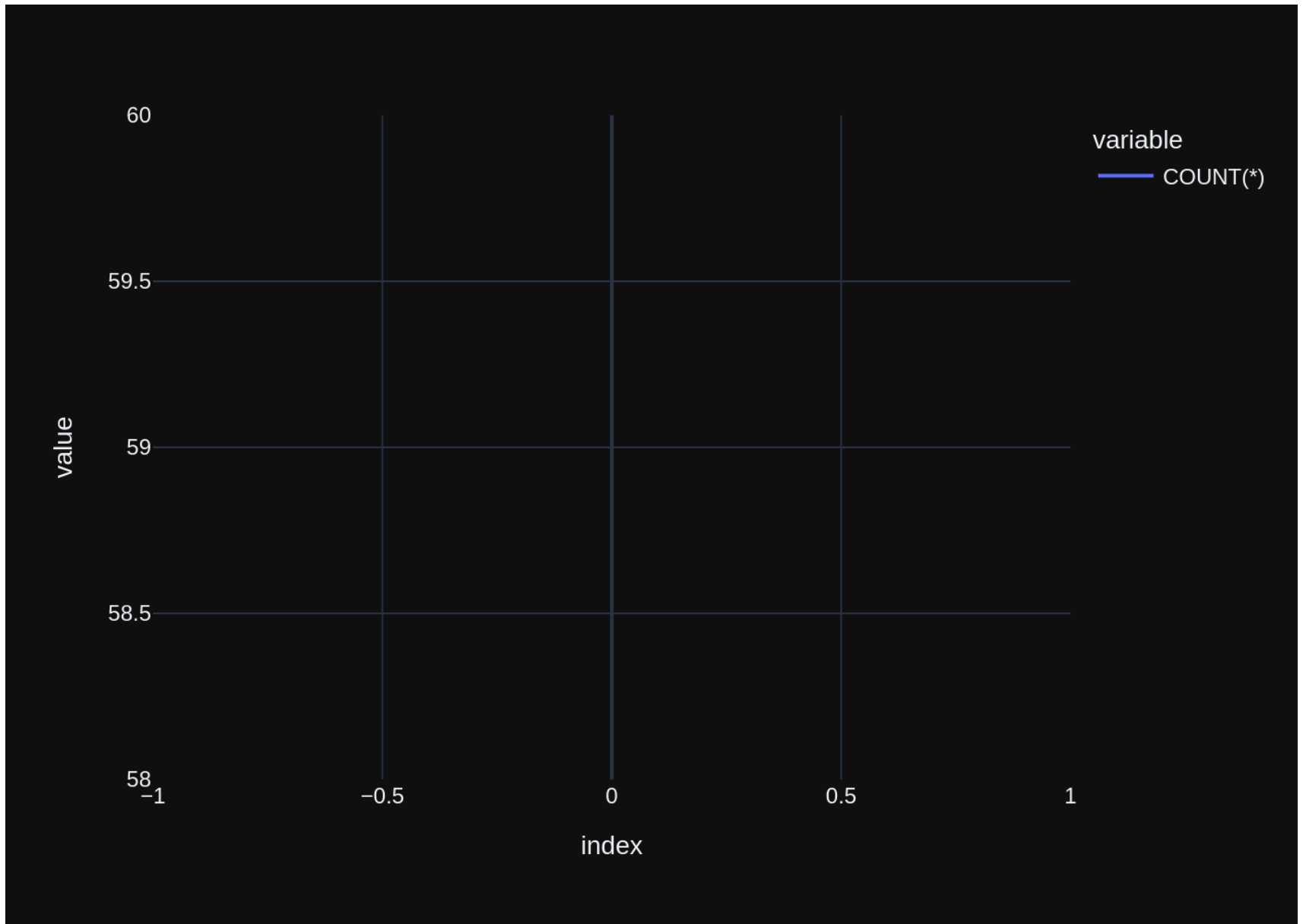
```
keep_alive=None
```

Prompt Content:

```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: 'How many customers are there'\n\nThe DataFrame was produced using this query: SELECT COUNT(*) FROM \"customers\"\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n COUNT(*)      int64\nndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:27:06.429904722Z', 'message': {'role': 'assistant', 'content': '\n\npython\nimport plotly.express as px\n\nfig = px.indicator(df, value='COUNT(*)', title='Number of Customers')\n\nfig.show()\n\n'}, 'done_reason': 'stop', 'done': True, 'total_duration': 16136673040, 'load_duration': 678517, 'prompt_eval_count': 117, 'prompt_eval_duration': 4852533000, 'eval_count': 45, 'eval_duration': 11193775000}
```



```

Out[19]: ('SELECT COUNT(*) FROM "customers"',
          COUNT(*)
          0      59,
          Figure({
            'data': [{'hovertemplate': 'variable=COUNT(*)<br>index=%{x}<br>value=%{y}<extra></extra>',
                      'legendgroup': 'COUNT(*)',
                      'line': {'color': '#636efa', 'dash': 'solid'},
                      'marker': {'symbol': 'circle'},
                      'mode': 'lines',
                      'name': 'COUNT(*)',
                      'orientation': 'v',
                      'showlegend': True,
                      'type': 'scatter',
                      'x': array([0]),
                      'xaxis': 'x',
                      'y': array([59]),
                      'yaxis': 'y'}],
            'layout': {'legend': {'title': {'text': 'variable'}, 'tracegroupgap': 0},
                      'margin': {'t': 60},
                      'template': '...',
                      'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'index'}},
                      'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'value'}}}
          )))

```

In []:

In [20]: `vn.ask(question="what are the top 5 countries that customers come from?")`

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

29/199

ery with a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before. \n'}, {'role': 'user', 'content': 'what are the top 5 countries that customers come from?'}, {'role': 'assistant', 'content': 'SELECT c.Count ry, COUNT(*) AS TotalCustomers\nFROM "customers" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Find the top 5 customers who spent the most money overall, \n \n Hint: order total can be found on invoices table, calculation using invoice_items detail table is un necessary \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM "cu stomers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DE SC\nLIMIT 5'}, {'role': 'user', 'content': ' \n List all customers from Canada and their email address es:\n'}, {'role': 'assistant', 'content': 'SELECT c.Email, c.Country\nFROM "customers" c\nWHERE c.Country = \'Canada\'}}, {'role': 'user', 'content': ' \n Hint: album quantity is found in invoice_items, \n \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "i nvoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Find the customer with the most invoices \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS T otalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n Hint: album quantity is found in invoice_items, \n \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS Total Albums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nOR DER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Find the total number of invoices per country:\n'}, {'role': 'assistant', 'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFRO M "invoices" i\nGROUP BY i.BillingCountry'}, {'role': 'user', 'content': ' \n Find the top 5 most expen sive tracks (based on unit price):\n'}, {'role': 'assistant', 'content': 'SELECT t.TrackId, t.Name, t.UnitP rice\nFROM "tracks" t\nORDER BY t.UnitPrice DESC\nLIMIT 5'}, {'role': 'user', 'content': 'How many customer s are there'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM "customers"'}, {'role': 'user', 'cont ent': ' \n Find the customer who bought the most albums in total quantity (across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(iii.TrackId) AS TotalAlbums\nFROM "customers" c \nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'role': 'user', 'content': 'what are the top 5 countries that customers come from?'}]

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and fo rmat instructions. \n===Tables\nCREATE TABLE \"invoices\"\n(\n    InvoiceId INTEGER PRIMARY KEY AUTOIN CREMENT NOT NULL,\n    CustomerId INTEGER NOT NULL,\n    InvoiceDate DATETIME NOT NULL,\n    Billin
```

```

gAddress NVARCHAR(70),\r\n    BillingCity NVARCHAR(40),\r\n    BillingState NVARCHAR(40),\r\n    BillingCou
ntry NVARCHAR(40),\r\n    BillingPostalCode NVARCHAR(10),\r\n    Total NUMERIC(10,2) NOT NULL,\r\n    FORE
IGN KEY (CustomerId) REFERENCES \"customers\" (CustomerId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION
\r\n)\n\nCREATE TABLE \"customers\"(\r\n    CustomerId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Fir
stName NVARCHAR(40) NOT NULL,\r\n    LastName NVARCHAR(20) NOT NULL,\r\n    Company NVARCHAR(80),\r\n    Ad
dress NVARCHAR(70),\r\n    City NVARCHAR(40),\r\n    State NVARCHAR(40),\r\n    Country NVARCHAR(40),\r\n
PostalCode NVARCHAR(10),\r\n    Phone NVARCHAR(24),\r\n    Fax NVARCHAR(24),\r\n    Email NVARCHAR(60) NOT
NULL,\r\n    SupportRepId INTEGER,\r\n    FOREIGN KEY (SupportRepId) REFERENCES \"employees\" (EmployeeId)
\r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"invoice_items\"(\r\n    InvoiceL
ineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    InvoiceId INTEGER NOT NULL,\r\n    TrackId INTEGE
R NOT NULL,\r\n    UnitPrice NUMERIC(10,2) NOT NULL,\r\n    Quantity INTEGER NOT NULL,\r\n    FOREIGN KE
Y (InvoiceId) REFERENCES \"invoices\" (InvoiceId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n    F
OREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n
\nCREATE TABLE \"media_types\"(\r\n    MediaTypeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    N
ame NVARCHAR(120)\r\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON \"customers\" (SupportRepId)\n\nCREATE T
ABLE \"employees\"(\r\n    EmployeeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    LastName NVARC
HAR(20) NOT NULL,\r\n    FirstName NVARCHAR(20) NOT NULL,\r\n    Title NVARCHAR(30),\r\n    ReportsTo INT
EGER,\r\n    BirthDate DATETIME,\r\n    HireDate DATETIME,\r\n    Address NVARCHAR(70),\r\n    City NVARCHA
R(40),\r\n    State NVARCHAR(40),\r\n    Country NVARCHAR(40),\r\n    PostalCode NVARCHAR(10),\r\n    Phone
NVARCHAR(24),\r\n    Fax NVARCHAR(24),\r\n    Email NVARCHAR(60),\r\n    FOREIGN KEY (ReportsTo) REFERENCES
\"employees\" (EmployeeId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"albums
\"(\r\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Title NVARCHAR(160) NOT NULL,\r\n
    ArtistId INTEGER NOT NULL,\r\n    FOREIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistId) \r\n\t\t\tO
N DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"playlist_track\"(\r\n    PlaylistId INTEG
ER NOT NULL,\r\n    TrackId INTEGER NOT NULL,\r\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistI
d, TrackId),\r\n    FOREIGN KEY (PlaylistId) REFERENCES \"playlists\" (PlaylistId) \r\n\t\t\tON DELETE NO ACT
ION ON UPDATE NO ACTION,\r\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\t\tON DELETE NO
ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE sqlite_sequence(name,seq)\n\nCREATE TABLE \"tracks\"(\r\n
    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Name NVARCHAR(200) NOT NULL,\r\n    Album
Id INTEGER,\r\n    MediaTypeId INTEGER NOT NULL,\r\n    GenreId INTEGER,\r\n    Composer NVARCHAR(220),\r\n
    Milliseconds INTEGER NOT NULL,\r\n    Bytes INTEGER,\r\n    UnitPrice NUMERIC(10,2) NOT NULL,\r\n
    FOREIGN KEY (AlbumId) REFERENCES \"albums\" (AlbumId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n
    FOREIGN KEY (GenreId) REFERENCES \"genres\" (GenreId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n
    FOREIGN KEY (MediaTypeId) REFERENCES \"media_types\" (MediaTypeId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO
ACTION\r\n)\n\n\n===Additional Context\n\nIn the SQLite database invoice means order\n\n===Response Guidel
ines\n1. If the provided context is sufficient, please generate a valid SQL query without any explanations
for the question.\n2. If the provided context is almost sufficient but requires knowledge of a specific st
ring in a particular column, please generate an intermediate SQL query to find the distinct strings in that
column. Prepend the query with a comment saying intermediate_sql\n3. If the provided context is insufficie
nt, please explain why it can't be generated.\n4. Please use the most relevant table(s).\n5. If the quest
ion has been asked and answered before, please repeat the answer exactly as it was given before.\n\"}, {\"ro
le\": \"user\", \"content\": \"what are the top 5 countries that customers come from?\"}, {\"role\": \"assistant\", \"c

```

```

ontent": "SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM \"customers\" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n      Find the top 5 customers who spent the most money overall, \n      \n      Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessary\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n      List all customers from Canada and their email addresses:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.Email, c.Country\nFROM \"customers\" c\nWHERE c.Country = 'Canada'\"}, {\"role\": \"user\", \"content\": \" \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n      Find the customer with the most invoices\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1\"}, {\"role\": \"user\", \"content\": \" \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n      Find the total number of invoices per country:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM \"invoices\" i\nGROUP BY i.BillingCountry\"}, {\"role\": \"user\", \"content\": \" \n      Find the top 5 most expensive tracks (based on unit price):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT t.TrackId, t.Name, t.UnitPrice\nFROM \"tracks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \"How many customers are there\"}, {\"role\": \"assistant\", \"content\": \"SELECT COUNT(*) FROM \"customers\"\"}, {\"role\": \"user\", \"content\": \" \n      Find the customer who bought the most albums in total quantity (across all invoices):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1\"}, {\"role\": \"user\", \"content\": \"what are the top 5 countries that customers come from?\"}]

```

Add of existing embedding ID: d8a37163-5ce5-58cd-a316-ea5598d44d27-sql

Insert of existing embedding ID: d8a37163-5ce5-58cd-a316-ea5598d44d27-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:28:30.487754072Z', 'message': {'role': 'assistant', 'content': 'SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM "customers" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5'}, 'done_reason': 'stop', 'done': True, 'total_duration': 83956663321, 'load_duration': 713196, 'prompt_eval_count': 1776, 'prompt_eval_duration': 74862620000, 'eval_count': 33, 'eval_duration': 8454393000}
```

```
SELECT c.Country, COUNT(*) AS TotalCustomers
FROM "customers" c
GROUP BY c.Country
ORDER BY TotalCustomers DESC
LIMIT 5
SELECT c.Country, COUNT(*) AS TotalCustomers
FROM "customers" c
GROUP BY c.Country
ORDER BY TotalCustomers DESC
LIMIT 5
```

	Country	TotalCustomers
0	USA	13
1	Canada	8
2	France	5
3	Brazil	5
4	Germany	4

Ollama parameters:

```
model=codegemma:latest,
options={},
keep_alive=None
```

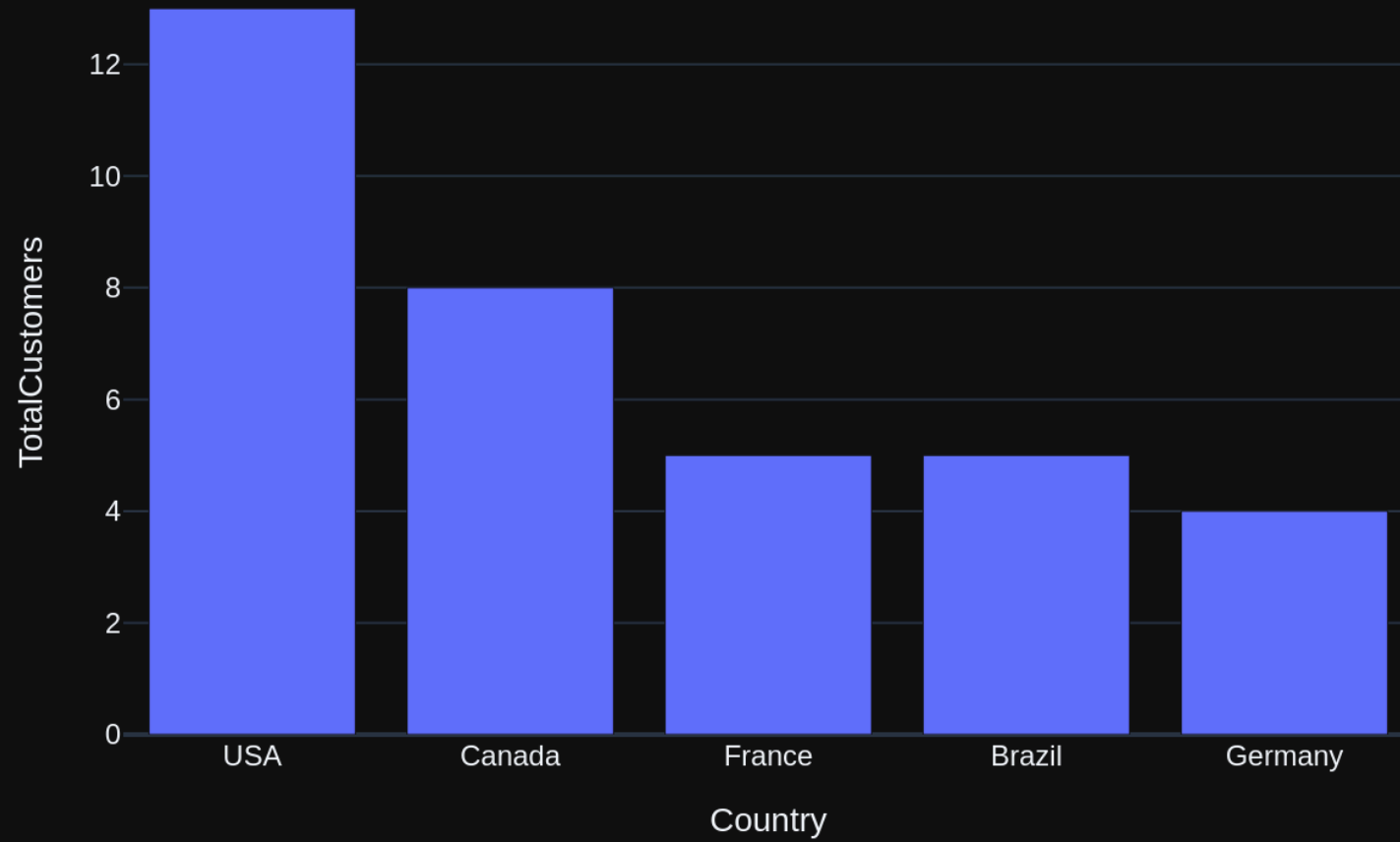
Prompt Content:

```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: 'what are the top 5 countries that customers come from?'\n\nThe DataFrame was produced using this query: SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM \"customers\" c\n\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n Country          object\nTotalCustomers    int64\nndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:28:49.347854475Z', 'message': {'role': 'assistant', 'content': "\n\npython\nimport plotly.express as px\n\nfig = px.bar(df, x='Country', y='TotalCustomers', title='Top 5 Countries of Customers')\n\nfig.show()\n\n"}, 'done_reason': 'stop', 'done': True, 'total_duration': 18834425107, 'load_duration': 695414, 'prompt_eval_count': 174, 'prompt_eval_duration': 7299655000, 'eval_count': 44, 'eval_duration': 11444574000}
```

Top 5 Countries of Customers



```
Out[20]: ('SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM "customers" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5',
Country TotalCustomers
0 USA 13
1 Canada 8
2 France 5
3 Brazil 5
4 Germany 4,
Figure({
  'data': [{'alignmentgroup': 'True',
            'hovertemplate': 'Country=%{x}<br>TotalCustomers=%{y}<extra></extra>',
            'legendgroup': '',
            'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
            'name': '',
            'offsetgroup': '',
            'orientation': 'v',
            'showlegend': False,
            'textposition': 'auto',
            'type': 'bar',
            'x': array(['USA', 'Canada', 'France', 'Brazil', 'Germany'], dtype=object),
            'xaxis': 'x',
            'y': array([13, 8, 5, 5, 4]),
            'yaxis': 'y'}],
  'layout': {'barmode': 'relative',
            'legend': {'tracegroupgap': 0},
            'template': '...',
            'title': {'text': 'Top 5 Countries of Customers'},
            'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Country'}},
            'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'TotalCustomers'}}}
}))
```

More SQL questions

see [sample-sql-queries-sqlite-chinook.ipynb](#)

```
In [21]: question = """
          List all albums and their corresponding artist names
          """

vn.ask(question=question)
```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

```
[{'role': 'system', 'content': 'You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables \nCREATE INDEX IFK_AlbumArtistId ON "albums" (ArtistId)\n\nCREATE TABLE "albums"\n\n(\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Title NVARCHAR(160) NOT NULL,\n    ArtistId INTEGER NOT NULL,\n    FOREIGN KEY (ArtistId) REFERENCES "artists" (ArtistId) \n\n)\n\nCREATE TABLE "tracks"\n\n(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES "albums" (AlbumId) \n\n)\n\nCREATE TABLE "artists"\n\n(\n    ArtistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n\n)\n\nCREATE TABLE "genres"\n\n(\n    GenreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n\n)\n\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\n\nCREATE INDEX IFK_TrackGenreId ON "tracks" (GenreId)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON "playlist_track" (TrackId)\n\nCREATE TABLE "playlists"\n\n(\n    PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n\n)\n\nCREATE INDEX IFK_TrackMediaTypeId ON "tracks" (MediaTypeId)\n\n\n===Additional Context \n\nIn the SQLite database invoice means order\n\n\n===Response Guidelines \n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question. \n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before. \n}', {'role': 'user', 'content': ' \n    List all albums and their corresponding artist names \n'}, {'role': 'assistant', 'content': 'SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM "albums" a\nJOIN "artists" ar ON a.ArtistId = ar.ArtistId'}, {'role': 'user', 'content': ' \n    There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\n    Can you find the top 10 most popular artists based on the number of tracks\n'}, {'role': 'assistant', 'content': 'SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nJOIN "tracks" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10'}, {'role': 'user', 'content': ' \n    Identify artists who have albums with tracks appearing in multiple genres:\n\n'}, {'role': 'assistant', 'content': 'SELECT a.ArtistId, a.Name AS ArtistName\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n    SELECT g2.GenreId\n    FROM "genres" g2\n    JOIN "tracks" t2 ON g2.GenreId = t2.GenreId\n    GROUP BY g2.GenreId\n    HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name'}, {'role': 'user', 'content': ' \n    Hint: album quantity is found in invoice_items, \n    Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n    Find the customer who bought the most albums in total quantity (across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.'}
```

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables \nCREATE INDEX IFK_AlbumArtistId ON \"albums\" (ArtistId)\n\nCREATE TABLE \"albums\"(\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Title NVARCHAR(160) NOT NULL,\n    ArtistId INTEGER NOT NULL,\n    FOREIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistId) \n\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n\nCREATE TABLE \"tracks\"(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES \"albums\" (AlbumId) \n\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES \"genres\" (GenreId) \n\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES \"media_types\" (MediaTypeId) \n\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n\nCREATE INDEX IFK_TrackAlbumId ON \"tracks\" (AlbumId)\n\nCREATE TABLE \"artists\"(\n    ArtistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n\nCREATE INDEX IFK_TrackGenreId ON \"tracks\" (GenreId)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON \"playlist_track\" (TrackId)\n\nCREATE TABLE \"playlists\"(\n    PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n\nCREATE TABLE \"genres\"(\n    GenreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n\nCREATE INDEX IFK_TrackMediaTypeId ON \"tracks\" (MediaTypeId)\n\n===Additional Context \n\nIn the SQLite database invoice means order\n\n===Response Guidelines \n1. If the provided conte
```

xt is sufficient, please generate a valid SQL query without any explanations for the question. \n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before. \n"}, {"role": "user", "content": "\n List all albums and their corresponding artist names \n"}, {"role": "assistant", "content": "SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM \"albums\" a\nJOIN \"artists\" ar ON a.ArtistId = ar.ArtistId"}, {"role": "user", "content": "\n There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\n Can you find the top 10 most popular artists based on the number of tracks\n"}, {"role": "assistant", "content": "SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nJOIN \"tracks\" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10"}, {"role": "user", "content": "\n Identify artists who have albums with tracks appearing in multiple genres:\n\n\n"}, {"role": "assistant", "content": "SELECT a.ArtistId, a.Name AS ArtistName\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n SELECT g2.GenreId\n FROM \"genres\" g2\n JOIN \"tracks\" t2 ON g2.GenreId = t2.GenreId\n GROUP BY g2.GenreId\n HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name"}, {"role": "user", "content": "\n Hint: album quantity is found in invoice_items, \n\n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": "\n Find the customer who bought the most albums in total quantity (across all invoices): \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1"}, {"role": "user", "content": "\n Hint: album quantity is found in invoice_items, \n\n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": "\n List all genres and the number of tracks in each genre:\n"}, {"role": "assistant", "content": "SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM \"genres\" g\nJOIN \"tracks\" t ON g.GenreId = t.GenreId\nGROUP BY g.Name"}, {"role": "user", "content": "\n Get all playlists containing at least 10 tracks and the total duration of those tracks:\n"}, {"role": "assistant", "content": "SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM \"playlist_track\" pt\nJOIN \"playlists\" p ON pt.PlaylistId = p.PlaylistId\nJOIN \"tracks\" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10"}, {"role": "user", "content": "\n Find the top 5 most expensive tracks (based on unit price):\n"}, {"role": "assistant", "content": "SELECT t.TrackId, t.Name, t.UnitPrice\nFROM \"tracks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5"}, {"role": "user", "content": "Can you list all tables in the SQLite database catalog?"}, {"role": "assistant", "content": "SELECT name FROM sqlite_master WHERE type='table'"}, {"role": "user", "content": "\n List all albums and their corresponding artist names \n"}]

Add of existing embedding ID: a7185c88-7417-5b75-a52e-4eae5f9deca-sql
Insert of existing embedding ID: a7185c88-7417-5b75-a52e-4eae5f9deca-sql


```
Ollama Response:
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:30:10.206279662Z', 'message': {'role': 'assistant', 'content': 'SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM "albums" a\nJOIN "artists" ar ON a.ArtistId = ar.ArtistId'}, 'done_reason': 'stop', 'done': True, 'total_duration': 80763921205, 'load_duration': 685847, 'prompt_eval_count': 1586, 'prompt_eval_duration': 70035942000, 'eval_count': 39, 'eval_duration': 10091699000}

SELECT a.Title, a.ArtistId, ar.Name AS ArtistName
FROM "albums" a
JOIN "artists" ar ON a.ArtistId = ar.ArtistId
SELECT a.Title, a.ArtistId, ar.Name AS ArtistName
FROM "albums" a
JOIN "artists" ar ON a.ArtistId = ar.ArtistId

      Title  ArtistId  \
0      For Those About To Rock We Salute You      1
1      Balls to the Wall      2
2      Restless and Wild      2
3      Let There Be Rock      1
4      Big Ones      3
..      ...      ...
342      Respighi:Pines of Rome      226
343 Schubert: The Late String Quartets & String Qu...      272
344      Monteverdi: L'Orfeo      273
345      Mozart: Chamber Music      274
346 Koyaanisqatsi (Soundtrack from the Motion Pict...      275

      ArtistName
0      AC/DC
1      Accept
2      Accept
3      AC/DC
4      Aerosmith
..      ...
342      Eugene Ormandy
343      Emerson String Quartet
344 C. Monteverdi, Nigel Rogers - Chiaroscuro; Lon...
345      Nash Ensemble
346      Philip Glass Ensemble

[347 rows x 3 columns]
Ollama parameters:
model=codegemma:latest,
options={},
```

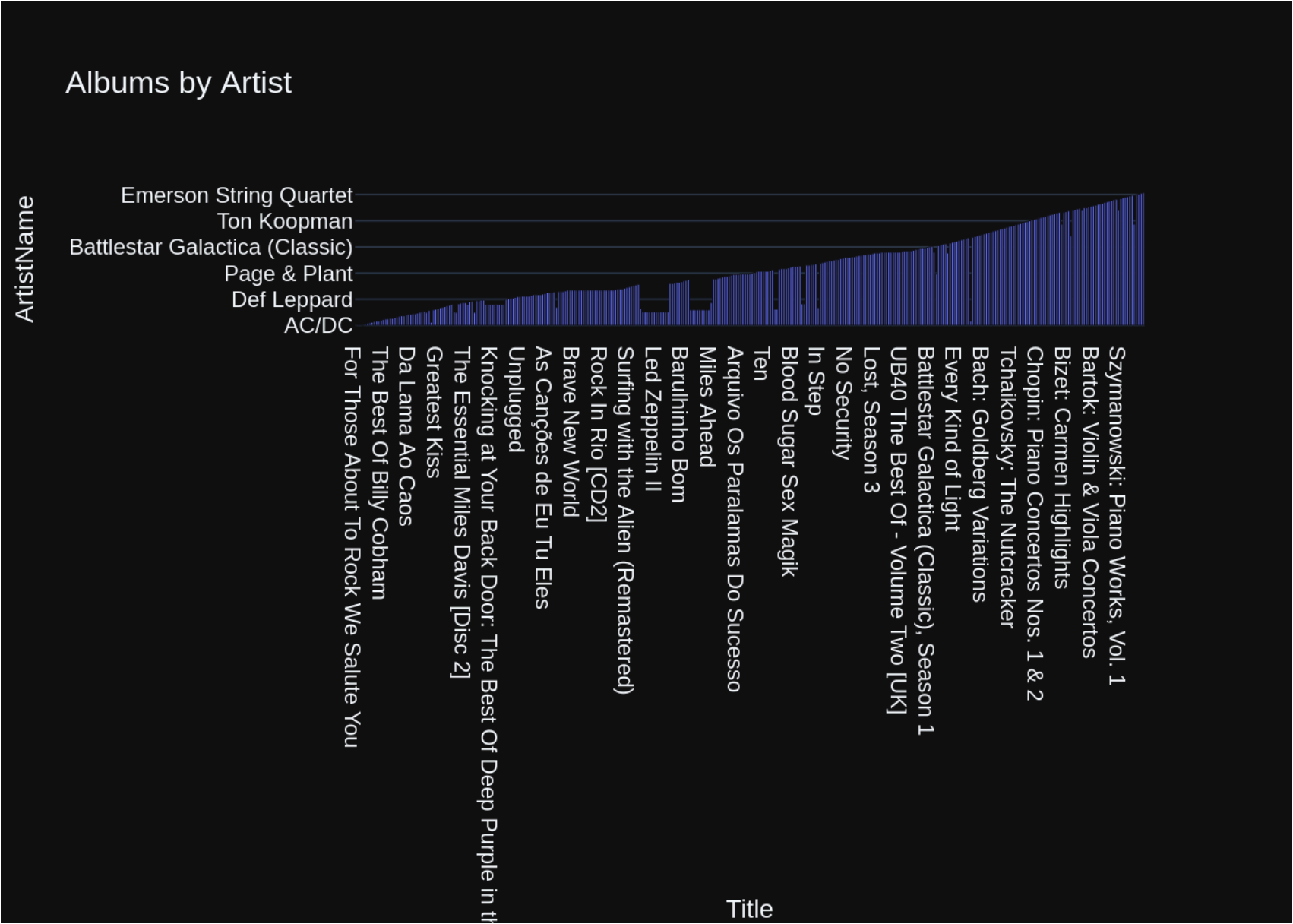
keep_alive=None

Prompt Content:

```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n      List all albums and their corresponding artist names \n'\n\nThe DataFrame was produced using this query: SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM \n\"albums\" a\nJOIN \"artists\" ar ON a.ArtistId = ar.ArtistId\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n Title          object\nArtistId        int64\nArtistName      object\nndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:30:29.114851912Z', 'message': {'role': 'assistant', 'content': "\n\npython\nimport plotly.express as px\n\nfig = px.bar(df, x='Title', y='ArtistName', title='Albums by Artist')\nfig.show()\n\n"}, 'done_reason': 'stop', 'done': True, 'total_duration': 18880064358, 'load_duration': 626179, 'prompt_eval_count': 187, 'prompt_eval_duration': 8143825000, 'eval_count': 41, 'eval_duration': 10647469000}
```



```
Out[21]: ('SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM "albums" a\nJOIN "artists" ar ON a.ArtistId = a\nr.ArtistId',
```

	Title	ArtistId	\
0	For Those About To Rock We Salute You	1	
1	Balls to the Wall	2	
2	Restless and Wild	2	
3	Let There Be Rock	1	
4	Big Ones	3	
..	
342	Respighi:Pines of Rome	226	
343	Schubert: The Late String Quartets & String Qu...	272	
344	Monteverdi: L'Orfeo	273	
345	Mozart: Chamber Music	274	
346	Koyaanisqatsi (Soundtrack from the Motion Pict...	275	

	ArtistName
0	AC/DC
1	Accept
2	Accept
3	AC/DC
4	Aerosmith
..	...
342	Eugene Ormandy
343	Emerson String Quartet
344	C. Monteverdi, Nigel Rogers - Chiaroscuro; Lon...
345	Nash Ensemble
346	Philip Glass Ensemble

```
[347 rows x 3 columns],
```

```
Figure({
  'data': [{'alignmentgroup': 'True',
            'hovertemplate': 'Title=%{x}<br>ArtistName=%{y}<extra></extra>',
            'legendgroup': '',
            'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
            'name': '',
            'offsetgroup': '',
            'orientation': 'v',
            'showlegend': False,
            'textposition': 'auto',
            'type': 'bar',
            'x': array(['For Those About To Rock We Salute You', 'Balls to the Wall',
                       'Restless and Wild', ..., 'Monteverdi: L'Orfeo',
```

```

        'Mozart: Chamber Music',
        'Koyaanisqatsi (Soundtrack from the Motion Picture)'], dtype=object),
    'xaxis': 'x',
    'y': array(['AC/DC', 'Accept', 'Accept', ...,
               'C. Monteverdi, Nigel Rogers - Chiaroscuro; London Baroque; London Cornett & Sa
ckbu',
               'Nash Ensemble', 'Philip Glass Ensemble'], dtype=object),
    'yaxis': 'y'}],
    'layout': {'barmode': 'relative',
               'legend': {'tracegroupgap': 0},
               'template': '...',
               'title': {'text': 'Albums by Artist'},
               'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Title'}},
               'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'ArtistName'}}}
    )))

```

```

In [22]: question = """
        Find all tracks with a name containing "What" (case-insensitive)
        """

        vn.ask(question=question)

```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

```
[{'role': 'system', 'content': 'You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables \nCREATE INDEX IFK_TrackGenreId ON "tracks" (GenreId)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON "playlist_track" (TrackId)\n\nCREATE TABLE "tracks"\n(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES "albums" (AlbumId) \n    \n    FOREIGN KEY (GenreId) REFERENCES "genres" (GenreId) \n    \n    FOREIGN KEY (MediaTypeId) REFERENCES "media_types" (MediaTypeId) \n    \n)\n\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\n\nCREATE INDEX IFK_TrackMediaTypeId ON "tracks" (MediaTypeId)\n\nCREATE TABLE "playlist_track"\n(\n    PlaylistId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId) \n    \n    FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \n    \n)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\n\nCREATE INDEX IFK_AlbumArtistId ON "albums" (ArtistId)\n\nCREATE TABLE "playlists"\n(\n    PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\n\nCREATE TABLE "genres"\n(\n    GenreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\n\n===Additional Context \n\nIn the SQLite database invoice means order\n\n===Response Guidelines \n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question. \n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before. \n'}], {'role': 'user', 'content': ' \n    Find all tracks with a name containing "What" (case-insensitive)\n'}, {'role': 'assistant', 'content': 'SELECT * \nFROM "tracks" \nWHERE LOWER(Name) LIKE \'%what%\''}, {'role': 'user', 'content': ' \n    Get all playlists containing at least 10 tracks and the total duration of those tracks:\n'}, {'role': 'assistant', 'content': 'SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM "playlist_track" pt\nJOIN "playlists" p ON pt.PlaylistId = p.PlaylistId\nJOIN "tracks" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10'}, {'role': 'user', 'content': ' \n    Identify artists who have albums with tracks appearing in multiple genres:\n\n\n'}, {'role': 'assistant', 'content': 'SELECT a.ArtistId, a.Name AS ArtistName\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n    SELECT g2.GenreId\n    FROM "genres" g2\n    JOIN "tracks" t2 ON g2.GenreId = t2.GenreId\n    GROUP BY g2.GenreId\n    HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name'}, {'role': 'user', 'content': ' \n    List all genres and the number of tracks in each genre:\n'}, {'role': 'assistant', 'content': 'SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM "genres" g\nJOIN "tracks" t ON g.GenreId = t.GenreId\nGROUP BY g.Name'}, {'role': 'user', 'content': ' \n    Find the top 5 most expensive tracks (based on unit price):\n'}, {'role': 'assistant', 'content': 'SELECT t.TrackId, t.Name, t.UnitPrice\nFROM "tracks" t\nORDER BY t.UnitPrice DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n    There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\n    Can you find the top 10 most
```

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions."}]

Tables

CREATE INDEX IFK_TrackGenreId ON `tracks` (GenreId)

CREATE INDEX IFK_PlaylistTrackTrackId ON `playlist_track` (TrackId)

CREATE TABLE `tracks` (
  TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
  Name NVARCHAR(200) NOT NULL,
  AlbumId INTEGER,
  MediaTypeId INTEGER NOT NULL,
  GenreId INTEGER,
  Composer NVARCHAR(220),
  Milliseconds INTEGER NOT NULL,
  Bytes INTEGER,
  UnitPrice NUMERIC(10,2) NOT NULL,
  FOREIGN KEY (AlbumId) REFERENCES `albums` (AlbumId) ON DELETE NO ACTION ON UPDATE NO ACTION,
  FOREIGN KEY (GenreId) REFERENCES `genres` (GenreId) ON DELETE NO ACTION ON UPDATE NO ACTION,
  FOREIGN KEY (MediaTypeId) REFERENCES `media_types` (MediaTypeId) ON DELETE NO ACTION ON UPDATE NO ACTION
)

CREATE INDEX IFK_TrackAlbumId ON `tracks` (AlbumId)

CREATE INDEX IFK_TrackMediaTypeId ON `tracks` (MediaTypeId)

CREATE TABLE `playlist_track` (
  PlaylistId INTEGER NOT NULL,
  TrackId INTEGER NOT NULL,
  CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),
  FOREIGN KEY (PlaylistId) REFERENCES `playlists` (PlaylistId) ON DELETE NO ACTION ON UPDATE NO ACTION,
  FOREIGN KEY (TrackId) REFERENCES `tracks` (TrackId) ON DELETE NO ACTION ON UPDATE NO ACTION
)

CREATE INDEX IFK_InvoiceLineTrackId ON `invoice_items` (TrackId)

CREATE INDEX IFK_AlbumArtistId ON `albums` (ArtistId)

CREATE TABLE `playlists` (
  PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
  Name NVARCHAR(120)
)

CREATE TABLE `genres` (
  GenreId INTEGER PRIMARY KEY
```

EY AUTOINCREMENT NOT NULL,\r\n Name NVARCHAR(120)\r\n)\n\n\n===Additional Context \n\nIn the SQLite data base invoice means order\n\n\n===Response Guidelines \n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question. \n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before. \n"}, {"role": "user", "content": " \n Find all tracks with a name containing \"What\" (case-insensitive)\n"}, {"role": "assistant", "content": "SELECT * \nFROM \"tracks\" \nWHERE LOWER(Name) LIKE '%what%'"}, {"role": "user", "content": " \n Get all playlists containing at least 10 tracks and the total duration of those tracks:\n"}, {"role": "assistant", "content": "SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM \"playlist_track\" pt\nJOIN \"playlists\" p ON pt.PlaylistId = p.PlaylistId\nJOIN \"tracks\" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10"}, {"role": "user", "content": " \n Identify artists who have albums with tracks appearing in multiple genres:\n\n\n"}, {"role": "assistant", "content": "SELECT a.ArtistId, a.Name AS ArtistName\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n SELECT g2.GenreId\n FROM \"genres\" g2\n JOIN \"tracks\" t2 ON g2.GenreId = t2.GenreId\n GROUP BY g2.GenreId\n HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name"}, {"role": "user", "content": " \n List all genres and the number of tracks in each genre:\n"}, {"role": "assistant", "content": "SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM \"genres\" g\nJOIN \"tracks\" t ON g.GenreId = t.GenreId\nGROUP BY g.Name"}, {"role": "user", "content": " \n Find the top 5 most expensive tracks (based on unit price):\n"}, {"role": "assistant", "content": "SELECT t.TrackId, t.Name, t.UnitPrice\nFROM \"tracks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5"}, {"role": "user", "content": " \n There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\n Can you find the top 10 most popular artists based on the number of tracks\n"}, {"role": "assistant", "content": "SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nJOIN \"tracks\" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10"}, {"role": "user", "content": " \n Find the customer who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1"}, {"role": "user", "content": " \n List all albums and their corresponding artist names \n"}, {"role": "assistant", "content": "SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM \"albums\" a\nJOIN \"artists\" ar ON a.ArtistId = ar.ArtistId"}, {"role": "user", "content": " \n Hint: album quantity is found in invoice_items, \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": " \n Hint: album quantity is found in invoice_items, \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}


```
N i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user",  
"content": " \n    Find all tracks with a name containing \"What\" (case-insensitive)\n"}]
```

Add of existing embedding ID: 6f22268c-5062-5f11-ba2d-8555f06b409d-sql

Insert of existing embedding ID: 6f22268c-5062-5f11-ba2d-8555f06b409d-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:31:45.769027519Z', 'message': {'role': 'assistant', 'content': 'SELECT * \nFROM "tracks" \nWHERE LOWER(Name) LIKE \'%what%\'}', 'done_reason': 'stop', 'done': True, 'total_duration': 76415815336, 'load_duration': 694892, 'prompt_eval_count': 1627, 'prompt_eval_duration': 70717840000, 'eval_count': 20, 'eval_duration': 5057237000}
```

```
SELECT *
FROM "tracks"
WHERE LOWER(Name) LIKE '%what%'
SELECT *
FROM "tracks"
WHERE LOWER(Name) LIKE '%what%'
```

	TrackId	Name	AlbumId	\
0	26	What It Takes	5	
1	88	What You Are	10	
2	130	Do what cha wanna	13	
3	342	What is and Should Never Be	30	
4	607	So What	48	
5	960	What A Day	76	
6	1000	What If I Do?	80	
7	1039	What Now My Love	83	
8	1145	Whatsername	89	
9	1440	Whatever It Is, I Just Can't Stop	116	
10	1469	Look What You've Done	119	
11	1470	Get What You Need	119	
12	1628	What Is And What Should Never Be	133	
13	1778	You're What's Happening (In The World Today)	146	
14	1823	So What	149	
15	2772	I Don't Know What To Do With Myself	223	
16	2884	What Kate Did	231	
17	2893	Whatever the Case May Be	230	
18	2992	I Still Haven't Found What I'm Looking for	237	
19	3007	I Still Haven't Found What I'm Looking For	238	
20	3258	Whatever Gets You Thru the Night	255	
21	3475	What Is It About Men	322	

	MediaTypeId	GenreId	Composer	\
0	1	1	Steven Tyler, Joe Perry, Desmond Child	
1	1	1	Audioslave/Chris Cornell	
2	1	2	George Duke	
3	1	1	Jimmy Page/Robert Plant	
4	1	2	Miles Davis	
5	1	1	Mike Bordin, Billy Gould, Mike Patton	

6	1	1	Dave Grohl, Taylor Hawkins, Nate Mendel, Chris...
7	1	12	carl sigman/gilbert becaud/pierre leroyer
8	1	4	Green Day
9	1	1	Jay Kay/Kay, Jay
10	1	4	N. Cester
11	1	4	C. Cester/C. Muncey/N. Cester
12	1	1	Jimmy Page, Robert Plant
13	1	14	Allen Story/George Gordy/Robert Gordy
14	1	3	Culmer/Exalt
15	1	7	None
16	3	19	None
17	3	19	None
18	1	1	Bono/Clayton, Adam/Mullen Jr., Larry/The Edge
19	1	1	U2
20	2	9	None
21	2	9	Delroy "Chris" Cooper, Donovan Jackson, Earl C...

	Milliseconds	Bytes	UnitPrice
0	310622	10144730	0.99
1	249391	5988186	0.99
2	274155	9018565	0.99
3	260675	8497116	0.99
4	564009	18360449	0.99
5	158275	5203430	0.99
6	302994	9929799	0.99
7	149995	4913383	0.99
8	252316	8244843	0.99
9	247222	8249453	0.99
10	230974	7517083	0.99
11	247719	8043765	0.99
12	287973	9369385	0.99
13	142027	4631104	0.99
14	189152	6162894	0.99
15	221387	7251478	0.99
16	2610250	484583988	1.99
17	2616410	183867185	1.99
18	353567	11542247	0.99
19	280764	9306737	0.99
20	215084	3499018	0.99
21	209573	3426106	0.99

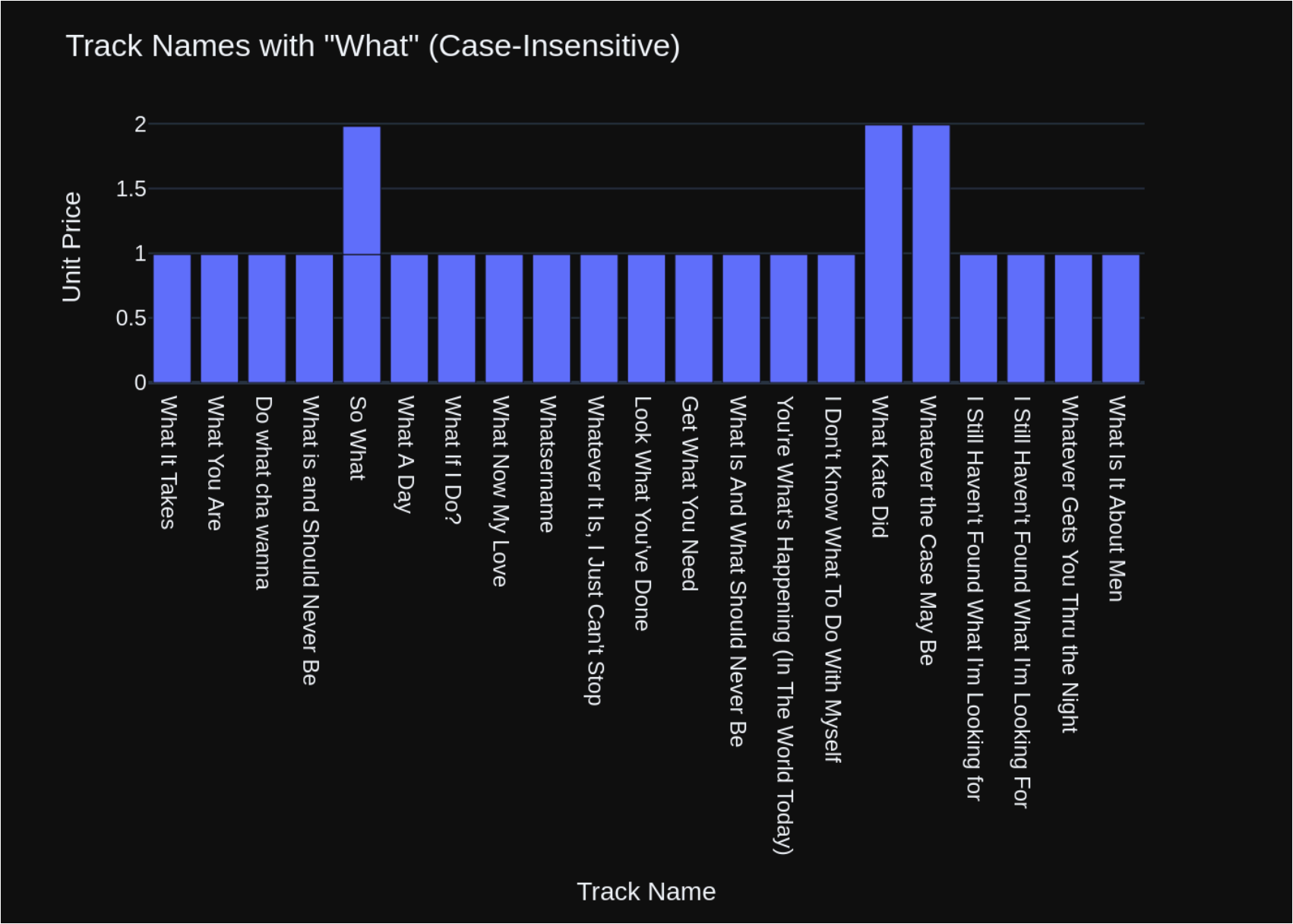
Ollama parameters:

model=codegemma:latest,

```

options={},
keep_alive=None
Prompt Content:
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n Find all tracks with a name containing \"What\" (case-insensitive)\n'\n\nThe DataFrame was produced using this query: SELECT * \nFROM \"tracks\" \nWHERE LOWER(Name) LIKE '%what%'\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n TrackId          int64\nName              object\nAlbumId          int64\nMediaTypeId      int64\nGenreId          int64\nComposer          object\nMilliseconds      int64\nBytes            int64\nUnitPrice        float64\nndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
Ollama Response:
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:32:13.22540461Z', 'message': {'role': 'assistant', 'content': '\n\npython\nimport plotly.express as px\n\nfig = px.bar(df, x=\'Name\', y=\'UnitPrice\')\n\nfig.update_layout(\n    title=\'Track Names with "What" (Case-Insensitive)\',\n    xaxis_title=\'Track Name\',\n    yaxis_title=\'Unit Price\'\n)\n\nfig.show()\n\n'}, 'done_reason': 'stop', 'done': True, 'total_duration': 27429692524, 'load_duration': 635948, 'prompt_eval_count': 212, 'prompt_eval_duration': 8487810000, 'eval_count': 77, 'eval_duration': 18846163000}

```

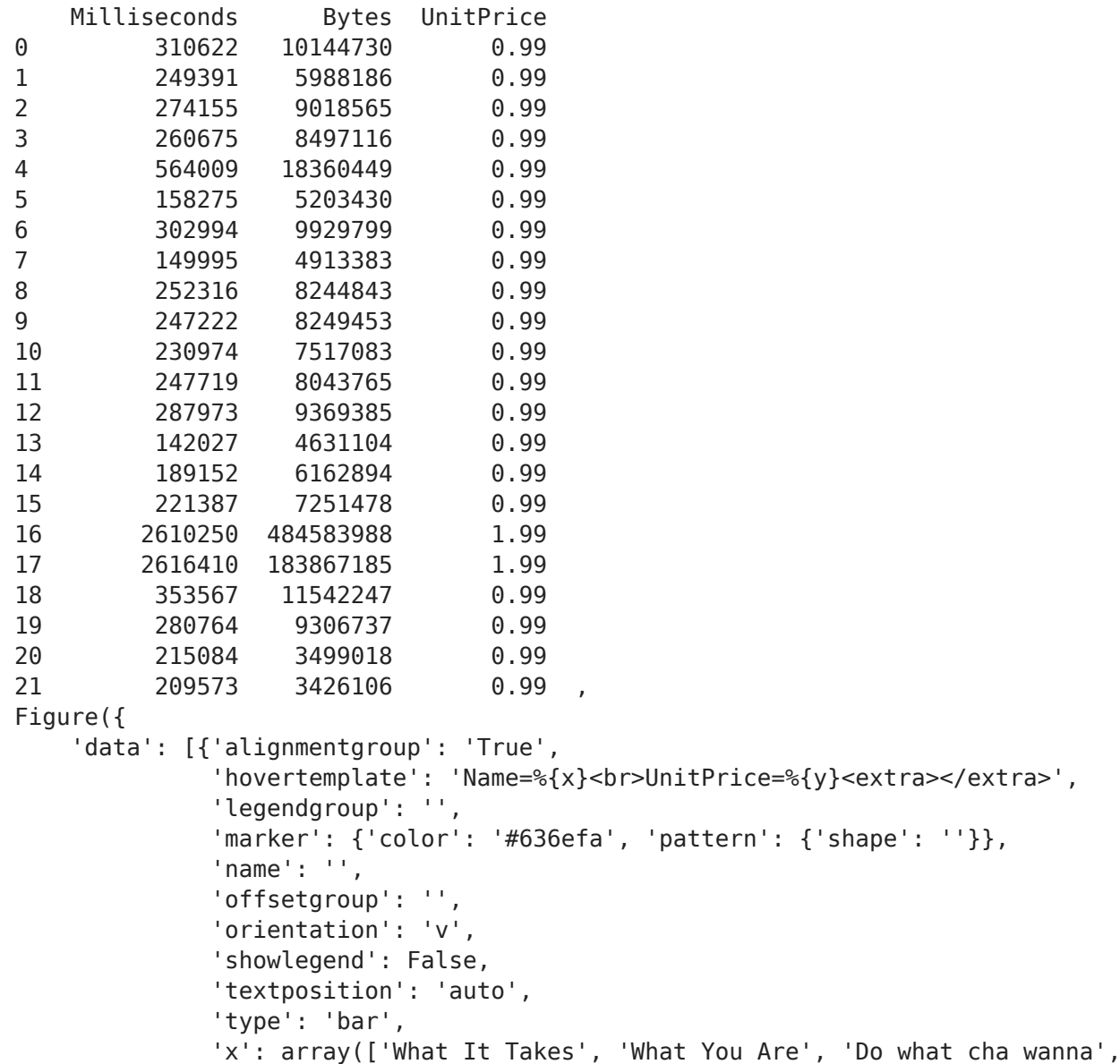


```
Out[22]: ('SELECT * \nFROM "tracks" \nWHERE LOWER(Name) LIKE \'%what%\',
```

	TrackId	Name	AlbumId \
0	26	What It Takes	5
1	88	What You Are	10
2	130	Do what cha wanna	13
3	342	What is and Should Never Be	30
4	607	So What	48
5	960	What A Day	76
6	1000	What If I Do?	80
7	1039	What Now My Love	83
8	1145	Whatsername	89
9	1440	Whatever It Is, I Just Can't Stop	116
10	1469	Look What You've Done	119
11	1470	Get What You Need	119
12	1628	What Is And What Should Never Be	133
13	1778	You're What's Happening (In The World Today)	146
14	1823	So What	149
15	2772	I Don't Know What To Do With Myself	223
16	2884	What Kate Did	231
17	2893	Whatever the Case May Be	230
18	2992	I Still Haven't Found What I'm Looking for	237
19	3007	I Still Haven't Found What I'm Looking For	238
20	3258	Whatever Gets You Thru the Night	255
21	3475	What Is It About Men	322

	MediaTypeId	GenreId	Composer \
0	1	1	Steven Tyler, Joe Perry, Desmond Child
1	1	1	Audioslave/Chris Cornell
2	1	2	George Duke
3	1	1	Jimmy Page/Robert Plant
4	1	2	Miles Davis
5	1	1	Mike Bordin, Billy Gould, Mike Patton
6	1	1	Dave Grohl, Taylor Hawkins, Nate Mendel, Chris...
7	1	12	carl sigman/gilbert becaud/pierre leroyer
8	1	4	Green Day
9	1	1	Jay Kay/Kay, Jay
10	1	4	N. Cester
11	1	4	C. Cester/C. Muncey/N. Cester
12	1	1	Jimmy Page, Robert Plant
13	1	14	Allen Story/George Gordy/Robert Gordy
14	1	3	Culmer/Exalt
15	1	7	None

16	3	19	None
17	3	19	None
18	1	1	Bono/Clayton, Adam/Mullen Jr., Larry/The Edge
19	1	1	U2
20	2	9	None
21	2	9	Delroy "Chris" Cooper, Donovan Jackson, Earl C...



```

        'What is and Should Never Be', 'So What', 'What A Day', 'What If I Do?',
        'What Now My Love', 'Whatsername', "Whatever It Is, I Just Can't Stop",
        "Look What You've Done", 'Get What You Need',
        'What Is And What Should Never Be',
        "You're What's Happening (In The World Today)", 'So What',
        "I Don't Know What To Do With Myself", 'What Kate Did',
        'Whatever the Case May Be',
        "I Still Haven't Found What I'm Looking for",
        "I Still Haven't Found What I'm Looking For",
        'Whatever Gets You Thru the Night', 'What Is It About Men'],
        dtype=object),
    'xaxis': 'x',
    'y': array([0.99, 0.99, 0.99, 0.99, 0.99, 0.99, 0.99, 0.99, 0.99, 0.99, 0.99, 0.99,
                0.99, 0.99, 0.99, 0.99, 1.99, 1.99, 0.99, 0.99, 0.99, 0.99]),
    'yaxis': 'y'}],
    'layout': {'barmode': 'relative',
               'legend': {'tracegroupgap': 0},
               'margin': {'t': 60},
               'template': '...',
               'title': {'text': 'Track Names with "What" (Case-Insensitive)'},
               'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Track Name'}},
               'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'Unit Price'}}}
    )))

```

```

In [23]: question = """
        Get the total number of invoices for each customer
        """

        vn.ask(question=question)

```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1


```
M "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}}, {'role': 'use
```

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables \nCREATE TABLE \"invoices\" \r\n\r\n    InvoiceId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    CustomerId INTEGER NOT NULL,\r\n    InvoiceDate DATETIME NOT NULL,\r\n    BillingAddress NVARCHAR(70),\r\n    BillingCity NVARCHAR(40),\r\n    BillingState NVARCHAR(40),\r\n    BillingCountry NVARCHAR(40),\r\n    BillingPostalCode NVARCHAR(10),\r\n    Total NUMERIC(10,2) NOT NULL,\r\n    FOREIGN KEY (CustomerId) REFERENCES \"customers\" (CustomerId) \r\n\r\nON DELETE NO ACTION ON UPDATE NO ACTION \r\n\r\n)\n\nCREATE INDEX IFK InvoiceCustomerId ON \"invoices\" (CustomerId)\n\nCREATE INDEX IFK InvoiceLineInv"}]
```

```

dia_types\" (MediaTypeId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\n===Additional Context\n\nIn the SQLite database invoice means order\n\n===Response Guidelines \n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question. \n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before. \n\"}, {\"role\": \"user\", \"content\": \" \n    Get the total number of invoices for each customer\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\"}, {\"role\": \"user\", \"content\": \" \n    Find the customer with the most invoices\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1\"}, {\"role\": \"user\", \"content\": \" \n    Get the average invoice total for each customer:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\"}, {\"role\": \"user\", \"content\": \" \n    Find the total number of invoices per country:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM \"invoices\" i\nGROUP BY i.BillingCountry\",

```

```
{
  "role": "user",
  "content": " \n      Find the customer who bought the most albums in total quantity (across all invoices):\n"},
  {
    "role": "assistant",
    "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1",
    "role": "user",
    "content": " \n      Find all invoices since 2010 and the total amount invoiced:\n"},
  {
    "role": "assistant",
    "content": "SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\nFROM \"invoices\" i\nWHERE i.InvoiceDate >= '2010-01-01'\nGROUP BY i.InvoiceDate",
    "role": "user",
    "content": " \n      Find the top 5 customers who spent the most money overall, \n      \n      Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessary\n"},
  {
    "role": "assistant",
    "content": "SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5",
    "role": "user",
    "content": " \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"},
  {
    "role": "assistant",
    "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5",
    "role": "user",
    "content": " \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"},
  {
    "role": "assistant",
    "content": "SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5",
    "role": "user",
    "content": " \n      List all invoices with a total exceeding $10:\n"},
  {
    "role": "assistant",
    "content": "SELECT * \nFROM \"invoices\" \nWHERE Total > 10.00",
    "role": "user",
    "content": " \n      Get the total number of invoices for each customer\n"}]
```

Add of existing embedding ID: 49e67df3-a604-51f8-ad01-b8f5a2043eac-sql

Insert of existing embedding ID: 49e67df3-a604-51f8-ad01-b8f5a2043eac-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:33:42.15362354Z', 'message': {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, 'done_reason': 'stop', 'done': True, 'total_duration': 88815844935, 'load_duration': 738381, 'prompt_eval_count': 1872, 'prompt_eval_duration': 77487299000, 'eval_count': 42, 'eval_duration': 10691605000}
```

```
SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices
FROM "customers" c
JOIN "invoices" i ON c.CustomerId = i.CustomerId
GROUP BY c.CustomerId
SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices
FROM "customers" c
JOIN "invoices" i ON c.CustomerId = i.CustomerId
GROUP BY c.CustomerId
```

	CustomerId	TotalInvoices
0	1	7
1	2	7
2	3	7
3	4	7
4	5	7
5	6	7
6	7	7
7	8	7
8	9	7
9	10	7
10	11	7
11	12	7
12	13	7
13	14	7
14	15	7
15	16	7
16	17	7
17	18	7
18	19	7
19	20	7
20	21	7
21	22	7
22	23	7
23	24	7
24	25	7
25	26	7
26	27	7

27	28	7
28	29	7
29	30	7
30	31	7
31	32	7
32	33	7
33	34	7
34	35	7
35	36	7
36	37	7
37	38	7
38	39	7
39	40	7
40	41	7
41	42	7
42	43	7
43	44	7
44	45	7
45	46	7
46	47	7
47	48	7
48	49	7
49	50	7
50	51	7
51	52	7
52	53	7
53	54	7
54	55	7
55	56	7
56	57	7
57	58	7
58	59	6

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

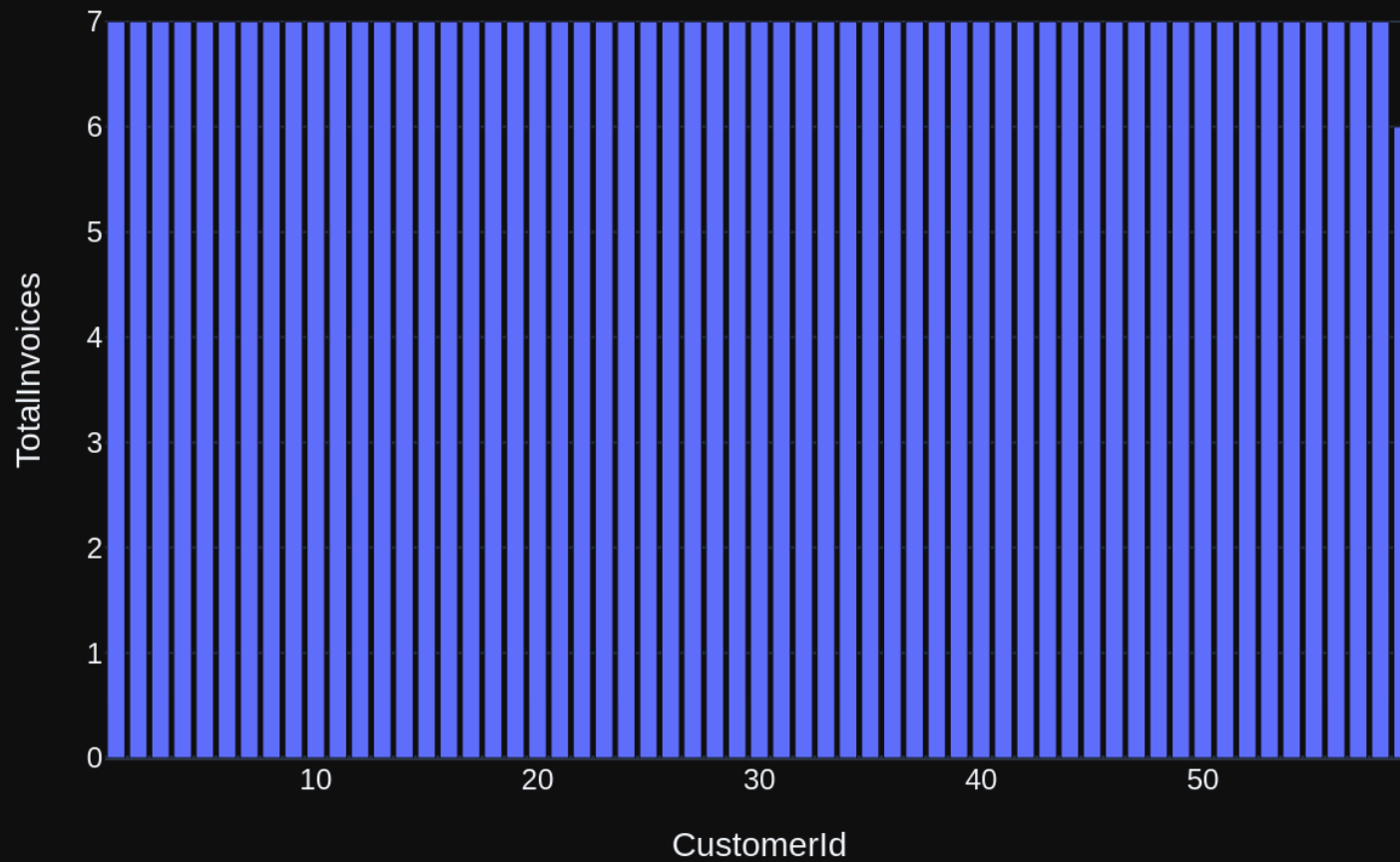
```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n    Get the total number of invoices for each customer\n\n\nThe DataFrame was produced using this query: SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\nCustomerId
```

```
int64\nTotalInvoices    int64\nndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:34:01.323337806Z', 'message': {'role': 'assistant', 'content': "\n\npython\nimport plotly.express as px\n\nfig = px.bar(df, x='CustomerId', y='TotalInvoices', title='Total Invoices per Customer')\nfig.show()\n\n"}, 'done_reason': 'stop', 'done': True, 'total_duration': 19143396917, 'load_duration': 708696, 'prompt_eval_count': 188, 'prompt_eval_duration': 8046387000, 'eval_count': 43, 'eval_duration': 11006526000}
```

Total Invoices per Customer




```
Out[23]: ('SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId',
```

	CustomerId	TotalInvoices
0	1	7
1	2	7
2	3	7
3	4	7
4	5	7
5	6	7
6	7	7
7	8	7
8	9	7
9	10	7
10	11	7
11	12	7
12	13	7
13	14	7
14	15	7
15	16	7
16	17	7
17	18	7
18	19	7
19	20	7
20	21	7
21	22	7
22	23	7
23	24	7
24	25	7
25	26	7
26	27	7
27	28	7
28	29	7
29	30	7
30	31	7
31	32	7
32	33	7
33	34	7
34	35	7
35	36	7
36	37	7
37	38	7
38	39	7

39	40	7
40	41	7
41	42	7
42	43	7
43	44	7
44	45	7
45	46	7
46	47	7
47	48	7
48	49	7
49	50	7
50	51	7
51	52	7
52	53	7
53	54	7
54	55	7
55	56	7
56	57	7
57	58	7
58	59	6,

```
Figure({
  'data': [{
    'alignmentgroup': 'True',
    'hovertemplate': 'CustomerId=%{x}<br>TotalInvoices=%{y}<extra></extra>',
    'legendgroup': '',
    'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
    'name': '',
    'offsetgroup': '',
    'orientation': 'v',
    'showlegend': False,
    'textposition': 'auto',
    'type': 'bar',
    'x': array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
                19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36,
                37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54,
                55, 56, 57, 58, 59]),
    'xaxis': 'x',
    'y': array([7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
                7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
                7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 6]),
    'yaxis': 'y'}],
  'layout': {'barmode': 'relative',
    'legend': {'tracegroupgap': 0},
```

```
        'template': '...',  
        'title': {'text': 'Total Invoices per Customer'},  
        'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'CustomerId'}},  
        'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'TotalInvoices'}}}  
    )))
```

```
In [24]: question = """  
        Find the total number of invoices per country:  
        """  
  
        vn.ask(question=question)
```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

s given before. \n'}, {'role': 'user', 'content': ' \n Find the total number of invoices per countr

```
y:\n'}, {'role': 'assistant', 'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "invoic
s" i\nGROUP BY i.BillingCountry'}, {'role': 'user', 'content': ' \n    Get the total number of invoices fo
r each customer\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvo
ices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'rol
e': 'user', 'content': ' \n    Find all invoices since 2010 and the total amount invoiced:\n'}, {'role':
'assistant', 'content': 'SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\nFROM "invoices" i\nWHERE i.Invo
iceDate >= \'2010-01-01\'\nGROUP BY i.InvoiceDate'}, {'role': 'user', 'content': ' \n    Find the custome
r with the most invoices\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS
TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId
\nORDER BY TotalInvoices DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n    Hint: album quantity is foun
d in invoice_items, \n    \n    Find the top 5 customers who bought the most albums in total quantity (acro
ss all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS To
talAlbums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId
\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n    List all invoices with a total
exceeding $10:\n'}, {'role': 'assistant', 'content': 'SELECT * \nFROM "invoices" \nWHERE Total > 10.00'},
{'role': 'user', 'content': ' \n    Hint: album quantity is found in invoice_items, \n    \n    Find the t
op 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistan
t', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices"
i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.Custome
rId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n    Find the top 5 customers w
ho spent the most money overall, \n    \n    Hint: order total can be found on invoices table, calculatio
n using invoice_items detail table is unnecessary\n'}, {'role': 'assistant', 'content': 'SELECT c.Customer
Id, SUM(i.Total) AS TotalSpent\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP
BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n    Find the custom
er who bought the most albums in total quantity (across all invoices): \n'}, {'role': 'assistant', 'conten
t': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.Cust
omerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER
BY TotalAlbums DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n    Get the average invoice total for each
customer:\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\n
FROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'us
er', 'content': ' \n    Find the total number of invoices per country:\n'}]
```

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the
question. Your response should ONLY be based on the given context and follow the response guidelines and fo
rmat instructions. \n===Tables\nCREATE TABLE \"invoices\"\n(\n    InvoiceId INTEGER PRIMARY KEY AUTOIN
CREMENT NOT NULL,\n    CustomerId INTEGER NOT NULL,\n    InvoiceDate DATETIME NOT NULL,\n    Billin
gAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCou
ntry NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2) NOT NULL,\n    FORE
```

```

IGN KEY (CustomerId) REFERENCES \"customers\" (CustomerId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION
\r\n)\n\nCREATE TABLE \"invoice_items\" \r\n(\r\n    InvoiceLineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
\r\n    InvoiceId INTEGER NOT NULL, \r\n    TrackId INTEGER NOT NULL, \r\n    UnitPrice NUMERIC(10,2) NOT
NULL, \r\n    Quantity INTEGER NOT NULL, \r\n    FOREIGN KEY (InvoiceId) REFERENCES \"invoices\" (InvoiceI
d) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION, \r\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (Tra
ckId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON \"invoi
ces\" (CustomerId)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON \"invoice_items\" (InvoiceId)\n\nCREATE INDE
X IFK_InvoiceLineTrackId ON \"invoice_items\" (TrackId)\n\nCREATE TABLE \"employees\" \r\n(\r\n    EmployeeI
d INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, \r\n    LastName NVARCHAR(20) NOT NULL, \r\n    FirstName NVAR
CHAR(20) NOT NULL, \r\n    Title NVARCHAR(30), \r\n    ReportsTo INTEGER, \r\n    BirthDate DATETIME, \r\n
HireDate DATETIME, \r\n    Address NVARCHAR(70), \r\n    City NVARCHAR(40), \r\n    State NVARCHAR(40), \r\n
Country NVARCHAR(40), \r\n    PostalCode NVARCHAR(10), \r\n    Phone NVARCHAR(24), \r\n    Fax NVARCHAR(24), \r
\n    Email NVARCHAR(60), \r\n    FOREIGN KEY (ReportsTo) REFERENCES \"employees\" (EmployeeId) \r\n\t\tON D
ELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"customers\" \r\n(\r\n    CustomerId INTEGER PRIMA
RY KEY AUTOINCREMENT NOT NULL, \r\n    FirstName NVARCHAR(40) NOT NULL, \r\n    LastName NVARCHAR(20) NOT N
ULL, \r\n    Company NVARCHAR(80), \r\n    Address NVARCHAR(70), \r\n    City NVARCHAR(40), \r\n    State NVARC
HAR(40), \r\n    Country NVARCHAR(40), \r\n    PostalCode NVARCHAR(10), \r\n    Phone NVARCHAR(24), \r\n    Fax
NVARCHAR(24), \r\n    Email NVARCHAR(60) NOT NULL, \r\n    SupportRepId INTEGER, \r\n    FOREIGN KEY (Support
RepId) REFERENCES \"employees\" (EmployeeId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE
TABLE \"albums\" \r\n(\r\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, \r\n    Title NVARCHAR(160)
NOT NULL, \r\n    ArtistId INTEGER NOT NULL, \r\n    FOREIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistI
d) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"tracks\" \r\n(\r\n    TrackId INTE
GER PRIMARY KEY AUTOINCREMENT NOT NULL, \r\n    Name NVARCHAR(200) NOT NULL, \r\n    AlbumId INTEGER, \r\n
MediaTypeId INTEGER NOT NULL, \r\n    GenreId INTEGER, \r\n    Composer NVARCHAR(220), \r\n    Milliseconds I
NTEGER NOT NULL, \r\n    Bytes INTEGER, \r\n    UnitPrice NUMERIC(10,2) NOT NULL, \r\n    FOREIGN KEY (Album
Id) REFERENCES \"albums\" (AlbumId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION, \r\n    FOREIGN KEY (Ge
nreId) REFERENCES \"genres\" (GenreId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION, \r\n    FOREIGN KEY
(MediaTypeId) REFERENCES \"media_types\" (MediaTypeId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)
\n\nCREATE INDEX IFK_EmployeeReportsTo ON \"employees\" (ReportsTo)\n\n\n===Additional Context\n\nIn the S
QLite database invoice means order\n\n===Response Guidelines\n1. If the provided context is sufficient, pl
ease generate a valid SQL query without any explanations for the question.\n2. If the provided context is
almost sufficient but requires knowledge of a specific string in a particular column, please generate an in
termediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying i
ntermediate_sql\n3. If the provided context is insufficient, please explain why it can't be generated.\n
4. Please use the most relevant table(s).\n5. If the question has been asked and answered before, please r
epeat the answer exactly as it was given before.\n\"}, {\"role\": \"user\", \"content\": \" \n    Find the total
number of invoices per country:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT i.BillingCountry, COUNT(*) AS
TotalInvoices\nFROM \"invoices\" i\nGROUP BY i.BillingCountry\"}, {\"role\": \"user\", \"content\": \" \n    Get t
he total number of invoices for each customer\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, C
OUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerI
d\nGROUP BY c.CustomerId\"}, {\"role\": \"user\", \"content\": \" \n    Find all invoices since 2010 and the total
amount invoiced:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\n

```

```

FROM "invoices" i\nWHERE i.InvoiceDate >= '2010-01-01'\nGROUP BY i.InvoiceDate"}, {"role": "user", "content": " \n      Find the customer with the most invoices \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1"}, {"role": "user", "content": " \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": " \n      List all invoices with a total exceeding $10:\n"}, {"role": "assistant", "content": "SELECT * \nFROM \"invoices\" \nWHERE Total > 10.00"}, {"role": "user", "content": " \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": " \n      Find the top 5 customers who spent the most money overall, \n      \n      Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessary \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5"}, {"role": "user", "content": " \n      Find the customer who bought the most albums in total quantity (across all invoices): \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1"}, {"role": "user", "content": " \n      Get the average invoice total for each customer:\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}, {"role": "user", "content": " \n      Find the total number of invoices per country:\n"}]

```

Add of existing embedding ID: dd282d7c-a4ef-5e3a-87e0-cb45fac50808-sql

Insert of existing embedding ID: dd282d7c-a4ef-5e3a-87e0-cb45fac50808-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:35:31.280288135Z', 'message': {'role': 'assistant', 'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "invoices" i\nGROUP BY i.BillingCountry'}, 'done_reason': 'stop', 'done': True, 'total_duration': 89864674283, 'load_duration': 750602, 'prompt_eval_count': 1926, 'prompt_eval_duration': 82750414000, 'eval_count': 25, 'eval_duration': 6478972000}
```

```
SELECT i.BillingCountry, COUNT(*) AS TotalInvoices
FROM "invoices" i
GROUP BY i.BillingCountry
SELECT i.BillingCountry, COUNT(*) AS TotalInvoices
FROM "invoices" i
GROUP BY i.BillingCountry
```

	BillingCountry	TotalInvoices
0	Argentina	7
1	Australia	7
2	Austria	7
3	Belgium	7
4	Brazil	35
5	Canada	56
6	Chile	7
7	Czech Republic	14
8	Denmark	7
9	Finland	7
10	France	35
11	Germany	28
12	Hungary	7
13	India	13
14	Ireland	7
15	Italy	7
16	Netherlands	7
17	Norway	7
18	Poland	7
19	Portugal	14
20	Spain	7
21	Sweden	7
22	USA	91
23	United Kingdom	21

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

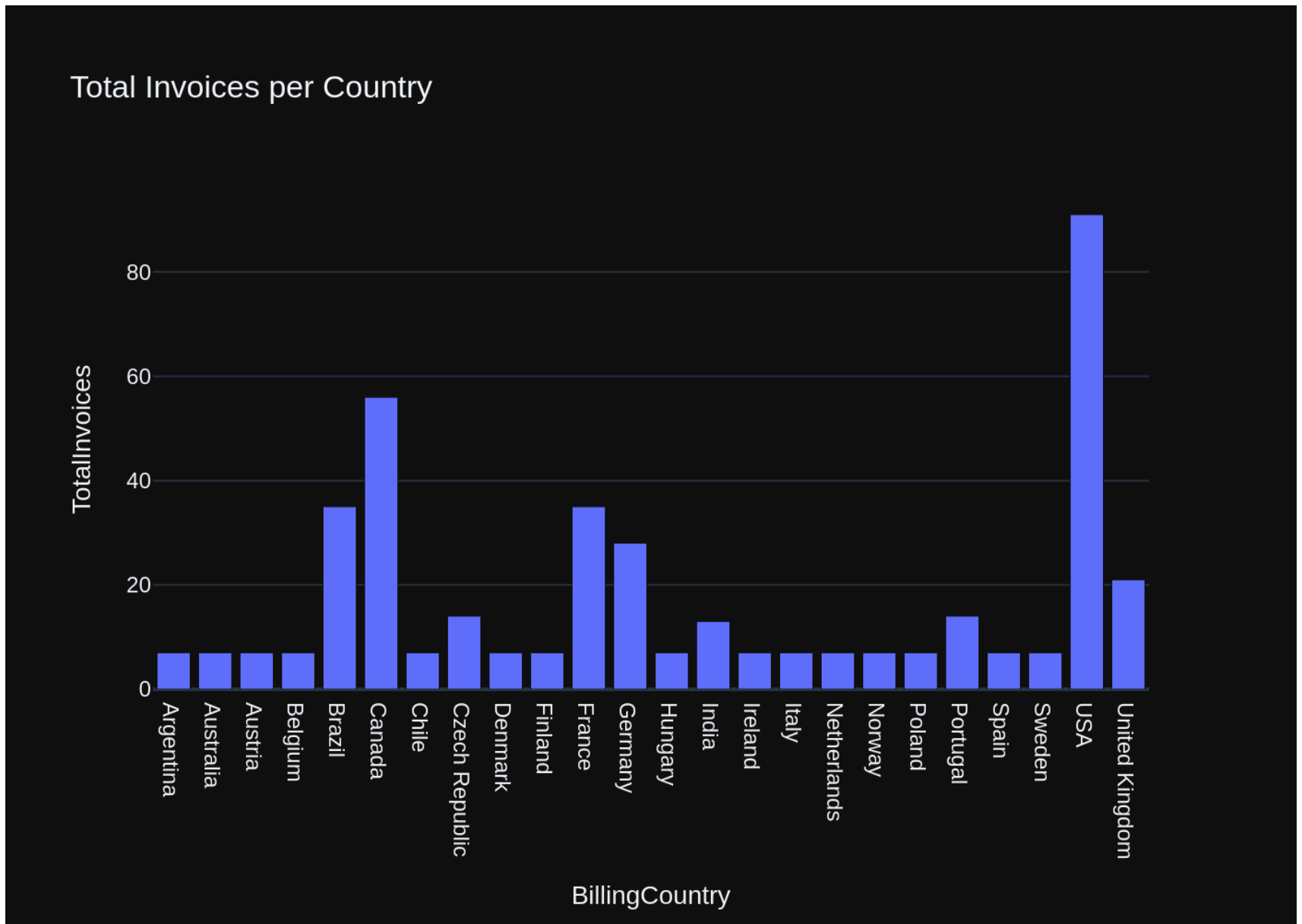
Prompt Content:

```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query
```


that answers the question the user asked: ' \n Find the total number of invoices per country:\n'\n\nThe DataFrame was produced using this query: SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM \"invoices\" i\nGROUP BY i.BillingCountry\n\nThe following is information about the resulting pandas DataFrame 'df':\nRunning df.dtypes gives:\n BillingCountry object\nTotalInvoices int64\ndtype: object\"}, {\"role\": \"user\", \"content\": \"Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code.\"}]

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:35:50.252772492Z', 'message': {'role': 'assistant', 'content': "\n\npython\nimport plotly.express as px\n\nfig = px.bar(df, x='BillingCountry', y='TotalInvoices', title='Total Invoices per Country')\n\nfig.show()\n\n\"}, 'done_reason': 'stop', 'done': True, 'total_duration': 18945260434, 'load_duration': 641791, 'prompt_eval_count': 169, 'prompt_eval_duration': 7327858000, 'eval_count': 44, 'eval_duration': 11529350000}
```



```
Out[24]: ('SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "invoices" i\nGROUP BY i.BillingCountry',
```

	BillingCountry	TotalInvoices
0	Argentina	7
1	Australia	7
2	Austria	7
3	Belgium	7
4	Brazil	35
5	Canada	56
6	Chile	7
7	Czech Republic	14
8	Denmark	7
9	Finland	7
10	France	35
11	Germany	28
12	Hungary	7
13	India	13
14	Ireland	7
15	Italy	7
16	Netherlands	7
17	Norway	7
18	Poland	7
19	Portugal	14
20	Spain	7
21	Sweden	7
22	USA	91
23	United Kingdom	21,

```
Figure({
  'data': [{ 'alignmentgroup': 'True',
    'hovertemplate': 'BillingCountry=%{x}<br>TotalInvoices=%{y}<extra></extra>',
    'legendgroup': '',
    'marker': { 'color': '#636efa', 'pattern': { 'shape': '' } },
    'name': '',
    'offsetgroup': '',
    'orientation': 'v',
    'showlegend': False,
    'textposition': 'auto',
    'type': 'bar',
    'x': array(['Argentina', 'Australia', 'Austria', 'Belgium', 'Brazil', 'Canada',
      'Chile', 'Czech Republic', 'Denmark', 'Finland', 'France', 'Germany',
      'Hungary', 'India', 'Ireland', 'Italy', 'Netherlands', 'Norway',
      'Poland', 'Portugal', 'Spain', 'Sweden', 'USA', 'United Kingdom'],
      dtype=object),
```

```

        'xaxis': 'x',
        'y': array([ 7,  7,  7,  7, 35, 56,  7, 14,  7,  7, 35, 28,  7, 13,  7,  7,  7,  7,
                    7, 14,  7,  7, 91, 21]),
        'yaxis': 'y']},
    'layout': {'barmode': 'relative',
               'legend': {'tracegroupgap': 0},
               'template': '...',
               'title': {'text': 'Total Invoices per Country'},
               'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'BillingCountry'}}},
               'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'TotalInvoices'}}}
    )))

```

```

In [25]: question = """
        List all invoices with a total exceeding $10:
        """

        vn.ask(question=question)

```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

```
[{'role': 'system', 'content': 'You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables\nCREATE TABLE "invoice_items"\n(\n    InvoiceLineId INTEGER PRIMARY KEY\n    AUTOINCREMENT NOT NULL,\n    InvoiceId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    Quantity INTEGER NOT NULL,\n    FOREIGN KEY (InvoiceId) REFERENCES\n    "invoices" (InvoiceId)\n)\nCREATE INDEX IFK_InvoiceLineInvoiceId ON "invoice_items" (InvoiceId)\nCREATE TABLE "invoices"\n(\n    InvoiceId INTEGER PRIMARY\n    KEY AUTOINCREMENT NOT NULL,\n    CustomerId INTEGER NOT NULL,\n    InvoiceDate DATETIME NOT NULL,\n    BillingAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (CustomerId) REFERENCES "customers" (CustomerId)\n)\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\nCREATE INDEX IFK_InvoiceCustomerId ON "invoices" (CustomerId)\nCREATE TABLE "tracks"\n(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT\n    NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT\n    NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES "albums"\n    (AlbumId)\n)\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\nCREATE INDEX IFK_TrackGenreId ON "tracks" (GenreId)\nCREATE INDEX IFK_TrackMediaTypeId ON "tracks" (MediaTypeId)\nCREATE TABLE "employees"\n(\n    EmployeeId INTEGER PRIMARY KEY\n    AUTOINCREMENT NOT NULL,\n    LastName NVARCHAR(20) NOT NULL,\n    FirstName NVARCHAR(20) NOT NULL,\n    Title NVARCHAR(30),\n    ReportsTo INTEGER,\n    BirthDate DATETIME,\n    HireDate DATETIME,\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60) NOT NULL,\n    SupportRepId INTEGER,\n    FOREIGN KEY (SupportRepId) REFERENCES\n    "employees" (EmployeeId)\n)\nCREATE INDEX IFK_EmployeeReportsTo ON "employees" (ReportsTo)\nCREATE TABLE "customers"\n(\n    CustomerId INTEGER PRIMARY KEY\n    AUTOINCREMENT NOT NULL,\n    Company NVARCHAR(80),\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60) NOT NULL,\n    SupportRepId INTEGER,\n    FOREIGN KEY (SupportRepId) REFERENCES\n    "employees" (EmployeeId)\n)\nCREATE INDEX IFK_CustomerSupportRepId ON\n    "customers" (SupportRepId)\n===Additional Context\nIn the SQLite database invoice means order\n===Response Guidelines\n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question.\n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql\n3. If the provided context is insufficient, please explain why it can't be generated.\n4. Please use the most relevant table(s).\n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before.\n'}], {'role': 'user', 'content': '\n    List all invoices with a total exceeding $10:\n'}, {'role': 'assistant', 'content': 'SELECT *\nFROM "invoices"\nWHERE Total > 10.00'}, {'role': 'user', 'content': '\n    Find all invoices since 2010 and the total amount invoiced:\n'}, {'role': 'assistant', 'content': 'SELECT *\nFROM "invoices"\nWHERE InvoiceDate >= 2010-01-01\nORDER BY Total DESC\nLIMIT 10'}
```

```
t': 'SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\nFROM "invoices" i\nWHERE i.InvoiceDate >= \'2010-01-01\'\nGROUP BY i.InvoiceDate'}, {'role': 'user', 'content': ' \n      Find the top 5 customers who spent the most money overall, \n      \n      Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessary \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Find the customer who bought the most albums in total quantity (across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n      Get the total number of invoices for each customer\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': ' \n      Find the customer with the most invoices \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n      Find the total number of invoices per country:\n'}, {'role': 'assistant', 'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "invoices" i\nGROUP BY i.BillingCountry'}, {'role': 'user', 'content': ' \n      Get the average invoice total for each customer:\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': ' \n      List all invoices with a total exceeding $10:\n'}]
```

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables\nCREATE TABLE \"invoice_items\"(\n    InvoiceLineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    InvoiceId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    Quantity INTEGER NOT NULL,\n    FOREIGN KEY (InvoiceId) REFERENCES \"invoices\" (InvoiceId)\n    ON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId)\n    ON DELETE NO ACTION ON UPDATE NO ACTION\n)\nCREATE INDEX IFK_InvoiceLineInvoiceId ON \"invoice_items\" (InvoiceId)\nCREATE TABLE \"invoices\"(\n    InvoiceId INTEGE
```

```

R PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    CustomerId INTEGER NOT NULL,\r\n    InvoiceDate DATETIME NOT
NULL,\r\n    BillingAddress NVARCHAR(70),\r\n    BillingCity NVARCHAR(40),\r\n    BillingState NVARCHAR(4
0),\r\n    BillingCountry NVARCHAR(40),\r\n    BillingPostalCode NVARCHAR(10),\r\n    Total NUMERIC(10,2)
NOT NULL,\r\n    FOREIGN KEY (CustomerId) REFERENCES \"customers\" (CustomerId) \r\n\r\n\t\tON DELETE NO ACTION
ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON \"invoice_items\" (TrackId)\n\nCREATE IN
DEX IFK_InvoiceCustomerId ON \"invoices\" (CustomerId)\n\nCREATE TABLE \"tracks\" \r\n(\r\n    TrackId INTEG
ER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Name NVARCHAR(200) NOT NULL,\r\n    AlbumId INTEGER,\r\n
MediaTypeId INTEGER NOT NULL,\r\n    GenreId INTEGER,\r\n    Composer NVARCHAR(220),\r\n    Milliseconds I
NTEGER NOT NULL,\r\n    Bytes INTEGER,\r\n    UnitPrice NUMERIC(10,2) NOT NULL,\r\n    FOREIGN KEY (Album
Id) REFERENCES \"albums\" (AlbumId) \r\n\r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n    FOREIGN KEY (Ge
nreId) REFERENCES \"genres\" (GenreId) \r\n\r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n    FOREIGN KEY
(MediaTypeId) REFERENCES \"media_types\" (MediaTypeId) \r\n\r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)
\n\nCREATE INDEX IFK_EmployeeReportsTo ON \"employees\" (ReportsTo)\n\nCREATE TABLE \"customers\" \r\n(\r\n
CustomerId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    FirstName NVARCHAR(40) NOT NULL,\r\n    Last
Name NVARCHAR(20) NOT NULL,\r\n    Company NVARCHAR(80),\r\n    Address NVARCHAR(70),\r\n    City NVARCHAR
(40),\r\n    State NVARCHAR(40),\r\n    Country NVARCHAR(40),\r\n    PostalCode NVARCHAR(10),\r\n    Phone
NVARCHAR(24),\r\n    Fax NVARCHAR(24),\r\n    Email NVARCHAR(60) NOT NULL,\r\n    SupportRepId INTEGER,\r
\n    FOREIGN KEY (SupportRepId) REFERENCES \"employees\" (EmployeeId) \r\n\r\n\t\tON DELETE NO ACTION ON UPDAT
E NO ACTION\r\n)\n\nCREATE TABLE \"employees\" \r\n(\r\n    EmployeeId INTEGER PRIMARY KEY AUTOINCREMENT NOT
NULL,\r\n    LastName NVARCHAR(20) NOT NULL,\r\n    FirstName NVARCHAR(20) NOT NULL,\r\n    Title NVARCHA
R(30),\r\n    ReportsTo INTEGER,\r\n    BirthDate DATETIME,\r\n    HireDate DATETIME,\r\n    Address NVARCH
AR(70),\r\n    City NVARCHAR(40),\r\n    State NVARCHAR(40),\r\n    Country NVARCHAR(40),\r\n    PostalCode
NVARCHAR(10),\r\n    Phone NVARCHAR(24),\r\n    Fax NVARCHAR(24),\r\n    Email NVARCHAR(60),\r\n    FOREIGN
KEY (ReportsTo) REFERENCES \"employees\" (EmployeeId) \r\n\r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)
\n\nCREATE INDEX IFK_CustomerSupportRepId ON \"customers\" (SupportRepId)\n\n\n===Additional Context\n\nIn
the SQLite database invoice means order\n\n===Response Guidelines\n1. If the provided context is sufficien
t, please generate a valid SQL query without any explanations for the question.\n2. If the provided contex
t is almost sufficient but requires knowledge of a specific string in a particular column, please generate
an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment say
ing intermediate_sql\n3. If the provided context is insufficient, please explain why it can't be generate
d.\n4. Please use the most relevant table(s).\n5. If the question has been asked and answered before, ple
ase repeat the answer exactly as it was given before.\n\"}, {\"role\": \"user\", \"content\": \" \n    List all i
nvoices with a total exceeding $10:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT * \nFROM \"invoices\" \nW
HERE Total > 10.00\"}, {\"role\": \"user\", \"content\": \" \n    Find all invoices since 2010 and the total amoun
t invoiced:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\nFROM
\"invoices\" i\nWHERE i.InvoiceDate >= '2010-01-01'\nGROUP BY i.InvoiceDate\"}, {\"role\": \"user\", \"content\":
\" \n    Find the top 5 customers who spent the most money overall, \n    \n    Hint: order total can be
found on invoices table, calculation using invoice_items detail table is unnecessary\n\"}, {\"role\": \"assist
ant\", \"content\": \"SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\"
i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5\"}, {\"role\": \"use
r\", \"content\": \" \n    Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 customer
s who bought the most albums in total quantity (across all invoices):\n\"}, {\"role\": \"assistant\", \"content\":

```

```
"SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" i
i ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5\"}, {\"role\": \"use
r\", \"content\": \" \n    Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 customer
s who bought the most albums in total quantity (across all invoices):\n\"}, {\"role\": \"assistant\", \"content\":
\"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.Cust
omerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDE
R BY TotalAlbums DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n    Find the customer who bought the mo
st albums in total quantity (across all invoices): \n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.Custome
rId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.Custome
rId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums
DESC\nLIMIT 1\"}, {\"role\": \"user\", \"content\": \" \n    Get the total number of invoices for each customer
\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"cu
stomers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\"}, {\"role\": \"user\",
\"content\": \" \n    Find the customer with the most invoices \n\"}, {\"role\": \"assistant\", \"content\": \"SELEC
T c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.Custome
rId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1\"}, {\"role\": \"user\", \"conten
t\": \" \n    Find the total number of invoices per country:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT
i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM \"invoices\" i\nGROUP BY i.BillingCountry\"}, {\"role\": \"us
er\", \"content\": \" \n    Get the average invoice total for each customer:\n\"}, {\"role\": \"assistant\", \"conte
nt\": \"SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM \"customers\" c\nJOIN \"invoices\" i O
N c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\"}, {\"role\": \"user\", \"content\": \" \n    List all invoic
es with a total exceeding $10:\n\"}]
```

Add of existing embedding ID: aea89953-21b2-55d1-9dda-431ee6033c3d-sql

Insert of existing embedding ID: aea89953-21b2-55d1-9dda-431ee6033c3d-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:37:17.94656018Z', 'message': {'role': 'assistant', 'content': 'SELECT * \nFROM "invoices" \nWHERE Total > 10.00'}, 'done_reason': 'stop', 'done': True, 'total_duration': 87588612562, 'load_duration': 1062109, 'prompt_eval_count': 1874, 'prompt_eval_duration': 81870508000, 'eval_count': 20, 'eval_duration': 5085360000}
```

```
SELECT *
FROM "invoices"
WHERE Total > 10.00
SELECT *
FROM "invoices"
WHERE Total > 10.00
```

	InvoiceId	CustomerId	InvoiceDate	BillingAddress \
0	5	23	2009-01-11 00:00:00	69 Salem Street
1	12	2	2009-02-11 00:00:00	Theodor-Heuss-Straße 34
2	19	40	2009-03-14 00:00:00	8, Rue Hanovre
3	26	19	2009-04-14 00:00:00	1 Infinite Loop
4	33	57	2009-05-15 00:00:00	Calle Lira, 198
..
59	383	10	2013-08-12 00:00:00	Rua Dr. Falcão Filho, 155
60	390	48	2013-09-12 00:00:00	Lijnbaansgracht 120bg
61	397	27	2013-10-13 00:00:00	1033 N Park Ave
62	404	6	2013-11-13 00:00:00	Rilská 3174/6
63	411	44	2013-12-14 00:00:00	Porthaninkatu 9

	BillingCity	BillingState	BillingCountry	BillingPostalCode	Total
0	Boston	MA	USA	2113	13.86
1	Stuttgart	None	Germany	70174	13.86
2	Paris	None	France	75002	13.86
3	Cupertino	CA	USA	95014	13.86
4	Santiago	None	Chile	None	13.86
..
59	São Paulo	SP	Brazil	01007-010	13.86
60	Amsterdam	VV	Netherlands	1016	13.86
61	Tucson	AZ	USA	85719	13.86
62	Prague	None	Czech Republic	14300	25.86
63	Helsinki	None	Finland	00530	13.86

[64 rows x 9 columns]

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n List all invoices with a total exceeding $10:\n'\n\nThe DataFrame was produced using this query: SELECT * \nFROM \"invoices\" \nWHERE Total > 10.00\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n InvoiceId\nint64\nCustomerId\nint64\nInvoiceDate\nobject\nBillingAddress\nobject\nBillingCity\nobject\nBillingState\nobject\nBillingCountry\nobject\nBillingPostalCode\nobject\nTotal\nfloat64\nndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:37:44.994516855Z', 'message': {'role': 'assistant', 'content': "\n\npython\nimport plotly.express as px\n\nfig = px.bar(df, x='InvoiceId', y='Total', title='Invoices with Total Exceeding $10')\n\nfig.update_layout(\n    xaxis_title='Invoice ID',\n    yaxis_title='Total',\n)\n\nfig.show()\n\n"}, 'done_reason': 'stop', 'done': True, 'total_duration': 27021081555, 'load_duration': 721488, 'prompt_eval_count': 203, 'prompt_eval_duration': 8514490000, 'eval_count': 72, 'eval_duration': 18416461000}
```

Invoices with Total Exceeding \$10



Out[25]: ('SELECT * \nFROM "invoices" \nWHERE Total > 10.00',

	InvoiceId	CustomerId	InvoiceDate	BillingAddress \
0	5	23	2009-01-11 00:00:00	69 Salem Street
1	12	2	2009-02-11 00:00:00	Theodor-Heuss-Straße 34
2	19	40	2009-03-14 00:00:00	8, Rue Hanovre
3	26	19	2009-04-14 00:00:00	1 Infinite Loop
4	33	57	2009-05-15 00:00:00	Calle Lira, 198
..
59	383	10	2013-08-12 00:00:00	Rua Dr. Falcão Filho, 155
60	390	48	2013-09-12 00:00:00	Lijnbaansgracht 120bg
61	397	27	2013-10-13 00:00:00	1033 N Park Ave
62	404	6	2013-11-13 00:00:00	Rilská 3174/6
63	411	44	2013-12-14 00:00:00	Porthaninkatu 9

	BillingCity	BillingState	BillingCountry	BillingPostalCode	Total
0	Boston	MA	USA	2113	13.86
1	Stuttgart	None	Germany	70174	13.86
2	Paris	None	France	75002	13.86
3	Cupertino	CA	USA	95014	13.86
4	Santiago	None	Chile	None	13.86
..
59	São Paulo	SP	Brazil	01007-010	13.86
60	Amsterdam	VV	Netherlands	1016	13.86
61	Tucson	AZ	USA	85719	13.86
62	Prague	None	Czech Republic	14300	25.86
63	Helsinki	None	Finland	00530	13.86

[64 rows x 9 columns],

Figure({

```

    'data': [{'alignmentgroup': 'True',
               'hovertemplate': 'InvoiceId=%{x}<br>Total=%{y}<extra></extra>',
               'legendgroup': '',
               'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
               'name': '',
               'offsetgroup': '',
               'orientation': 'v',
               'showlegend': False,
               'textposition': 'auto',
               'type': 'bar',
               'x': array([ 5, 12, 19, 26, 33, 40, 47, 54, 61, 68, 75, 82, 88, 89,
                           96, 103, 110, 117, 124, 131, 138, 145, 152, 159, 166, 173, 180, 187,
                           193, 194, 201, 208, 215, 222, 229, 236, 243, 250, 257, 264, 271, 278,
```

```

                285, 292, 298, 299, 306, 311, 312, 313, 320, 327, 334, 341, 348, 355,
                362, 369, 376, 383, 390, 397, 404, 411]),
        'xaxis': 'x',
        'y': array([13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86,
                    13.86, 13.86, 17.91, 18.86, 21.86, 15.86, 13.86, 13.86, 13.86, 13.86,
                    13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 14.91, 21.86,
                    18.86, 15.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86,
                    13.86, 13.86, 13.86, 13.86, 10.91, 23.86, 16.86, 11.94, 10.91, 16.86,
                    13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86,
                    13.86, 13.86, 25.86, 13.86]),
        'yaxis': 'y'}],
    'layout': {'barmode': 'relative',
               'legend': {'tracegroupgap': 0},
               'template': '...',
               'title': {'text': 'Invoices with Total Exceeding $10'},
               'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Invoice ID'}},
               'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'Total'}}
    })

```

```

In [26]: question = """
        Find all invoices since 2010 and the total amount invoiced:
        """

        vn.ask(question=question)

```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

86/199

_sql \n3. If the provided context is insufficient, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before. \n'}, {'role': 'user', 'content': ' \n Find all invoices since 2010 and the total amount invoiced:\n'}, {'role': 'assistant', 'content': 'SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\nFROM "invoices" i\nWHERE i.InvoiceDate >= \'2010-01-01\'\nGROUP BY i.InvoiceDate'}, {'role': 'user', 'content': ' \n List all invoices with a total exceeding \$10:\n'}, {'role': 'assistant', 'content': 'SELECT * \nFROM "invoices" \nWHERE Total > 10.00'}, {'role': 'user', 'content': ' \n Find the total number of invoices per country:\n'}, {'role': 'assistant', 'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "invoices" i\nGROUP BY i.BillingCountry'}, {'role': 'user', 'content': ' \n Get the total number of invoices for each customer\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': ' \n Find the top 5 customers who spent the most money overall, \n \n Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessary \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Find the customer who bought the most albums in total quantity (across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n Hint: album quantity is found in invoice_items, \n \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Find the customer with the most invoices \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n Hint: album quantity is found in invoice_items, \n \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Get the average invoice total for each customer:\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': ' \n Find all invoices since 2010 and the total amount invoiced:\n'}]

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables \nCREATE TABLE \"invoices\"(\n    InvoiceId INTEGER PRIMARY KEY AUTOIN
```

tional Context \n\nIn the SQLite database invoice means order\n\n===Response Guidelines \n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question. \n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before. \n"}\n{"role": "user", "content":


```
" \n    Find all invoices since 2010 and the total amount invoiced:\n"}, {"role": "assistant", "content":
"SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\nFROM \"invoices\" i\nWHERE i.InvoiceDate >= '2010-01-0
1'\nGROUP BY i.InvoiceDate"}, {"role": "user", "content": " \n    List all invoices with a total exceeding
$10:\n"}, {"role": "assistant", "content": "SELECT * \nFROM \"invoices\" \nWHERE Total > 10.00"}, {"role":
"user", "content": " \n    Find the total number of invoices per country:\n"}, {"role": "assistant", "cont
ent": "SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM \"invoices\" i\nGROUP BY i.BillingCountr
y"}, {"role": "user", "content": " \n    Get the total number of invoices for each customer\n"}, {"role":
"assistant", "content": "SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJO
IN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}, {"role": "user", "content": "
\n    Find the top 5 customers who spent the most money overall, \n    \n    Hint: order total can be fo
und on invoices table, calculation using invoice_items detail table is unnecessary \n"}, {"role": "assistan
t", "content": "SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\" i
ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5"}, {"role": "use
r", "content": " \n    Find the customer who bought the most albums in total quantity (across all invoice
s): \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"c
ustomers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId
= ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1"}, {"role": "user", "content": "
\n    Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 customers who bought the m
ost albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT i.Custome
rId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId
= ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": "
\n    Find the customer with the most invoices \n"}, {"role": "assistant", "content": "SELECT c.CustomerI
d, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.Custo
merId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1"}, {"role": "user", "content": " \n
Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 customers who bought the most al
bums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, C
OUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\n
JOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\n
LIMIT 5"}, {"role": "user", "content": " \n    Get the average invoice total for each customer:\n"}, {"rol
e": "assistant", "content": "SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM \"customers\" c
\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}, {"role": "user", "content":
" \n    Find all invoices since 2010 and the total amount invoiced:\n"}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:39:19.136611169Z', 'message': {'role': 'assistan
t', 'content': 'SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\nFROM \"invoices\" i\nWHERE i', 'done_reas
on': 'stop', 'done': True, 'total_duration': 94046624476, 'load_duration': 745062, 'prompt_eval_count': 202
0, 'prompt_eval_duration': 86711387000, 'eval_count': 25, 'eval_duration': 6698290000}
```

```
SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount
FROM \"invoices\" i
WHERE i
SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount
FROM \"invoices\" i
```

```
WHERE i
Couldn't run sql: Execution failed on sql 'SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount
FROM "invoices" i
WHERE i': no such column: i
```

```
In [27]: question = """
        List all employees and their reporting manager's name (if any):
        """

        vn.ask(question=question)
```

```
Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1
```

```
[{'role': 'system', 'content': 'You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables \nCREATE INDEX IFK_EmployeeReportsTo ON "employees" (ReportsTo)\n\nCREATE TABLE "employees"\n\n(\n    EmployeeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    LastName NVARCHAR(20) NOT NULL,\n    FirstName NVARCHAR(20) NOT NULL,\n    Title NVARCHAR(30),\n    ReportsTo INTEGER,\n    BirthDate DATETIME,\n    HireDate DATETIME,\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60),\n    FOREIGN KEY (ReportsTo) REFERENCES "employees" (EmployeeId) \n)\n\nON DELETE NO ACTION ON UPDATE NO ACTION\n\nCREATE TABLE "customers"\n\n(\n    CustomerId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    FirstName NVARCHAR(40) NOT NULL,\n    LastName NVARCHAR(20) NOT NULL,\n    Company NVARCHAR(80),\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60) NOT NULL,\n    SupportRepId INTEGER,\n    FOREIGN KEY (SupportRepId) REFERENCES "employees" (EmployeeId) \n)\n\nON DELETE NO ACTION ON UPDATE NO ACTION\n\nCREATE INDEX IFK_CustomerSupportRepId ON "customers" (SupportRepId)\n\nCREATE TABLE "invoices"\n\n(\n    InvoiceId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    CustomerId INTEGER NOT NULL,\n    InvoiceDate DATETIME NOT NULL,\n    BillingAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (CustomerId) REFERENCES "customers" (CustomerId) \n)\n\nON DELETE NO ACTION ON UPDATE NO ACTION\n\nCREATE TABLE "invoice_items"\n\n(\n    InvoiceLineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    InvoiceId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    Quantity INTEGER NOT NULL,\n    FOREIGN KEY (InvoiceId) REFERENCES "invoices" (InvoiceId) \n)\n\nON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \n)\n\nON DELETE NO ACTION ON UPDATE NO ACTION\n\nCREATE TABLE "artists"\n\n(\n    ArtistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\n\nCREATE TABLE "tracks"\n\n(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES "albums" (AlbumId) \n)\n\nON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES "genres" (GenreId) \n)\n\nON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES "media_types" (MediaTypeId) \n)\n\nON DELETE NO ACTION ON UPDATE NO ACTION\n\nCREATE TABLE "albums"\n\n(\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Title NVARCHAR(160) NOT NULL,\n    ArtistId INTEGER NOT NULL,\n    FOREIGN KEY (ArtistId) REFERENCES "artists" (ArtistId) \n)\n\nON DELETE NO ACTION ON UPDATE NO ACTION\n\nCREATE TABLE sqlite_stat1(tbl,idx,stat)\n\n\n===Additional Context \n\nIn the SQLite database invoice means order\n\n===Response Guidelines \n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question. \n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before. \n'}], {'role': 'user', 'content': " \n    List all
```

```

employees and their reporting manager's name (if any):\n"}, {'role': 'assistant', 'content': 'SELECT e.Firs
tName, e.LastName, mt.FirstName AS ManagerFirstName, mt.LastName AS ManagerLastName\nFROM "employees" e\nLE
FT JOIN "employees" mt ON e.ReportsTo = mt.EmployeeId'}, {'role': 'user', 'content': ' \n      Find the top
5 customers who spent the most money overall, \n      \n      Hint: order total can be found on invoices tabl
e, calculation using invoice_items detail table is unnecessary \n'}, {'role': 'assistant', 'content': 'SELE
CT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.Cust
omerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      F
ind the customer with the most invoices \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT
(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP
BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1'}, {'role': 'user', 'content': 'what are the top 5 co
untries that customers come from?'}, {'role': 'assistant', 'content': 'SELECT c.Country, COUNT(*) AS TotalC
ustomers\nFROM "customers" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5'}, {'role': 'user',
'content': ' \n      List all customers from Canada and their email addresses:\n'}, {'role': 'assistant',
'content': 'SELECT c.Email, c.Country\nFROM "customers" c\nWHERE c.Country = \'Canada\''}, {'role': 'user',
'content': ' \n      Get the total number of invoices for each customer\n'}, {'role': 'assistant', 'conten
t': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.C
ustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': ' \n      Find all invoices s
ince 2010 and the total amount invoiced:\n'}, {'role': 'assistant', 'content': 'SELECT i.InvoiceDate, SUM
(i.Total) AS TotalAmount\nFROM "invoices" i\nWHERE i.InvoiceDate >= \'2010-01-01\'\nGROUP BY i.InvoiceDat
e'}, {'role': 'user', 'content': ' \n      Find the customer who bought the most albums in total quantity
(across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS To
talAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON
i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'role': 'user',
'content': ' \n      Get the average invoice total for each customer:\n'}, {'role': 'assistant', 'content':
'SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM "customers" c\nJOIN "invoices" i ON c.Custo
merId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': ' \n      Hint: album quantity is
found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity
(across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS Tot
alAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON
i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user',
'content': ' \n      List all employees and their reporting manager's name (if any):\n"}]

```

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```

[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the
question. Your response should ONLY be based on the given context and follow the response guidelines and fo
rmat instructions. \n===Tables \nCREATE INDEX IFK_EmployeeReportsTo ON \"employees\" (ReportsTo)\n\nCREATE
TABLE \"employees\"(\r\n(\r\n      EmployeeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n      LastName NVAR
CHAR(20) NOT NULL,\r\n      FirstName NVARCHAR(20) NOT NULL,\r\n      Title NVARCHAR(30),\r\n      ReportsTo IN
TEGER,\r\n      BirthDate DATETIME,\r\n      HireDate DATETIME,\r\n      Address NVARCHAR(70),\r\n      City NVARCH

```

```

AR(40),\r\n    State NVARCHAR(40),\r\n    Country NVARCHAR(40),\r\n    PostalCode NVARCHAR(10),\r\n    Phon
e NVARCHAR(24),\r\n    Fax NVARCHAR(24),\r\n    Email NVARCHAR(60),\r\n    FOREIGN KEY (ReportsTo) REFERENC
ES \"employees\" (EmployeeId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"custom
ers\" \r\n(\r\n    CustomerId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    FirstName NVARCHAR(40) NOT
NULL,\r\n    LastName NVARCHAR(20) NOT NULL,\r\n    Company NVARCHAR(80),\r\n    Address NVARCHAR(70),\r\n
City NVARCHAR(40),\r\n    State NVARCHAR(40),\r\n    Country NVARCHAR(40),\r\n    PostalCode NVARCHAR(1
0),\r\n    Phone NVARCHAR(24),\r\n    Fax NVARCHAR(24),\r\n    Email NVARCHAR(60) NOT NULL,\r\n    Support
RepId INTEGER,\r\n    FOREIGN KEY (SupportRepId) REFERENCES \"employees\" (EmployeeId) \r\n\t\t\tON DELETE NO
ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON \"customers\" (SupportRepId)\n
\nCREATE TABLE \"invoices\" \r\n(\r\n    InvoiceId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Custom
erId INTEGER NOT NULL,\r\n    InvoiceDate DATETIME NOT NULL,\r\n    BillingAddress NVARCHAR(70),\r\n    B
illingCity NVARCHAR(40),\r\n    BillingState NVARCHAR(40),\r\n    BillingCountry NVARCHAR(40),\r\n    Billi
ngPostalCode NVARCHAR(10),\r\n    Total NUMERIC(10,2) NOT NULL,\r\n    FOREIGN KEY (CustomerId) REFERENCES
\"customers\" (CustomerId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"invoice_i
tems\" \r\n(\r\n    InvoiceLineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    InvoiceId INTEGER NOT
NULL,\r\n    TrackId INTEGER NOT NULL,\r\n    UnitPrice NUMERIC(10,2) NOT NULL,\r\n    Quantity INTEGER
NOT NULL,\r\n    FOREIGN KEY (InvoiceId) REFERENCES \"invoices\" (InvoiceId) \r\n\t\t\tON DELETE NO ACTION ON
UPDATE NO ACTION,\r\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\t\tON DELETE NO ACTION
ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"artists\" \r\n(\r\n    ArtistId INTEGER PRIMARY KEY AUTOINCREMENT
NOT NULL,\r\n    Name NVARCHAR(120)\r\n)\n\nCREATE TABLE \"tracks\" \r\n(\r\n    TrackId INTEGER PRIMARY KEY
AUTOINCREMENT NOT NULL,\r\n    Name NVARCHAR(200) NOT NULL,\r\n    AlbumId INTEGER,\r\n    MediaTypeId INT
EGER NOT NULL,\r\n    GenreId INTEGER,\r\n    Composer NVARCHAR(220),\r\n    Milliseconds INTEGER NOT NUL
L,\r\n    Bytes INTEGER,\r\n    UnitPrice NUMERIC(10,2) NOT NULL,\r\n    FOREIGN KEY (AlbumId) REFERENCES
\"albums\" (AlbumId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n    FOREIGN KEY (GenreId) REFERENC
ES \"genres\" (GenreId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n    FOREIGN KEY (MediaTypeId) R
EFERENCES \"media_types\" (MediaTypeId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABL
E \"albums\" \r\n(\r\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Title NVARCHAR(160) NO
T NULL,\r\n    ArtistId INTEGER NOT NULL,\r\n    FOREIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistId)
\r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE sqlite_stat1(tbl,idx,stat)\n\n\n===Add
itional Context\n\nIn the SQLite database invoice means order\n\n===Response Guidelines\n1. If the provid
ed context is sufficient, please generate a valid SQL query without any explanations for the question. \n2.
If the provided context is almost sufficient but requires knowledge of a specific string in a particular co
lumn, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the qu
ery with a comment saying intermediate_sql\n3. If the provided context is insufficient, please explain why
it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and
answered before, please repeat the answer exactly as it was given before. \n\"}, {\"role\": \"user\", \"content\":
\" \n    List all employees and their reporting manager's name (if any):\n\"}, {\"role\": \"assistant\", \"conten
t\": \"SELECT e.FirstName, e.LastName, mt.FirstName AS ManagerFirstName, mt.LastName AS ManagerLastName\nFROM
\"employees\" e\nLEFT JOIN \"employees\" mt ON e.ReportsTo = mt.EmployeeId\"}, {\"role\": \"user\", \"content\": \"
\n    Find the top 5 customers who spent the most money overall, \n    \n    Hint: order total can be fo
und on invoices table, calculation using invoice_items detail table is unnecessary\n\"}, {\"role\": \"assistan
t\", \"content\": \"SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\" i

```

```

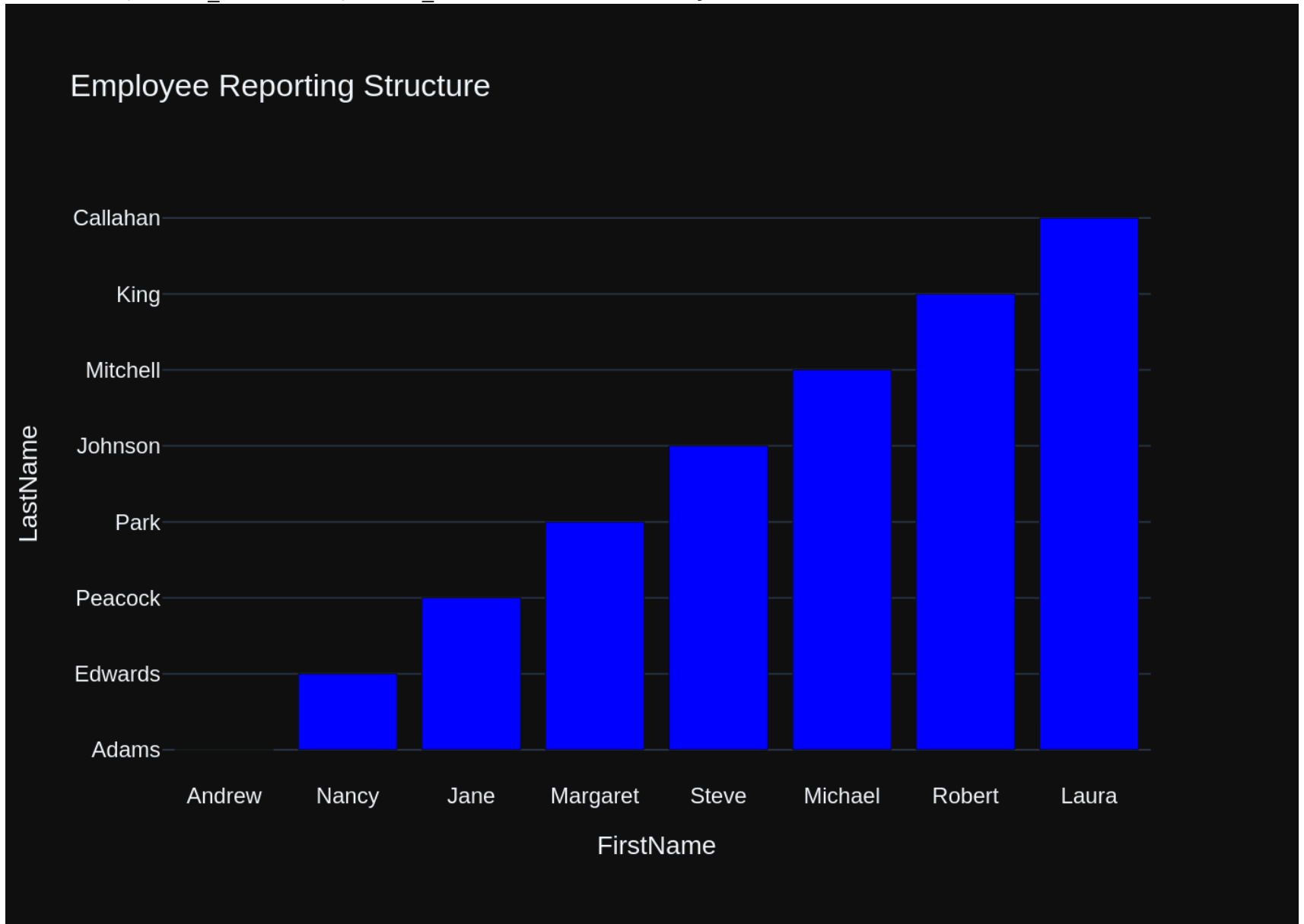
ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5"}], {"role": "user", "content": "\n Find the customer with the most invoices\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1"}], {"role": "user", "content": "what are the top 5 countries that customers come from?"}, {"role": "assistant", "content": "SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM \"customers\" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5"}], {"role": "user", "content": "\n List all customers from Canada and their email addresses\n"}, {"role": "assistant", "content": "SELECT c.Email, c.Country\nFROM \"customers\" c\nWHERE c.Country = 'Canada'"}], {"role": "user", "content": "\n Get the total number of invoices for each customer\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}], {"role": "user", "content": "\n Find all invoices since 2010 and the total amount invoiced:\n"}, {"role": "assistant", "content": "SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\nFROM \"invoices\" i\nWHERE i.InvoiceDate >= '2010-01-01'\nGROUP BY i.InvoiceDate"}], {"role": "user", "content": "\n Find the customer who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1"}], {"role": "user", "content": "\n Get the average invoice total for each customer:\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}], {"role": "user", "content": "\n Hint: album quantity is found in invoice_items, \n \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}], {"role": "user", "content": "\n List all employees and their reporting manager's name (if any):\n"}]]

```

Add of existing embedding ID: fd25ebba-4066-5a0f-8613-7b1c2ace0339-sql

Insert of existing embedding ID: fd25ebba-4066-5a0f-8613-7b1c2ace0339-sql


```
'total_duration': 26690848586, 'load_duration': 659533, 'prompt_eval_count': 173, 'prompt_eval_duration': 7894262000, 'eval_count': 69, 'eval_duration': 18704873000}
```




```
Out[27]: ('SELECT e.FirstName, e.LastName, mt.FirstName AS ManagerFirstName, mt.LastName AS ManagerLastName\nFROM
"employees" e\nLEFT JOIN "employees" mt ON e.ReportsTo = mt.EmployeeId',
```

	FirstName	LastName	ManagerFirstName	ManagerLastName
0	Andrew	Adams	None	None
1	Nancy	Edwards	Andrew	Adams
2	Jane	Peacock	Nancy	Edwards
3	Margaret	Park	Nancy	Edwards
4	Steve	Johnson	Nancy	Edwards
5	Michael	Mitchell	Andrew	Adams
6	Robert	King	Michael	Mitchell
7	Laura	Callahan	Michael	Mitchell,

```
Figure({
  'data': [{'alignmentgroup': 'True',
            'customdata': array([[None],
                                ['Adams'],
                                ['Edwards'],
                                ['Edwards'],
                                ['Edwards'],
                                ['Adams'],
                                ['Mitchell'],
                                ['Mitchell']], dtype=object),
            'hovertemplate': ('<b>{%hovertext}</b><br><br>Fir' ... '{customdata[0]}<extra></extra>'),
            'hovertext': array([None, 'Andrew', 'Nancy', 'Nancy', 'Nancy', 'Andrew', 'Michael',
                                'Michael'], dtype=object),
            'legendgroup': '',
            'marker': {'color': 'blue', 'pattern': {'shape': ''}},
            'name': '',
            'offsetgroup': '',
            'orientation': 'v',
            'showlegend': False,
            'textposition': 'auto',
            'type': 'bar',
            'x': array(['Andrew', 'Nancy', 'Jane', 'Margaret', 'Steve', 'Michael', 'Robert',
                        'Laura'], dtype=object),
            'xaxis': 'x',
            'y': array(['Adams', 'Edwards', 'Peacock', 'Park', 'Johnson', 'Mitchell', 'King',
                        'Callahan'], dtype=object),
            'yaxis': 'y'}],
  'layout': {'barmode': 'relative',
             'legend': {'tracegroupgap': 0},
             'template': '...',
             'title': {'text': 'Employee Reporting Structure'},
```

```
        'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'FirstName'}},  
        'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'LastName'}}  
    )))
```

```
In [28]: question = """  
        Get the average invoice total for each customer:  
        """  
  
        vn.ask(question=question)
```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

99/199

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions."}]
```

```
===Tables\nCREATE TABLE \"invoices\"\n(\n    InvoiceId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    CustomerId INTEGER NOT NULL,\n    InvoiceDate DATETIME NOT NULL,\n    BillingAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (CustomerId) REFERENCES \"customers\" (CustomerId)\nON DELETE NO ACTION ON UPDATE NO ACTION\n)\nCREATE INDEX IFK_InvoiceCustomerId ON \"invoices\" (CustomerId)\nCREATE INDEX IFK_InvoiceLineInvoiceId ON \"invoice_items\" (InvoiceId)\nCREATE TABLE \"invoice_items\"\n(\n    InvoiceLineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    InvoiceId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    Quantity INTEGER NOT NULL,\n    FOREIGN KEY (InvoiceId) REFERENCES \"invoices\" (InvoiceId)\nON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId)\nON DELETE NO ACTION ON UPDATE NO ACTION\n)\nCREATE INDEX IFK_InvoiceLineTrackId ON \"invoice_items\" (TrackId)\nCREATE TABLE sqlite_stat1(tbl,idx,stat)\nCREATE
```

```

INDEX IFK_CustomerSupportRepId ON \"customers\" (SupportRepId)\n\nCREATE TABLE \"customers\"(\n  CustomerId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n  FirstName NVARCHAR(40) NOT NULL,\n  LastName NVARCHAR(20) NOT NULL,\n  Company NVARCHAR(80),\n  Address NVARCHAR(70),\n  City NVARCHAR(40),\n  State NVARCHAR(40),\n  Country NVARCHAR(40),\n  PostalCode NVARCHAR(10),\n  Phone NVARCHAR(24),\n  Fax NVARCHAR(24),\n  Email NVARCHAR(60) NOT NULL,\n  SupportRepId INTEGER,\n  FOREIGN KEY (SupportRepId) REFERENCES \"employees\" (EmployeeId) \nON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_EmployeeReportsTo ON \"employees\" (ReportsTo)\n\nCREATE TABLE \"employees\"(\n  EmployeeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n  LastName NVARCHAR(20) NOT NULL,\n  FirstName NVARCHAR(20) NOT NULL,\n  Title NVARCHAR(30),\n  ReportsTo INTEGER,\n  BirthDate DATETIME,\n  HireDate DATETIME,\n  Address NVARCHAR(70),\n  City NVARCHAR(40),\n  State NVARCHAR(40),\n  Country NVARCHAR(40),\n  PostalCode NVARCHAR(10),\n  Phone NVARCHAR(24),\n  Fax NVARCHAR(24),\n  Email NVARCHAR(60),\n  FOREIGN KEY (ReportsTo) REFERENCES \"employees\" (EmployeeId) \nON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n===Additional Context\n\nIn the SQLite database invoice means order\n\n===Response Guidelines\n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question.\n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql\n3. If the provided context is insufficient, please explain why it can't be generated.\n4. Please use the most relevant table(s).\n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before.\n\"}, {\"role\": \"user\", \"content\": \"\n  Get the average invoice total for each customer:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\"}, {\"role\": \"user\", \"content\": \"\n  Get the total number of invoices for each customer:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\"}, {\"role\": \"user\", \"content\": \"\n  Find the top 5 customers who spent the most money overall,\n  \n  Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessary\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \"\n  Find the customer with the most invoices\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1\"}, {\"role\": \"user\", \"content\": \"\n  Find all invoices since 2010 and the total amount invoiced:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\nFROM \"invoices\" i\nWHERE i.InvoiceDate >= '2010-01-01'\nGROUP BY i.InvoiceDate\"}, {\"role\": \"user\", \"content\": \"\n  Find the total number of invoices per country:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM \"invoices\" i\nGROUP BY i.BillingCountry\"}, {\"role\": \"user\", \"content\": \"\n  Hint: album quantity is found in invoice_items,\n  \n  Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \"\n  Find the customer who bought the most albums in total quantity (across a

```

```
ll invoices): \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.\nInvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1"}, {"role": "user", "co\nntent": " \n    Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 customers who b\nought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELEC\nT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON\ni.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user",\n\"content\": " \n    List all invoices with a total exceeding $10:\n"}, {"role": "assistant", "content": "SE\nLECT * \nFROM \"invoices\" \nWHERE Total > 10.00"}, {"role": "user", "content": " \n    Get the average in\nvoice total for each customer:\n"}]
```

Add of existing embedding ID: 6bed484b-9a80-57f4-ad89-5f775b5df252-sql

Insert of existing embedding ID: 6bed484b-9a80-57f4-ad89-5f775b5df252-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:42:47.621940418Z', 'message': {'role': 'assistant', 'content': 'SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, 'done_reason': 'stop', 'done': True, 'total_duration': 86293220440, 'load_duration': 717957, 'prompt_eval_count': 1704, 'prompt_eval_duration': 74453778000, 'eval_count': 42, 'eval_duration': 11216094000}
```

```
SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal
FROM "customers" c
JOIN "invoices" i ON c.CustomerId = i.CustomerId
GROUP BY c.CustomerId
SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal
FROM "customers" c
JOIN "invoices" i ON c.CustomerId = i.CustomerId
GROUP BY c.CustomerId
```

	CustomerId	AverageInvoiceTotal
0	1	5.660000
1	2	5.374286
2	3	5.660000
3	4	5.660000
4	5	5.802857
5	6	7.088571
6	7	6.088571
7	8	5.374286
8	9	5.374286
9	10	5.374286
10	11	5.374286
11	12	5.374286
12	13	5.374286
13	14	5.374286
14	15	5.517143
15	16	5.374286
16	17	5.660000
17	18	5.374286
18	19	5.517143
19	20	5.660000
20	21	5.374286
21	22	5.660000
22	23	5.374286
23	24	6.231429
24	25	6.088571
25	26	6.802857
26	27	5.374286

27	28	6.231429
28	29	5.374286
29	30	5.374286
30	31	5.374286
31	32	5.374286
32	33	5.374286
33	34	5.660000
34	35	5.374286
35	36	5.374286
36	37	6.231429
37	38	5.374286
38	39	5.517143
39	40	5.517143
40	41	5.374286
41	42	5.660000
42	43	5.802857
43	44	5.945714
44	45	6.517143
45	46	6.517143
46	47	5.374286
47	48	5.802857
48	49	5.374286
49	50	5.374286
50	51	5.517143
51	52	5.374286
52	53	5.374286
53	54	5.374286
54	55	5.374286
55	56	5.374286
56	57	6.660000
57	58	5.517143
58	59	6.106667

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n    Get the average invoice total for each customer:\n\n\nT\nhe DataFrame was produced using this query: SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM\n\n\"customers\" c\nJOIN\n\n\"invoices\" i\nON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\n\n\nThe following\nis information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\nCustomerId
```

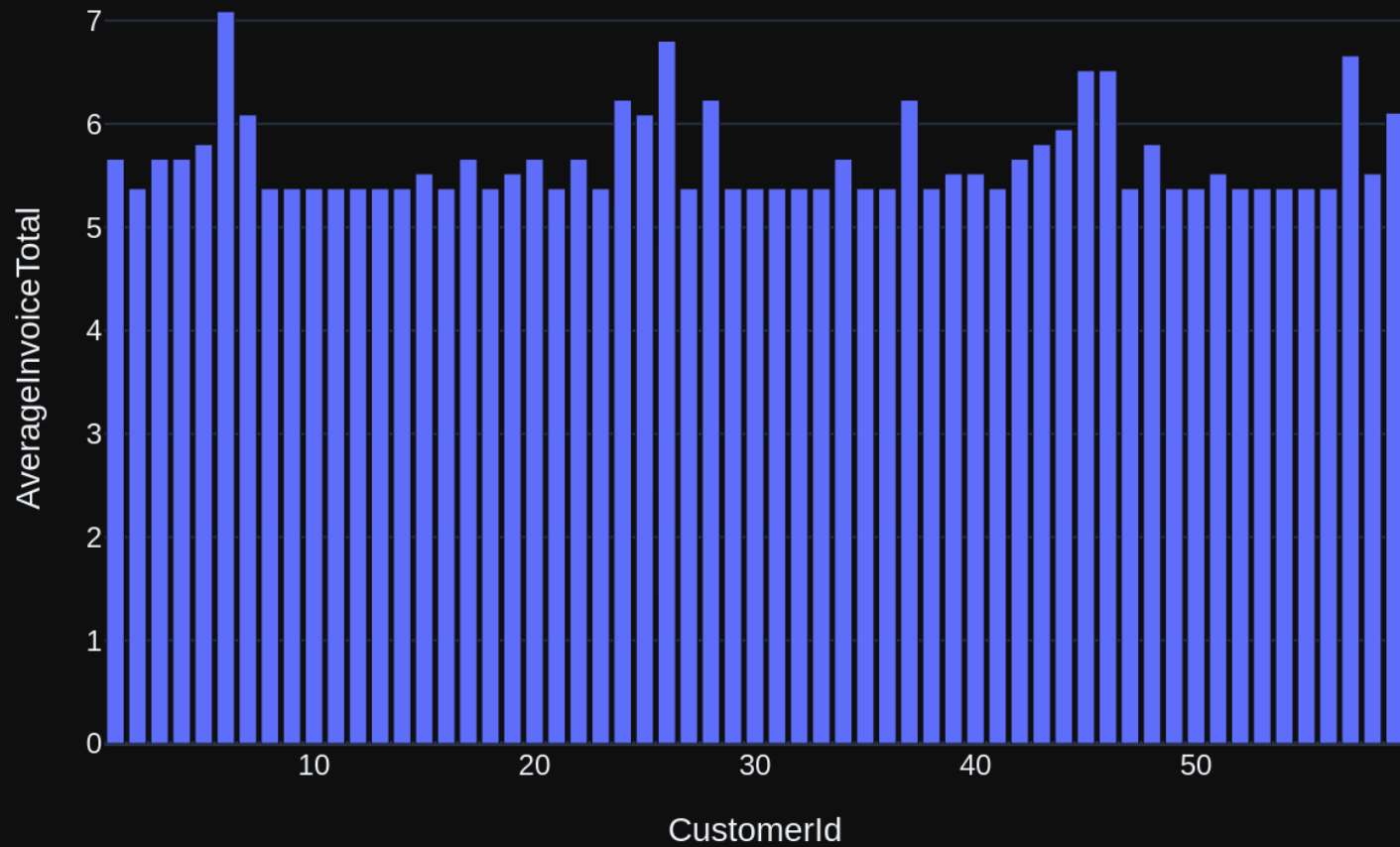


```
int64\nAverageInvoiceTotal    float64\nndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:43:07.578432644Z', 'message': {'role': 'assistant', 'content': "\n\npython\nimport plotly.express as px\n\nfig = px.bar(df, x='CustomerId', y='AverageInvoiceTotal', title='Average Invoice Total by Customer')\n\nfig.show()\n\n"}, 'done_reason': 'stop', 'done': True, 'total_duration': 19928640860, 'load_duration': 640644, 'prompt_eval_count': 189, 'prompt_eval_duration': 8189889000, 'eval_count': 44, 'eval_duration': 11648984000}
```

Average Invoice Total by Customer



```
Out[28]: ('SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM "customers" c\nJOIN "invoices" i ON c.Cus  
tomerId = i.CustomerId\nGROUP BY c.CustomerId',
```

	CustomerId	AverageInvoiceTotal
0	1	5.660000
1	2	5.374286
2	3	5.660000
3	4	5.660000
4	5	5.802857
5	6	7.088571
6	7	6.088571
7	8	5.374286
8	9	5.374286
9	10	5.374286
10	11	5.374286
11	12	5.374286
12	13	5.374286
13	14	5.374286
14	15	5.517143
15	16	5.374286
16	17	5.660000
17	18	5.374286
18	19	5.517143
19	20	5.660000
20	21	5.374286
21	22	5.660000
22	23	5.374286
23	24	6.231429
24	25	6.088571
25	26	6.802857
26	27	5.374286
27	28	6.231429
28	29	5.374286
29	30	5.374286
30	31	5.374286
31	32	5.374286
32	33	5.374286
33	34	5.660000
34	35	5.374286
35	36	5.374286
36	37	6.231429
37	38	5.374286
38	39	5.517143

39	40	5.517143
40	41	5.374286
41	42	5.660000
42	43	5.802857
43	44	5.945714
44	45	6.517143
45	46	6.517143
46	47	5.374286
47	48	5.802857
48	49	5.374286
49	50	5.374286
50	51	5.517143
51	52	5.374286
52	53	5.374286
53	54	5.374286
54	55	5.374286
55	56	5.374286
56	57	6.660000
57	58	5.517143
58	59	6.106667,

```
Figure({
  'data': [{ 'alignmentgroup': 'True',
    'hovertemplate': 'CustomerId=%{x}<br>AverageInvoiceTotal=%{y}<extra></extra>',
    'legendgroup': '',
    'marker': { 'color': '#636efa', 'pattern': { 'shape': '' } },
    'name': '',
    'offsetgroup': '',
    'orientation': 'v',
    'showlegend': False,
    'textposition': 'auto',
    'type': 'bar',
    'x': array([ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
      19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36,
      37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54,
      55, 56, 57, 58, 59]),
    'xaxis': 'x',
    'y': array([5.66, 5.37428571, 5.66, 5.66, 5.80285714, 7.08857143,
      6.08857143, 5.37428571, 5.37428571, 5.37428571, 5.37428571, 5.37428571,
      5.37428571, 5.37428571, 5.51714286, 5.37428571, 5.66, 5.37428571,
      5.51714286, 5.66, 5.37428571, 5.66, 5.37428571, 6.23142857,
      6.08857143, 6.80285714, 5.37428571, 6.23142857, 5.37428571, 5.37428571,
      5.37428571, 5.37428571, 5.37428571, 5.66, 5.37428571, 5.37428571,
```

```

        6.23142857, 5.37428571, 5.51714286, 5.51714286, 5.37428571, 5.66      ,
        5.80285714, 5.94571429, 6.51714286, 6.51714286, 5.37428571, 5.80285714,
        5.37428571, 5.37428571, 5.51714286, 5.37428571, 5.37428571, 5.37428571,
        5.37428571, 5.37428571, 6.66      , 5.51714286, 6.10666667]),
    'yaxis': 'y'}],
    'layout': {'barmode': 'relative',
    'legend': {'tracegroupgap': 0},
    'template': '...',
    'title': {'text': 'Average Invoice Total by Customer'},
    'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'CustomerId'}},
    'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'AverageInvoiceTotal'}}}
    )))

```

```

In [29]: question = """
        Find the top 5 most expensive tracks (based on unit price):
        """

        vn.ask(question=question)

```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

```
[{'role': 'system', 'content': 'You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables\nCREATE TABLE "tracks"\n(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES "albums" (AlbumId)\n)\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\nCREATE INDEX IFK_TrackGenreId ON "tracks" (GenreId)\nCREATE INDEX IFK_PlaylistTrackTrackId ON "playlist_track" (TrackId)\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\nCREATE INDEX IFK_TrackMediaTypeId ON "tracks" (MediaTypeId)\nCREATE TABLE "invoice_items"\n(\n    InvoiceLineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    InvoiceId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    Quantity INTEGER NOT NULL,\n    FOREIGN KEY (InvoiceId) REFERENCES "invoices" (InvoiceId)\n)\nCREATE TABLE "playlist_track"\n(\n    PlaylistId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId)\n)\nCREATE TABLE "albums"\n(\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Title NVARCHAR(160) NOT NULL,\n    ArtistId INTEGER NOT NULL,\n    FOREIGN KEY (ArtistId) REFERENCES "artists" (ArtistId)\n)\n===Additional Context\nIn the SQLite database invoice means order\n===Response Guidelines\n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question.\n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql\n3. If the provided context is insufficient, please explain why it can't be generated.\n4. Please use the most relevant table(s).\n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before.\n'}], {'role': 'user', 'content': '\n    Find the top 5 most expensive tracks (based on unit price):\n'}, {'role': 'assistant', 'content': 'SELECT t.TrackId, t.Name, t.UnitPrice\nFROM "tracks" t\nORDER BY t.UnitPrice DESC\nLIMIT 5'}, {'role': 'user', 'content': '\n    There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\n    Can you find the top 10 most popular artists based on the number of tracks\n'}, {'role': 'assistant', 'content': 'SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nJOIN "tracks" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10'}, {'role': 'user', 'content': '\n    Hint: album quantity is found in invoice_items,\n    Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': '\n    Hint: album quantity is found in invoice_items,\n    \n    '}
```

```
Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role':
'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices" i\nJO
IN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMI
T 5'}, {'role': 'user', 'content': ' \n      Find the top 5 customers who spent the most money overall, \n
\n      Hint: order total can be found on invoices table, calculation using invoice_items detail table is un
necessary \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM "cu
stomers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DE
SC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Find the customer who bought the most albums in total
quantity (across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.Track
Id) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_ite
ms" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'role':
'user', 'content': ' \n      Get all playlists containing at least 10 tracks and the total duration of thos
e tracks:\n'}, {'role': 'assistant', 'content': 'SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Millis
econds) AS TotalDuration\nFROM "playlist_track" pt\nJOIN "playlists" p ON pt.PlaylistId = p.PlaylistId\nJOI
N "tracks" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10'},
{'role': 'user', 'content': ' \n      Identify artists who have albums with tracks appearing in multiple ge
nres:\n\n\n'}, {'role': 'assistant', 'content': 'SELECT a.ArtistId, a.Name AS ArtistName\nFROM "artists" a
\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n  SELECT g2.GenreId\n  FROM "genres"
g2\n  JOIN "tracks" t2 ON g2.GenreId = t2.GenreId\n  GROUP BY g2.GenreId\n  HAVING COUNT(g2.GenreId) > 1\n)
\nGROUP BY a.ArtistId, a.Name'}, {'role': 'user', 'content': ' \n      List all genres and the number of tra
cks in each genre:\n'}, {'role': 'assistant', 'content': 'SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nF
ROM "genres" g\nJOIN "tracks" t ON g.GenreId = t.GenreId\nGROUP BY g.Name'}, {'role': 'user', 'content': '
\n      Find all tracks with a name containing "What" (case-insensitive)\n'}, {'role': 'assistant', 'conten
t': 'SELECT * \nFROM "tracks" \nWHERE LOWER(Name) LIKE \'%what%\'}}, {'role': 'user', 'content': ' \n      F
ind the top 5 most expensive tracks (based on unit price):\n'}]
```

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the
question. Your response should ONLY be based on the given context and follow the response guidelines and fo
rmat instructions. \n===Tables\nCREATE TABLE \"tracks\"(\n  TrackId INTEGER PRIMARY KEY AUTOINCREM
ENT NOT NULL,\n  Name NVARCHAR(200) NOT NULL,\n  AlbumId INTEGER,\n  MediaTypeId INTEGER NOT
NULL,\n  GenreId INTEGER,\n  Composer NVARCHAR(220),\n  Milliseconds INTEGER NOT NULL,\n  Bytes INTEGER,\n  UnitPrice NUMERIC(10,2) NOT NULL,\n  FOREIGN KEY (AlbumId) REFERENCES \"albums\"
(AlbumId) \n\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n  FOREIGN KEY (GenreId) REFERENCES \"genres
\" (GenreId) \n\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n  FOREIGN KEY (MediaTypeId) REFERENCES
\"media_types\" (MediaTypeId) \n\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n\n)\n\nCREATE INDEX IFK_Trac
kAlbumId ON \"tracks\" (AlbumId)\n\nCREATE INDEX IFK_TrackGenreId ON \"tracks\" (GenreId)\n\nCREATE INDEX I
FK_PlaylistTrackTrackId ON \"playlist_track\" (TrackId)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON \"invoice
_items\" (TrackId)\n\nCREATE INDEX IFK_TrackMediaTypeId ON \"tracks\" (MediaTypeId)\n\nCREATE TABLE \"invoi
```

```

ce_items"\r\n(\r\n    InvoiceLineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    InvoiceId INTEGER
NOT NULL,\r\n    TrackId INTEGER NOT NULL,\r\n    UnitPrice NUMERIC(10,2) NOT NULL,\r\n    Quantity INTEG
ER NOT NULL,\r\n    FOREIGN KEY (InvoiceId) REFERENCES \"invoices\" (InvoiceId) \r\n\t\t\tON DELETE NO ACTIO
N ON UPDATE NO ACTION,\r\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\t\tON DELETE NO AC
TION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"playlist_track\"\r\n(\r\n    PlaylistId INTEGER NOT NUL
L,\r\n    TrackId INTEGER NOT NULL,\r\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\r\n
    FOREIGN KEY (PlaylistId) REFERENCES \"playlists\" (PlaylistId) \r\n\t\t\tON DELETE NO ACTION ON UP
DATE NO ACTION,\r\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\t\tON DELETE NO ACTION ON
UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_AlbumArtistId ON \"albums\" (ArtistId)\n\nCREATE TABLE \"albums
\"\r\n(\r\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Title NVARCHAR(160) NOT NULL,\r
\n    ArtistId INTEGER NOT NULL,\r\n    FOREIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistId) \r\n\t\t\tO
N DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\n\n===Additional Context \n\nIn the SQLite database invoice m
eans order\n\n===Response Guidelines \n1. If the provided context is sufficient, please generate a valid SQ
L query without any explanations for the question. \n2. If the provided context is almost sufficient but re
quires knowledge of a specific string in a particular column, please generate an intermediate SQL query to
find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql \n3. If
the provided context is insufficient, please explain why it can't be generated. \n4. Please use the most re
levant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly
as it was given before. \n\"}, {\"role\": \"user\", \"content\": \" \n    Find the top 5 most expensive tracks (ba
sed on unit price):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT t.TrackId, t.Name, t.UnitPrice\nFROM \"tr
acks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n    There are 3 tables: art
ists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by A
lbumId,\n    Can you find the top 10 most popular artists based on the number of tracks\n\"}, {\"role\": \"assi
stant\", \"content\": \"SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM \"artists\" a\nJOIN \"albums\" al
ON a.ArtistId = al.ArtistId\nJOIN \"tracks\" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTr
acks DESC\nLIMIT 10\"}, {\"role\": \"user\", \"content\": \" \n    Hint: album quantity is found in invoice_items,
\n    \n    Find the top 5 customers who bought the most albums in total quantity (across all invoice
s):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"cu
stomers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId =
ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \"
\n    Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 customers who bought the m
ost albums in total quantity (across all invoices):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT i.Custome
rId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId
= ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \"
\n    Find the top 5 customers who spent the most money overall, \n    \n    Hint: order total can be fo
und on invoices table, calculation using invoice_items detail table is unnecessary \n\"}, {\"role\": \"assistan
t\", \"content\": \"SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\" i
ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5\"}, {\"role\": \"use
r\", \"content\": \" \n    Find the customer who bought the most albums in total quantity (across all invoice
s): \n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"c
ustomers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId
= ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1\"}, {\"role\": \"user\", \"content\": \"

```



```

\n      Get all playlists containing at least 10 tracks and the total duration of those tracks:\n"}, {"role": "assistant", "content": "SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM \"playlist_track\" pt\nJOIN \"playlists\" p ON pt.PlaylistId = p.PlaylistId\nJOIN \"tracks\" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10"}, {"role": "user", "content": "  \n      Identify artists who have albums with tracks appearing in multiple genres:\n\n\n"}, {"role": "assistant", "content": "SELECT a.ArtistId, a.Name AS ArtistName\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n  SELECT g2.GenreId\n  FROM \"genres\" g2\n  JOIN \"tracks\" t2 ON g2.GenreId = t2.GenreId\n  GROUP BY g2.GenreId\n  HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name"}, {"role": "user", "content": "  \n      List all genres and the number of tracks in each genre:\n"}, {"role": "assistant", "content": "SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM \"genres\" g\nJOIN \"tracks\" t ON g.GenreId = t.GenreId\nGROUP BY g.Name"}, {"role": "user", "content": "  \n      Find all tracks with a name containing \"What\" (case-insensitive)\n"}, {"role": "assistant", "content": "SELECT * \nFROM \"tracks\" \nWHERE LOWER(Name) LIKE '%what%'"}, {"role": "user", "content": "  \n      Find the top 5 most expensive tracks (based on unit price):\n"}]

```

Add of existing embedding ID: f33f8cb6-1b12-5ea7-8d9a-aef8166b9970-sql

Insert of existing embedding ID: f33f8cb6-1b12-5ea7-8d9a-aef8166b9970-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:44:36.098204214Z', 'message': {'role': 'assistant', 'content': 'SELECT t.TrackId, t.Name, t.UnitPrice\nFROM "tracks" t\nORDER BY t.UnitPrice DESC\nLIMIT 5'}, 'done_reason': 'stop', 'done': True, 'total_duration': 88423453404, 'load_duration': 758094, 'prompt_eval_count': 1799, 'prompt_eval_duration': 79604055000, 'eval_count': 31, 'eval_duration': 8171465000}
```

```
SELECT t.TrackId, t.Name, t.UnitPrice
FROM "tracks" t
ORDER BY t.UnitPrice DESC
LIMIT 5
SELECT t.TrackId, t.Name, t.UnitPrice
FROM "tracks" t
ORDER BY t.UnitPrice DESC
LIMIT 5
```

	TrackId	Name	UnitPrice
0	2819	Battlestar Galactica: The Story So Far	1.99
1	2820	Occupation / Precipice	1.99
2	2821	Exodus, Pt. 1	1.99
3	2822	Exodus, Pt. 2	1.99
4	2823	Collaborators	1.99

Ollama parameters:

```
model=codegemma:latest,
options={},
keep_alive=None
```

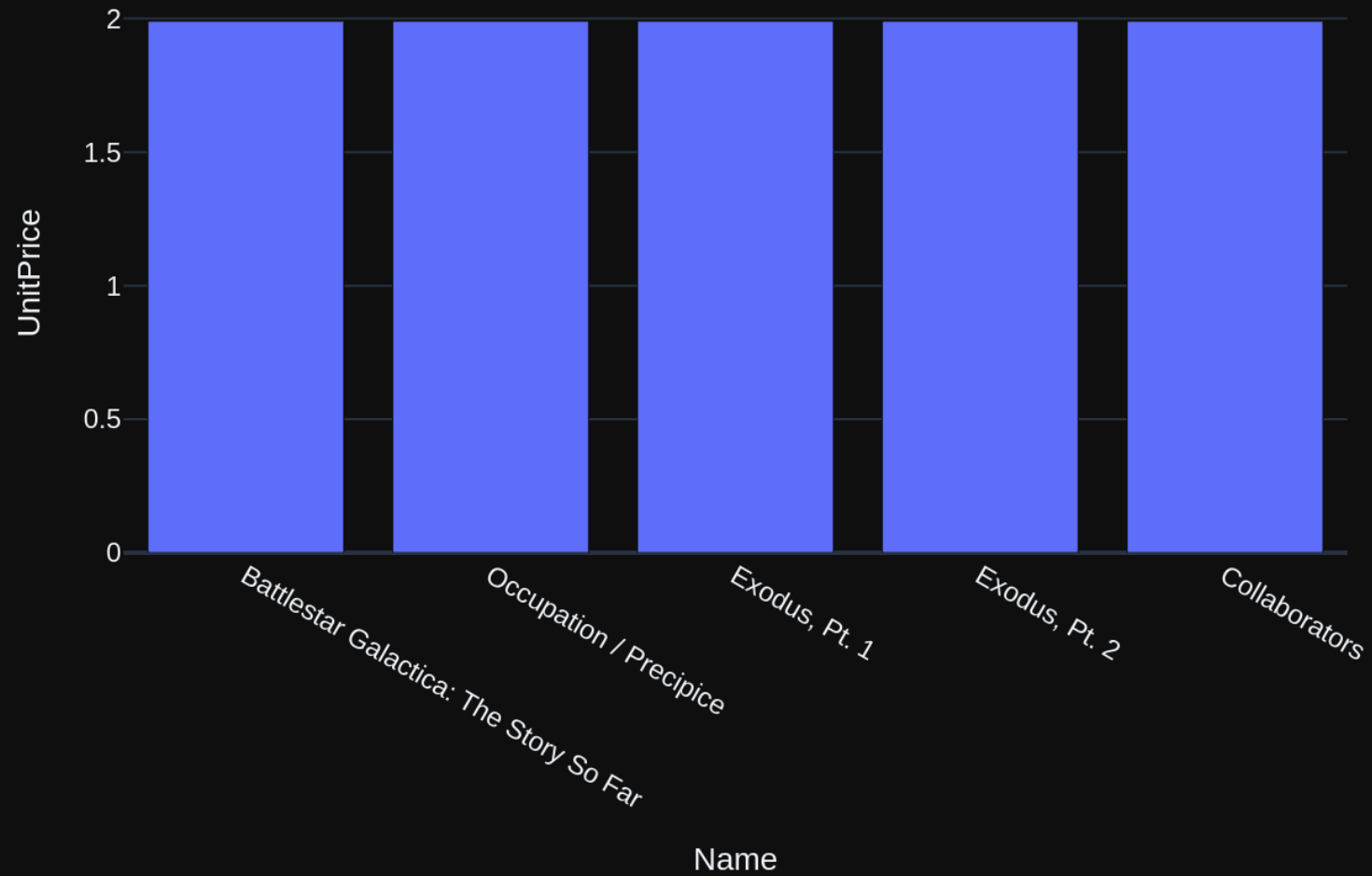
Prompt Content:

```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n Find the top 5 most expensive tracks (based on unit price):\n'\n\nThe DataFrame was produced using this query: SELECT t.TrackId, t.Name, t.UnitPrice\nFROM \"tracks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n TrackId          int64\nName              object\nUnitPrice         float64\ndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:44:53.983570949Z', 'message': {'role': 'assistant', 'content': "\n\npython\nimport plotly.express as px\n\nfig = px.bar(df, x='Name', y='UnitPrice', title='Top 5 Most Expensive Tracks')\nfig.show()\n\n"}, 'done_reason': 'stop', 'done': True, 'total_duration': 17858013273, 'load_duration': 628930, 'prompt_eval_count': 185, 'prompt_eval_duration': 7458965000, 'eval_count': 43, 'eval_duration': 10306053000}
```

Top 5 Most Expensive Tracks



```
Out[29]: ('SELECT t.TrackId, t.Name, t.UnitPrice\nFROM "tracks" t\nORDER BY t.UnitPrice DESC\nLIMIT 5',
          TrackId      Name      UnitPrice
0      2819  Battlestar Galactica: The Story So Far      1.99
1      2820                      Occupation / Precipice      1.99
2      2821                      Exodus, Pt. 1      1.99
3      2822                      Exodus, Pt. 2      1.99
4      2823                      Collaborators      1.99,
Figure({
  'data': [{'alignmentgroup': 'True',
            'hovertemplate': 'Name=%{x}<br>UnitPrice=%{y}<extra></extra>',
            'legendgroup': '',
            'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
            'name': '',
            'offsetgroup': '',
            'orientation': 'v',
            'showlegend': False,
            'textposition': 'auto',
            'type': 'bar',
            'x': array(['Battlestar Galactica: The Story So Far', 'Occupation / Precipice',
                        'Exodus, Pt. 1', 'Exodus, Pt. 2', 'Collaborators'], dtype=object),
            'xaxis': 'x',
            'y': array([1.99, 1.99, 1.99, 1.99, 1.99]),
            'yaxis': 'y'}],
  'layout': {'barmode': 'relative',
             'legend': {'tracegroupgap': 0},
             'template': '...',
             'title': {'text': 'Top 5 Most Expensive Tracks'},
             'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Name'}},
             'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'UnitPrice'}}}
  )))
```

```
In [30]: question = """
          List all genres and the number of tracks in each genre:
          """

          vn.ask(question=question)
```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

```
[{'role': 'system', 'content': 'You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables\nCREATE TABLE "tracks"\n(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES "albums" (AlbumId)\n)\nCREATE INDEX IFK_TrackGenreId ON "tracks" (GenreId)\nCREATE TABLE "genres"\n(\n    GenreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\nCREATE INDEX IFK_PlaylistTrackTrackId ON "playlist_track" (TrackId)\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\nCREATE TABLE "playlists"\n(\n    PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\nCREATE INDEX IFK_TrackMediaTypeId ON "tracks" (MediaTypeId)\nCREATE TABLE "playlist_track"\n(\n    PlaylistId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId)\n)\nCREATE INDEX IFK_TrackTrackId ON "playlist_track" (TrackId)\nCREATE TABLE "albums"\n(\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Title NVARCHAR(160) NOT NULL,\n    ArtistId INTEGER NOT NULL,\n    FOREIGN KEY (ArtistId) REFERENCES "artists" (ArtistId)\n)\nCREATE INDEX IFK_AlbumArtistId ON "albums" (ArtistId)\n\n===Additional Context\nIn the SQLite database invoice means order\n\n===Response Guidelines\n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question.\n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql\n3. If the provided context is insufficient, please explain why it can't be generated.\n4. Please use the most relevant table(s).\n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before.\n\n}', {'role': 'user', 'content': '\n    List all genres and the number of tracks in each genre:\n'}, {'role': 'assistant', 'content': 'SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM "genres" g\nJOIN "tracks" t ON g.GenreId = t.GenreId\nGROUP BY g.Name'}, {'role': 'user', 'content': '\n    There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\n    Can you find the top 10 most popular artists based on the number of tracks\n'}, {'role': 'assistant', 'content': 'SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nJOIN "tracks" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10'}, {'role': 'user', 'content': '\n    Identify artists who have albums with tracks appearing in multiple genres:\n\n'}, {'role': 'assistant', 'content': 'SELECT a.ArtistId, a.Name AS ArtistName\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n    SELECT g2.GenreId\n    FROM "genres" g2\n    JOIN "tracks" t2 ON g2.GenreId = t2.GenreId\n    GROUP BY g2.GenreId\n    HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name'}, {'role': 'user', 'content': '\n    Get all playlists containing at least 10 tracks and the total duration of those tracks:\n'}, {'role': 'assistant', 'content': 'SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM "playlist_track" pt\nJOIN "playlists" p ON pt.PlaylistId = p.PlaylistId\nJOIN "tracks" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nORDER BY TotalDuration DESC\nLIMIT 10'}
```

```

pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10'}}, {'role': 'user', 'content': ' \n Find the top
5 most expensive tracks (based on unit price):\n'}, {'role': 'assistant', 'content': 'SELECT t.TrackId, t.N
ame, t.UnitPrice\nFROM "tracks" t\nORDER BY t.UnitPrice DESC\nLIMIT 5'}}, {'role': 'user', 'content': ' \n
Hint: album quantity is found in invoice_items, \n \n Find the top 5 customers who bought the most al
bums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, C
OUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN
"invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT
5'}}, {'role': 'user', 'content': ' \n List all albums and their corresponding artist names \n'}, {'rol
e': 'assistant', 'content': 'SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM "albums" a\nJOIN "arti
sts" ar ON a.ArtistId = ar.ArtistId'}, {'role': 'user', 'content': ' \n Find the customer who bought t
he most albums in total quantity (across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT c.Cu
stomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.Custo
merId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums D
ESC\nLIMIT 1'}}, {'role': 'user', 'content': ' \n Hint: album quantity is found in invoice_items, \n
\n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'r
ole': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices"
i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC
\nLIMIT 5'}}, {'role': 'user', 'content': ' \n Find all tracks with a name containing "What" (case-insen
sitive)\n'}, {'role': 'assistant', 'content': 'SELECT * \nFROM "tracks" \nWHERE LOWER(Name) LIKE \'%what%
\''}, {'role': 'user', 'content': ' \n List all genres and the number of tracks in each genre:\n'}]

```

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```

[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the
question. Your response should ONLY be based on the given context and follow the response guidelines and fo
rmat instructions. \n===Tables\nCREATE TABLE \"tracks\"(\n    TrackId INTEGER PRIMARY KEY AUTOINCREM
ENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT
NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    B
ytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES \"albums\"
(AlbumId) \n\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES \"genres
\" (GenreId) \n\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES
\"media_types\" (MediaTypeId) \n\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n\n)\n\nCREATE INDEX IFK_Trac
kGenreId ON \"tracks\" (GenreId)\n\nCREATE TABLE \"genres\"(\n    GenreId INTEGER PRIMARY KEY AUTOINC
REMENT NOT NULL,\n    Name NVARCHAR(120)\n\n)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON \"playlist_trac
k\" (TrackId)\n\nCREATE INDEX IFK_TrackAlbumId ON \"tracks\" (AlbumId)\n\nCREATE TABLE \"playlists\"(\n
    PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n\n)\n\nCREATE INDEX
IFK_TrackMediaTypeId ON \"tracks\" (MediaTypeId)\n\nCREATE TABLE \"playlist_track\"(\n    PlaylistId
INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (Playl
istId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES \"playlists\" (PlaylistId) \n\n\t\tON DELETE NO
ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \n\n\t\tON DELETE

```

```

NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"albums\"(\r\n(\r\n    AlbumId INTEGER PRIMARY KEY AUTOI
NCREMENT NOT NULL,\r\n    Title NVARCHAR(160) NOT NULL,\r\n    ArtistId INTEGER NOT NULL,\r\n    FOREIGN
KEY (ArtistId) REFERENCES \"artists\" (ArtistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCR
EATE INDEX IFK_AlbumArtistId ON \"albums\" (ArtistId)\n\n\n===Additional Context\n\n\nIn the SQLite database
invoice means order\n\n===Response Guidelines\n1. If the provided context is sufficient, please generate a
valid SQL query without any explanations for the question.\n2. If the provided context is almost sufficien
t but requires knowledge of a specific string in a particular column, please generate an intermediate SQL q
uery to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql
\n3. If the provided context is insufficient, please explain why it can't be generated.\n4. Please use the
most relevant table(s).\n5. If the question has been asked and answered before, please repeat the answer e
xactly as it was given before.\n\"}, {\"role\": \"user\", \"content\": \" \n    List all genres and the number of
tracks in each genre:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT g.Name, COUNT(t.GenreId) AS TotalTracks
\nFROM \"genres\" g\nJOIN \"tracks\" t ON g.GenreId = t.GenreId\nGROUP BY g.Name\"}, {\"role\": \"user\", \"conte
nt\": \" \n    There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistI
d, albums and tracks are linked by AlbumId,\n    Can you find the top 10 most popular artists based on the
number of tracks\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFRO
M \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nJOIN \"tracks\" t ON al.AlbumId = t.AlbumI
d\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10\"}, {\"role\": \"user\", \"content\": \" \n    Identify a
rtists who have albums with tracks appearing in multiple genres:\n\n\n\"}, {\"role\": \"assistant\", \"content\":
\"SELECT a.ArtistId, a.Name AS ArtistName\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistI
d\nWHERE a.ArtistId IN (\n    SELECT g2.GenreId\n    FROM \"genres\" g2\n    JOIN \"tracks\" t2 ON g2.GenreId = t
2.GenreId\n    GROUP BY g2.GenreId\n    HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name\"}, {\"rol
e\": \"user\", \"content\": \" \n    Get all playlists containing at least 10 tracks and the total duration of
those tracks:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Mi
lliseconds) AS TotalDuration\nFROM \"playlist_track\" pt\nJOIN \"playlists\" p ON pt.PlaylistId = p.Playlis
tId\nJOIN \"tracks\" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId)
>= 10\"}, {\"role\": \"user\", \"content\": \" \n    Find the top 5 most expensive tracks (based on unit pric
e):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT t.TrackId, t.Name, t.UnitPrice\nFROM \"tracks\" t\nORDER
BY t.UnitPrice DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n    Hint: album quantity is found in invoic
e_items, \n    \n    Find the top 5 customers who bought the most albums in total quantity (across all inv
oices):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM
\"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.Invoice
Id = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\":
\" \n    List all albums and their corresponding artist names \n\"}, {\"role\": \"assistant\", \"content\": \"SELE
CT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM \"albums\" a\nJOIN \"artists\" ar ON a.ArtistId = ar.Ar
tistId\"}, {\"role\": \"user\", \"content\": \" \n    Find the customer who bought the most albums in total quant
ity (across all invoices): \n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) A
S TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_ite
ms\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1\"}, {\"rol
e\": \"user\", \"content\": \" \n    Hint: album quantity is found in invoice_items, \n    \n    Find the top 5
customers who bought the most albums in total quantity (across all invoices):\n\"}, {\"role\": \"assistant\", \"c
ontent\": \"SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_

```

```
items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n    Find all tracks with a name containing \\\"What\\\" (case-insensitive)\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT * \nFROM \\\"tracks\\\" \nWHERE LOWER(Name) LIKE '%what%'\"}, {\"role\": \"user\", \"content\": \" \n    List all genres and the number of tracks in each genre:\n\"}]
```

Add of existing embedding ID: f626b681-4d8f-563a-beee-1ea759baaa82-sql

Insert of existing embedding ID: f626b681-4d8f-563a-beee-1ea759baaa82-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:46:15.309456819Z', 'message': {'role': 'assistant', 'content': 'SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM "genres" g\nJOIN "tracks" t ON g.GenreId = t.GenreId\nGROUP BY g.Name'}, 'done_reason': 'stop', 'done': True, 'total_duration': 81233749327, 'load_duration': 702412, 'prompt_eval_count': 1677, 'prompt_eval_duration': 69448836000, 'eval_count': 44, 'eval_duration': 11158073000}
```

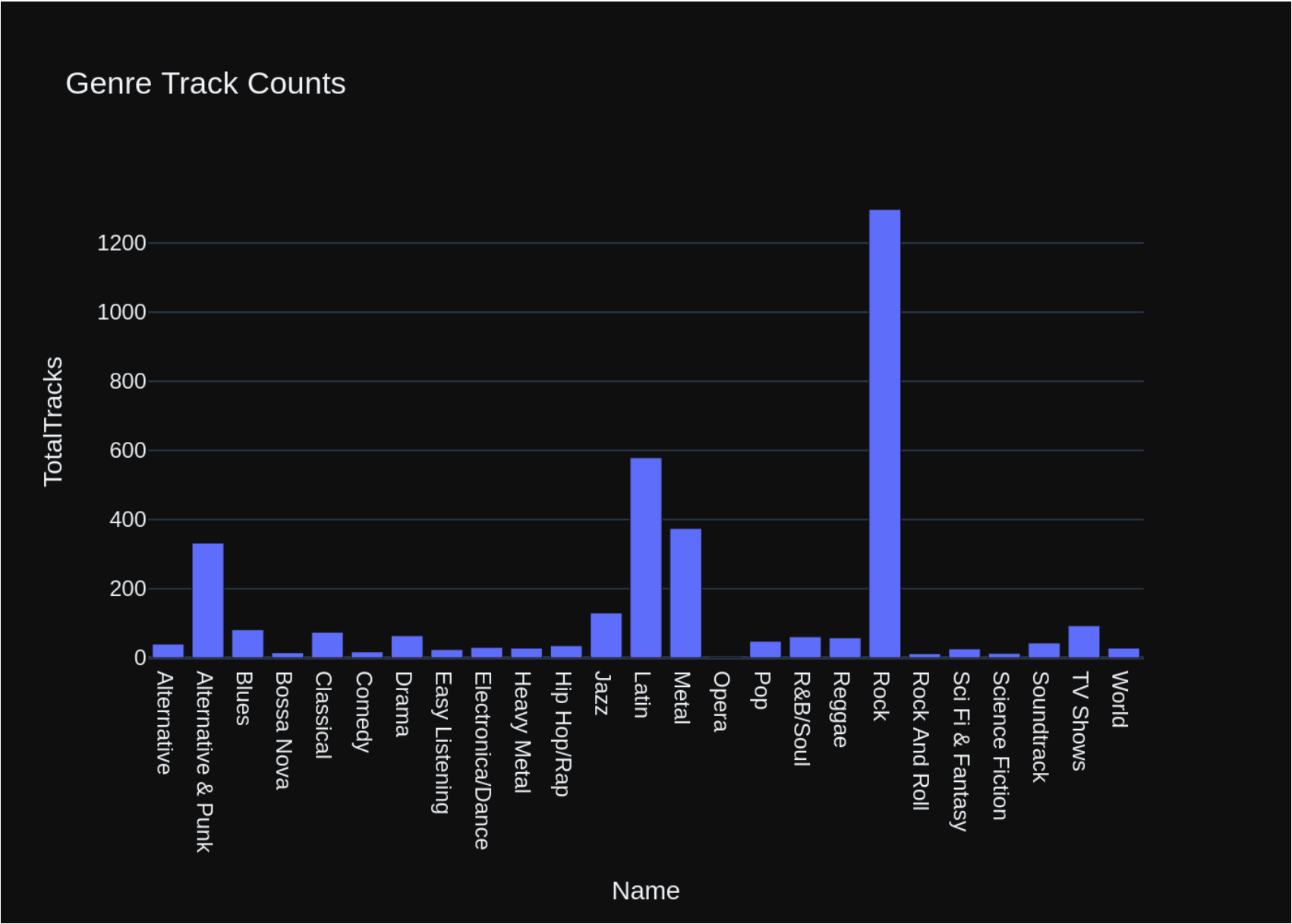
```
SELECT g.Name, COUNT(t.GenreId) AS TotalTracks
FROM "genres" g
JOIN "tracks" t ON g.GenreId = t.GenreId
GROUP BY g.Name
SELECT g.Name, COUNT(t.GenreId) AS TotalTracks
FROM "genres" g
JOIN "tracks" t ON g.GenreId = t.GenreId
GROUP BY g.Name
```

	Name	TotalTracks
0	Alternative	40
1	Alternative & Punk	332
2	Blues	81
3	Bossa Nova	15
4	Classical	74
5	Comedy	17
6	Drama	64
7	Easy Listening	24
8	Electronica/Dance	30
9	Heavy Metal	28
10	Hip Hop/Rap	35
11	Jazz	130
12	Latin	579
13	Metal	374
14	Opera	1
15	Pop	48
16	R&B/Soul	61
17	Reggae	58
18	Rock	1297
19	Rock And Roll	12
20	Sci Fi & Fantasy	26
21	Science Fiction	13
22	Soundtrack	43
23	TV Shows	93
24	World	28

Ollama parameters:

model=codegemma:latest,

```
options={},
keep_alive=None
Prompt Content:
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n      List all genres and the number of tracks in each genre:\n'\n\nThe DataFrame was produced using this query: SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM\n\"genres\" g\nJOIN \"tracks\" t ON g.GenreId = t.GenreId\nGROUP BY g.Name\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n Name                object\nTotalTracks      int64\nndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
Ollama Response:
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:46:33.829070714Z', 'message': {'role': 'assistant', 'content': "\n\npython\nimport plotly.express as px\n\nnfig = px.bar(df, x='Name', y='TotalTracks', title='Genre Track Counts')\n\nnfig.show()\n\n"}, 'done_reason': 'stop', 'done': True, 'total_duration': 18492923528, 'load_duration': 676388, 'prompt_eval_count': 190, 'prompt_eval_duration': 8060324000, 'eval_count': 41, 'eval_duration': 10342276000}
```



```
Out[30]: ('SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM "genres" g\nJOIN "tracks" t ON g.GenreId = t.GenreI\nGROUP BY g.Name',
```

	Name	TotalTracks
0	Alternative	40
1	Alternative & Punk	332
2	Blues	81
3	Bossa Nova	15
4	Classical	74
5	Comedy	17
6	Drama	64
7	Easy Listening	24
8	Electronica/Dance	30
9	Heavy Metal	28
10	Hip Hop/Rap	35
11	Jazz	130
12	Latin	579
13	Metal	374
14	Opera	1
15	Pop	48
16	R&B/Soul	61
17	Reggae	58
18	Rock	1297
19	Rock And Roll	12
20	Sci Fi & Fantasy	26
21	Science Fiction	13
22	Soundtrack	43
23	TV Shows	93
24	World	28,

```
Figure({
  'data': [{'alignmentgroup': 'True',
            'hovertemplate': 'Name={x}<br>TotalTracks={y}<extra></extra>',
            'legendgroup': '',
            'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
            'name': '',
            'offsetgroup': '',
            'orientation': 'v',
            'showlegend': False,
            'textposition': 'auto',
            'type': 'bar',
            'x': array(['Alternative', 'Alternative & Punk', 'Blues', 'Bossa Nova', 'Classical',
                        'Comedy', 'Drama', 'Easy Listening', 'Electronica/Dance', 'Heavy Metal',
                        'Hip Hop/Rap', 'Jazz', 'Latin', 'Metal', 'Opera', 'Pop', 'R&B/Soul',
```

```

        'Reggae', 'Rock', 'Rock And Roll', 'Sci Fi & Fantasy',
        'Science Fiction', 'Soundtrack', 'TV Shows', 'World'], dtype=object),
    'xaxis': 'x',
    'y': array([ 40, 332, 81, 15, 74, 17, 64, 24, 30, 28, 35, 130,
                579, 374, 1, 48, 61, 58, 1297, 12, 26, 13, 43, 93,
                28]),
    'yaxis': 'y'}],
    'layout': {'barmode': 'relative',
               'legend': {'tracegroupgap': 0},
               'template': '...',
               'title': {'text': 'Genre Track Counts'},
               'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Name'}},
               'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'TotalTracks'}}}
    ))

```

```

In [31]: question = """
        Get all genres that do not have any tracks associated with them:
        """

        vn.ask(question=question)

```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

```
ist track" pt\nJOIN "playlists" p ON pt.PlaylistId = p.PlaylistId\nJOIN "tracks" t ON pt.TrackId = t.TrackId
```

```

d\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10'}, {'role': 'user', 'content': ' \n  F
ind all tracks with a name containing "What" (case-insensitive)\n'}, {'role': 'assistant', 'content': 'SELE
CT * \nFROM "tracks" \nWHERE LOWER(Name) LIKE \'%what%\'}}, {'role': 'user', 'content': ' \n  List all a
lbums and their corresponding artist names \n'}, {'role': 'assistant', 'content': 'SELECT a.Title, a.Artis
tId, ar.Name AS ArtistName\nFROM "albums" a\nJOIN "artists" ar ON a.ArtistId = ar.ArtistId'}, {'role': 'use
r', 'content': ' \n  Find the top 5 most expensive tracks (based on unit price):\n'}, {'role': 'assistan
t', 'content': 'SELECT t.TrackId, t.Name, t.UnitPrice\nFROM "tracks" t\nORDER BY t.UnitPrice DESC\nLIMIT
5'}, {'role': 'user', 'content': ' \n  Find the customer who bought the most albums in total quantity
(across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS To
talAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON
i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'role': 'user',
'content': ' \n  Hint: album quantity is found in invoice_items, \n  \n  Find the top 5 customers wh
o bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SE
LECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId
= i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY Tot
alAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n  Hint: album quantity is found in invoice_item
s, \n  \n  Find the top 5 customers who bought the most albums in total quantity (across all invoice
s):\n'}, {'role': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFRO
M "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY Tot
alAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n  Get all genres that do not have any tracks as
sociated with them:\n'}]

```

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```

[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the
question. Your response should ONLY be based on the given context and follow the response guidelines and fo
rmat instructions. \n===Tables \nCREATE INDEX IFK_TrackGenreId ON \"tracks\" (GenreId)\n\nCREATE TABLE \"tr
acks\"(\n\n  TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n\n  Name NVARCHAR(200) NOT NUL
L,\n\n  AlbumId INTEGER,\n\n  MediaTypeId INTEGER NOT NULL,\n\n  GenreId INTEGER,\n\n  Composer NV
ARCHAR(220),\n\n  Milliseconds INTEGER NOT NULL,\n\n  Bytes INTEGER,\n\n  UnitPrice NUMERIC(10,2) N
OT NULL,\n\n  FOREIGN KEY (AlbumId) REFERENCES \"albums\" (AlbumId) \n\n\t\t\tON DELETE NO ACTION ON UPDATE
NO ACTION,\n\n  FOREIGN KEY (GenreId) REFERENCES \"genres\" (GenreId) \n\n\t\t\tON DELETE NO ACTION ON UPDA
TE NO ACTION,\n\n  FOREIGN KEY (MediaTypeId) REFERENCES \"media_types\" (MediaTypeId) \n\n\t\t\tON DELETE N
O ACTION ON UPDATE NO ACTION\n\n)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON \"playlist_track\" (TrackId)
\n\nCREATE INDEX IFK_TrackMediaTypeId ON \"tracks\" (MediaTypeId)\n\nCREATE INDEX IFK_TrackAlbumId ON \"tra
cks\" (AlbumId)\n\nCREATE TABLE \"genres\"(\n\n  GenreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NUL
L,\n\n  Name NVARCHAR(120)\n\n)\n\nCREATE TABLE \"albums\"(\n\n  AlbumId INTEGER PRIMARY KEY AUTOIN
CREMENT NOT NULL,\n\n  Title NVARCHAR(160) NOT NULL,\n\n  ArtistId INTEGER NOT NULL,\n\n  FOREIGN K
EY (ArtistId) REFERENCES \"artists\" (ArtistId) \n\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n\n)\n\nCRE
ATE TABLE \"playlist_track\"(\n\n  PlaylistId INTEGER NOT NULL,\n\n  TrackId INTEGER NOT NULL,\n

```

```

\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\r\n    FOREIGN KEY (PlaylistId) REFER
ENCES \"playlists\" (PlaylistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n    FOREIGN KEY (Track
Id) REFERENCES \"tracks\" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IF
K_AlbumArtistId ON \"albums\" (ArtistId)\n\nCREATE TABLE \"playlists\"(\r\n(\r\n    PlaylistId INTEGER PRIMA
RY KEY AUTOINCREMENT NOT NULL,\r\n    Name NVARCHAR(120)\r\n)\n\n\n===Additional Context \n\nIn the SQLite
database invoice means order\n\n===Response Guidelines \n1. If the provided context is sufficient, please g
enerate a valid SQL query without any explanations for the question. \n2. If the provided context is almost
sufficient but requires knowledge of a specific string in a particular column, please generate an intermedi
ate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermed
iate_sql \n3. If the provided context is insufficient, please explain why it can't be generated. \n4. Pleas
e use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat th
e answer exactly as it was given before. \n\"}, {\"role\": \"user\", \"content\": \" \n    Identify artists who h
ave albums with tracks appearing in multiple genres:\n\n\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT a.Ar
tistId, a.Name AS ArtistName\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nWHERE a.A
rtistId IN (\n SELECT g2.GenreId\n FROM \"genres\" g2\n JOIN \"tracks\" t2 ON g2.GenreId = t2.GenreId\n
GROUP BY g2.GenreId\n HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name\"}, {\"role\": \"user\", \"co
ntent\": \" \n    There are 3 tables: artists, albums and tracks, where albums and artists are linked by Arti
stId, albums and tracks are linked by AlbumId,\n    Can you find the top 10 most popular artists based on t
he number of tracks\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\n
FROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nJOIN \"tracks\" t ON al.AlbumId = t.Alb
umId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10\"}, {\"role\": \"user\", \"content\": \" \n    List all
genres and the number of tracks in each genre:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT g.Name, COUNT
(t.GenreId) AS TotalTracks\nFROM \"genres\" g\nJOIN \"tracks\" t ON g.GenreId = t.GenreId\nGROUP BY g.Nam
e\"}, {\"role\": \"user\", \"content\": \" \n    Get all playlists containing at least 10 tracks and the total du
ration of those tracks:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT pt.PlaylistId, p.Name AS PlaylistNam
e, SUM(t.Milliseconds) AS TotalDuration\nFROM \"playlist_track\" pt\nJOIN \"playlists\" p ON pt.PlaylistId
= p.PlaylistId\nJOIN \"tracks\" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(p
t.TrackId) >= 10\"}, {\"role\": \"user\", \"content\": \" \n    Find all tracks with a name containing \"What\" (c
ase-insensitive)\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT * \nFROM \"tracks\" \nWHERE LOWER(Name) LIKE
'what%'\"}, {\"role\": \"user\", \"content\": \" \n    List all albums and their corresponding artist names
\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM \"albums\"
a\nJOIN \"artists\" ar ON a.ArtistId = ar.ArtistId\"}, {\"role\": \"user\", \"content\": \" \n    Find the top 5 m
ost expensive tracks (based on unit price):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT t.TrackId, t.Nam
e, t.UnitPrice\nFROM \"tracks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n
Find the customer who bought the most albums in total quantity (across all invoices): \n\"}, {\"role\": \"assis
tant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invo
ices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY
c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1\"}, {\"role\": \"user\", \"content\": \" \n    Hint: album quanti
ty is found in invoice_items, \n    \n    Find the top 5 customers who bought the most albums in total quan
tity (across all invoices):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) A
S TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_ite
ms\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5\"}, {\"rol

```



```
e": "user", "content": "  \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": "  \n      Get all genres that do not have any tracks associated with them:\n"}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:47:56.878839054Z', 'message': {'role': 'assistant', 'content': 'SELECT g.GenreId, g.Name\nFROM "genres" g\nLEFT JOIN "tracks" t ON g.GenreId = t.GenreId\nWHERE t.GenreId IS NULL;'}, 'done_reason': 'stop', 'done': True, 'total_duration': 82936127468, 'load_duration': 695066, 'prompt_eval_count': 1678, 'prompt_eval_duration': 71718525000, 'eval_count': 42, 'eval_duration': 10569740000}
```

```
SELECT g.GenreId, g.Name
FROM "genres" g
LEFT JOIN "tracks" t ON g.GenreId = t.GenreId
WHERE t.GenreId IS NULL;
```

Output from LLM: `SELECT q.GenreId, q.Name`

```
FROM "genres" g
LEFT JOIN "tracks" t ON g.GenreId = t.GenreId
WHERE t.GenreId IS NULL;
```

Extracted SQL: SELECT g.GenreId, g.Name

```
FROM "genres" g
LEFT JOIN "tracks" t ON g.GenreId = t.GenreId
WHERE t.GenreId IS NULL
```

```
SELECT g.GenreId, g.Name
FROM "genres" g
LEFT JOIN "tracks" t ON g.GenreId = t.GenreId
WHERE t.GenreId IS NULL
```

Empty DataFrame

```
Columns: [GenreId, Name]
```

Index: []

Ollama parameters:

```
model=codegemma:latest,
```

```
options={},
```

```
keep alive=None
```

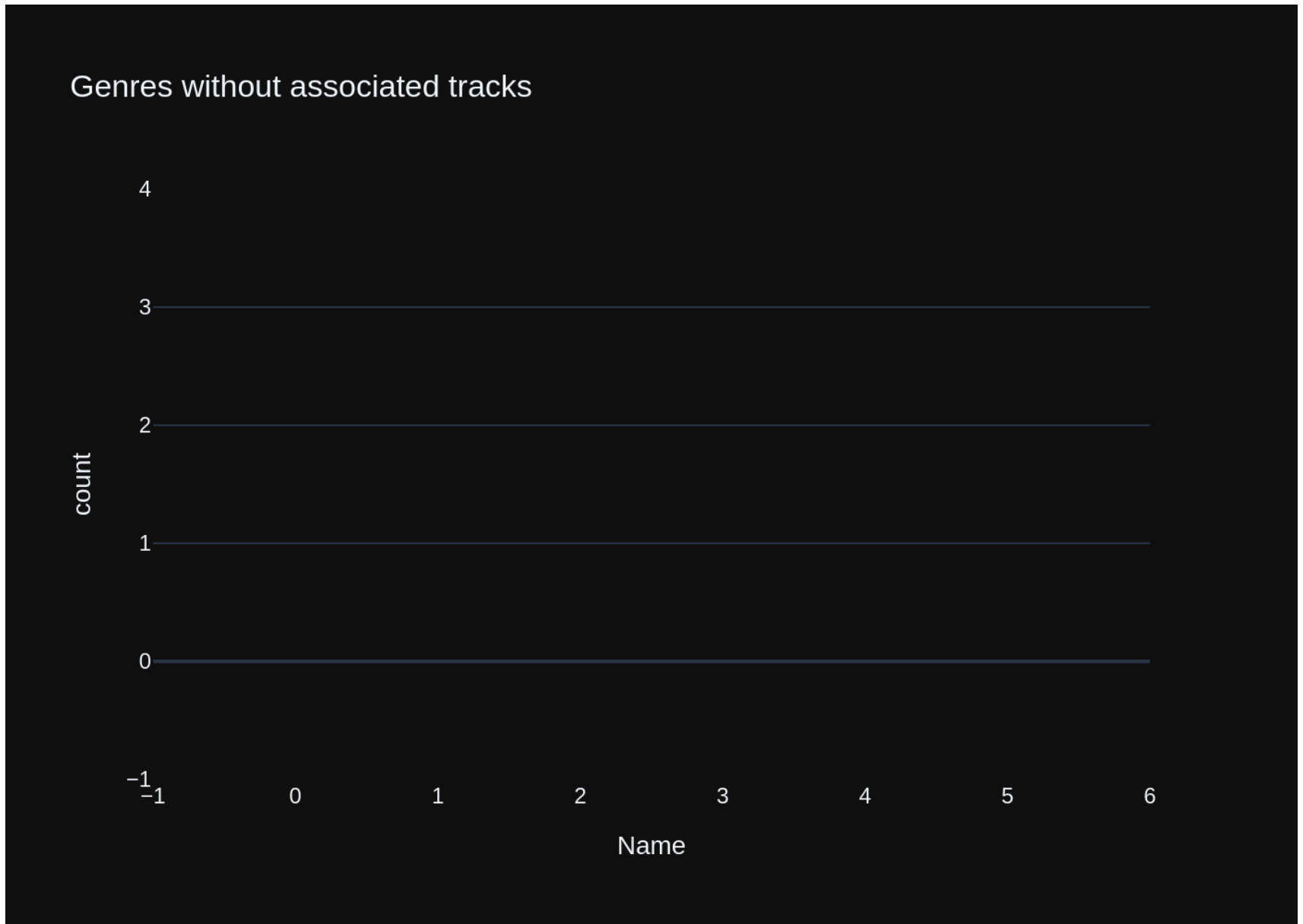
Prompt Content:

```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n      Get all genres that do not have any tracks associated with them:\n'\n\nThe DataFrame was produced using this query: SELECT g.GenreId, g.Name\nFROM \"genres\" g\nLEFT JOIN \"tracks\" t ON g.GenreId = t.GenreId\nWHERE t.GenreId IS NULL\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\nGenreId      object\nName          object\ndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the da"}]
```

taframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:48:13.08422513Z', 'message': {'role': 'assistant', 'content': "\n\npython\nimport plotly.express as px\n\nfig = px.bar(df, x='Name', title='Genres without associated tracks')\nfig.show()\n\n"}, 'done_reason': 'stop', 'done': True, 'total_duration': 16202935152, 'load_duration': 768206, 'prompt_eval_count': 186, 'prompt_eval_duration': 7374055000, 'eval_count': 37, 'eval_duration': 8739586000}
```



```

Out[31]: ('SELECT g.GenreId, g.Name\nFROM "genres" g\nLEFT JOIN "tracks" t ON g.GenreId = t.GenreId\nWHERE t.GenreI
d IS NULL',
Empty DataFrame
Columns: [GenreId, Name]
Index: [],
Figure({
  'data': [{ 'alignmentgroup': 'True',
    'hovertemplate': 'Name=%{x}<br>count=%{y}<extra></extra>',
    'legendgroup': '',
    'marker': { 'color': '#636efa', 'pattern': { 'shape': '' } },
    'name': '',
    'offsetgroup': '',
    'orientation': 'v',
    'showlegend': False,
    'textposition': 'auto',
    'type': 'bar',
    'x': array([], dtype=object),
    'xaxis': 'x',
    'y': array([], dtype=int64),
    'yaxis': 'y' } ],
  'layout': { 'barmode': 'relative',
    'legend': { 'tracegroupgap': 0 },
    'template': '...',
    'title': { 'text': 'Genres without associated tracks' },
    'xaxis': { 'anchor': 'y', 'domain': [0.0, 1.0], 'title': { 'text': 'Name' } },
    'yaxis': { 'anchor': 'x', 'domain': [0.0, 1.0], 'title': { 'text': 'count' } } }
}))

```

```

In [32]: question = """
        List all customers who have not placed any orders:
        """

vn.ask(question=question)

```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions."}, {"role": "user", "content": "===Tables\n\nCREATE TABLE \"invoices\"\n\n    InvoiceId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n\n    CustomerId INTEGER NOT NULL,\n\n    InvoiceDate DATETIME NOT NULL,\n\n    BillingAddress NVARCHAR(70),\n\n    BillingCity NVARCHAR(40),\n\n    BillingState NVARCHAR(40),\n\n    BillingCountry NVARCHAR(40),\n\n    BillingPostalCode NVARCHAR(10),\n\n    Total NUMERIC(10,2) NOT NULL,\n\n    FOREIGN KEY (CustomerId) REFERENCES \"customers\" (CustomerId) \n\nON DELETE NO ACTION ON UPDATE NO ACTION\n\n\nCREATE TABLE \"customers\"\n\n    CustomerId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n\n    FirstName NVARCHAR(40) NOT NULL,\n\n    LastName NVARCHAR(20) NOT NULL,\n\n    Company NVARCHAR(80),\n\n    Address NVARCHAR(70),\n\n    City NVARCHAR(40),\n\n    State NVARCHAR(40),\n\n    Country NVARCHAR(40),\n\n    PostalCode NVARCHAR(10),\n\n    Phone NVARCHAR(24),\n\n    Fax NVARCHAR(24),\n\n    Email NVARCHAR(60) NOT NULL,\n\n    SupportRepId INTEGER,\n\n    FOREIGN KEY (SupportRepId) REFERENCES \"employees\" (EmployeeId) \n\nON DELETE NO ACTION ON UPDATE NO ACTION\n\n\nCREATE TABLE \"invoice_items\"\n\n    InvoiceLineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n\n    InvoiceId INTEGER NOT NULL,\n\n    TrackId INTEGER NOT NULL,\n\n    UnitPrice NUMERIC(10,2) NOT NULL,\n\n    Quantity INTEGER NOT NULL,\n\n    FOREIGN KEY (InvoiceId) REFERENCES \"invoices\" (InvoiceId) \n\nON DELETE NO ACTION ON UPDATE NO ACTION,\n\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \n\nON DELETE NO ACTION ON UPDATE NO ACTION\n\n\nCREATE TABLE \"employees\"\n\n    EmployeeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n\n    LastName NVARCHAR(20) NOT NULL,\n\n    FirstName NVARCHAR(20) NOT NULL,\n\n    Title NVARCHAR(30),\n\n    ReportsTo INTEGER,\n\n    BirthDate DATETIME,\n\n    HireDate DATETIME,\n\n    Address NVARCHAR(70),\n\n    City NVARCHAR(40),\n\n    State NVARCHAR(40),\n\n    Country NVARCHAR(40),\n\n    PostalCode NVARCHAR(10),\n\n    Phone NVARCHAR(24),\n\n    Fax NVARCHAR(24),\n\n    Email NVARCHAR(60),\n\n    FOREIGN KEY (ReportsTo) REFERENCES \"employees\" (EmployeeId) \n\nON DELETE NO ACTION ON UPDATE NO ACTION\n\n\nCREATE TABLE \"playlist_track\"\n\n    PlaylistId INTEGER NOT NULL,\n\n    TrackId INTEGER NOT NULL,\n\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\n\n    FOREIGN KEY (PlaylistId) REFERENCES \"playlists\" (PlaylistId) \n\nON DELETE NO ACTION ON UPDATE NO ACTION,\n\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \n\nON DELETE NO ACTION ON UPDATE NO ACTION\n\n\nCREATE TABLE \"albums\"\n\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n\n    Title NVARCHAR(160) NOT NULL,\n\n    ArtistId INTEGER NOT NULL,\n\n    FOREIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistId) \n\nON DELETE NO ACTION ON UPDATE NO ACTION\n\n\nCREATE INDEX IFK_CustomerSupportRepId ON \"customers\" (SupportRepId)\n\n\nCREATE TABLE \"playlists\"\n\n    PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n\n    Name NVARCHAR(120)\n\n\nCREATE TABLE \"tracks\"\n\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n\n    Name NVARCHAR(200) NOT NULL,\n\n    AlbumId INTEGER,\n\n    MediaTypeId INTEGER NOT NULL,\n\n    GenreId INTEGER,\n\n    Composer NVARCHAR(220),\n\n    Milliseconds INTEGER NOT NULL,\n\n    Bytes INTEGER,\n\n    UnitPrice NUMERIC(10,2) NOT NULL,\n\n    FOREIGN KEY (AlbumId) REFERENCES \"albums\" (AlbumId) \n\nON DELETE NO ACTION ON UPDATE NO ACTION,\n\n    FOREIGN KEY (GenreId) REFERENCES \"genres\" (GenreId) \n\nON DELETE NO ACTION ON UPDATE NO ACTION,\n\n    FOREIGN KEY (MediaTypeId) REFERENCES \"media_types\" (MediaTypeId) \n\nON DELETE NO ACTION ON UPDATE NO ACTION\n\n\nCREATE INDEX IFK_InvoiceCustomerId ON \"invoices\" (CustomerId)\n\n\n===Additional Context\n\nIn the SQLite database invoice means order\n\n===Response Guidelines\n\n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question.\n\n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepare"}]
```

nd the query with a comment saying `intermediate_sql \n3`. If the provided context is insufficient, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before. \n'}, {'role': 'user', 'content': ' \n Find the customer who bought the most albums in total quantity (across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n Hint: album quantity is found in invoice_items, \n \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Find the customer with the most invoices \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n Find the top 5 customers who spent the most money overall, \n \n Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessary \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Hint: album quantity is found in invoice_items, \n \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n List all customers from Canada and their email addresses:\n'}, {'role': 'assistant', 'content': 'SELECT c.Email, c.Country\nFROM "customers" c\nWHERE c.Country = \'Canada\''}, {'role': 'user', 'content': 'what are the top 5 countries that customers come from?'}, {'role': 'assistant', 'content': 'SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM "customers" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Get the total number of invoices for each customer\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': 'How many customers are there'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM "customers"'}, {'role': 'user', 'content': ' \n Get the average invoice total for each customer:\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': ' \n List all customers who have not placed any orders:\n'}]

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables\nCREATE TABLE `invoices`\n(\n    InvoiceId INTEGER PRIMARY KEY AUTOIN
```

```

CREMENT NOT NULL,\r\n      CustomerId INTEGER NOT NULL,\r\n      InvoiceDate DATETIME NOT NULL,\r\n      Billin
gAddress NVARCHAR(70),\r\n      BillingCity NVARCHAR(40),\r\n      BillingState NVARCHAR(40),\r\n      BillingCou
ntry NVARCHAR(40),\r\n      BillingPostalCode NVARCHAR(10),\r\n      Total NUMERIC(10,2) NOT NULL,\r\n      FORE
IGN KEY (CustomerId) REFERENCES \"customers\" (CustomerId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION
\r\n)\n\nCREATE TABLE \"customers\"\r\n(\r\n      CustomerId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n      Fir
stName NVARCHAR(40) NOT NULL,\r\n      LastName NVARCHAR(20) NOT NULL,\r\n      Company NVARCHAR(80),\r\n      Ad
dress NVARCHAR(70),\r\n      City NVARCHAR(40),\r\n      State NVARCHAR(40),\r\n      Country NVARCHAR(40),\r\n
PostalCode NVARCHAR(10),\r\n      Phone NVARCHAR(24),\r\n      Fax NVARCHAR(24),\r\n      Email NVARCHAR(60) NOT
NULL,\r\n      SupportRepId INTEGER,\r\n      FOREIGN KEY (SupportRepId) REFERENCES \"employees\" (EmployeeId)
\r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"invoice_items\"\r\n(\r\n      InvoiceL
ineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n      InvoiceId INTEGER NOT NULL,\r\n      TrackId INTEGE
R NOT NULL,\r\n      UnitPrice NUMERIC(10,2) NOT NULL,\r\n      Quantity INTEGER NOT NULL,\r\n      FOREIGN KE
Y (InvoiceId) REFERENCES \"invoices\" (InvoiceId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n      F
OREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n
\nCREATE TABLE \"employees\"\r\n(\r\n      EmployeeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n      Last
Name NVARCHAR(20) NOT NULL,\r\n      FirstName NVARCHAR(20) NOT NULL,\r\n      Title NVARCHAR(30),\r\n      Rep
ortsTo INTEGER,\r\n      BirthDate DATETIME,\r\n      HireDate DATETIME,\r\n      Address NVARCHAR(70),\r\n      Ci
ty NVARCHAR(40),\r\n      State NVARCHAR(40),\r\n      Country NVARCHAR(40),\r\n      PostalCode NVARCHAR(10),\r
\n      Phone NVARCHAR(24),\r\n      Fax NVARCHAR(24),\r\n      Email NVARCHAR(60),\r\n      FOREIGN KEY (ReportsT
o) REFERENCES \"employees\" (EmployeeId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TAB
LE \"playlist_track\"\r\n(\r\n      PlaylistId INTEGER NOT NULL,\r\n      TrackId INTEGER NOT NULL,\r\n      CO
NSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\r\n      FOREIGN KEY (PlaylistId) REFERENCES
\"playlists\" (PlaylistId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n      FOREIGN KEY (TrackId) RE
FERENCES \"tracks\" (TrackId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"albums
\"\r\n(\r\n      AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n      Title NVARCHAR(160) NOT NULL,\r
\n      ArtistId INTEGER NOT NULL,\r\n      FOREIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistId) \r\n\t\t\tO
N DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON \"customers\" (Supp
ortRepId)\n\nCREATE TABLE \"playlists\"\r\n(\r\n      PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NUL
L,\r\n      Name NVARCHAR(120)\r\n)\n\nCREATE TABLE \"tracks\"\r\n(\r\n      TrackId INTEGER PRIMARY KEY AUTOIN
CREMENT NOT NULL,\r\n      Name NVARCHAR(200) NOT NULL,\r\n      AlbumId INTEGER,\r\n      MediaTypeId INTEGER
NOT NULL,\r\n      GenreId INTEGER,\r\n      Composer NVARCHAR(220),\r\n      Milliseconds INTEGER NOT NULL,\r\n
Bytes INTEGER,\r\n      UnitPrice NUMERIC(10,2) NOT NULL,\r\n      FOREIGN KEY (AlbumId) REFERENCES \"albums\"
(AlbumId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n      FOREIGN KEY (GenreId) REFERENCES \"genres
\" (GenreId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n      FOREIGN KEY (MediaTypeId) REFERENCES
\"media_types\" (MediaTypeId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_Invo
iceCustomerId ON \"invoices\" (CustomerId)\n\n\n===Additional Context\n\nIn the SQLite database invoice me
ans order\n\n===Response Guidelines\n1. If the provided context is sufficient, please generate a valid SQL
query without any explanations for the question.\n2. If the provided context is almost sufficient but requ
ires knowledge of a specific string in a particular column, please generate an intermediate SQL query to fi
nd the distinct strings in that column. Prepend the query with a comment saying intermediate_sql\n3. If th
e provided context is insufficient, please explain why it can't be generated.\n4. Please use the most rele
vant table(s).\n5. If the question has been asked and answered before, please repeat the answer exactly as

```

```

it was given before. \n"}, {"role": "user", "content": " \n      Find the customer who bought the most albums in total quantity (across all invoices): \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1"}, {"role": "user", "content": " \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices): \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": " \n      Find the customer with the most invoices \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1"}, {"role": "user", "content": " \n      Find the top 5 customers who spent the most money overall, \n      \n      Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessary \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5"}, {"role": "user", "content": " \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices): \n"}, {"role": "assistant", "content": "SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": " \n      List all customers from Canada and their email addresses: \n"}, {"role": "assistant", "content": "SELECT c.Email, c.Country\nFROM \"customers\" c\nWHERE c.Country = 'Canada'"}, {"role": "user", "content": "what are the top 5 countries that customers come from?"}, {"role": "assistant", "content": "SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM \"customers\" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5"}, {"role": "user", "content": " \n      Get the total number of invoices for each customer \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}, {"role": "user", "content": "How many customers are there?"}, {"role": "assistant", "content": "SELECT COUNT(*) FROM \"customers\""}, {"role": "user", "content": " \n      Get the average invoice total for each customer: \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}, {"role": "user", "content": " \n      List all customers who have not placed any orders: \n"}]

```

Ollama Response:

```

{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:49:45.187326399Z', 'message': {'role': 'assistant', 'content': 'SELECT c.CustomerId, c.FirstName, c.LastName\nFROM "customers" c\nLEFT JOIN "invoices" i ON c.CustomerId = i.CustomerId\nWHERE i.InvoiceId IS NULL;', 'done_reason': 'stop', 'done': True, 'total_duration': 92006776545, 'load_duration': 1103996, 'prompt_eval_count': 1980, 'prompt_eval_duration': 80713712000, 'eval_count': 43, 'eval_duration': 10662746000}
SELECT c.CustomerId, c.FirstName, c.LastName
FROM "customers" c
LEFT JOIN "invoices" i ON c.CustomerId = i.CustomerId
WHERE i.InvoiceId IS NULL;

```



```
Output from LLM: SELECT c.CustomerId, c.FirstName, c.LastName
FROM "customers" c
LEFT JOIN "invoices" i ON c.CustomerId = i.CustomerId
WHERE i.InvoiceId IS NULL;
```

```
Extracted SQL: SELECT c.CustomerId, c.FirstName, c.LastName
FROM "customers" c
LEFT JOIN "invoices" i ON c.CustomerId = i.CustomerId
WHERE i.InvoiceId IS NULL
SELECT c.CustomerId, c.FirstName, c.LastName
FROM "customers" c
LEFT JOIN "invoices" i ON c.CustomerId = i.CustomerId
WHERE i.InvoiceId IS NULL
```

Empty DataFrame

Columns: [CustomerId, FirstName, LastName]

Index: []

Ollama parameters:

model=codegemma:latest,

options={},

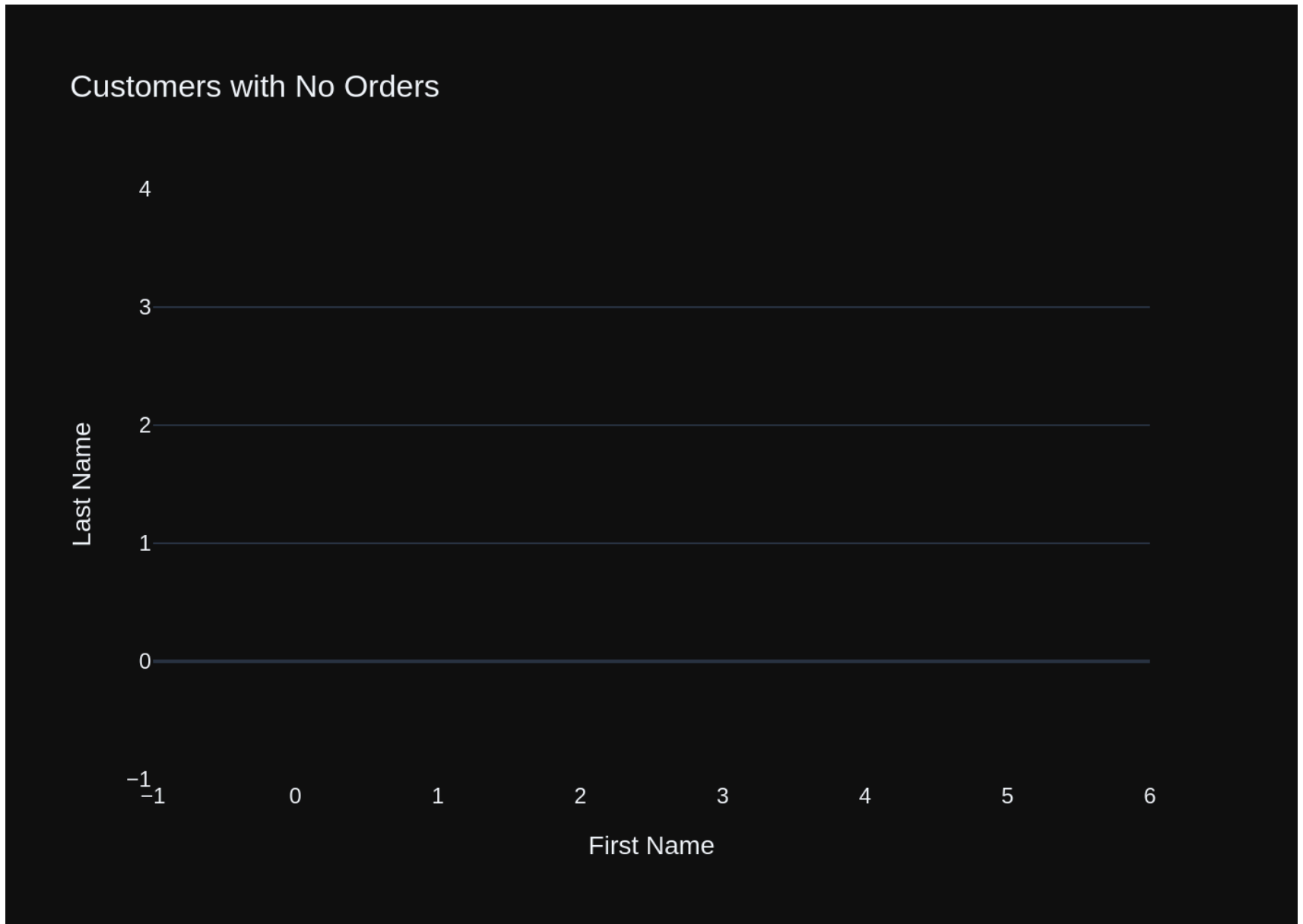
keep_alive=None

Prompt Content:

```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n List all customers who have not placed any orders:\n'\n\nThe DataFrame was produced using this query: SELECT c.CustomerId, c.FirstName, c.LastName\nFROM \"customers\" c\nLEFT JOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nWHERE i.InvoiceId IS NULL\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\nCustomerId    object\nFirstName      object\nLastName       object\ndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:50:08.734276407Z', 'message': {'role': 'assistant', 'content': "```python\nimport plotly.express as px\n\nfig = px.bar(df, x='FirstName', y='LastName', title='Customers with No Orders')\n\nfig.update_layout(xaxis_title='First Name', yaxis_title='Last Name')\n\nfig.show()\n```", 'done_reason': 'stop', 'done': True, 'total_duration': 23544587659, 'load_duration': 920830, 'prompt_eval_count': 188, 'prompt_eval_duration': 7656835000, 'eval_count': 63, 'eval_duration': 15794551000}
```



```

Out[32]: ('SELECT c.CustomerId, c.FirstName, c.LastName\nFROM "customers" c\nLEFT JOIN "invoices" i ON c.CustomerId
= i.CustomerId\nWHERE i.InvoiceId IS NULL',
Empty DataFrame
Columns: [CustomerId, FirstName, LastName]
Index: [],
Figure({
  'data': [{ 'alignmentgroup': 'True',
    'hovertemplate': 'FirstName=%{x}<br>LastName=%{y}<extra></extra>',
    'legendgroup': '',
    'marker': { 'color': '#636efa', 'pattern': { 'shape': '' } },
    'name': '',
    'offsetgroup': '',
    'orientation': 'v',
    'showlegend': False,
    'textposition': 'auto',
    'type': 'bar',
    'x': array([], dtype=object),
    'xaxis': 'x',
    'y': array([], dtype=object),
    'yaxis': 'y' } ],
  'layout': { 'barmode': 'relative',
    'legend': { 'tracegroupgap': 0 },
    'template': '...',
    'title': { 'text': 'Customers with No Orders' },
    'xaxis': { 'anchor': 'y', 'domain': [0.0, 1.0], 'title': { 'text': 'First Name' } },
    'yaxis': { 'anchor': 'x', 'domain': [0.0, 1.0], 'title': { 'text': 'Last Name' } } }
}))

```

```

In [33]: question = """
    There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, album:
    Can you find the top 10 most popular artists based on the number of tracks
    """

vn.ask(question=question)

```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

```
[{'role': 'system', 'content': 'You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables\nCREATE TABLE "tracks"\n(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES "albums" (AlbumId)\n)\nCREATE TABLE "albums"\n(\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Title NVARCHAR(160) NOT NULL,\n    ArtistId INTEGER NOT NULL,\n    FOREIGN KEY (ArtistId) REFERENCES "artists" (ArtistId)\n)\nCREATE TABLE "artists"\n(\n    ArtistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\nCREATE INDEX IFK_AlbumArtistId ON "albums" (ArtistId)\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\nCREATE TABLE "playlists"\n(\n    PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\nCREATE TABLE "genres"\n(\n    GenreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\nCREATE TABLE "playlist_track"\n(\n    PlaylistId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId)\n)\nCREATE INDEX IFK_TrackGenreId ON "tracks" (GenreId)\nCREATE INDEX IFK_PlaylistTrackTrackId ON "playlist_track" (TrackId)\n===Additional Context\nIn the SQLite database invoice means order\n===Response Guidelines\n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question.\n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql\n3. If the provided context is insufficient, please explain why it can't be generated.\n4. Please use the most relevant table(s).\n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before.\n'}], {'role': 'user', 'content': '\nThere are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\nCan you find the top 10 most popular artists based on the number of tracks\n'}, {'role': 'assistant', 'content': 'SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nJOIN "tracks" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10'}, {'role': 'user', 'content': '\nIdentify artists who have albums with tracks appearing in multiple genres:\n\n'}, {'role': 'assistant', 'content': 'SELECT a.ArtistId, a.Name AS ArtistName\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\nSELECT g2.GenreId\nFROM "genres" g2\nJOIN "tracks" t2 ON g2.GenreId = t2.GenreId\nGROUP BY g2.GenreId\nHAVING COUNT(g2.GenreId) > 1)\nGROUP BY a.ArtistId, a.Name'}, {'role': 'user', 'content': '\nHint: album quantity is found in invoice_items,\nFind the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': '\nList all albums and their corresponding artist names\n'}, {'role': 'assistant', 'content': 'SELECT al.Title, a.Name\nFROM "albums" al\nJOIN "artists" a ON al.ArtistId = a.ArtistId\nORDER BY al.Title\nLIMIT 10'}]
```

```
e': 'assistant', 'content': 'SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM "albums" a\nJOIN "artists" ar ON a.ArtistId = ar.ArtistId', {'role': 'user', 'content': ' \n Hint: album quantity is found in invoice_items, \n \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n List all genres and the number of tracks in each genre:\n'}, {'role': 'assistant', 'content': 'SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM "genres" g\nJOIN "tracks" t ON g.GenreId = t.GenreId\nGROUP BY g.Name'}, {'role': 'user', 'content': ' \n Find the top 5 most expensive tracks (based on unit price):\n'}, {'role': 'assistant', 'content': 'SELECT t.TrackId, t.Name, t.UnitPrice\nFROM "tracks" t\nORDER BY t.UnitPrice DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Find the customer who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n Find the top 5 customers who spent the most money overall, \n \n Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessary\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n Get all playlists containing at least 10 tracks and the total duration of those tracks:\n'}, {'role': 'assistant', 'content': 'SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM "playlist_track" pt\nJOIN "playlists" p ON pt.PlaylistId = p.PlaylistId\nJOIN "tracks" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10'}, {'role': 'user', 'content': ' \n There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\n Can you find the top 10 most popular artists based on the number of tracks\n'}]
```

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables\nCREATE TABLE \"tracks\"(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES \"albums\" (AlbumId)\nON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES \"genres\" (GenreId)\nON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES \"media_types\" (MediaTypeId)\nON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE \"albums\"(\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Title NVARCHAR(160) NOT NULL,\n    ArtistId INTEGER NOT NULL,\n    FOREIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistId)\nON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE \"artists\"(\n    ArtistId INTEGER PRIMAR
```

```

Y KEY AUTOINCREMENT NOT NULL,\r\n    Name NVARCHAR(120)\r\n)\n\nCREATE INDEX IFK_AlbumArtistId ON \"albums
\" (ArtistId)\n\nCREATE INDEX IFK_TrackAlbumId ON \"tracks\" (AlbumId)\n\nCREATE TABLE \"playlists\" \r\n(\r\n
    PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Name NVARCHAR(120)\r\n)\n\nCREATE TABLE
\"genres\" \r\n(\r\n    GenreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Name NVARCHAR(120)\r\n)\n
\nCREATE TABLE \"playlist_track\" \r\n(\r\n    PlaylistId INTEGER NOT NULL,\r\n    TrackId INTEGER NOT NUL
L,\r\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\r\n    FOREIGN KEY (PlaylistId) R
EFERENCES \"playlists\" (PlaylistId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n    FOREIGN KEY (T
rackId) REFERENCES \"tracks\" (TrackId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDE
X IFK_TrackGenreId ON \"tracks\" (GenreId)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON \"playlist_track\"
(TrackId)\n\n\n===Additional Context \n\nIn the SQLite database invoice means order\n\n===Response Guidelin
es \n1. If the provided context is sufficient, please generate a valid SQL query without any explanations f
or the question. \n2. If the provided context is almost sufficient but requires knowledge of a specific str
ing in a particular column, please generate an intermediate SQL query to find the distinct strings in that
column. Prepend the query with a comment saying intermediate_sql \n3. If the provided context is insufficie
nt, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the quest
ion has been asked and answered before, please repeat the answer exactly as it was given before. \n\"}, {\"ro
le\": \"user\", \"content\": \" \n    There are 3 tables: artists, albums and tracks, where albums and artists ar
e linked by ArtistId, albums and tracks are linked by AlbumId,\n    Can you find the top 10 most popular ar
tists based on the number of tracks\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT a.Name, COUNT(t.TrackId)
AS TotalTracks\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nJOIN \"tracks\" t ON a
l.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10\"}, {\"role\": \"user\", \"content\":
\" \n    Identify artists who have albums with tracks appearing in multiple genres:\n\n\"}, {\"role\": \"ass
istant\", \"content\": \"SELECT a.ArtistId, a.Name AS ArtistName\nFROM \"artists\" a\nJOIN \"albums\" al ON a.A
rtistId = al.ArtistId\nWHERE a.ArtistId IN (\n    SELECT g2.GenreId\n    FROM \"genres\" g2\n    JOIN \"tracks\"
t2 ON g2.GenreId = t2.GenreId\n    GROUP BY g2.GenreId\n    HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistI
d, a.Name\"}, {\"role\": \"user\", \"content\": \" \n    Hint: album quantity is found in invoice_items, \n    \n
Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n\"}, {\"role\":
\"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN
\"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGRO
UP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n    List all alb
ums and their corresponding artist names \n\"}, {\"role\": \"assistant\", \"content\": \"SELECT a.Title, a.ArtistI
d, ar.Name AS ArtistName\nFROM \"albums\" a\nJOIN \"artists\" ar ON a.ArtistId = ar.ArtistId\"}, {\"role\": \"u
ser\", \"content\": \" \n    Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 custom
ers who bought the most albums in total quantity (across all invoices):\n\"}, {\"role\": \"assistant\", \"conten
t\": \"SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items
\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5\"}, {\"role\":
\"user\", \"content\": \" \n    List all genres and the number of tracks in each genre:\n\"}, {\"role\": \"assistan
t\", \"content\": \"SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM \"genres\" g\nJOIN \"tracks\" t ON g.G
enreId = t.GenreId\nGROUP BY g.Name\"}, {\"role\": \"user\", \"content\": \" \n    Find the top 5 most expensive t
racks (based on unit price):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT t.TrackId, t.Name, t.UnitPrice\n
FROM \"tracks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n    Find the cus
tomer who bought the most albums in total quantity (across all invoices): \n\"}, {\"role\": \"assistant\", \"cont

```

```
ent": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1\"}, {\"role\": \"user\", \"content\": \" \n      Find the top 5 customers who spent the most money overall, \n      \n      Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessary \n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n      Get all playlists containing at least 10 tracks and the total duration of those tracks:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM \"playlist_track\" pt\nJOIN \"playlists\" p ON pt.PlaylistId = p.PlaylistId\nJOIN \"tracks\" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10\"}, {\"role\": \"user\", \"content\": \" \n      There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\n      Can you find the top 10 most popular artists based on the number of tracks\n\"}]
```

Add of existing embedding ID: 127fd4bd-b9af-539d-9313-1d0234d073b7-sql

Insert of existing embedding ID: 127fd4bd-b9af-539d-9313-1d0234d073b7-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:51:46.333396555Z', 'message': {'role': 'assistant', 'content': 'SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nJOIN "tracks" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10'}, 'done_reason': 'stop', 'done': True, 'total_duration': 97504584705, 'load_duration': 678215, 'prompt_eval_count': 1785, 'prompt_eval_duration': 78310493000, 'eval_count': 71, 'eval_duration': 18560680000}
```

```
SELECT a.Name, COUNT(t.TrackId) AS TotalTracks
FROM "artists" a
JOIN "albums" al ON a.ArtistId = al.ArtistId
JOIN "tracks" t ON al.AlbumId = t.AlbumId
GROUP BY a.Name
ORDER BY TotalTracks DESC
LIMIT 10
SELECT a.Name, COUNT(t.TrackId) AS TotalTracks
FROM "artists" a
JOIN "albums" al ON a.ArtistId = al.ArtistId
JOIN "tracks" t ON al.AlbumId = t.AlbumId
GROUP BY a.Name
ORDER BY TotalTracks DESC
LIMIT 10
```

	Name	TotalTracks
0	Iron Maiden	213
1	U2	135
2	Led Zeppelin	114
3	Metallica	112
4	Lost	92
5	Deep Purple	92
6	Pearl Jam	67
7	Lenny Kravitz	57
8	Various Artists	56
9	The Office	53

Ollama parameters:

```
model=codegemma:latest,
options={},
keep_alive=None
```

Prompt Content:

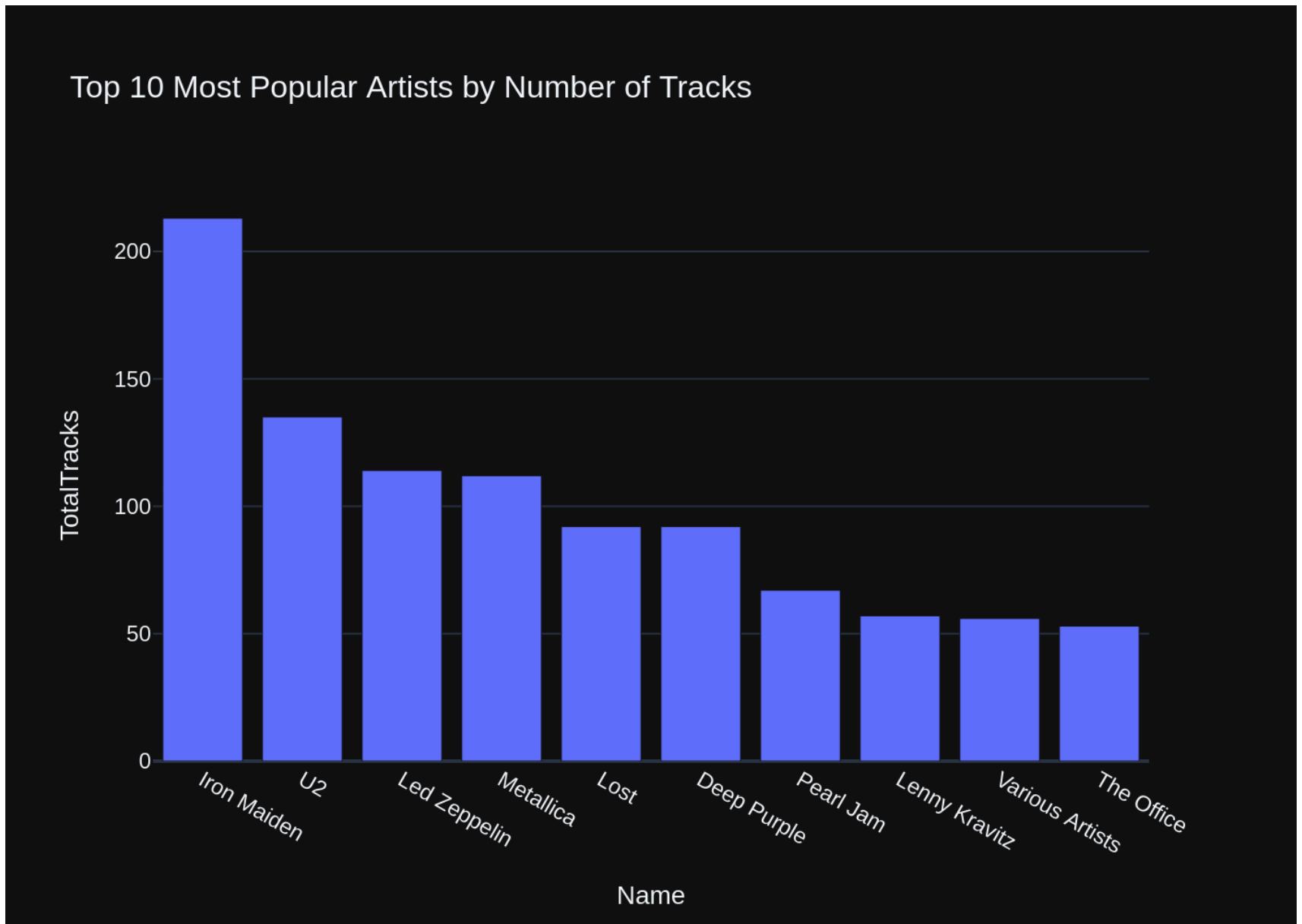
```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\n Can you find the top 10 most popular artists based on the number of tracks\n\n\nThe DataFrame was produced using this query: SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nJOIN \"tracks\" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10"}]
```



```
istId\nJOIN \"tracks\" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n Name\nobject\nTotalTracks      int64\nndtype: object\"}, {\"role\": \"user\", \"content\": \"Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code.\"}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:52:09.922011487Z', 'message': {'role': 'assistant', 'content': \"```python\nimport plotly.express as px\n\nfig = px.bar(df, x='Name', y='TotalTracks', title='Top 10 Most Popular Artists by Number of Tracks')\n\nfig.show()\n```\"}, 'done_reason': 'stop', 'done': True, 'total_duration': 23559313956, 'load_duration': 703828, 'prompt_eval_count': 255, 'prompt_eval_duration': 10827833000, 'eval_count': 49, 'eval_duration': 12643232000}
```



```
Out[33]: ('SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.Ar  
tistId\nJOIN "tracks" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10',
```

```
      Name  TotalTracks
0      Iron Maiden      213
1              U2      135
2      Led Zeppelin      114
3      Metallica      112
4              Lost      92
5      Deep Purple      92
6      Pearl Jam      67
7      Lenny Kravitz      57
8  Various Artists      56
9      The Office      53,
```

```
Figure({
  'data': [{'alignmentgroup': 'True',
            'hovertemplate': 'Name=%{x}<br>TotalTracks=%{y}<extra></extra>',
            'legendgroup': '',
            'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
            'name': '',
            'offsetgroup': '',
            'orientation': 'v',
            'showlegend': False,
            'textposition': 'auto',
            'type': 'bar',
            'x': array(['Iron Maiden', 'U2', 'Led Zeppelin', 'Metallica', 'Lost', 'Deep Purple',
                        'Pearl Jam', 'Lenny Kravitz', 'Various Artists', 'The Office'],
                       dtype=object),
            'xaxis': 'x',
            'y': array([213, 135, 114, 112, 92, 92, 67, 57, 56, 53]),
            'yaxis': 'y'}],
  'layout': {'barmode': 'relative',
             'legend': {'tracegroupgap': 0},
             'template': '...',
             'title': {'text': 'Top 10 Most Popular Artists by Number of Tracks'},
             'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Name'}},
             'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'TotalTracks'}}})
```

```
In [34]: question = """
          List all customers from Canada and their email addresses:
          """
```

```
vn.ask(question=question)
```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

149/199

```

customers" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5'}}, {'role': 'user', 'content': ' \n
Find the customer with the most invoices \n'}}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT
(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP
BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1'}}, {'role': 'user', 'content': ' \n    Get the total
number of invoices for each customer\n'}}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.In
voiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY
c.CustomerId'}}, {'role': 'user', 'content': ' \n    Find the customer who bought the most albums in total
quantity (across all invoices): \n'}}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.Track
Id) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_ite
ms" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}}, {'role':
'user', 'content': 'How many customers are there'}}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM
"customers"'}, {'role': 'user', 'content': ' \n    Find the top 5 customers who spent the most money over
all, \n    \n    Hint: order total can be found on invoices table, calculation using invoice_items detail
table is unnecessary \n'}}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY To
talSpent DESC\nLIMIT 5'}}, {'role': 'user', 'content': ' \n    Find the total number of invoices per countr
y:\n'}}, {'role': 'assistant', 'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "invoic
es" i\nGROUP BY i.BillingCountry'}}, {'role': 'user', 'content': ' \n    Hint: album quantity is found in in
voice_items, \n    \n    Find the top 5 customers who bought the most albums in total quantity (across all
invoices):\n'}}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFR
OM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId
= ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}}, {'role': 'user', 'content': '
\n    Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 customers who bought the m
ost albums in total quantity (across all invoices):\n'}}, {'role': 'assistant', 'content': 'SELECT i.Custome
rId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = i
i.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}}, {'role': 'user', 'content': ' \n
List all customers from Canada and their email addresses:\n'}]}

```

Ollama parameters:

```
model=codegemma:latest,
```

```
options={},
```

```
keep alive=None
```

Prompt Content:

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables \nCREATE INDEX IFK_CustomerSupportRepId ON \"customers\" (SupportRepId)\n\nCREATE TABLE \"customers\"(\n    CustomerId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    FirstName NVARCHAR(40) NOT NULL,\n    LastName NVARCHAR(20) NOT NULL,\n    Company NVARCHAR(80),\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60) NOT NULL,\n    SupportRepId INTEGER,\n    FOREIGN KEY (SupportRepId) REFERENCES \"employees\" (EmployeeId)\n    ON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE \"invoices\"(\n    InvoiceId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    CustomerId INTEGER NOT NULL,\n    InvoiceDate DATETIME
```

```

NOT NULL,\r\n    BillingAddress NVARCHAR(70),\r\n    BillingCity NVARCHAR(40),\r\n    BillingState NVARCHAR
(40),\r\n    BillingCountry NVARCHAR(40),\r\n    BillingPostalCode NVARCHAR(10),\r\n    Total NUMERIC(10,2)
NOT NULL,\r\n    FOREIGN KEY (CustomerId) REFERENCES \"customers\" (CustomerId) \r\n\t\t\tON DELETE NO ACTION
ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON \"invoices\" (CustomerId)\n\nCREATE TABLE
\"employees\" \r\n(\r\n    EmployeeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    LastName NVARCHAR(2
0) NOT NULL,\r\n    FirstName NVARCHAR(20) NOT NULL,\r\n    Title NVARCHAR(30),\r\n    ReportsTo INTEGE
R,\r\n    BirthDate DATETIME,\r\n    HireDate DATETIME,\r\n    Address NVARCHAR(70),\r\n    City NVARCHAR(4
0),\r\n    State NVARCHAR(40),\r\n    Country NVARCHAR(40),\r\n    PostalCode NVARCHAR(10),\r\n    Phone NV
ARCHAR(24),\r\n    Fax NVARCHAR(24),\r\n    Email NVARCHAR(60),\r\n    FOREIGN KEY (ReportsTo) REFERENCES
\"employees\" (EmployeeId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"invoice_i
tems\" \r\n(\r\n    InvoiceLineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    InvoiceId INTEGER NOT
NULL,\r\n    TrackId INTEGER NOT NULL,\r\n    UnitPrice NUMERIC(10,2) NOT NULL,\r\n    Quantity INTEGER
NOT NULL,\r\n    FOREIGN KEY (InvoiceId) REFERENCES \"invoices\" (InvoiceId) \r\n\t\t\tON DELETE NO ACTION ON
UPDATE NO ACTION,\r\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\t\tON DELETE NO ACTION
ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE sqlite_sequence(name,seq)\n\nCREATE TABLE \"playlist_track\" \r\n
(\r\n    PlaylistId INTEGER NOT NULL,\r\n    TrackId INTEGER NOT NULL,\r\n    CONSTRAINT PK_PlaylistTrack
PRIMARY KEY (PlaylistId, TrackId),\r\n    FOREIGN KEY (PlaylistId) REFERENCES \"playlists\" (PlaylistId)
\r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackI
d) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_EmployeeReportsTo ON \"employee
s\" (ReportsTo)\n\nCREATE TABLE \"albums\" \r\n(\r\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NUL
L,\r\n    Title NVARCHAR(160) NOT NULL,\r\n    ArtistId INTEGER NOT NULL,\r\n    FOREIGN KEY (ArtistId) R
EFERENCES \"artists\" (ArtistId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\n\n===Additional Co
ntext \n\nIn the SQLite database invoice means order\n\n===Response Guidelines \n1. If the provided context
is sufficient, please generate a valid SQL query without any explanations for the question. \n2. If the pro
vided context is almost sufficient but requires knowledge of a specific string in a particular column, plea
se generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a
comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it can't b
e generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered b
efore, please repeat the answer exactly as it was given before. \n\"}, {\"role\": \"user\", \"content\": \" \n
List all customers from Canada and their email addresses:\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.E
mail, c.Country\nFROM \"customers\" c\nWHERE c.Country = 'Canada'\"}, {\"role\": \"user\", \"content\": \"what are
the top 5 countries that customers come from?\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.Country, COUNT
(*) AS TotalCustomers\nFROM \"customers\" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5\"},
{\"role\": \"user\", \"content\": \" \n    Find the customer with the most invoices \n\"}, {\"role\": \"assistant\",
\"content\": \"SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices
\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1\"}, {\"rol
e\": \"user\", \"content\": \" \n    Get the total number of invoices for each customer\n\"}, {\"role\": \"assistan
t\", \"content\": \"SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invo
ices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\"}, {\"role\": \"user\", \"content\": \" \n    Fin
d the customer who bought the most albums in total quantity (across all invoices): \n\"}, {\"role\": \"assistan
t\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoice
s\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.C

```

```

ustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1"}], {"role": "user", "content": "How many customers are ther
e"}, {"role": "assistant", "content": "SELECT COUNT(*) FROM \"customers\""}, {"role": "user", "content": "
\n      Find the top 5 customers who spent the most money overall, \n      \n      Hint: order total can be fo
und on invoices table, calculation using invoice_items detail table is unnecessary \n"}, {"role": "assistan
t", "content": "SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\" i
ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5"}, {"role": "use
r", "content": " \n      Find the total number of invoices per country:\n"}, {"role": "assistant", "conten
t": "SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM \"invoices\" i\nGROUP BY i.BillingCountry"},
{"role": "user", "content": " \n      Hint: album quantity is found in invoice_items, \n      \n      Find the t
op 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistan
t", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoice
s\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.C
ustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": " \n      Hint: album quantity
is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantit
y (across all invoices):\n"}, {"role": "assistant", "content": "SELECT i.CustomerId, COUNT(ii.InvoiceLineI
d) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY
i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": " \n      List all customer
s from Canada and their email addresses:\n"}]]

```

Add of existing embedding ID: 584873f8-1904-50f1-8f80-7ccf08059264-sql

Insert of existing embedding ID: 584873f8-1904-50f1-8f80-7ccf08059264-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:53:33.792084896Z', 'message': {'role': 'assistant', 'content': 'SELECT c.Email, c.Country\nFROM "customers" c\nWHERE c.Country = \'Canada\''}, 'done_reason': 'stop', 'done': True, 'total_duration': 83738425537, 'load_duration': 723557, 'prompt_eval_count': 1778, 'prompt_eval_duration': 76853445000, 'eval_count': 24, 'eval_duration': 6207894000}
```

```
SELECT c.Email, c.Country
FROM "customers" c
WHERE c.Country = 'Canada'
SELECT c.Email, c.Country
FROM "customers" c
WHERE c.Country = 'Canada'
```

	Email	Country
0	ftremblay@gmail.com	Canada
1	mphilips12@shaw.ca	Canada
2	jenniferp@rogers.ca	Canada
3	robbrown@shaw.ca	Canada
4	edfrancis@yachoo.ca	Canada
5	marthasilk@gmail.com	Canada
6	aaronmitchell@yahoo.ca	Canada
7	ellie.sullivan@shaw.ca	Canada

Ollama parameters:

```
model=codegemma:latest,
options={},
keep_alive=None
```

Prompt Content:

```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n      List all customers from Canada and their email addresses:\n'\n\nThe DataFrame was produced using this query: SELECT c.Email, c.Country\nFROM \"customers\" c\nWHERE c.Country = 'Canada'\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\nEmail      object\nCountry    object\ndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:53:52.826971592Z', 'message': {'role': 'assistant', 'content': "\n\npython\nimport plotly.express as px\n\nfig = px.bar(df, x='Email', y='Country', color='Country', title='Canadian Customer Emails')\n\nfig.show()\n\n", 'done_reason': 'stop', 'done': True, 'total_duration': 19008008243, 'load_duration': 759889, 'prompt_eval_count': 165, 'prompt_eval_duration': 7294272000, 'eval_count': 44, 'eval_duration': 11582078000}
```



```

Out[34]: ('SELECT c.Email, c.Country\nFROM "customers" c\nWHERE c.Country = \'Canada\'',
          Email Country
0      ftremblay@gmail.com Canada
1      mphilips12@shaw.ca Canada
2      jenniferp@rogers.ca Canada
3      robbrown@shaw.ca Canada
4      edfrancis@yachoo.ca Canada
5      marthasilk@gmail.com Canada
6      aaronmitchell@yahoo.ca Canada
7      ellie.sullivan@shaw.ca Canada,
Figure({
  'data': [{'alignmentgroup': 'True',
            'hovertemplate': 'Country=%{y}<br>Email=%{x}<extra></extra>',
            'legendgroup': 'Canada',
            'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
            'name': 'Canada',
            'offsetgroup': 'Canada',
            'orientation': 'v',
            'showlegend': True,
            'textposition': 'auto',
            'type': 'bar',
            'x': array(['ftremblay@gmail.com', 'mphilips12@shaw.ca', 'jenniferp@rogers.ca',
                       'robbrown@shaw.ca', 'edfrancis@yachoo.ca', 'marthasilk@gmail.com',
                       'aaronmitchell@yahoo.ca', 'ellie.sullivan@shaw.ca'], dtype=object),
            'xaxis': 'x',
            'y': array(['Canada', 'Canada', 'Canada', 'Canada', 'Canada', 'Canada', 'Canada', 'Canada',
                       'Canada'], dtype=object),
            'yaxis': 'y'}],
  'layout': {'barmode': 'relative',
             'legend': {'title': {'text': 'Country'}, 'tracegroupgap': 0},
             'template': '...',
             'title': {'text': 'Canadian Customer Emails'},
             'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Email'}},
             'yaxis': {'anchor': 'x',
                       'categoryarray': [Canada],
                       'categoryorder': 'array',
                       'domain': [0.0, 1.0],
                       'title': {'text': 'Country'}}}
}))

```

```

In [35]: question = """
          Find the customer with the most invoices

```

```
"""
```

```
vn.ask(question=question)
```

```
Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1
```

```
tomers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices
```

```
DESC\LIMIT 1'}, {'role': 'user', 'content': ' \n      Find the top 5 customers who spent the most money over all, \n      \n      Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessary \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Find the customer who bought the most albums in total quantity (across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Get the total number of invoices for each customer\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': ' \n      Get the average invoice total for each customer:\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': ' \n      List all invoices with a total exceeding $10:\n'}, {'role': 'assistant', 'content': 'SELECT * \nFROM "invoices" \nWHERE Total > 10.00'}, {'role': 'user', 'content': ' \n      Find the total number of invoices per country:\n'}, {'role': 'assistant', 'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "invoices" i\nGROUP BY i.BillingCountry'}, {'role': 'user', 'content': ' \n      Find all invoices since 2010 and the total amount invoiced:\n'}, {'role': 'assistant', 'content': 'SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\nFROM "invoices" i\nWHERE i.InvoiceDate >= \'2010-01-01\'\nGROUP BY i.InvoiceDate'}, {'role': 'user', 'content': ' \n      Find the customer with the most invoices \n'}]
```

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables\nCREATE TABLE \"invoices\"\n(\n  InvoiceId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n  CustomerId INTEGER NOT NULL,\n  InvoiceDate DATETIME NOT NULL,\n  BillingAddress NVARCHAR(70),\n  BillingCity NVARCHAR(40),\n  BillingState NVARCHAR(40),\n  BillingCountry NVARCHAR(40),\n  BillingPostalCode NVARCHAR(10),\n  Total NUMERIC(10,2) NOT NULL,\n  FOREIGN KEY (CustomerId) REFERENCES \"customers\" (CustomerId)\n)\nON DELETE NO ACTION ON UPDATE NO ACTION\n\nCREATE INDEX IFK_InvoiceCustomerId ON \"invoices\" (CustomerId)\n\nCREATE INDEX IFK_InvoiceLineInv
```

dia_types\" (MediaTypeId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\n===Additional Context
 \n\nIn the SQLite database invoice means order\n\n===Response Guidelines \n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question. \n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before. \n"}, {"role": "user", "content": " \n Find the customer with the most invoices \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1"}, {"role": "user", "content": " \n Find the top 5 customers who spent the most money overall, \n \n Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessary \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5"}, {"role": "user", "content": " \n Find the customer who bought the most albums in total quantity (across all invoices): \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY

```

c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1"}], {"role": "user", "content": " \n    Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}], {"role": "user", "content": " \n    Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}], {"role": "user", "content": " \n    Get the total number of invoices for each customer\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}, {"role": "user", "content": " \n    Get the average invoice total for each customer:\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}, {"role": "user", "content": " \n    List all invoices with a total exceeding $10:\n"}, {"role": "assistant", "content": "SELECT * \nFROM \"invoices\" \nWHERE Total > 10.00"}, {"role": "user", "content": " \n    Find the total number of invoices per country:\n"}, {"role": "assistant", "content": "SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM \"invoices\" i\nGROUP BY i.BillingCountry"}, {"role": "user", "content": " \n    Find all invoices since 2010 and the total amount invoiced:\n"}, {"role": "assistant", "content": "SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\nFROM \"invoices\" i\nWHERE i.InvoiceDate >= '2010-01-01'\nGROUP BY i.InvoiceDate"}, {"role": "user", "content": " \n    Find the customer with the most invoices \n"}]

```

Add of existing embedding ID: 3013d1b4-feb2-519d-bfb9-114500436e3d-sql

Insert of existing embedding ID: 3013d1b4-feb2-519d-bfb9-114500436e3d-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:55:29.88488508Z', 'message': {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1'}, 'done_reason': 'stop', 'done': True, 'total_duration': 96948971632, 'load_duration': 748683, 'prompt_eval_count': 1871, 'prompt_eval_duration': 82554741000, 'eval_count': 52, 'eval_duration': 13731415000}
SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices
FROM "customers" c
JOIN "invoices" i ON c.CustomerId = i.CustomerId
GROUP BY c.CustomerId
ORDER BY TotalInvoices DESC
LIMIT 1
```

```
CustomerId TotalInvoices
0          1          7
```

Ollama parameters:

```
model=codegemma:latest,
options={},
keep_alive=None
```

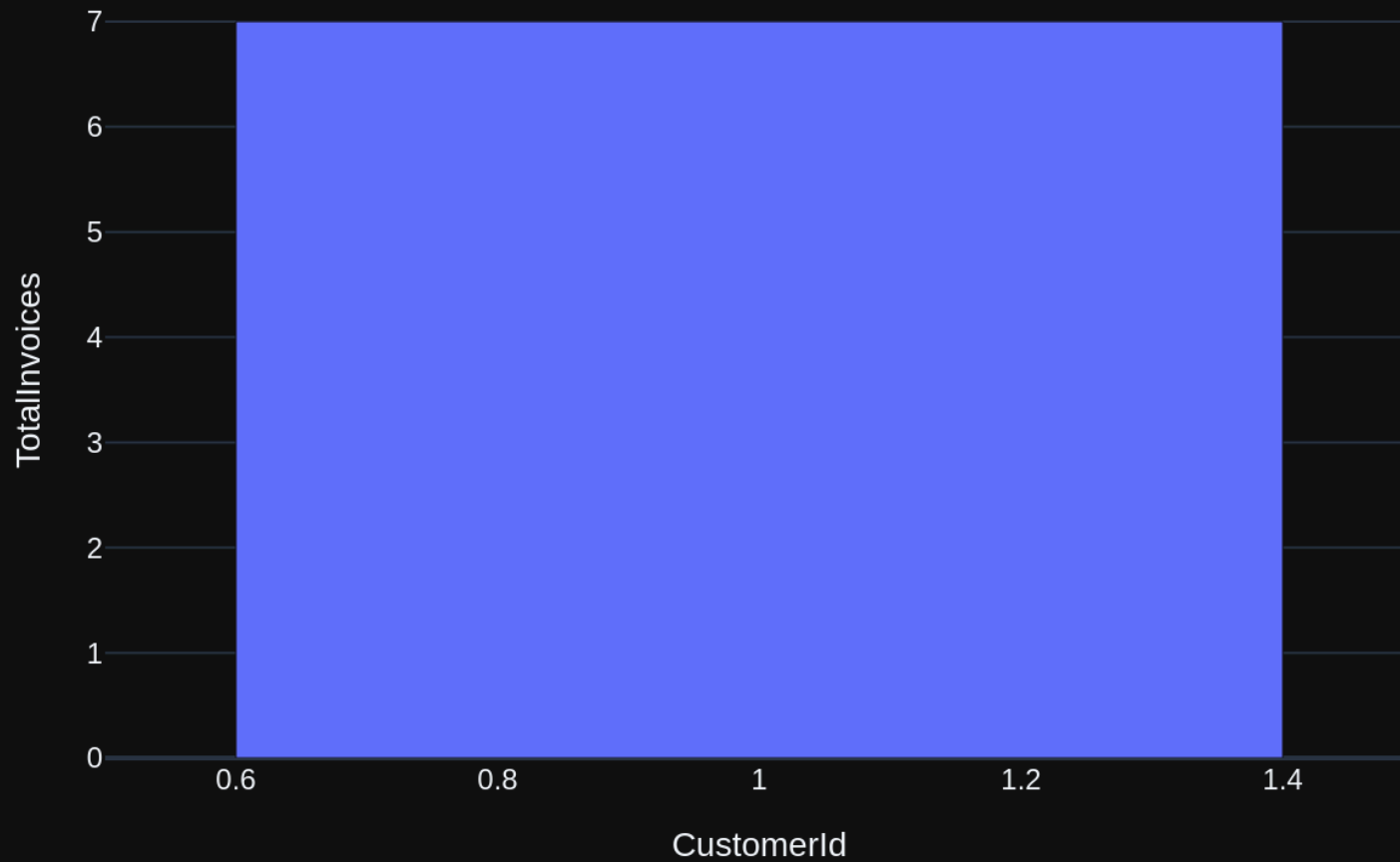
Prompt Content:

```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n      Find the customer with the most invoices \n\n\nThe DataFrame was produced using this query: SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\nCustomerId          int64\nTotalInvoices        int64\ndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:55:51.931418411Z', 'message': {'role': 'assistant', 'content': '\n\npython\nimport plotly.express as px\n\nfig = px.bar(\n    df,\n    x='CustomerId',\n    y='TotalInvoices',\n    title='Customer with Most Invoices',\n)\n\nfig.show()\n\n"}, 'done_reason': 'stop', 'done': True, 'total_duration': 22019543816, 'load_duration': 797435, 'prompt_eval_count': 197, 'prompt_eval_duration': 8399675000, 'eval_count': 53, 'eval_duration': 13489103000}
```

Customer with Most Invoices



```

Out[35]: ('SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1',
         CustomerId  TotalInvoices
         0           1           7,
         Figure({
           'data': [{'alignmentgroup': 'True',
                     'hvertemplate': 'CustomerId=%{x}<br>TotalInvoices=%{y}<extra></extra>',
                     'legendgroup': '',
                     'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
                     'name': '',
                     'offsetgroup': '',
                     'orientation': 'v',
                     'showlegend': False,
                     'textposition': 'auto',
                     'type': 'bar',
                     'x': array([1]),
                     'xaxis': 'x',
                     'y': array([7]),
                     'yaxis': 'y'}],
           'layout': {'barmode': 'relative',
                      'legend': {'tracegroupgap': 0},
                      'template': '...',
                      'title': {'text': 'Customer with Most Invoices'},
                      'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'CustomerId'}},
                      'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'TotalInvoices'}}}
         )))

```

In []:

Advanced SQL questions

```

In [36]: question = """
         Find the customer who bought the most albums in total quantity (across all invoices):
         """
         vn.ask(question=question)

```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

```
[{'role': 'system', 'content': 'You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables\nCREATE TABLE "tracks"\n(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES "albums" (AlbumId)\n)\nCREATE TABLE "genres"\n(\n    GenreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(100) NOT NULL,\n    FOREIGN KEY (MediaTypeId) REFERENCES "media_types" (MediaTypeId)\n)\nCREATE TABLE "media_types"\n(\n    MediaTypeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(100) NOT NULL,\n    FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId)\n)\nCREATE TABLE "albums"\n(\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Title NVARCHAR(160) NOT NULL,\n    ArtistId INTEGER NOT NULL,\n    FOREIGN KEY (ArtistId) REFERENCES "artists" (ArtistId)\n)\nCREATE INDEX IFK_AlbumArtistId ON "albums" (ArtistId)\nCREATE TABLE "invoices"\n(\n    InvoiceId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    CustomerId INTEGER NOT NULL,\n    InvoiceDate DATETIME NOT NULL,\n    BillingAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (CustomerId) REFERENCES "customers" (CustomerId)\n)\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\nCREATE INDEX IFK_InvoiceLineInvoiceId ON "invoice_items" (InvoiceId)\nCREATE INDEX IFK_InvoiceCustomerId ON "invoices" (CustomerId)\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\nCREATE TABLE "artists"\n(\n    ArtistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\n\n===Additional Context\nIn the SQLite database invoice means order\n\n===Response Guidelines\n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question.\n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying in_terminmediate_sql\n3. If the provided context is insufficient, please explain why it can't be generated.\n4. Please use the most relevant table(s).\n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before.\n\n}', {'role': 'user', 'content': 'Find the customer who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'role': 'user', 'content': 'Hint: album quantity is found in invoice_items,\n\nFind the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = i.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': 'Hint: album quantity is found in invoice_items,\n\nFind the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT i.CustomerId, C
```

Ollama parameters:

```
options={},
```

Prompt Content:

```

OT NULL,\r\n    TrackId INTEGER NOT NULL,\r\n    UnitPrice NUMERIC(10,2) NOT NULL,\r\n    Quantity INTEGE
R NOT NULL,\r\n    FOREIGN KEY (InvoiceId) REFERENCES \"invoices\" (InvoiceId) \r\n\t\t\tON DELETE NO ACTION
ON UPDATE NO ACTION,\r\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\t\tON DELETE NO ACTI
ON ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"albums\"(\r\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMEN
T NOT NULL,\r\n    Title NVARCHAR(160) NOT NULL,\r\n    ArtistId INTEGER NOT NULL,\r\n    FOREIGN KEY (Ar
tistId) REFERENCES \"artists\" (ArtistId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE IN
DEX IFK_AlbumArtistId ON \"albums\" (ArtistId)\n\nCREATE TABLE \"invoices\"(\r\n    InvoiceId INTEGER P
RIMARY KEY AUTOINCREMENT NOT NULL,\r\n    CustomerId INTEGER NOT NULL,\r\n    InvoiceDate DATETIME NOT NU
LL,\r\n    BillingAddress NVARCHAR(70),\r\n    BillingCity NVARCHAR(40),\r\n    BillingState NVARCHAR(4
0),\r\n    BillingCountry NVARCHAR(40),\r\n    BillingPostalCode NVARCHAR(10),\r\n    Total NUMERIC(10,2)
NOT NULL,\r\n    FOREIGN KEY (CustomerId) REFERENCES \"customers\" (CustomerId) \r\n\t\t\tON DELETE NO ACTION
ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON \"invoice_items\" (TrackId)\n\nCREATE IN
DEX IFK_InvoiceLineInvoiceId ON \"invoice_items\" (InvoiceId)\n\nCREATE INDEX IFK_InvoiceCustomerId ON \"in
voices\" (CustomerId)\n\nCREATE INDEX IFK_TrackAlbumId ON \"tracks\" (AlbumId)\n\nCREATE TABLE \"artists
\"(\r\n    ArtistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Name NVARCHAR(120)\r\n)\n\n\n===
Additional Context \n\nIn the SQLite database invoice means order\n\n===Response Guidelines \n1. If the pro
vided context is sufficient, please generate a valid SQL query without any explanations for the question.
\n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particul
ar column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend t
he query with a comment saying intermediate_sql \n3. If the provided context is insufficient, please explai
n why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been aske
d and answered before, please repeat the answer exactly as it was given before. \n\"}, {\"role\": \"user\", \"con
tent\": \" \n    Find the customer who bought the most albums in total quantity (across all invoices):
\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"custo
mers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = i
i.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1\"}, {\"role\": \"user\", \"content\": \" \n
Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 customers who bought the most al
bums in total quantity (across all invoices):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, C
OUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\n
JOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\n
LIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n    Hint: album quantity is found in invoice_items, \n    \n
Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n\"}, {\"role\":
\"assistant\", \"content\": \"SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\n
JOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\n
LIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n    Find the customer with the most invoices \n\"}, {\"role\": \"a
ssistant\", \"content\": \"SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN
\"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT
1\"}, {\"role\": \"user\", \"content\": \" \n    Find the top 5 customers who spent the most money overall, \n
\n    Hint: order total can be found on invoices table, calculation using invoice_items detail table is un
necessary \n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"c
ustomers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent
DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n    There are 3 tables: artists, albums and tracks, wher

```

e albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\n Can you find the top 10 most popular artists based on the number of tracks\n"}, {"role": "assistant", "content": "SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nJOIN \"tracks\" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10"}, {"role": "user", "content": " \n Find all invoices since 2010 and the total amount invoiced:\n"}, {"role": "assistant", "content": "SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\nFROM \"invoices\" i\nWHERE i.InvoiceDate >= '2010-01-01'\nGROUP BY i.InvoiceDate"}, {"role": "user", "content": " \n Get the total number of invoices for each customer\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}, {"role": "user", "content": " \n Find the top 5 most expensive tracks (based on unit price):\n"}, {"role": "assistant", "content": "SELECT t.TrackId, t.Name, t.UnitPrice\nFROM \"tracks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5"}, {"role": "user", "content": " \n Identify artists who have albums with tracks appearing in multiple genres:\n\n\n"}, {"role": "assistant", "content": "SELECT a.ArtistId, a.Name AS ArtistName\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n SELECT g2.GenreId\n FROM \"genres\" g2\n JOIN \"tracks\" t2 ON g2.GenreId = t2.GenreId\n GROUP BY g2.GenreId\n HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name"}, {"role": "user", "content": " \n Find the customer who bought the most albums in total quantity (across all invoices): \n"}]

Add of existing embedding ID: d1d70c18-f5d9-5970-a32c-914deecal087-sql

Insert of existing embedding ID: d1d70c18-f5d9-5970-a32c-914deecal087-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:57:29.060958875Z', 'message': {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, 'done_reason': 'stop', 'done': True, 'total_duration': 97038916904, 'load_duration': 832273, 'prompt_eval_count': 1820, 'prompt_eval_duration': 78017885000, 'eval_count': 70, 'eval_duration': 18332291000}
```

```
SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums
FROM "customers" c
JOIN "invoices" i ON c.CustomerId = i.CustomerId
JOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId
GROUP BY c.CustomerId
ORDER BY TotalAlbums DESC
LIMIT 1
```

```
SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums
FROM "customers" c
JOIN "invoices" i ON c.CustomerId = i.CustomerId
JOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId
GROUP BY c.CustomerId
ORDER BY TotalAlbums DESC
LIMIT 1
```

CustomerId	TotalAlbums
0	1 38

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

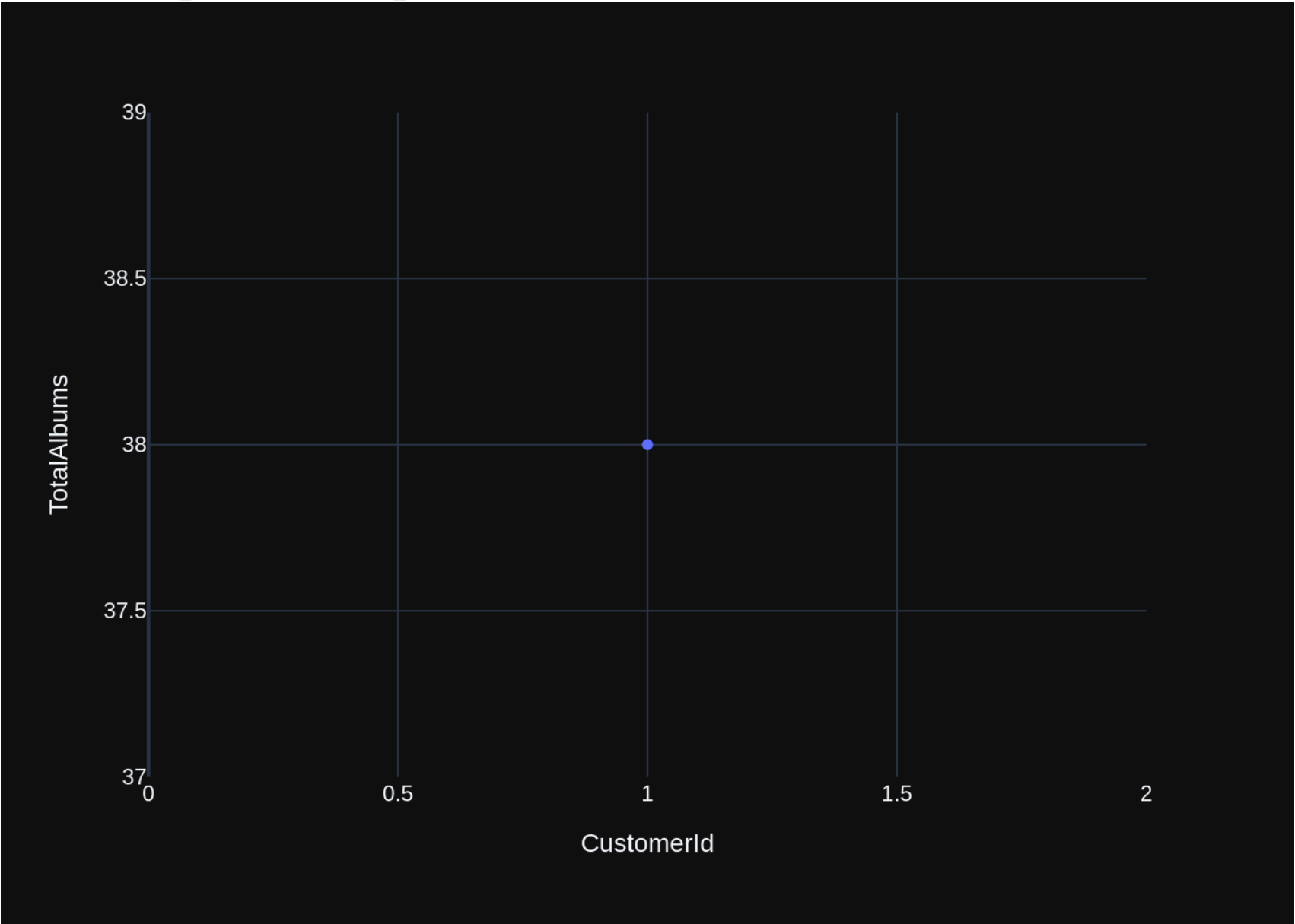
```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n      Find the customer who bought the most albums in total quantity (across all invoices): \n'\n\nThe DataFrame was produced using this query: SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\nCustomerId      int64\nTotalAlbums      int64\nndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:57:59.45195276Z', 'message': {'role': 'assistant', 'content': "\n\npython\nimport plotly.express as px\n\nfig = px.bar(df, x='CustomerId', y='TotalAlbums', title='Customer Album Purchases')\n\nif len(df) == 1:\n    fig.add_trace(px.indicator(value=df['TotalAlbums']
```



```
s'].iloc[0], title='Total Albums Purchased'))\n\nfig.show()\n``"}}, 'done_reason': 'stop', 'done': True, 'total_duration': 30363457561, 'load_duration': 757730, 'prompt_eval_count': 224, 'prompt_eval_duration': 9353209000, 'eval_count': 80, 'eval_duration': 20880106000}
```



```

Out[36]: ('SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1',
          CustomerId TotalAlbums
          0          1          38,
          Figure({
            'data': [{'hovertemplate': 'CustomerId=%{x}<br>TotalAlbums=%{y}<extra></extra>',
                      'legendgroup': '',
                      'marker': {'color': '#636efa', 'symbol': 'circle'},
                      'mode': 'markers',
                      'name': '',
                      'orientation': 'v',
                      'showlegend': False,
                      'type': 'scatter',
                      'x': array([1]),
                      'xaxis': 'x',
                      'y': array([38]),
                      'yaxis': 'y'}],
            'layout': {'legend': {'tracegroupgap': 0},
                      'margin': {'t': 60},
                      'template': '...',
                      'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'CustomerId'}},
                      'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'TotalAlbums'}}}
          })

```

```

In [37]: question = """
          Hint: album quantity is found in invoice_items,

          Find the top 5 customers who bought the most albums in total quantity (across all invoices):
          """
          vn.ask(question=question)

```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

```
[{'role': 'system', 'content': 'You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables\nCREATE TABLE "invoice_items"\n(\n    InvoiceLineId INTEGER PRIMARY KEY\n    AUTOINCREMENT NOT NULL,\n    InvoiceId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    Quantity INTEGER NOT NULL,\n    FOREIGN KEY (InvoiceId) REFERENCES\n    "invoices" (InvoiceId)\n)\nCREATE TABLE "tracks"\n(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES "albums" (AlbumId)\n)\nCREATE TABLE "albums"\n(\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Title NVARCHAR(160) NOT NULL,\n    ArtistId INTEGER NOT NULL,\n    FOREIGN KEY (ArtistId) REFERENCES "artists" (ArtistId)\n)\nCREATE INDEX IFK_AlbumArtistId ON "albums" (ArtistId)\nCREATE INDEX IFK_InvoiceLineInvoiceId ON "invoice_items" (InvoiceId)\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\nCREATE TABLE "invoices"\n(\n    InvoiceId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    CustomerId INTEGER NOT NULL,\n    InvoiceDate DATETIME NOT NULL,\n    BillingAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (CustomerId) REFERENCES "customers" (CustomerId)\n)\nCREATE INDEX IFK_InvoiceCustomerId ON "invoices" (CustomerId)\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\nCREATE TABLE "artists"\n(\n    ArtistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\n===Additional Context\nIn the SQLite database invoice means order\n===Response Guidelines\n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question.\n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying in intermediate_sql\n3. If the provided context is insufficient, please explain why it can't be generated.\n4. Please use the most relevant table(s).\n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before.\n'}], {'role': 'user', 'content': '\n    Hint: album quantity is found in invoice_items,\n    Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': '\n    Hint: album quantity is found in invoice_items,\n    Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': '\n    Find the customer who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "c'}
```

```

"customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}}, {'role': 'user', 'content': ' \n Find the top 5 customers who spent the most money overall, \n \n Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessary \n'}}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(ii.Total) AS TotalSpent\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5'}}, {'role': 'user', 'content': ' \n There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\n Can you find the top 10 most popular artists based on the number of tracks\n'}}, {'role': 'assistant', 'content': 'SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nJOIN "tracks" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10'}}, {'role': 'user', 'content': ' \n Find the customer with the most invoices \n'}}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1'}}, {'role': 'user', 'content': ' \n Find the top 5 most expensive tracks (based on unit price):\n'}}, {'role': 'assistant', 'content': 'SELECT t.TrackId, t.Name, t.UnitPrice\nFROM "tracks" t\nORDER BY t.UnitPrice DESC\nLIMIT 5'}}, {'role': 'user', 'content': ' \n Identify artists who have albums with tracks appearing in multiple genres:\n\n\n'}}, {'role': 'assistant', 'content': 'SELECT a.ArtistId, a.Name AS ArtistName\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n SELECT g2.GenreId\n FROM "genres" g2\n JOIN "tracks" t2 ON g2.GenreId = t2.GenreId\n GROUP BY g2.GenreId\n HAVING COUNT(g2.GenreId) > 1)\nGROUP BY a.ArtistId, a.Name'}}, {'role': 'user', 'content': ' \n Get the total number of invoices for each customer\n'}}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}}, {'role': 'user', 'content': ' \n Find the total number of invoices per country:\n'}}, {'role': 'assistant', 'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "invoices" i\nGROUP BY i.BillingCountry'}}, {'role': 'user', 'content': ' \n Hint: album quantity is found in invoice_items, \n \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}}

```

Ollama parameters:

```
model=codegemma:latest,
```

```
options={},
```

```
keep alive=None
```

Prompt Content:

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables\nCREATE TABLE \"invoice_items\"(\n    InvoiceLineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    InvoiceId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    Quantity INTEGER NOT NULL,\n    FOREIGN KEY (InvoiceId) REFERENCES \"invoices\" (InvoiceId)\nON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId)\nON DELETE NO ACTION ON UPDATE NO ACTION)\nCREATE TABLE \"tracks\"(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL
```

```

L,\r\n    FOREIGN KEY (AlbumId) REFERENCES \"albums\" (AlbumId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO AC
TION,\r\n    FOREIGN KEY (GenreId) REFERENCES \"genres\" (GenreId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO
ACTION,\r\n    FOREIGN KEY (MediaTypeId) REFERENCES \"media_types\" (MediaTypeId) \r\n\t\t\tON DELETE NO ACTI
ON ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"albums\"(\r\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMEN
T NOT NULL,\r\n    Title NVARCHAR(160) NOT NULL,\r\n    ArtistId INTEGER NOT NULL,\r\n    FOREIGN KEY (Ar
tistId) REFERENCES \"artists\" (ArtistId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE IN
DEX IFK_AlbumArtistId ON \"albums\" (ArtistId)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON \"invoice_items
\" (InvoiceId)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON \"invoice_items\" (TrackId)\n\nCREATE TABLE \"invo
ices\"(\r\n    InvoiceId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    CustomerId INTEGER NOT NUL
L,\r\n    InvoiceDate DATETIME NOT NULL,\r\n    BillingAddress NVARCHAR(70),\r\n    BillingCity NVARCHAR(4
0),\r\n    BillingState NVARCHAR(40),\r\n    BillingCountry NVARCHAR(40),\r\n    BillingPostalCode NVARCHAR
(10),\r\n    Total NUMERIC(10,2) NOT NULL,\r\n    FOREIGN KEY (CustomerId) REFERENCES \"customers\" (Custo
merId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON \"invo
ices\" (CustomerId)\n\nCREATE INDEX IFK_TrackAlbumId ON \"tracks\" (AlbumId)\n\nCREATE TABLE \"artists\"(\r
\n    ArtistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Name NVARCHAR(120)\r\n)\n\n\n===Addi
tional Context \n\nIn the SQLite database invoice means order\n\n===Response Guidelines \n1. If the provide
d context is sufficient, please generate a valid SQL query without any explanations for the question. \n2.
If the provided context is almost sufficient but requires knowledge of a specific string in a particular co
lumn, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the qu
ery with a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why
it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and
answered before, please repeat the answer exactly as it was given before. \n\"}, {\"role\": \"user\", \"content\":
\" \n    Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 customers who bought th
e most albums in total quantity (across all invoices):\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.Cust
omerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.Cus
tomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbu
ms DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n    Hint: album quantity is found in invoice_items, \n
\n    Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n\"}, {\"r
ole\": \"assistant\", \"content\": \"SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices
\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums
DESC\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n    Find the customer who bought the most albums in tota
l quantity (across all invoices): \n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.Tra
ckId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invo
ice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1\"},
{\"role\": \"user\", \"content\": \" \n    Find the top 5 customers who spent the most money overall, \n    \n
Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessa
ry \n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customer
s\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC
\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \n    There are 3 tables: artists, albums and tracks, where albu
ms and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\n    Can you find the top 1
0 most popular artists based on the number of tracks\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT a.Name,
COUNT(t.TrackId) AS TotalTracks\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nJOIN

```

```
\"tracks\" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10\"}, {\"role\": \"u
ser\", \"content\": \" \n      Find the customer with the most invoices \n\"}, {\"role\": \"assistant\", \"content\":
\"SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.C
ustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1\"}, {\"role\": \"user\",
\"content\": \" \n      Find the top 5 most expensive tracks (based on unit price):\n\"}, {\"role\": \"assistant\",
\"content\": \"SELECT t.TrackId, t.Name, t.UnitPrice\nFROM \"tracks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5\"},
{\"role\": \"user\", \"content\": \" \n      Identify artists who have albums with tracks appearing in multiple ge
nres:\n\n\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT a.ArtistId, a.Name AS ArtistName\nFROM \"artists\"
a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n  SELECT g2.GenreId\n  FROM \"gen
res\" g2\n  JOIN \"tracks\" t2 ON g2.GenreId = t2.GenreId\n  GROUP BY g2.GenreId\n  HAVING COUNT(g2.GenreI
d) > 1\n)\nGROUP BY a.ArtistId, a.Name\"}, {\"role\": \"user\", \"content\": \" \n      Get the total number of invo
ices for each customer\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT(i.InvoiceId) AS To
talInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerI
d\"}, {\"role\": \"user\", \"content\": \" \n      Find the total number of invoices per country:\n\"}, {\"role\": \"ass
istant\", \"content\": \"SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM \"invoices\" i\nGROUP BY i.Bi
llingCountry\"}, {\"role\": \"user\", \"content\": \" \n      Hint: album quantity is found in invoice_items, \n
\n      Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n\"}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T22:59:35.63141622Z', 'message': {'role': 'assistan
t', 'content': 'SELECT c.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoi
ce_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, 'd
one_reason': 'stop', 'done': True, 'total_duration': 96082450718, 'load_duration': 829857, 'prompt_eval_cou
nt': 1808, 'prompt_eval_duration': 80020504000, 'eval_count': 57, 'eval_duration': 15375800000}
```

```
SELECT c.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums
FROM \"invoices\" i
JOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId
GROUP BY c.CustomerId
ORDER BY TotalAlbums DESC
LIMIT 5
SELECT c.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums
FROM \"invoices\" i
JOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId
GROUP BY c.CustomerId
ORDER BY TotalAlbums DESC
LIMIT 5
```

```
Couldn't run sql: Execution failed on sql 'SELECT c.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums
FROM \"invoices\" i
JOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId
GROUP BY c.CustomerId
ORDER BY TotalAlbums DESC
LIMIT 5': no such column: c.CustomerId
```

```
SELECT c.CustomerId, SUM(il.Quantity) AS TotalAlbums
FROM Customers c
JOIN invoices i ON c.CustomerId = i.CustomerId
JOIN invoice_items il ON i.InvoiceId = il.InvoiceId
GROUP BY c.CustomerId
ORDER BY TotalAlbums DESC
LIMIT 5
```

```
In [38]: question = """
        Find the top 5 customers who spent the most money overall,

        Hint: order total can be found on invoices table, calculation using invoice_items detail table is unneces
        """

        vn.ask(question=question)
```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

projects/wqong/py4kids/lesson-18-ai/vanna/docs/ollama-codegemma-chromadb-sqlite-test-1.html 176/199


```

repeat the answer exactly as it was given before. \n'}, {'role': 'user', 'content': ' \n      Find the top 5
customers who spent the most money overall, \n      \n      Hint: order total can be found on invoices table,
calculation using invoice_items detail table is unnecessary \n'}, {'role': 'assistant', 'content': 'SELECT
c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.Custome
rId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Hint:
album quantity is found in invoice_items, \n      \n      Find the top 5 customers who bought the most albums i
n total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i
i.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invo
ice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'},
{'role': 'user', 'content': ' \n      Hint: album quantity is found in invoice_items, \n      \n      Find the t
op 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistan
t', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices" i\nJOIN "invoi
ce_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'},
{'role': 'user', 'content': ' \n      Find the customer with the most invoices \n'}, {'role': 'assistant',
'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i
ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1'}, {'role': 'us
er', 'content': ' \n      Find the customer who bought the most albums in total quantity (across all invoic
es): \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "c
ustomers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.
InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n
Get the average invoice total for each customer:\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerI
d, AVG(i.Total) AS AverageInvoiceTotal\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerI
d\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': ' \n      Get the total number of invoices for each
customer\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nF
ROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'use
r', 'content': ' \n      Find the top 5 most expensive tracks (based on unit price):\n'}, {'role': 'assistan
t', 'content': 'SELECT t.TrackId, t.Name, t.UnitPrice\nFROM "tracks" t\nORDER BY t.UnitPrice DESC\nLIMIT
5'}, {'role': 'user', 'content': 'what are the top 5 countries that customers come from?'}, {'role': 'assis
tant', 'content': 'SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM "customers" c\nGROUP BY c.Country\nOR
DER BY TotalCustomers DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Find all invoices since 2010 an
d the total amount invoiced:\n'}, {'role': 'assistant', 'content': 'SELECT i.InvoiceDate, SUM(i.Total) AS T
otalAmount\nFROM "invoices" i\nWHERE i.InvoiceDate >= \'2010-01-01\'\nGROUP BY i.InvoiceDate'}, {'role': 'u
ser', 'content': ' \n      Find the top 5 customers who spent the most money overall, \n      \n      Hint: o
rder total can be found on invoices table, calculation using invoice_items detail table is unnecessary
\n'}]

```

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```

[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the
question. Your response should ONLY be based on the given context and follow the response guidelines and fo

```

```

rmat instructions. \n===Tables \nCREATE TABLE \"invoices\"(\r\n\r\n    InvoiceId INTEGER PRIMARY KEY AUTOIN
CREMENT NOT NULL,\r\n    CustomerId INTEGER NOT NULL,\r\n    InvoiceDate DATETIME NOT NULL,\r\n    Billin
gAddress NVARCHAR(70),\r\n    BillingCity NVARCHAR(40),\r\n    BillingState NVARCHAR(40),\r\n    BillingCou
ntry NVARCHAR(40),\r\n    BillingPostalCode NVARCHAR(10),\r\n    Total NUMERIC(10,2) NOT NULL,\r\n    FORE
IGN KEY (CustomerId) REFERENCES \"customers\" (CustomerId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION
\r\n)\n\nCREATE TABLE \"invoice_items\"(\r\n\r\n    InvoiceLineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NUL
L,\r\n    InvoiceId INTEGER NOT NULL,\r\n    TrackId INTEGER NOT NULL,\r\n    UnitPrice NUMERIC(10,2) NO
T NULL,\r\n    Quantity INTEGER NOT NULL,\r\n    FOREIGN KEY (InvoiceId) REFERENCES \"invoices\" (InvoiceI
d) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (Tra
ckId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON \"in
voice_items\" (InvoiceId)\n\nCREATE INDEX IFK_InvoiceCustomerId ON \"invoices\" (CustomerId)\n\nCREATE INDE
X IFK_InvoiceLineTrackId ON \"invoice_items\" (TrackId)\n\nCREATE TABLE \"customers\"(\r\n\r\n    CustomerI
d INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    FirstName NVARCHAR(40) NOT NULL,\r\n    LastName NVAR
CHAR(20) NOT NULL,\r\n    Company NVARCHAR(80),\r\n    Address NVARCHAR(70),\r\n    City NVARCHAR(40),\r\n
State NVARCHAR(40),\r\n    Country NVARCHAR(40),\r\n    PostalCode NVARCHAR(10),\r\n    Phone NVARCHAR(2
4),\r\n    Fax NVARCHAR(24),\r\n    Email NVARCHAR(60) NOT NULL,\r\n    SupportRepId INTEGER,\r\n    FOREI
GN KEY (SupportRepId) REFERENCES \"employees\" (EmployeeId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION
\r\n)\n\nCREATE TABLE \"employees\"(\r\n\r\n    EmployeeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n
LastName NVARCHAR(20) NOT NULL,\r\n    FirstName NVARCHAR(20) NOT NULL,\r\n    Title NVARCHAR(30),\r\n
ReportsTo INTEGER,\r\n    BirthDate DATETIME,\r\n    HireDate DATETIME,\r\n    Address NVARCHAR(70),\r\n
City NVARCHAR(40),\r\n    State NVARCHAR(40),\r\n    Country NVARCHAR(40),\r\n    PostalCode NVARCHAR(1
0),\r\n    Phone NVARCHAR(24),\r\n    Fax NVARCHAR(24),\r\n    Email NVARCHAR(60),\r\n    FOREIGN KEY (Repo
rtsTo) REFERENCES \"employees\" (EmployeeId) \r\n\t\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE
TABLE \"tracks\"(\r\n\r\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n    Name NVARCHAR(200)
NOT NULL,\r\n    AlbumId INTEGER,\r\n    MediaTypeId INTEGER NOT NULL,\r\n    GenreId INTEGER,\r\n    Comp
oser NVARCHAR(220),\r\n    Milliseconds INTEGER NOT NULL,\r\n    Bytes INTEGER,\r\n    UnitPrice NUMERIC(1
0,2) NOT NULL,\r\n    FOREIGN KEY (AlbumId) REFERENCES \"albums\" (AlbumId) \r\n\t\t\tON DELETE NO ACTION ON
UPDATE NO ACTION,\r\n    FOREIGN KEY (GenreId) REFERENCES \"genres\" (GenreId) \r\n\t\t\tON DELETE NO ACTION
ON UPDATE NO ACTION,\r\n    FOREIGN KEY (MediaTypeId) REFERENCES \"media_types\" (MediaTypeId) \r\n\t\t\tON D
ELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"playlist_track\"(\r\n\r\n    PlaylistId INTEGER
NOT NULL,\r\n    TrackId INTEGER NOT NULL,\r\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, Tr
ackId),\r\n    FOREIGN KEY (PlaylistId) REFERENCES \"playlists\" (PlaylistId) \r\n\t\t\tON DELETE NO ACTION O
N UPDATE NO ACTION,\r\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\t\tON DELETE NO ACTIO
N ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_EmployeeReportsTo ON \"employees\" (ReportsTo)\n\n\n===Addit
ional Context \n\nIn the SQLite database invoice means order\n\n===Response Guidelines \n1. If the provided
context is sufficient, please generate a valid SQL query without any explanations for the question. \n2. If
the provided context is almost sufficient but requires knowledge of a specific string in a particular colum
n, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query
with a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it
can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and ans
wered before, please repeat the answer exactly as it was given before. \n\"}, {\"role\": \"user\", \"content\": \"
\n    Find the top 5 customers who spent the most money overall, \n    \n    Hint: order total can be fo

```

```

und on invoices table, calculation using invoice_items detail table is unnecessary \n"}}, {"role": "assistant", "content": "SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5"}}, {"role": "user", "content": " \n Hint: album quantity is found in invoice_items, \n \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}}, {"role": "user", "content": " \n Hint: album quantity is found in invoice_items, \n \n Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}}, {"role": "assistant", "content": "SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}}, {"role": "user", "content": " \n Find the customer with the most invoices \n"}}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1"}}, {"role": "user", "content": " \n Find the customer who bought the most albums in total quantity (across all invoices): \n"}}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1"}}, {"role": "user", "content": " \n Get the average invoice total for each customer:\n"}}, {"role": "assistant", "content": "SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}}, {"role": "user", "content": " \n Get the total number of invoices for each customer\n"}}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}}, {"role": "user", "content": " \n Find the top 5 most expensive tracks (based on unit price):\n"}}, {"role": "assistant", "content": "SELECT t.TrackId, t.Name, t.UnitPrice\nFROM \"tracks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5"}}, {"role": "user", "content": "what are the top 5 countries that customers come from?"}, {"role": "assistant", "content": "SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM \"customers\" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5"}}, {"role": "user", "content": " \n Find all invoices since 2010 and the total amount invoiced:\n"}}, {"role": "assistant", "content": "SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\nFROM \"invoices\" i\nWHERE i.InvoiceDate >= '2010-01-01'\nGROUP BY i.InvoiceDate"}}, {"role": "user", "content": " \n Find the top 5 customers who spent the most money overall, \n \n Hint: order total can be found on invoices table, calculation using invoice_items detail table is unnecessary \n"}]}

```

Ollama Response:

```

{'model': 'codegemma:latest', 'created_at': '2024-06-13T23:01:14.386175427Z', 'message': {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c'}, 'done_reason': 'stop', 'done': True, 'total_duration': 98713492756, 'load_duration': 924926, 'prompt_eval_count': 1962, 'prompt_eval_duration': 87403438000, 'eval_count': 39, 'eval_duration': 10639920000}
SELECT c.CustomerId, SUM(i.Total) AS TotalSpent
FROM \"customers\" c
JOIN \"invoices\" i ON c.CustomerId = i.CustomerId

```

```
GROUP BY c
SELECT c.CustomerId, SUM(i.Total) AS TotalSpent
FROM "customers" c
JOIN "invoices" i ON c.CustomerId = i.CustomerId
GROUP BY c
Couldn't run sql: Execution failed on sql 'SELECT c.CustomerId, SUM(i.Total) AS TotalSpent
FROM "customers" c
JOIN "invoices" i ON c.CustomerId = i.CustomerId
GROUP BY c': no such column: c
```

```
In [39]: question = """
        Get all playlists containing at least 10 tracks and the total duration of those tracks:
        """
        vn.ask(question=question)
```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

```
q2.GenreId\n FROM "genres" q2\n JOIN "tracks" t2 ON q2.GenreId = t2.GenreId\n GROUP BY q2.GenreId\n HAV
```

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables \nCREATE INDEX IFK_PlaylistTrackTrackId ON \"playlist_track\" (TrackId)\n\nCREATE TABLE \"playlists\"\n(\n    PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\n\nCREATE TABLE \"playlist_track\"\n(\n    PlaylistId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES \"playlists\" (PlaylistId) \n    \n    ON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \n    \n    ON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE \"tracks\"\n(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES \"albums\" (AlbumId) \n    \n    ON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES \"genres\" (GenreId) \n    \n    ON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES \"media_types\" (MediaTypeId) \n    \n    ON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_TrackGenreId ON \"tracks\" (GenreId)\n\nCREATE INDEX IFK_TrackAlbumId ON \"tracks\" (AlbumId)\n\nCREATE INDEX IFK_TrackMediaTypeId ON \"tr
```

```

acks\" (MediaTypeId)\n\nCREATE INDEX IFK_AlbumArtistId ON \"albums\" (ArtistId)\n\nCREATE TABLE \"albums\"
\"\\r\\n(\\r\\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\\r\\n    Title NVARCHAR(160) NOT NULL,\\r
\\n    ArtistId INTEGER NOT NULL,\\r\\n    FOREIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistId) \\r\\n\\t\\tO
N DELETE NO ACTION ON UPDATE NO ACTION\\r\\n)\n\nCREATE TABLE \"genres\"\\r\\n(\\r\\n    GenreId INTEGER PRIMARY
KEY AUTOINCREMENT NOT NULL,\\r\\n    Name NVARCHAR(120)\\r\\n)\n\n===Additional Context \\n\\nIn the SQLite dat
abase invoice means order\\n\\n===Response Guidelines \\n1. If the provided context is sufficient, please gene
rate a valid SQL query without any explanations for the question. \\n2. If the provided context is almost su
fficient but requires knowledge of a specific string in a particular column, please generate an intermediat
e SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermedia
te_sql \\n3. If the provided context is insufficient, please explain why it can't be generated. \\n4. Please
use the most relevant table(s). \\n5. If the question has been asked and answered before, please repeat the
answer exactly as it was given before. \\n\"}, {\"role\": \"user\", \"content\": \" \\n    Get all playlists contain
ing at least 10 tracks and the total duration of those tracks:\\n\"}, {\"role\": \"assistant\", \"content\": \"SELE
CT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\\nFROM \"playlist_track\" pt
\\nJOIN \"playlists\" p ON pt.PlaylistId = p.PlaylistId\\nJOIN \"tracks\" t ON pt.TrackId = t.TrackId\\nGROUP
BY pt.PlaylistId, p.Name\\nHAVING COUNT(pt.TrackId) >= 10\"}, {\"role\": \"user\", \"content\": \" \\n    List all g
enres and the number of tracks in each genre:\\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT g.Name, COUNT
(t.GenreId) AS TotalTracks\\nFROM \"genres\" g\\nJOIN \"tracks\" t ON g.GenreId = t.GenreId\\nGROUP BY g.Nam
e\"}, {\"role\": \"user\", \"content\": \" \\n    There are 3 tables: artists, albums and tracks, where albums and a
rtists are linked by ArtistId, albums and tracks are linked by AlbumId,\\n    Can you find the top 10 most p
opular artists based on the number of tracks\\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT a.Name, COUNT(t.
TrackId) AS TotalTracks\\nFROM \"artists\" a\\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\\nJOIN \"tracks
\" t ON al.AlbumId = t.AlbumId\\nGROUP BY a.Name\\nORDER BY TotalTracks DESC\\nLIMIT 10\"}, {\"role\": \"user\", \"c
ontent\": \" \\n    Identify artists who have albums with tracks appearing in multiple genres:\\n\\n\\n\"}, {\"ro
le\": \"assistant\", \"content\": \"SELECT a.ArtistId, a.Name AS ArtistName\\nFROM \"artists\" a\\nJOIN \"albums\"
al ON a.ArtistId = al.ArtistId\\nWHERE a.ArtistId IN (\\n    SELECT g2.GenreId\\n    FROM \"genres\" g2\\n    JOIN
\"tracks\" t2 ON g2.GenreId = t2.GenreId\\n    GROUP BY g2.GenreId\\n    HAVING COUNT(g2.GenreId) > 1\\n)\\nGROUP B
Y a.ArtistId, a.Name\"}, {\"role\": \"user\", \"content\": \" \\n    Find the customer who bought the most albums
in total quantity (across all invoices): \\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT c.CustomerId, COUNT
(ii.TrackId) AS TotalAlbums\\nFROM \"customers\" c\\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\\nJOIN
\"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\\nGROUP BY c.CustomerId\\nORDER BY TotalAlbums DESC\\nLIMIT
1\"}, {\"role\": \"user\", \"content\": \" \\n    Hint: album quantity is found in invoice_items, \\n    \\n    Find
the top 5 customers who bought the most albums in total quantity (across all invoices):\\n\"}, {\"role\": \"assi
stant\", \"content\": \"SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\\nFROM \"customers\" c\\nJOIN \"inv
oices\" i ON c.CustomerId = i.CustomerId\\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\\nGROUP BY
c.CustomerId\\nORDER BY TotalAlbums DESC\\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \\n    Find all tracks wi
th a name containing \"What\" (case-insensitive)\\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT * \\nFROM \"t
racks\" \\nWHERE LOWER(Name) LIKE '%what%'\"}, {\"role\": \"user\", \"content\": \" \\n    Hint: album quantity is f
ound in invoice_items, \\n    \\n    Find the top 5 customers who bought the most albums in total quantity (a
cross all invoices):\\n\"}, {\"role\": \"assistant\", \"content\": \"SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS
TotalAlbums\\nFROM \"invoices\" i\\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\\nGROUP BY i.Custo
merId\\nORDER BY TotalAlbums DESC\\nLIMIT 5\"}, {\"role\": \"user\", \"content\": \" \\n    Find the top 5 most expen

```

```
sive tracks (based on unit price):\n"}, {"role": "assistant", "content": "SELECT t.TrackId, t.Name, t.UnitP\nrice\nFROM \"tracks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5"}, {"role": "user", "content": " \n  List al\nl albums and their corresponding artist names \n"}, {"role": "assistant", "content": "SELECT a.Title, a.Ar\nlistId, ar.Name AS ArtistName\nFROM \"albums\" a\nJOIN \"artists\" ar ON a.ArtistId = ar.ArtistId"}, {"rol\ne": "user", "content": " \n  Get all playlists containing at least 10 tracks and the total duration of\nthose tracks:\n"}]
```

Add of existing embedding ID: e7c4b3aa-664f-5f87-8b25-449a4482f3fd-sql

Insert of existing embedding ID: e7c4b3aa-664f-5f87-8b25-449a4482f3fd-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T23:02:57.347300303Z', 'message': {'role': 'assistant', 'content': 'SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM "playlist_track" pt\nJOIN "playlists" p ON pt.PlaylistId = p.PlaylistId\nJOIN "tracks" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10'}, 'done_reason': 'stop', 'done': True, 'total_duration': 102920738722, 'load_duration': 821971, 'prompt_eval_count': 1683, 'prompt_eval_duration': 77459932000, 'eval_count': 88, 'eval_duration': 24780736000}
```

```
SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration
FROM "playlist_track" pt
JOIN "playlists" p ON pt.PlaylistId = p.PlaylistId
JOIN "tracks" t ON pt.TrackId = t.TrackId
GROUP BY pt.PlaylistId, p.Name
HAVING COUNT(pt.TrackId) >= 10
SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration
FROM "playlist_track" pt
JOIN "playlists" p ON pt.PlaylistId = p.PlaylistId
JOIN "tracks" t ON pt.TrackId = t.TrackId
GROUP BY pt.PlaylistId, p.Name
HAVING COUNT(pt.TrackId) >= 10
```

	PlaylistId	PlaylistName	TotalDuration
0	1	Music	877683083
1	3	TV Shows	501094957
2	5	90's Music	398705153
3	8	Music	877683083
4	10	TV Shows	501094957
5	11	Brazilian Music	9486559
6	12	Classical	21770592
7	13	Classical 101 - Deep Cuts	6755730
8	14	Classical 101 - Next Steps	7575051
9	15	Classical 101 - The Basics	7439811
10	16	Grunge	4122018
11	17	Heavy Metal Classic	8206312

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n      Get all playlists containing at least 10 tracks and the total duration of those tracks:\n'\n\nThe DataFrame was produced using this query: SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM \"playlist_track\" pt\nJOIN \"playlists\" p ON pt.PlaylistId = p.PlaylistId\nJOIN \"tracks\" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.N
```

```

ame\nHAVING COUNT(pt.TrackId) >= 10\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n PlaylistId          int64\n PlaylistName      object\n TotalDuration      int64\nndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]

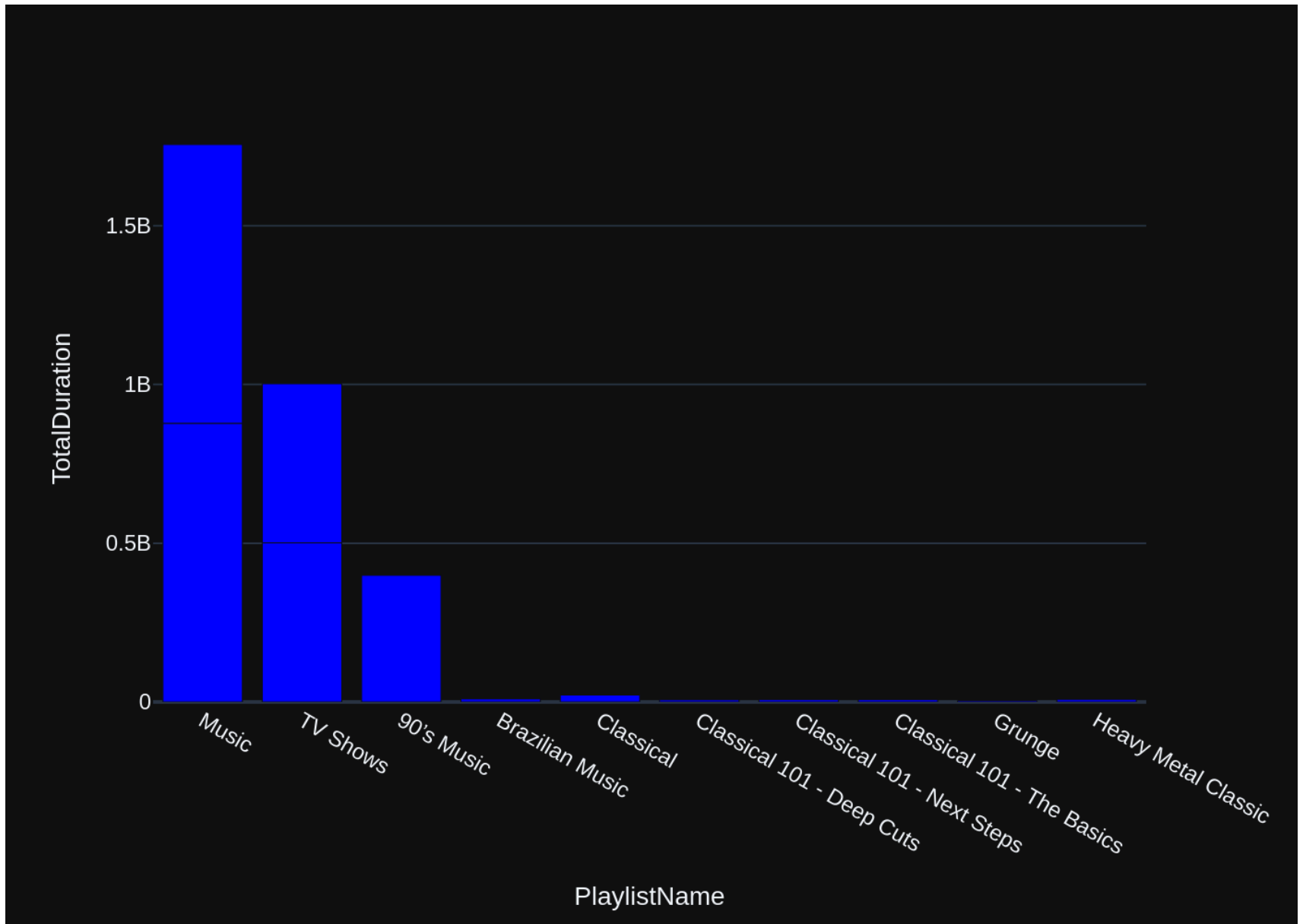
```

Ollama Response:

```

{'model': 'codegemma:latest', 'created_at': '2024-06-13T23:03:24.715454197Z', 'message': {'role': 'assistant', 'content': "\n\npython\nimport plotly.express as px\n\nfig = px.bar(df, x='PlaylistName', y='TotalDuration', hover_name='PlaylistName')\nfig.update_traces(marker_color='blue')\nfig.show()\n\n"}, 'done_reason': 'stop', 'done': True, 'total_duration': 27340345217, 'load_duration': 700342, 'prompt_eval_count': 248, 'prompt_eval_duration': 11523796000, 'eval_count': 56, 'eval_duration': 15684832000}

```



```
Out[39]: ('SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM "playlist_tracks" pt\nJOIN "playlists" p ON pt.PlaylistId = p.PlaylistId\nJOIN "tracks" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10',
```

	PlaylistId	PlaylistName	TotalDuration
0	1	Music	877683083
1	3	TV Shows	501094957
2	5	90's Music	398705153
3	8	Music	877683083
4	10	TV Shows	501094957
5	11	Brazilian Music	9486559
6	12	Classical	21770592
7	13	Classical 101 - Deep Cuts	6755730
8	14	Classical 101 - Next Steps	7575051
9	15	Classical 101 - The Basics	7439811
10	16	Grunge	4122018
11	17	Heavy Metal Classic	8206312

```
Figure({
  'data': [{'alignmentgroup': 'True',
    'hovertemplate': ('<b>{hovertext}</b><br><br>Pla' ... 'alDuration=%{y}<extra></extra>'),
    'hovertext': array(['Music', 'TV Shows', '90's Music', 'Music', 'TV Shows',
      'Brazilian Music', 'Classical', 'Classical 101 - Deep Cuts',
      'Classical 101 - Next Steps', 'Classical 101 - The Basics', 'Grunge',
      'Heavy Metal Classic'], dtype=object),
    'legendgroup': '',
    'marker': {'color': 'blue', 'pattern': {'shape': ''}},
    'name': '',
    'offsetgroup': '',
    'orientation': 'v',
    'showlegend': False,
    'textposition': 'auto',
    'type': 'bar',
    'x': array(['Music', 'TV Shows', '90's Music', 'Music', 'TV Shows',
      'Brazilian Music', 'Classical', 'Classical 101 - Deep Cuts',
      'Classical 101 - Next Steps', 'Classical 101 - The Basics', 'Grunge',
      'Heavy Metal Classic'], dtype=object),
    'xaxis': 'x',
    'y': array([877683083, 501094957, 398705153, 877683083, 501094957, 9486559,
      21770592, 6755730, 7575051, 7439811, 4122018, 8206312]),
    'yaxis': 'y'}],
  'layout': {'barmode': 'relative',
    'legend': {'tracegroupgap': 0},
    'margin': {'t': 60},
```

```
        'template': '...',  
        'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'PlaylistName'}},  
        'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'TotalDuration'}}}  
    )))
```

```
In [40]: question = """  
        Identify artists who have albums with tracks appearing in multiple genres:  
  
        """  
  
        vn.ask(question=question)
```

Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1

```
[{'role': 'system', 'content': 'You are a SQLite expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables\nCREATE TABLE "tracks"\n(\n    TrackId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(200) NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2) NOT NULL,\n    FOREIGN KEY (AlbumId) REFERENCES "albums" (AlbumId)\n)\nCREATE INDEX IFK_AlbumArtistId ON "albums" (ArtistId)\nCREATE INDEX IFK_TrackGenreId ON "tracks" (GenreId)\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\nCREATE TABLE "albums"\n(\n    AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Title NVARCHAR(160) NOT NULL,\n    ArtistId INTEGER NOT NULL,\n    FOREIGN KEY (ArtistId) REFERENCES "artists" (ArtistId)\n)\nCREATE INDEX IFK_TrackMediaTypeId ON "tracks" (MediaTypeId)\nCREATE TABLE "genres"\n(\n    GenreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\nCREATE INDEX IFK_PlaylistTrackTrackId ON "playlist_track" (TrackId)\nCREATE TABLE "artists"\n(\n    ArtistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n    Name NVARCHAR(120)\n)\nCREATE TABLE "playlist_track"\n(\n    PlaylistId INTEGER NOT NULL,\n    TrackId INTEGER NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId)\n)\nCREATE INDEX IFK_PlaylistTrackTrackId ON "playlist_track" (TrackId)\nCREATE INDEX IFK_TrackGenreId ON "tracks" (GenreId)\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\n\n===Additional Context\n\nIn the SQLite database invoice means order\n\n===Response Guidelines\n\n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question.\n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql\n3. If the provided context is insufficient, please explain why it can't be generated.\n4. Please use the most relevant table(s).\n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before.\n\n}', {'role': 'user', 'content': '\n    Identify artists who have albums with tracks appearing in multiple genres:\n\n\n'}, {'role': 'assistant', 'content': 'SELECT a.ArtistId, a.Name AS ArtistName\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n    SELECT g2.GenreId\n    FROM "genres" g2\n    JOIN "tracks" t2 ON g2.GenreId = t2.GenreId\n    GROUP BY g2.GenreId\n    HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name'}, {'role': 'user', 'content': '\n    There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\n    Can you find the top 10 most popular artists based on the number of tracks\n\n'}, {'role': 'assistant', 'content': 'SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nJOIN "tracks" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10'}, {'role': 'user', 'content': '\n    List all albums and their corresponding artist names\n\n'}, {'role': 'assistant', 'content': 'SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM "albums" a\nJOIN "artists" ar ON a.ArtistId = ar.ArtistId'}, {'role': 'user', 'content': '\n    List all genres and the number of tracks in each genre:\n\n'}, {'role': 'assistant', 'content': 'SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM "genres" g\nJOIN "tracks" t ON g.GenreId = t.GenreId\nGROUP BY g.Name'}, {'role': 'user', 'content': '\n    Hint: album quantity is found in invoice_items, \n    \n    Find the top 5 customers who bought
```

```
t the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT c.
CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.Cus
tomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums
DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Find the customer who bought the most albums in tota
l quantity (across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(ii.Tra
ckId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_i
tems" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'rol
e': 'user', 'content': ' \n      Hint: album quantity is found in invoice_items, \n      \n      Find the top 5
customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'c
ontent': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices" i\nJOIN "invoice_ite
ms" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role':
'user', 'content': ' \n      Get all playlists containing at least 10 tracks and the total duration of thos
e tracks:\n'}, {'role': 'assistant', 'content': 'SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Millis
econds) AS TotalDuration\nFROM "playlist_track" pt\nJOIN "playlists" p ON pt.PlaylistId = p.PlaylistId\nJOI
N "tracks" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10'},
{'role': 'user', 'content': ' \n      Find the top 5 most expensive tracks (based on unit price):\n'}, {'rol
e': 'assistant', 'content': 'SELECT t.TrackId, t.Name, t.UnitPrice\nFROM "tracks" t\nORDER BY t.UnitPrice D
ESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n      Find all tracks with a name containing "What" (case-in
sensitive)\n'}, {'role': 'assistant', 'content': 'SELECT * \nFROM "tracks" \nWHERE LOWER(Name) LIKE \'%wha
t%\'}}, {'role': 'user', 'content': ' \n      Identify artists who have albums with tracks appearing in mul
tiple genres:\n\n\n'}]
```

Ollama parameters:

model=codegemma:latest,

options={},

keep_alive=None

Prompt Content:

```
[{"role": "system", "content": "You are a SQLite expert. Please help to generate a SQL query to answer the
question. Your response should ONLY be based on the given context and follow the response guidelines and fo
rmat instructions. \n===Tables\nCREATE TABLE \"tracks\"(\n\n      TrackId INTEGER PRIMARY KEY AUTOINCREM
ENT NOT NULL,\n\n      Name NVARCHAR(200) NOT NULL,\n\n      AlbumId INTEGER,\n\n      MediaTypeId INTEGER NOT
NULL,\n\n      GenreId INTEGER,\n\n      Composer NVARCHAR(220),\n\n      Milliseconds INTEGER NOT NULL,\n\n
Bytes INTEGER,\n\n      UnitPrice NUMERIC(10,2) NOT NULL,\n\n      FOREIGN KEY (AlbumId) REFERENCES \"albums\"
(AlbumId) \n\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n\n      FOREIGN KEY (GenreId) REFERENCES \"genres
\" (GenreId) \n\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n\n      FOREIGN KEY (MediaTypeId) REFERENCES
\"media_types\" (MediaTypeId) \n\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n\n)\n\nCREATE INDEX IFK_Albu
mArtistId ON \"albums\" (ArtistId)\n\nCREATE INDEX IFK_TrackGenreId ON \"tracks\" (GenreId)\n\nCREATE INDEX
IFK_TrackAlbumId ON \"tracks\" (AlbumId)\n\nCREATE TABLE \"albums\"(\n\n      AlbumId INTEGER PRIMARY KEY
AUTOINCREMENT NOT NULL,\n\n      Title NVARCHAR(160) NOT NULL,\n\n      ArtistId INTEGER NOT NULL,\n\n      FOR
EIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistId) \n\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n\n)\n\n
CREATE INDEX IFK_TrackMediaTypeId ON \"tracks\" (MediaTypeId)\n\nCREATE TABLE \"genres\"(\n\n      Ge
nreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\n\n      Name NVARCHAR(120)\n\n)\n\nCREATE INDEX IFK_Playli
stTrackTrackId ON \"playlist_track\" (TrackId)\n\nCREATE TABLE \"artists\"(\n\n      ArtistId INTEGER PRI
```

```
MARY KEY AUTOINCREMENT NOT NULL,\r\n      Name NVARCHAR(120)\r\n)\n\r\nCREATE TABLE \"playlist_track\"\r\n(\r\n    PlaylistId INTEGER NOT NULL,\r\n    TrackId INTEGER NOT NULL,\r\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\r\n    FOREIGN KEY (PlaylistId) REFERENCES \"playlists\" (PlaylistId) \r\n)\n\r\nDELETE NO ACTION ON UPDATE NO ACTION,\r\n    FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n)\n\r\nDELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\r\n\r\n===Additional Context \n\nIn the SQLite database invoice means order\n\n===Response Guidelines \n1. If the provided context is sufficient, please generate a valid SQL query without any explanations for the question. \n2. If the provided context is almost sufficient but requires knowledge of a specific string in a particular column, please generate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment saying intermediate_sql\n3. If the provided context is insufficient, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before. \n"}, {"role": "user", "content": "\nIdentify artists who have albums with tracks appearing in multiple genres:\n\n\n"}, {"role": "assistant", "content": "SELECT a.ArtistId, a.Name AS ArtistName\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\nSELECT g2.GenreId\nFROM \"genres\" g2\nJOIN \"tracks\" t2 ON g2.GenreId = t2.GenreId\nGROUP BY g2.GenreId\nHAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name"}, {"role": "user", "content": "\nThere are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by AlbumId,\nCan you find the top 10 most popular artists based on the number of tracks\n"}, {"role": "assistant", "content": "SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nJOIN \"tracks\" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10"}, {"role": "user", "content": "\nList all albums and their corresponding artist names\n"}, {"role": "assistant", "content": "SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM \"albums\" a\nJOIN \"artists\" ar ON a.ArtistId = ar.ArtistId"}, {"role": "user", "content": "\nList all genres and the number of tracks in each genre:\n"}, {"role": "assistant", "content": "SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM \"genres\" g\nJOIN \"tracks\" t ON g.GenreId = t.GenreId\nGROUP BY g.Name"}, {"role": "user", "content": "\nHint: album quantity is found in invoice_items,\nFind the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": "\nFind the customer who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1"}, {"role": "user", "content": "\nHint: album quantity is found in invoice_items,\nFind the top 5 customers who bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": "\nGet all playlists containing at least 10 tracks and the total duration of those tracks:\n"}, {"role": "assistant", "content": "SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM \"playlist_track\" pt\nJOIN \"playlists\" p ON pt.PlaylistId = p.PlaylistId\nJOIN \"tracks\" t ON pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10"}, {"role": "user", "content": "\n"}]
```



```
Find the top 5 most expensive tracks (based on unit price):\n"}, {"role": "assistant", "content": "SELECT\n t.TrackId, t.Name, t.UnitPrice\nFROM \"tracks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5"}, {"role": "user",\n"content": "  \n  Find all tracks with a name containing \"What\" (case-insensitive)\n"}, {"role": "assis\ntant", "content": "SELECT * \nFROM \"tracks\" \nWHERE LOWER(Name) LIKE '%what%'"}, {"role": "user", "conten\nt": "  \n  Identify artists who have albums with tracks appearing in multiple genres:\n\n\n"}]
```

Add of existing embedding ID: 9a9c970b-b94c-5f22-b54c-b86921a38b65-sql

Insert of existing embedding ID: 9a9c970b-b94c-5f22-b54c-b86921a38b65-sql

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T23:05:17.391736048Z', 'message': {'role': 'assistant', 'content': 'SELECT a.ArtistId, a.Name AS ArtistName\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n  SELECT g2.GenreId\n  FROM "genres" g2\n  JOIN "tracks" t2 ON g2.GenreId = t2.GenreId\n  GROUP BY g2.GenreId\n  HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name'}, 'done_reason': 'stop', 'done': True, 'total_duration': 112573084080, 'load_duration': 784922, 'prompt_eval_count': 1676, 'prompt_eval_duration': 78939042000, 'eval_count': 116, 'eval_duration': 32953725000}
```

```
SELECT a.ArtistId, a.Name AS ArtistName
FROM "artists" a
JOIN "albums" al ON a.ArtistId = al.ArtistId
WHERE a.ArtistId IN (
  SELECT g2.GenreId
  FROM "genres" g2
  JOIN "tracks" t2 ON g2.GenreId = t2.GenreId
  GROUP BY g2.GenreId
  HAVING COUNT(g2.GenreId) > 1
)
GROUP BY a.ArtistId, a.Name
SELECT a.ArtistId, a.Name AS ArtistName
FROM "artists" a
JOIN "albums" al ON a.ArtistId = al.ArtistId
WHERE a.ArtistId IN (
  SELECT g2.GenreId
  FROM "genres" g2
  JOIN "tracks" t2 ON g2.GenreId = t2.GenreId
  GROUP BY g2.GenreId
  HAVING COUNT(g2.GenreId) > 1
)
GROUP BY a.ArtistId, a.Name
```

	ArtistId	ArtistName
0	1	AC/DC
1	2	Accept
2	3	Aerosmith
3	4	Alanis Morissette
4	5	Alice In Chains
5	6	Antônio Carlos Jobim
6	7	Apocalyptica
7	8	Audioslave
8	9	BackBeat
9	10	Billy Cobham
10	11	Black Label Society
11	12	Black Sabbath

12	13	Body Count
13	14	Bruce Dickinson
14	15	Buddy Guy
15	16	Caetano Veloso
16	17	Chico Buarque
17	18	Chico Science & Nação Zumbi
18	19	Cidade Negra
19	20	Cláudio Zoli
20	21	Various Artists
21	22	Led Zeppelin
22	23	Frank Zappa & Captain Beefheart
23	24	Marcos Valle

Ollama parameters:

model=codegemma:latest,

options={},

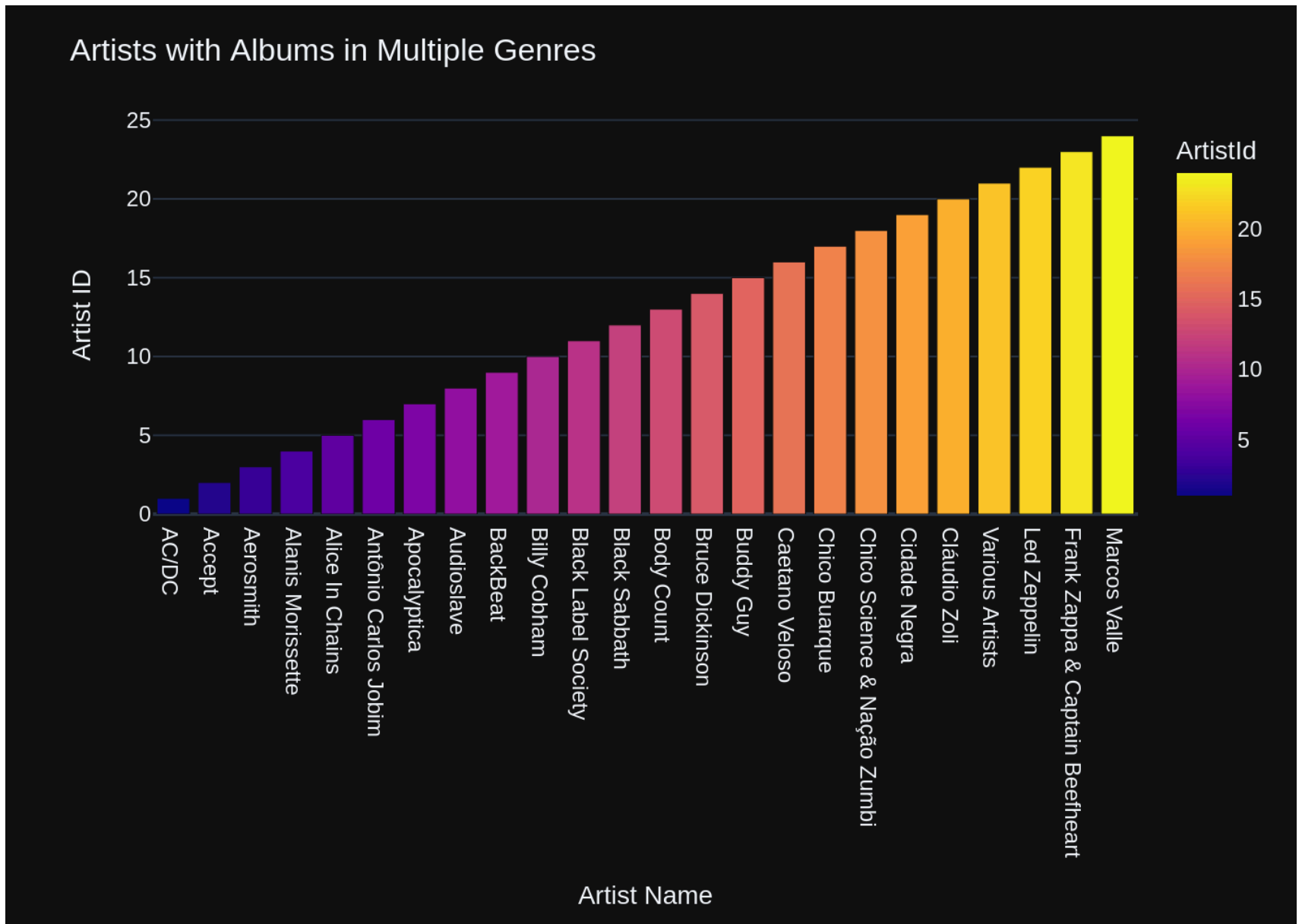
keep_alive=None

Prompt Content:

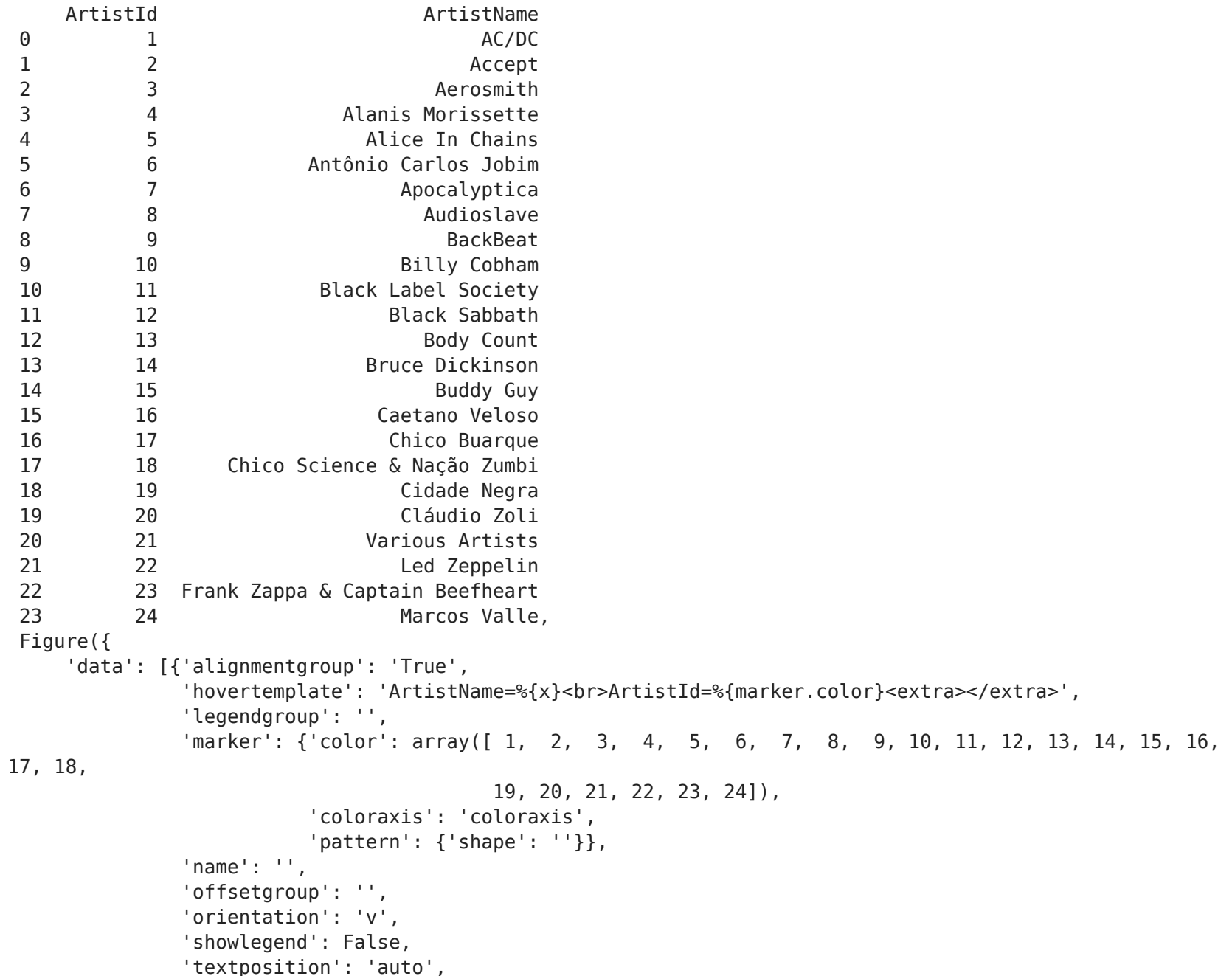
```
[{"role": "system", "content": "The following is a pandas DataFrame that contains the results of the query that answers the question the user asked: ' \n      Identify artists who have albums with tracks appearing in multiple genres:\n\n\n\n\n\nThe DataFrame was produced using this query: SELECT a.ArtistId, a.Name AS ArtistName\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n SELECT g2.GenreId\n FROM \"genres\" g2\n JOIN \"tracks\" t2 ON g2.GenreId = t2.GenreId\n GROUP BY g2.GenreId\n HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\nArtistId      int64\nArtistName    object\nndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the code."}]
```

Ollama Response:

```
{'model': 'codegemma:latest', 'created_at': '2024-06-13T23:05:47.300414838Z', 'message': {'role': 'assistant', 'content': "\n\npython\nimport plotly.express as px\n\nfig = px.bar(df, x='ArtistName', y='ArtistId', color='ArtistId')\n\nfig.update_layout(\n    title='Artists with Albums in Multiple Genres',\n    xaxis_title='Artist Name',\n    yaxis_title='Artist ID'\n)\n\nfig.show()\n\n"}, 'done_reason': 'stop', 'done': True, 'total_duration': 29882475579, 'load_duration': 748829, 'prompt_eval_count': 263, 'prompt_eval_duration': 10711043000, 'eval_count': 79, 'eval_duration': 19038715000}
```



```
Out[40]: ('SELECT a.ArtistId, a.Name AS ArtistName\nFROM "artists" a\nJOIN "albums" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n SELECT g2.GenreId\n FROM "genres" g2\n JOIN "tracks" t2 ON g2.GenreId = t2.GenreId\n GROUP BY g2.GenreId\n HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name',
```



```

        'type': 'bar',
        'x': array(['AC/DC', 'Accept', 'Aerosmith', 'Alanis Morissette', 'Alice In Chains',
                    'Antônio Carlos Jobim', 'Apocalyptica', 'Audioslave', 'BackBeat',
                    'Billy Cobham', 'Black Label Society', 'Black Sabbath', 'Body Count',
                    'Bruce Dickinson', 'Buddy Guy', 'Caetano Veloso', 'Chico Buarque',
                    'Chico Science & Nação Zumbi', 'Cidade Negra', 'Cláudio Zoli',
                    'Various Artists', 'Led Zeppelin', 'Frank Zappa & Captain Beefheart',
                    'Marcos Valle'], dtype=object),
        'xaxis': 'x',
        'y': array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
                    19, 20, 21, 22, 23, 24]),
        'yaxis': 'y'}],
    'layout': {'barmode': 'relative',
               'coloraxis': {'colorbar': {'title': {'text': 'ArtistId'}}},
               'colorscale': [[0.0, '#0d0887'], [0.11111111111111111,
                    '#46039f'], [0.22222222222222222,
                    '#7201a8'], [0.33333333333333333,
                    '#9c179e'], [0.44444444444444444,
                    '#bd3786'], [0.55555555555555556,
                    '#d8576b'], [0.66666666666666666,
                    '#ed7953'], [0.77777777777777778,
                    '#fb9f3a'], [0.88888888888888888,
                    '#fdca26'], [1.0, '#f0f921']]],
               'legend': {'tracegroupgap': 0},
               'margin': {'t': 60},
               'template': '...',
               'title': {'text': 'Artists with Albums in Multiple Genres'},
               'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Artist Name'}},
               'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'Artist ID'}}
    )))

```

Check completion time

In []:

```

In [41]: ts_stop = time()

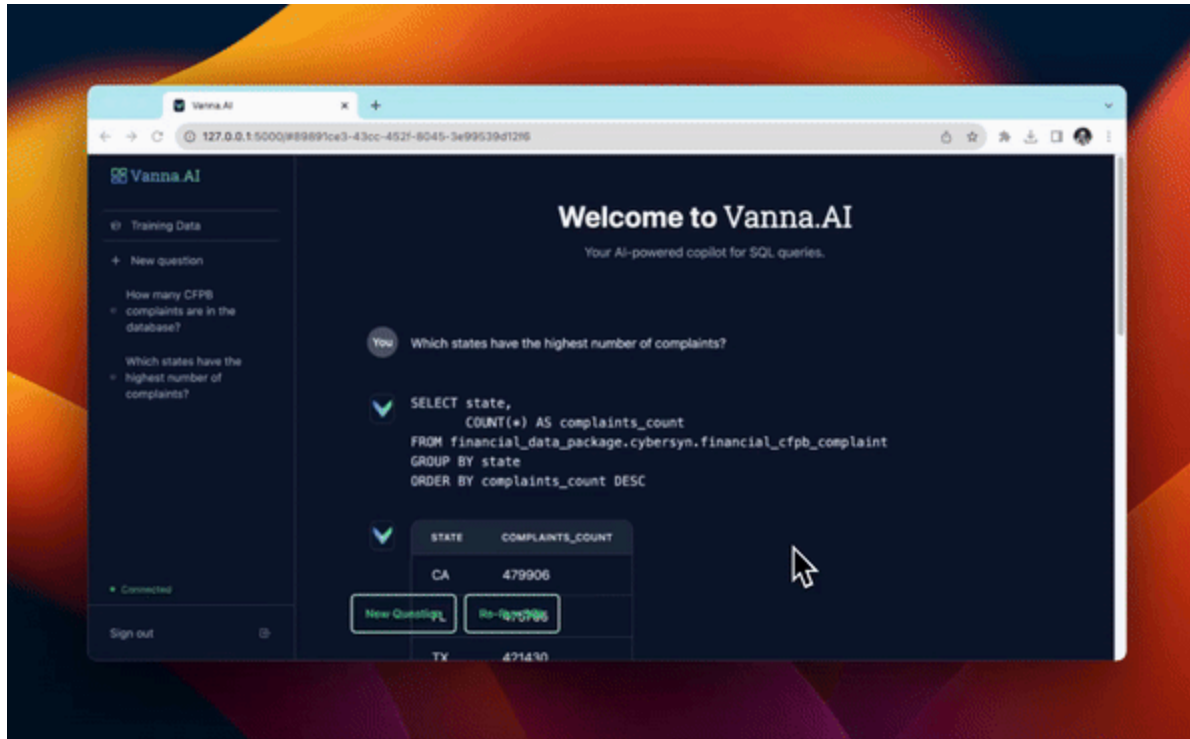
elapsed_time = ts_stop - ts_start
print(f"test running on '{hostname}' with '{model_name}' LLM took : {elapsed_time:.2f} sec")

```

test running on 'ducklover1' with 'codegemma' LLM took : 2572.97 sec

In []:

Launch the User Interface



```
from vanna.flask import VannaFlaskApp app = VannaFlaskApp(vn) app.run()
```

Next Steps

Using Vanna via Jupyter notebooks is great for getting started but check out additional customizable interfaces like the

- [Streamlit app](#)
- [Flask app](#)
- [Slackbot](#)