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.AIs hosted vector database (pgvector) for free. This is usable across machines with no additional setup.
- [Selected] ChromaDB
  Use ChromaDBs 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 = 'gemma'
```

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 invoi...	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 invoi...	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 invoi...	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-4eaef5f9deca-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"\r\n(\r\n ArtistId IN...	ddl
1	0db84e3d-ef41-563c-803e-21c1b985dc19-ddl	None	CREATE TABLE "invoices"\r\n(\r\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"\r\n(\r\n AlbumId INTE...	ddl
7	458debc8-8082-5450-a17a-66028bd55ace-ddl	None	CREATE TABLE "playlists"\r\n(\r\n PlaylistI...	ddl
8	4815f3fd-925b-53ce-9dfa-0e4285d5abd3-ddl	None	CREATE TABLE "invoice_items"\r\n(\r\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\n EmployeeI...	ddl
12	65df0648-bf05-5f75-9365-c21f54b2302d-ddl	None	CREATE TABLE "media_types"\r\n(\r\n MediaTy...	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
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: d8a37163-5ce5-58cd-a316-ea5598d44d27-sql
Add of existing embedding ID: a7185c88-7417-5b75-a52e-4eaef5f9deca-sql
Add of existing embedding ID: 6f22268c-5062-5f11-ba2d-8555f06b409d-sql
Add of existing embedding ID: 49e67df3-a604-51f8-ad01-b8f5a2043eac-sql
Add of existing embedding ID: dd282d7c-a4ef-5e3a-87e0-cb45fac50808-sql
Add of existing embedding ID: aea89953-21b2-55d1-9dda-431ee6033c3d-sql
Add of existing embedding ID: fd25ebba-4066-5a0f-8613-7b1c2ace0339-sql
Add of existing embedding ID: 6bed484b-9a80-57f4-ad89-5f775b5df252-sql
Add of existing embedding ID: f33f8cb6-1b12-5ea7-8d9a-aef8166b9970-sql
Add of existing embedding ID: f626b681-4d8f-563a-beee-1ea759baaa82-sql
Add of existing embedding ID: 127fd4bd-b9af-539d-9313-1d0234d073b7-sql
Add of existing embedding ID: 584873f8-1904-50f1-8f80-7ccf08059264-sql
Add of existing embedding ID: 3013d1b4-feb2-519d-bfb9-114500436e3d-sql
Add of existing embedding ID: d1d70c18-f5d9-5970-a32c-914deeca1087-sql
Add of existing embedding ID: e7c4b3aa-664f-5f87-8b25-449a4482f3fd-sql
Add of existing embedding ID: 9a9c970b-b94c-5f22-b54c-b86921a38b65-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"\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 "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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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 TABLE "media_types"\r\n(\r\n   MediaTypeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   Name NVARCHAR(120)\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 "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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE 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, TrackId),\r\n   FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \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" (ArtistId) \r\n\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 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=gemma: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\"\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 AUTOINCREM
ENT 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
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) REFERENC
ES \"albums\" (AlbumId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) REFER
ENCES \"genres\" (GenreId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (MediaTypeI
d) REFERENCES \"media_types\" (MediaTypeId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE
TABLE \"media_types\"\r\n(\r\n   MediaTypeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   Name NVARC
HAR(120)\r\n)\n\nCREATE TABLE \"artists\"\r\n(\r\n   ArtistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NUL
L,\r\n   Name NVARCHAR(120)\r\n)\n\nCREATE TABLE \"invoice_items\"\r\n(\r\n   InvoiceLineId INTEGER PRIMA

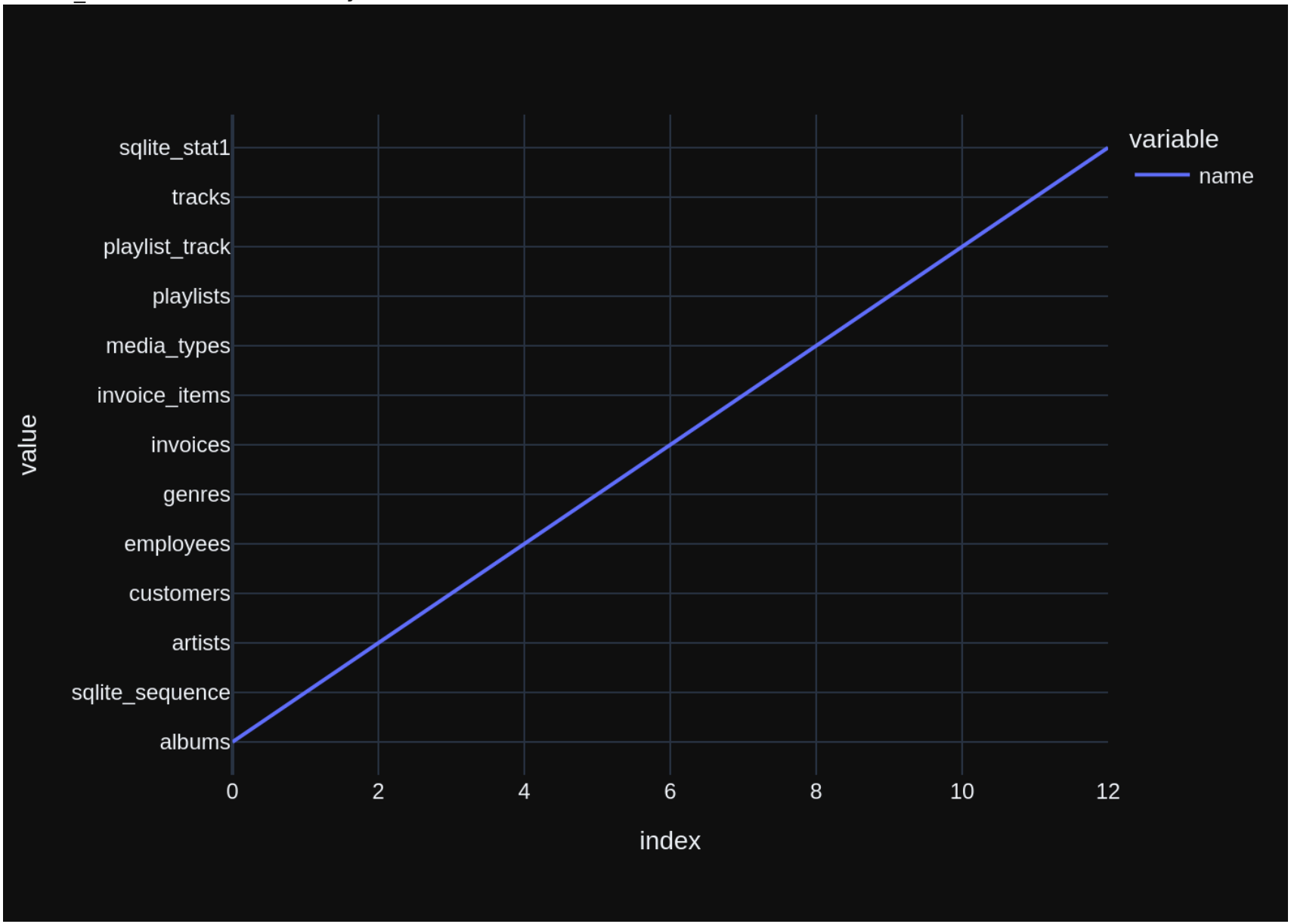
RY KEY AUTOINCREMENT NOT NULL,\r\n   InvoiceId INTEGER NOT NULL,\r\n   TrackId INTEGER NOT NULL,\r\n
UnitPrice NUMERIC(10,2) NOT NULL,\r\n   Quantity INTEGER NOT NULL,\r\n   FOREIGN KEY (InvoiceId) REFERE
NCES \"invoices\" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId)
REFERENCES \"tracks\" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"play
list_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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES
\"tracks\" (TrackId) \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   Arti
stId INTEGER NOT NULL,\r\n   FOREIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistId) \r\n\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.ReportsT
o = mt.EmployeeId"}, {"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   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   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": "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': 'gemma:latest', 'created_at': '2024-06-14T01:43:16.641981421Z', 'message': {'role': 'assistant',
'content': "```sql\nSELECT name FROM sqlite_master WHERE type='table'```"}, 'done_reason': 'stop', 'done':
True, 'total_duration': 61845601716, 'load_duration': 1918383735, 'prompt_eval_count': 1658, 'prompt_eval_d
uration': 56012059000, 'eval_count': 16, 'eval_duration': 3270915000}
```sql
SELECT name FROM sqlite_master WHERE type='table'```
Output from LLM: ```sql
SELECT name FROM sqlite_master WHERE type='table'```
Extracted SQL: 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=gemma: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\ndtyp
e: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the results of t
he dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value in the dataf
rame, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the cod
e."}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T01:43:40.176635623Z', 'message': {'role': 'assistant',
'content': "```python\nimport plotly.graph_objs as go\n\nif df.shape[0] == 1:\n    go.Indicator(go.layout.I
ndicator(title='Tables in Database', value=df['name'].iat[0]))\nelse:\n    go.Bar(x=df['name']).update_trac
```
```

es(marker={'color': 'lightgrey'})\n```"}, 'done_reason': 'stop', 'done': True, 'total_duration': 23508851459, 'load_duration': 605057, 'prompt_eval_count': 149, 'prompt_eval_duration': 5709056000, 'eval_count': 81, 'eval_duration': 17708355000}



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': [{'hovertemplate': 'variable=name<br>index=%{x}<br>value=%{y}<extra></extra>',
              'legendgroup': 'name',
              'line': {'color': '#636efa', 'dash': 'solid'},
              'marker': {'symbol': 'circle'},
              'mode': 'lines',
              'name': 'name',
              'orientation': 'v',
              'showlegend': True,
              'type': 'scatter',
              'x': array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12]),
              'xaxis': 'x',
              'y': array(['albums', 'sqlite_sequence', 'artists', 'customers', 'employees',
                          'genres', 'invoices', 'invoice_items', 'media_types', 'playlists',
                          'playlist_track', 'tracks', 'sqlite_stat1'], dtype=object),
              '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 [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
```

[{'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\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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\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   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(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\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(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 DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE sqlite_sequence(name,seq)\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 sqlite_stat1(tbl,idx,stat)\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\tON DELETE 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, TrackId),\r\n   FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE "media_types"\r\n(\r\n   MediaTypeId 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 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 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 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.Custom
erId\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.CustomerI
d'}, {'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   List all customers from Canada
and their email addresses:\n'}, {'role': 'assistant', 'content': 'SELECT c.Email, c.Country\nFROM "customer
s" c\nWHERE c.Country = \'Canada\''}, {'role': 'user', 'content': ' \n   Get the average invoice total fo
r each customer:\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, AVG(i.Total) AS AverageInvoice
Total\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'rol
e': '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.BillingCountr
y'}, {'role': 'user', 'content': "which table stores customer's orders"}]
Ollama parameters:
model=gemma: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\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\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 TABLE \"customers\"\r\n(\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\tON DELETE NO ACTION ON UPDATE NO A
CTION\r\n)\n\nCREATE TABLE \"employees\"\r\n(\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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)
\n\nCREATE TABLE sqlite_sequence(name,seq)\n\nCREATE TABLE \"playlists\"\r\n(\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(\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\tON DELETE 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, TrackId),\r\n   FOREIGN KEY (PlaylistId) REFERENCES \"playlists\" (PlaylistId)
\r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackI
d) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"media_types\"\r\n(\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   Fin
d 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 tota
l quantity (across all invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.Trac
kId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoi
ce_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 \"in
voice_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\nJO
IN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}, {"role": "user", "content": "wha
t 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 = 'Canad
a'"}, {"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\nJO
IN \"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.Billin
gCountry, COUNT(*) AS TotalInvoices\nFROM \"invoices\" i\nGROUP BY i.BillingCountry"}, {"role": "user", "co
ntent": "which table stores customer's orders"}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T01:45:07.317458045Z', 'message': {'role': 'assistant',
'content': 'The "invoices" table stores customers\' orders. It contains information about each invoice made
by a customer, including the customer ID, invoice date, billing address, total amount, and foreign key refe
rences to the customer and invoice items.'}, 'done_reason': 'stop', 'done': True, 'total_duration': 8669843
2705, 'load_duration': 631466, 'prompt_eval_count': 1846, 'prompt_eval_duration': 74529857000, 'eval_coun
t': 47, 'eval_duration': 11532965000}
The "invoices" table stores customers' orders. It contains information about each invoice made by a custome
r, including the customer ID, invoice date, billing address, total amount, and foreign key references to th
e customer and invoice items.
The "invoices" table stores customers' orders. It contains information about each invoice made by a custome
r, including the customer ID, invoice date, billing address, total amount, and foreign key references to th
e customer and invoice items.
Couldn't run sql:  Execution failed on sql 'The "invoices" table stores customers' orders. It contains info
rmation about each invoice made by a customer, including the customer ID, invoice date, billing address, to
tal amount, and foreign key references to the customer and invoice items.': 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
```

[{'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\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON "customers" (SupportRepId)\n\nCREATE TABLE "customers"\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(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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON "invoices" (CustomerId)\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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON "invoice_items" (InvoiceId)\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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\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(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 DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE "playlists"\r\n(\r\n   PlaylistId 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 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': 'How many customers are there'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM "customers"'}, {'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 deta

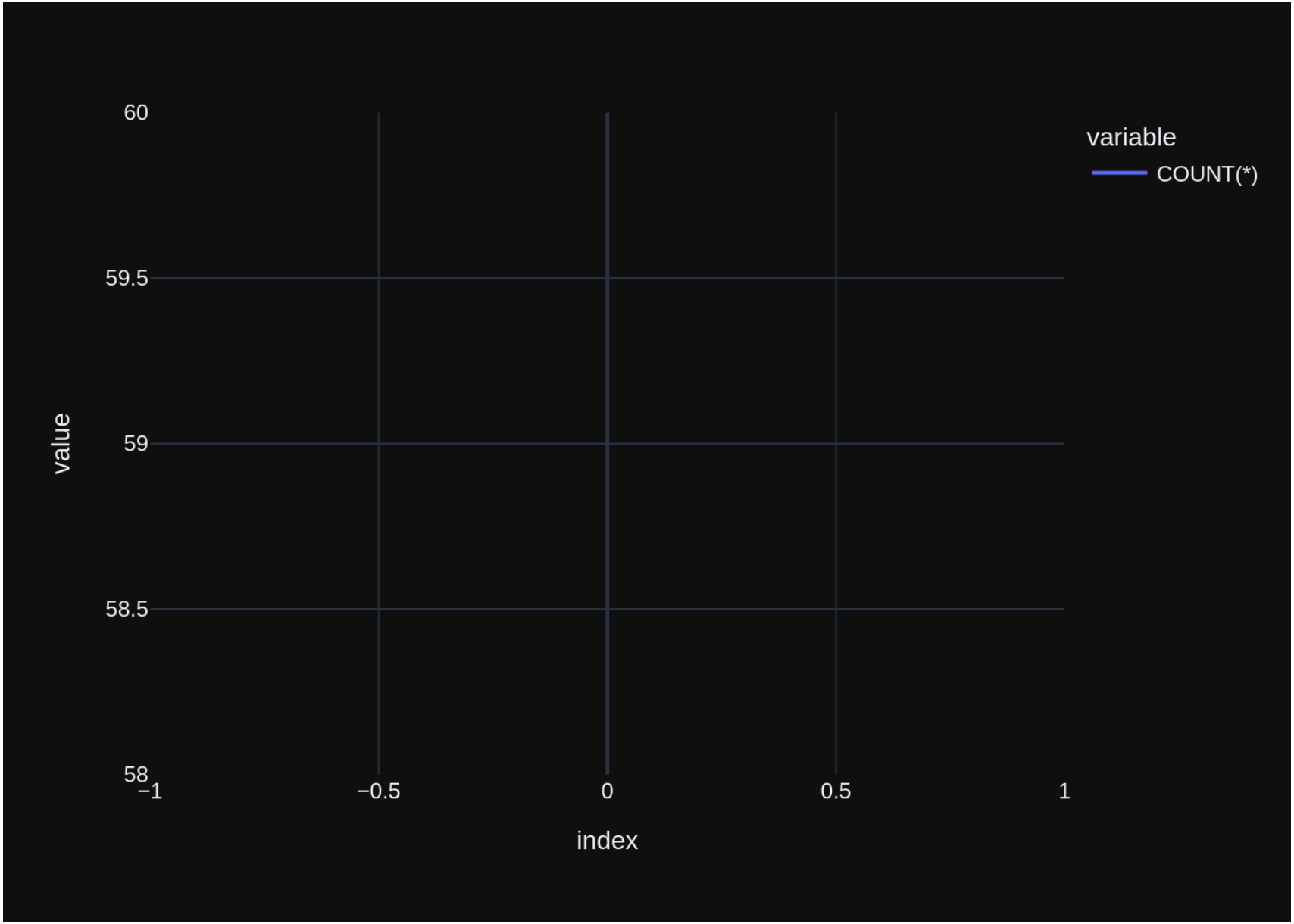
il table is unnecessary \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(i.Total) AS TotalS
pent\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY
TotalSpent DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n   Find the customer who bought the most 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   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'}, {'r
ole': '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\n
ORDER 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 al
l invoices):\n'}, {'role': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAl
bums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDE
R BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n   Get the average invoice total for ea
ch customer:\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTota
l\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', 'cont
ent': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "invoices" i\nGROUP BY i.BillingCountry'},
{'role': 'user', 'content': 'How many customers are there'}]
Ollama parameters:
model=gemma: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\tON DELETE NO ACTION ON UPDATE NO ACTION
\r\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON \"customers\" (SupportRepId)\n\nCREATE TABLE \"customers
\"\r\n(\r\n   CustomerId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   FirstName NVARCHAR(40) NOT NU
LL,\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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON \"invoices\" (CustomerId)\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\" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON \"invoice_items\" (InvoiceId)\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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON \"invoice_items\" (TrackId)\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(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 DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"playlists\"\r\n(\r\n   PlaylistId 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 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": "How many customers are there"}, {"role": "assistant", "content": "SELECT COUNT(*) FROM \"customers\""}, {"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   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 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", "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"}, {"ro
le": "user", "content": " \n   Get the average invoice total for each customer:\n"}, {"role": "assistan
t", "content": "SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM \"customers\" c\nJOIN \"invo
ices\" 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, C
OUNT(*) AS TotalInvoices\nFROM \"invoices\" i\nGROUP BY i.BillingCountry"}, {"role": "user", "content": "Ho
w 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': 'gemma:latest', 'created_at': '2024-06-14T01:46:19.636027514Z', 'message': {'role': 'assistant',
'content': 'SELECT COUNT(*) FROM "customers"'}, 'done_reason': 'stop', 'done': True, 'total_duration': 7227
7371005, 'load_duration': 794034, 'prompt_eval_count': 1681, 'prompt_eval_duration': 69852826000, 'eval_cou
nt': 8, 'eval_duration': 1780277000}
SELECT COUNT(*) FROM "customers"
SELECT COUNT(*) FROM "customers"
   COUNT(*)
0        59
Ollama parameters:
model=gemma: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 usin
g this query: SELECT COUNT(*) FROM \"customers\"\n\nThe following is information about the resulting pandas
DataFrame 'df': \nRunning df.dtypes gives:\n COUNT(*)    int64\ndtype: object"}, {"role": "user", "conten
t": "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 o
nly Python code. Do not answer with any explanations -- just the code."}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T01:46:42.304211296Z', 'message': {'role': 'assistant',
'content': "```python\nimport plotly.express as px\n\nif df.shape[0] == 1:\n    px.indicator(df['COUNT
(*)']).update_traces(title='Customers', mode='number')\nelse:\n    px.bar(df, x='COUNT(*)'), update_traces
(title='Customers')\n```"}, 'done_reason': 'stop', 'done': True, 'total_duration': 22642634374, 'load_durat
ion': 663834, 'prompt_eval_count': 117, 'prompt_eval_duration': 4920836000, 'eval_count': 71, 'eval_duratio
n': 17631591000}
```


60
59.5
59
58.5
58
value
−1
−0.5
0
0.5
1
index
variable
COUNT(*)

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
```

[{'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\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   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(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\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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE "media_types"\r\n(\r\n   MediaTypeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   Name NVARCHAR(120)\r\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON "customers" (SupportRepId)\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(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 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\tON DELETE 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, TrackId),\r\n   FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE sqlite_sequence(name,seq)\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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\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 qu

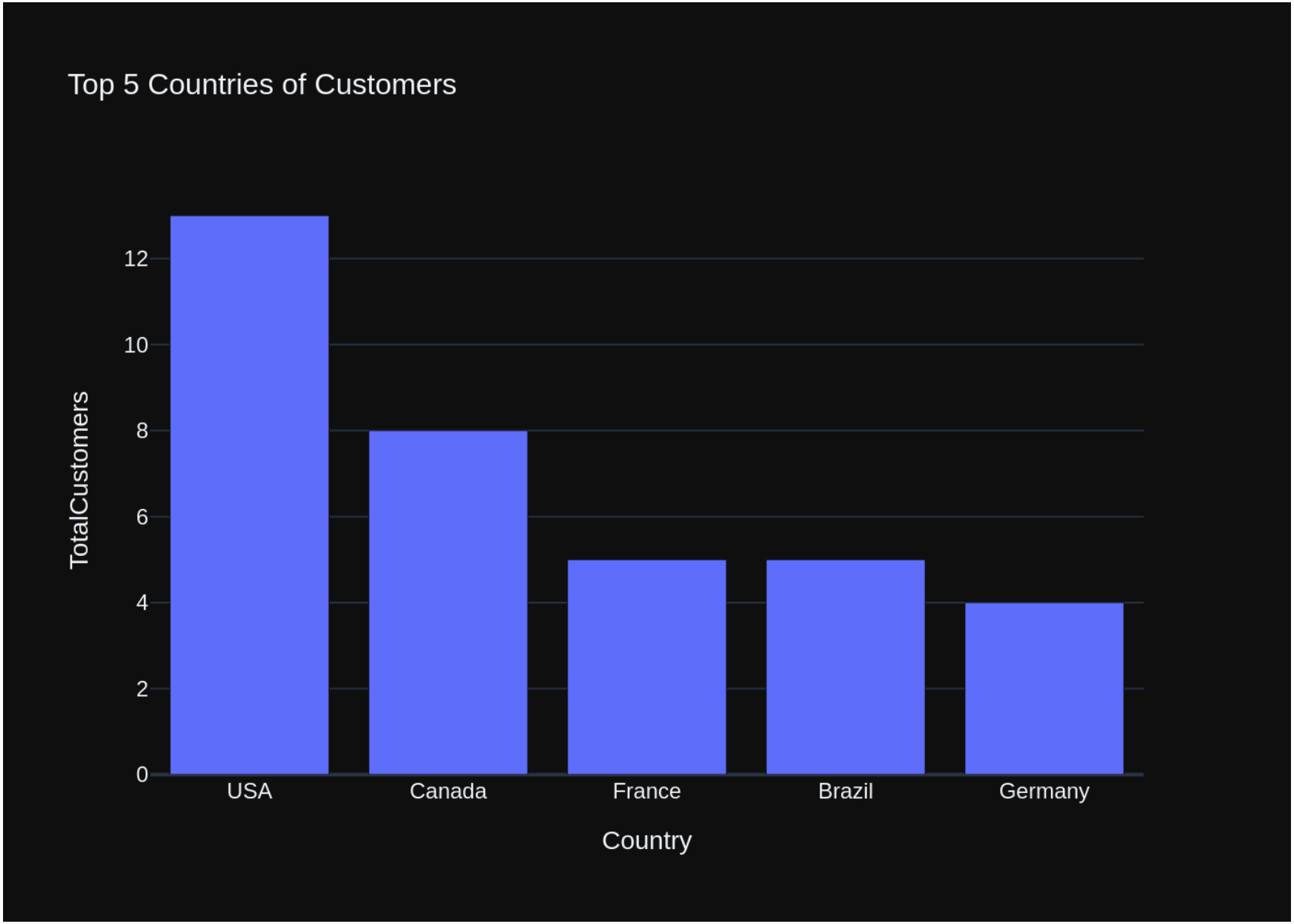
ery with a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it can\'t be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the answer exactly as it was given before. \n'}, {'role': 'user', 'content': '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 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?'}]
Ollama parameters:
model=gemma: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\"\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   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\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
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(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\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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   F
OREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n
\nCREATE TABLE \"media_types\"\r\n(\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(\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\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\tO
N DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"playlist_track\"\r\n(\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\tON DELETE NO ACT
ION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\tON DELETE NO
ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE sqlite_sequence(name,seq)\n\nCREATE TABLE \"tracks\"\r\n(\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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n
FOREIGN KEY (GenreId) 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\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': 'gemma:latest', 'created_at': '2024-06-14T01:48:05.223259061Z', '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': 82791509950, 'load_d
uration': 786017, 'prompt_eval_count': 1776, 'prompt_eval_duration': 73986003000, 'eval_count': 33, 'eval_d
uration': 8182578000}
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=gemma: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 D
ataFrame was produced using this query: SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM \"customers\" c
\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5\n\nThe following is information about the resul
ting pandas DataFrame 'df': \nRunning df.dtypes gives:\n Country           object\nTotalCustomers     int64
\ndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the result
s 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': 'gemma:latest', 'created_at': '2024-06-14T01:48:27.645489335Z', 'message': {'role': 'assistant',
'content': "```python\nimport plotly.express as px\n\nfig = px.bar(df, x='Country', y='TotalCustomers', tit
le='Top 5 Countries of Customers')\nfig.update_layout(title_font_size=16)\nfig.show()\n```"}, 'done_reaso
n': 'stop', 'done': True, 'total_duration': 22396705296, 'load_duration': 1006975, 'prompt_eval_count': 17
4, 'prompt_eval_duration': 7255732000, 'eval_count': 60, 'eval_duration': 15050146000}
```


Top 5 Countries of Customers
TotalCustomers
0
2
4
6
8
10
12
USA
Canada
France
Brazil
Germany
Country

Out[20]:
```
('SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM "customers" c\nGROUP BY c.Country\nORDER BY TotalCust
omers 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': {'font': {'size': 16}, '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"\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\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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_TrackAlbumId ON "tracks" (AlbumId)\n\nCREATE TABLE "artists"\r\n(\r\n   ArtistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   Name NVARCHAR(120)\r\n)\n\nCREATE INDEX IFK_TrackGenreId ON "tracks" (GenreId)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON "playlist_track" (TrackId)\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 INDEX IFK_TrackMediaTypeId ON "tracks" (MediaTypeId)\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 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 "invoic
e_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'r
ole': '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_i
tems" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'rol
e': 'user', 'content': ' \n   List all genres and the number of tracks in each genre:\n'}, {'role': 'assi
stant', '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   Get all playlists containing a
t 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 "p
laylists" p ON pt.PlaylistId = p.PlaylistId\nJOIN "tracks" t ON pt.TrackId = t.TrackId\nGROUP BY pt.Playlis
tId, p.Name\nHAVING COUNT(pt.TrackId) >= 10'}, {'role': 'user', 'content': ' \n   Find the top 5 most exp
ensive tracks (based on unit price):\n'}, {'role': 'assistant', 'content': 'SELECT t.TrackId, t.Name, t.Uni
tPrice\nFROM "tracks" t\nORDER BY t.UnitPrice DESC\nLIMIT 5'}, {'role': 'user', 'content': 'Can you list al
l tables in the SQLite database catalog?'}, {'role': 'assistant', 'content': "SELECT name FROM sqlite_maste
r WHERE type='table'"}, {'role': 'user', 'content': ' \n   List all albums and their corresponding artist
names \n'}]
Ollama parameters:
model=gemma: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_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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"tracks\"\r\n(\r\n   TrackId INTEGER P
RIMARY KEY AUTOINCREMENT NOT NULL,\r\n   Name NVARCHAR(200) NOT NULL,\r\n   AlbumId INTEGER,\r\n   Medi
aTypeId INTEGER NOT NULL,\r\n   GenreId INTEGER,\r\n   Composer NVARCHAR(220),\r\n   Milliseconds INTEG
ER 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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreI
d) REFERENCES \"genres\" (GenreId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (Med
iaTypeId) REFERENCES \"media_types\" (MediaTypeId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\n
CREATE INDEX IFK_TrackAlbumId ON \"tracks\" (AlbumId)\n\nCREATE TABLE \"artists\"\r\n(\r\n   ArtistId INTE
GER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   Name NVARCHAR(120)\r\n)\n\nCREATE INDEX IFK_TrackGenreId ON
\"tracks\" (GenreId)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON \"playlist_track\" (TrackId)\n\nCREATE TAB
LE \"playlists\"\r\n(\r\n   PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   Name NVARCHAR(12
0)\r\n)\n\nCREATE TABLE \"genres\"\r\n(\r\n   GenreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   N
ame NVARCHAR(120)\r\n)\n\nCREATE INDEX IFK_TrackMediaTypeId ON \"tracks\" (MediaTypeId)\n\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"}]

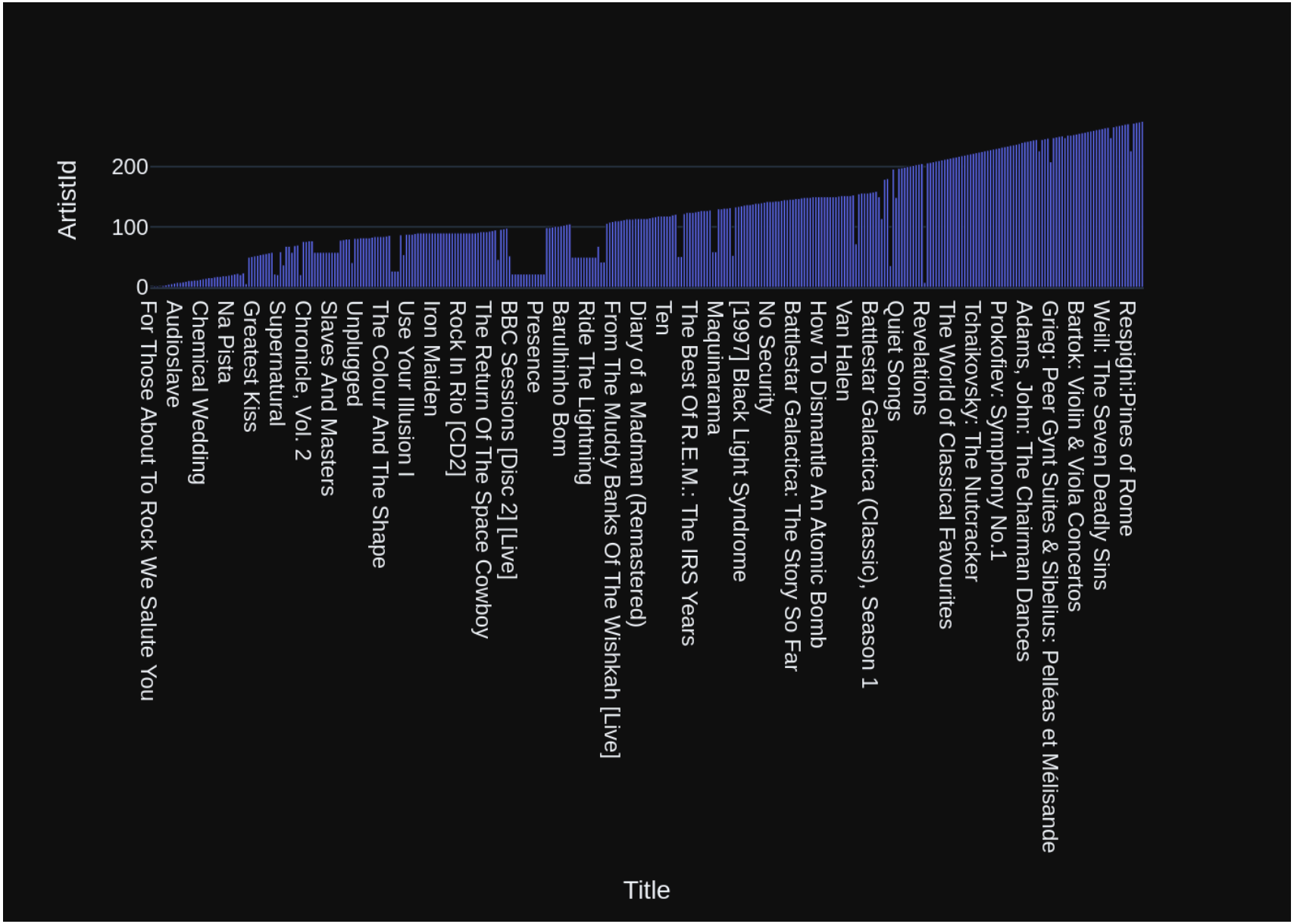
```
Add of existing embedding ID: a7185c88-7417-5b75-a52e-4eaef5f9deca-sql
Insert of existing embedding ID: a7185c88-7417-5b75-a52e-4eaef5f9deca-sql
```

```
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T01:49:46.213561936Z', 'message': {'role': 'assistant',
'content': 'SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM "albums" a\nJOIN "artists" ar ON a.Arti
stId = ar.ArtistId'}, 'done_reason': 'stop', 'done': True, 'total_duration': 78471621697, 'load_duration':
721745, 'prompt_eval_count': 1586, 'prompt_eval_duration': 67707829000, 'eval_count': 39, 'eval_duration':
10119021000}
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=gemma: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
\"albums\" a\nJOIN \"artists\" ar ON a.ArtistId = ar.ArtistId\n\nThe following is information about the res
ulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n Title         object\nArtistId      int64\nArti
stName    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 explana
tions -- just the code."}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T01:50:09.572452981Z', 'message': {'role': 'assistant',
'content': "```python\nimport plotly.express as px\n\nfig = px.bar(df, x='Title', y='ArtistName', color='Ar
tistName', title='Albums and Artists')\nfig.update_traces(hovermode='closest')\nfig.show()\n```"}, 'done_re
ason': 'stop', 'done': True, 'total_duration': 23330272367, 'load_duration': 1134632, 'prompt_eval_count':
187, 'prompt_eval_duration': 8077654000, 'eval_count': 58, 'eval_duration': 15156355000}
```


ArtistId
200
100
0
Title
Respighi:Pines of Rome
Weill: The Seven Deadly Sins
Bartok: Violin & Viola Concertos
Grieg: Peer Gynt Suites & Sibelius: Pelléas et Mélisande
Adams, John: The Chairman Dances
Prokofiev: Symphony No.1
Tchaikovsky: The Nutcracker
The World of Classical Favourites
Revelations
Quiet Songs
Battlestar Galactica (Classic), Season 1
Van Halen
How To Dismantle An Atomic Bomb
Battlestar Galactica: The Story So Far
No Security
[1997] Black Light Syndrome
Maquinarama
The Best Of R.E.M.: The IRS Years
Ten
Diary of a Madman (Remastered)
From The Muddy Banks Of The Wishkah [Live]
Ride The Lightning
Barulhinho Bom
Presence
BBC Sessions [Disc 2] [Live]
The Return Of The Space Cowboy
Rock In Rio [CD2]
Iron Maiden
Use Your Illusion I
The Colour And The Shape
Unplugged
Slaves And Masters
Chronicle, Vol. 2
Supernatural
Greatest Kiss
Na Pista
Chemical Wedding
Audioslave
For Those About To Rock We Salute You

Out[21]:
```
('SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM "albums" a\nJOIN "artists" ar ON a.ArtistId = a
r.ArtistId',
                                                 Title  ArtistId  \
 0                For 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>ArtistId=%{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([  1,   2,   2, ..., 273, 274, 275]),
                  'yaxis': 'y'}],
      'layout': {'barmode': 'relative',
                 'legend': {'tracegroupgap': 0},
                 'margin': {'t': 60},
                 'template': '...',
                 'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Title'}},
                 'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'ArtistId'}}}
 }))
```

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"\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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_TrackAlbumId ON "tracks" (AlbumId)\n\nCREATE INDEX IFK_TrackMediaTypeId ON "tracks" (MediaTypeId)\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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\n\nCREATE INDEX IFK_AlbumArtistId ON "albums" (ArtistId)\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\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

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', 'conten
t': ' \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\n
GROUP 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.Arti
stId, ar.Name AS ArtistName\nFROM "albums" a\nJOIN "artists" ar ON a.ArtistId = ar.ArtistId'}, {'role': 'us
er', 'content': ' \n   Hint: album quantity is found in invoice_items, \n   \n   Find the top 5 custome
rs 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 5'}, {'role': 'user', 'content': ' \n   Hint: album quantity is found in invoi
ce_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 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 "W
hat" (case-insensitive)\n'}]
Ollama parameters:
model=gemma: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 INDEX IFK_
PlaylistTrackTrackId ON \"playlist_track\" (TrackId)\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   Med
iaTypeId INTEGER NOT NULL,\r\n   GenreId INTEGER,\r\n   Composer NVARCHAR(220),\r\n   Milliseconds INTE
GER 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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreI
d) REFERENCES \"genres\" (GenreId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (Med
iaTypeId) REFERENCES \"media_types\" (MediaTypeId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\n
CREATE INDEX IFK_TrackAlbumId ON \"tracks\" (AlbumId)\n\nCREATE INDEX IFK_TrackMediaTypeId ON \"tracks\" (M
ediaTypeId)\n\nCREATE TABLE \"playlist_track\"\r\n(\r\n   PlaylistId INTEGER NOT NULL,\r\n   TrackId INT
EGER NOT NULL,\r\n   CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\r\n   FOREIGN KEY
(PlaylistId) REFERENCES \"playlists\" (PlaylistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n
FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)
\n\nCREATE INDEX IFK_InvoiceLineTrackId ON \"invoice_items\" (TrackId)\n\nCREATE INDEX IFK_AlbumArtistId ON
\"albums\" (ArtistId)\n\nCREATE TABLE \"playlists\"\r\n(\r\n   PlaylistId INTEGER PRIMARY KEY AUTOINCREMEN
T NOT NULL,\r\n   Name NVARCHAR(120)\r\n)\n\nCREATE TABLE \"genres\"\r\n(\r\n   GenreId INTEGER PRIMARY K

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===Response Guidelines \n1. If the provided context is sufficient, please gener ate a valid SQL query without any explanations for the question. \n2. If the provided context is almost suf ficient 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 us e the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the an swer 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\" \n WHERE LOWER(Name) LIKE '%what%'"}, {"role": "user", "content": " \n   Get all playlists containing at le ast 10 tracks and the total duration of those tracks:\n"}, {"role": "assistant", "content": "SELECT pt.Play listId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM \"playlist_track\" pt\nJOIN \"pl aylists\" p ON pt.PlaylistId = p.PlaylistId\nJOIN \"tracks\" t ON pt.TrackId = t.TrackId\nGROUP BY pt.Playl istId, 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.A rtistId, 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", "co ntent": " \n   List all genres and the number of tracks in each genre:\n"}, {"role": "assistant", "conten t": "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 (base d on unit price):\n"}, {"role": "assistant", "content": "SELECT t.TrackId, t.Name, t.UnitPrice\nFROM \"trac ks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5"}, {"role": "user", "content": " \n   There are 3 tables: artis ts, albums and tracks, where albums and artists are linked by ArtistId, albums and tracks are linked by Alb umId,\n   Can you find the top 10 most popular artists based on the number of tracks\n"}, {"role": "assist ant", "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 TotalTrack s DESC\nLIMIT 10"}, {"role": "user", "content": " \n   Find the customer who bought the most albums in t otal 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 \"i nvoice_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"}, {"rol e": "assistant", "content": "SELECT a.Title, a.ArtistId, ar.Name AS ArtistName\nFROM \"albums\" a\nJOIN \"a rtists\" ar ON a.ArtistId = ar.ArtistId"}, {"role": "user", "content": " \n   Hint: album quantity is fou nd in invoice_items, \n   \n   Find the top 5 customers who bought the most albums in total quantity (acr oss all invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAl bums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii O N 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 wh o bought the most albums in total quantity (across all invoices):\n"}, {"role": "assistant", "content": "SE LECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii O

```
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': 'gemma:latest', 'created_at': '2024-06-14T01:51:27.520356395Z', 'message': {'role': 'assistant',
'content': 'SELECT * \nFROM "tracks" \nWHERE LOWER(Name) LIKE \'%what%\''}, 'done_reason': 'stop', 'done':
True, 'total_duration': 77566997593, 'load_duration': 696071, 'prompt_eval_count': 1627, 'prompt_eval_durat
ion': 71856214000, 'eval_count': 20, 'eval_duration': 5073473000}
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=gemma: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-in
sensitive)\n'\n\nThe DataFrame was produced using this query: SELECT * \nFROM \"tracks\" \nWHERE LOWER(Nam
e) LIKE '%what%'\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dt
ypes gives:\n TrackId           int64\nName             object\nAlbumId           int64\nMediaTypeId
int64\nGenreId           int64\nComposer         object\nMilliseconds      int64\nBytes             int64\n
UnitPrice       float64\ndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly c
ode to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there i
s 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': 'gemma:latest', 'created_at': '2024-06-14T01:51:52.393866587Z', 'message': {'role': 'assistant',
'content': '```python\nimport plotly.express as px\n\nif df.shape[0] == 1:\n    px.indicator(df[\'Name\']
[0], label=\'Track Name\')\n\nelse:\n    px.bar(df, x=\'Name\', title=\'Tracks with "What" in Name\')\n``
`'}, 'done_reason': 'stop', 'done': True, 'total_duration': 24846398154, 'load_duration': 1555130, 'prompt_
eval_count': 212, 'prompt_eval_duration': 8626775000, 'eval_count': 66, 'eval_duration': 16127518000}
Couldn't run plotly code:  'NoneType' object has no attribute 'show'
Traceback (most recent call last):
  File "/home/gongai/anaconda3/envs/vanna/lib/python3.11/site-packages/vanna/base/base.py", line 1683, in a
sk
    img_bytes = fig.to_image(format="png", scale=2)
                ^^^^^^^^^^^^
AttributeError: 'NoneType' object has no attribute 'to_image'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "/home/gongai/anaconda3/envs/vanna/lib/python3.11/site-packages/vanna/base/base.py", line 1686, in a
sk
    fig.show()
    ^^^^^^^^
AttributeError: 'NoneType' object has no attribute 'show'
```

In [23]:
```python
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
```

[{'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\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON "invoices" (CustomerId)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON "invoice_items" (InvoiceId)\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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\n\nCREATE TABLE "customers"\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(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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON "customers" (SupportRepId)\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(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 DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_EmployeeReportsTo ON "employees" (ReportsTo)\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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\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': 'use

r', 'content': ' \n   Find the customer with the most invoices \n'}, {'role': 'assistant', 'content': 'S
ELECT 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   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 invoic
es per country:\n'}, {'role': 'assistant', 'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\n
FROM "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': 'SEL
ECT 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 TotalA
lbums DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n   Find all invoices since 2010 and the total amou
nt 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': '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 Tota
lAlbums 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 Tota
lAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n   List all invoices with a total exceeding $1
0:\n'}, {'role': 'assistant', 'content': 'SELECT * \nFROM "invoices" \nWHERE Total > 10.00'}, {'role': 'use
r', 'content': ' \n   Get the total number of invoices for each customer\n'}]
Ollama parameters:
model=gemma: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\tON DELETE NO ACTION ON UPDATE NO ACTION
\r\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON \"invoices\" (CustomerId)\n\nCREATE INDEX IFK_InvoiceLineInv

oiceId ON \"invoice_items\" (InvoiceId)\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\" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (Tr
ackId) REFERENCES \"tracks\" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX
IFK_InvoiceLineTrackId ON \"invoice_items\" (TrackId)\n\nCREATE TABLE \"customers\"\r\n(\r\n   CustomerId
INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   FirstName NVARCHAR(40) NOT NULL,\r\n   LastName NVARCH
AR(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\tON DELETE NO ACTION ON UPDATE NO ACTION
\r\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON \"customers\" (SupportRepId)\n\nCREATE TABLE \"employees
\"\r\n(\r\n   EmployeeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   LastName NVARCHAR(20) NOT NUL
L,\r\n   FirstName NVARCHAR(20) NOT NULL,\r\n   Title NVARCHAR(30),\r\n   ReportsTo INTEGER,\r\n   Bir
thDate DATETIME,\r\n   HireDate DATETIME,\r\n   Address NVARCHAR(70),\r\n   City NVARCHAR(40),\r\n   St
ate 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 DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_EmployeeReportsTo ON
\"employees\" (ReportsTo)\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 NUL
L,\r\n   GenreId INTEGER,\r\n   Composer NVARCHAR(220),\r\n   Milliseconds INTEGER NOT NULL,\r\n   Byt
es INTEGER,\r\n   UnitPrice NUMERIC(10,2) NOT NULL,\r\n   FOREIGN KEY (AlbumId) REFERENCES \"albums\" (A
lbumId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) REFERENCES \"genres\"
(GenreId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (MediaTypeId) REFERENCES \"me
dia_types\" (MediaTypeId) \r\n\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 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   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.Customer
Id\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 \"cu
stomers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoi
ces DESC\nLIMIT 1"}, {"role": "user", "content": " \n   Get the average invoice total for each custome
r:\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": "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"},

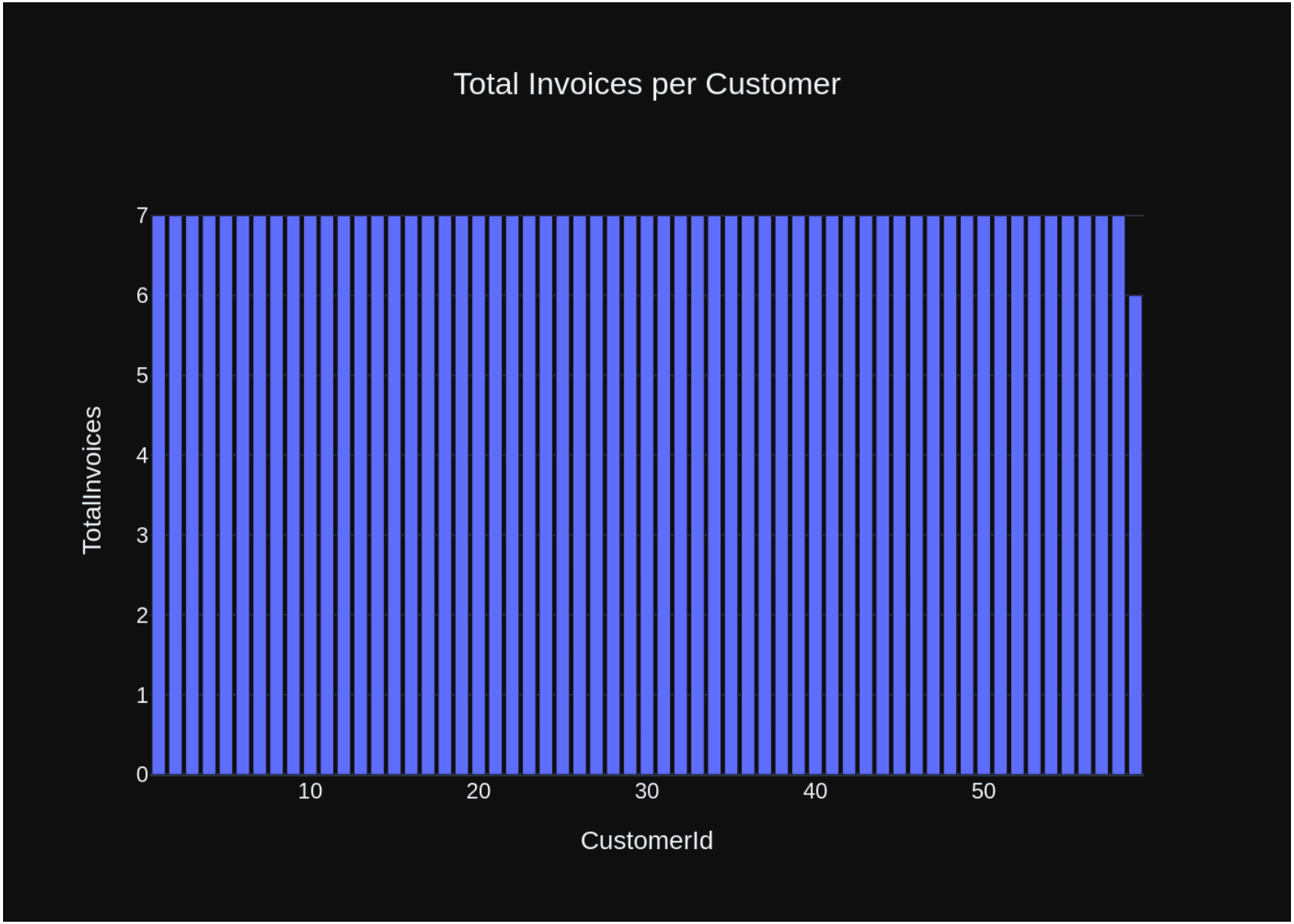
{"role": "user", "content": " \n   Find the customer who bought the most albums in total quantity (acros
s all invoices): \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlb
ums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" 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 u
sing 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 quan
tity is found in invoice_items, \n   \n   Find the top 5 customers who bought the most albums in total qu
antity (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_it
ems\" 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", "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"}, {"ro
le": "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': 'gemma:latest', 'created_at': '2024-06-14T01:53:21.653605262Z', '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_durati
on': 89201764037, 'load_duration': 792168, 'prompt_eval_count': 1872, 'prompt_eval_duration': 77895291000,
'eval_count': 42, 'eval_duration': 10667807000}
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=gemma: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\nFR
OM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\n\nThe follow
ing is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n CustomerId
```

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': 'gemma:latest', 'created_at': '2024-06-14T01:53:51.768281863Z', 'message': {'role': 'assistant', 'content': "```python\nimport plotly.express as px\n\nfig = px.bar(df, x='CustomerId', y='TotalInvoices', title='Total Invoices per Customer')\nfig.update_layout(title_x=0.5)\n\nif df.shape[0] == 1:\n   fig = px.indicator(text='No customers found with invoices.')\n\nfig.show()\n```"}, 'done_reason': 'stop', 'done': True, 'total_duration': 30088415068, 'load_duration': 693177, 'prompt_eval_count': 188, 'prompt_eval_duration': 7946501000, 'eval_count': 86, 'eval_duration': 22046103000}


Total Invoices per Customer
TotalInvoices
0
1
2
3
4
5
6
7
10
20
30
40
50
CustomerId

Out[23]: ('SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.Cus
tomerId = 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, 6]),
              'yaxis': 'y'}],
    'layout': {'barmode': 'relative',
               'legend': {'tracegroupgap': 0},
```

```
                'template': '...',
                'title': {'text': 'Total Invoices per Customer', 'x': 0.5},
                '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
```

[{'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\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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON "invoices" (CustomerId)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON "invoice_items" (InvoiceId)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\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(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 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   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(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\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\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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 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 total number of invoices per countr

y:\n'}, {'role': 'assistant', 'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "invoice
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=gemma: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

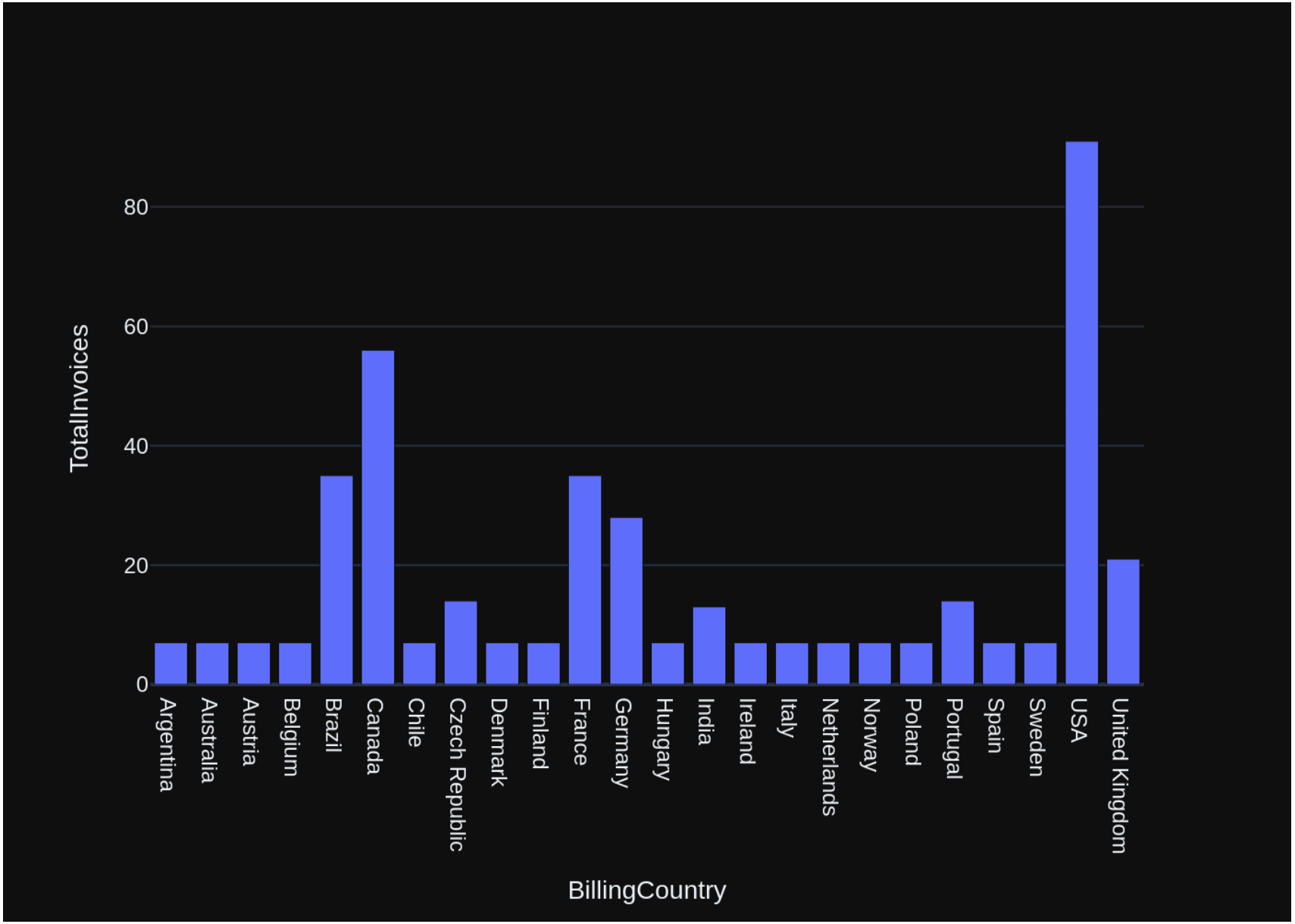
IGN KEY (CustomerId) REFERENCES \"customers\" (CustomerId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION
\r\n)\n\nCREATE TABLE \"invoice_items\"\r\n(\r\n   InvoiceLineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NUL
L,\r\n   InvoiceId INTEGER NOT NULL,\r\n   TrackId INTEGER NOT NULL,\r\n   UnitPrice NUMERIC(10,2) NO
T NULL,\r\n   Quantity INTEGER NOT NULL,\r\n   FOREIGN KEY (InvoiceId) REFERENCES \"invoices\" (InvoiceI
d) \r\n\t\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", "conte
nt": " \n   Find the customer with the most invoices \n"}, {"role": "assistant", "content": "SELECT c.Cu
stomerId, 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 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   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 al
l invoices):\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\n
FROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.In
voiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "cont
ent": " \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 \"invo
ices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5"}, {"rol
e": "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\nF
ROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.Inv
oiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1"}, {"role": "user", "conte
nt": " \n   Get the average invoice total for each customer:\n"}, {"role": "assistant", "content": "SELEC
T c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.Custome
rId = 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': 'gemma:latest', 'created_at': '2024-06-14T01:55:21.838316376Z', 'message': {'role': 'assistant',
'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "invoices" i\nGROUP BY i.BillingCountr
y'}, 'done_reason': 'stop', 'done': True, 'total_duration': 89970392045, 'load_duration': 969859, 'prompt_e
val_count': 1926, 'prompt_eval_duration': 82916356000, 'eval_count': 25, 'eval_duration': 6403318000}
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=gemma: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': 'gemma:latest', 'created_at': '2024-06-14T01:55:53.937467318Z', 'message': {'role': 'assistant', 'content': "```python\nimport plotly.express as px\n\nfig = px.bar(df, x='BillingCountry', y='TotalInvoices')\nfig.update_traces(marker={'color': 'auto'})\nfig.update_layout(title='Total Invoices per Country')\n\nif df.shape[0] == 1:\n   fig = px.indicator(df['TotalInvoices'][0], label='Total Invoices')\n```"}, 'done_reason': 'stop', 'done': True, 'total_duration': 32079573129, 'load_duration': 671617, 'prompt_eval_count': 169, 'prompt_eval_duration': 7416137000, 'eval_count': 93, 'eval_duration': 24570845000}


TotalInvoices
0
20
40
60
80
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
BillingCountry

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},
                  'margin': {'t': 60},
                  'template': '...',
                  '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"\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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON "invoice_items" (InvoiceId)\n\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\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\n\nCREATE INDEX IFK_InvoiceCustomerId ON "invoices" (CustomerId)\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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\nCREATE TABLE "customers"\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(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\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(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 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 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', 'conten

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 t
he most money overall, \n   \n   Hint: order total can be found on invoices table, calculation using in
voice_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.Cust
omerId\nORDER BY TotalSpent DESC\nLIMIT 5'}, {'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.CustomerI
d\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.Invoic
eId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'conten
t': ' \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\n
GROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n   Get the t
otal 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'}, {'r
ole': '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\nLIM
IT 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 custome
r:\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM "c
ustomers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'c
ontent': ' \n   List all invoices with a total exceeding $10:\n'}]
Ollama parameters:
model=gemma: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 \"invoice_items\"\r\n(\r\n   InvoiceLineId INTEGER PRIMARY K
EY AUTOINCREMENT NOT NULL,\r\n   InvoiceId INTEGER NOT NULL,\r\n   TrackId INTEGER NOT NULL,\r\n   Uni
tPrice NUMERIC(10,2) NOT NULL,\r\n   Quantity INTEGER NOT NULL,\r\n   FOREIGN KEY (InvoiceId) REFERENCE
S \"invoices\" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) RE
FERENCES \"tracks\" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_Invo
iceLineInvoiceId ON \"invoice_items\" (InvoiceId)\n\nCREATE TABLE \"invoices\"\r\n(\r\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(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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON \"invoice_items\" (TrackId)\n\nCREATE INDEX IFK_InvoiceCustomerId ON \"invoices\" (CustomerId)\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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\nCREATE TABLE \"customers\"\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(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\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(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 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 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 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\" 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.Custom
erId\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 invoi
ces with a total exceeding $10:\n"}]

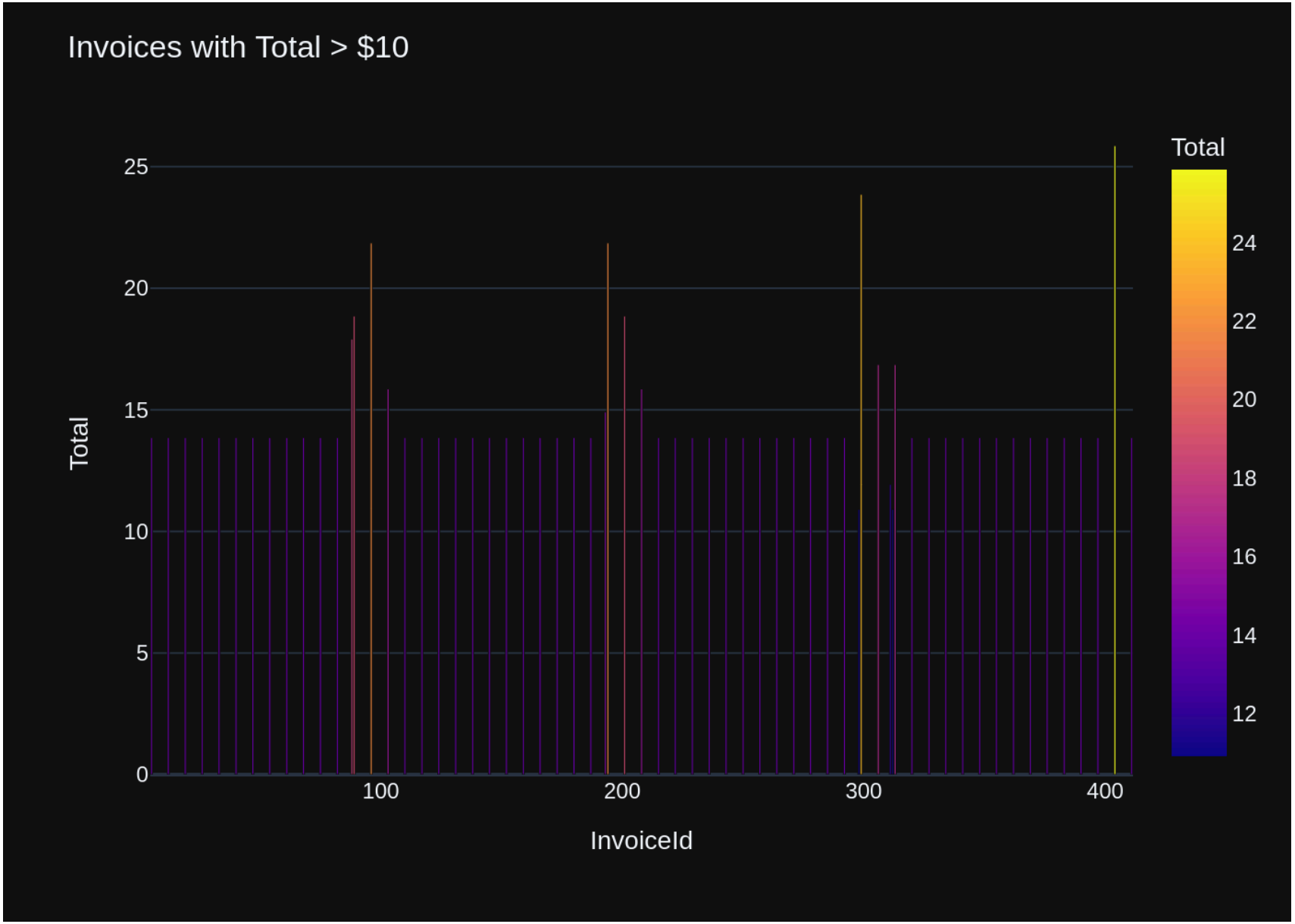
Add of existing embedding ID: aea89953-21b2-55d1-9dda-431ee6033c3d-sql
Insert of existing embedding ID: aea89953-21b2-55d1-9dda-431ee6033c3d-sql

```
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T01:57:21.90023111Z', 'message': {'role': 'assistant',
'content': 'SELECT * \nFROM "invoices" \nWHERE Total > 10.00'}, 'done_reason': 'stop', 'done': True, 'total
_duration': 87845580087, 'load_duration': 709192, 'prompt_eval_count': 1874, 'prompt_eval_duration': 821988
67000, 'eval_count': 20, 'eval_duration': 5018664000}
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=gemma: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 followin
g is information about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n InvoiceId
int64\nCustomerId             int64\nInvoiceDate           object\nBillingAddress       object\nBillingCit
y           object\nBillingState          object\nBillingCountry       object\nBillingPostalCode     objec
t\nTotal                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 wi
th any explanations -- just the code."}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T01:57:59.730702585Z', 'message': {'role': 'assistant',
'content': "```python\nimport plotly.express as px\n\nfig = px.bar(df, x='InvoiceId', y='Total', color='Tot
al', barmode='group')\nfig.update_traces(name='Invoices')\n\nif df.shape[0] == 1:\n    fig.update_traces(ty
pe='indicator', mode='number', value=df['Total'].iloc[0], label='Total')\n\nfig.update_layout(title='Invoic
es with Total > $10')\nfig.show()\n```"}, 'done_reason': 'stop', 'done': True, 'total_duration': 3780738633
4, 'load_duration': 668159, 'prompt_eval_count': 203, 'prompt_eval_duration': 8643886000, 'eval_count': 11
4, 'eval_duration': 29071532000}
```


Invoices with Total > $10
Total
25
20
15
10
5
0
100
200
300
400
InvoiceId
Total
24
22
20
18
16
14
12

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

	InvoiceId	CustomerId	InvoiceDate	BillingAddress	BillingCity	BillingState	BillingCountry	BillingPostalCode	Total
0	5	23	2009-01-11 00:00:00	69 Salem Street	Boston	MA	USA	2113	13.86
1	12	2	2009-02-11 00:00:00	Theodor-Heuss-Straße 34	Stuttgart	None	Germany	70174	13.86
2	19	40	2009-03-14 00:00:00	8, Rue Hanovre	Paris	None	France	75002	13.86
3	26	19	2009-04-14 00:00:00	1 Infinite Loop	Cupertino	CA	USA	95014	13.86
4	33	57	2009-05-15 00:00:00	Calle Lira, 198	Santiago	None	Chile	None	13.86
..	...	...	...	...	...	...	...	...	...
59	383	10	2013-08-12 00:00:00	Rua Dr. Falcão Filho, 155	São Paulo	SP	Brazil	01007-010	13.86
60	390	48	2013-09-12 00:00:00	Lijnbaansgracht 120bg	Amsterdam	VV	Netherlands	1016	13.86
61	397	27	2013-10-13 00:00:00	1033 N Park Ave	Tucson	AZ	USA	85719	13.86
62	404	6	2013-11-13 00:00:00	Rilská 3174/6	Prague	None	Czech Republic	14300	25.86
63	411	44	2013-12-14 00:00:00	Porthaninkatu 9	Helsinki	None	Finland	00530	13.86

[64 rows x 9 columns],
Figure({
'data': [{'alignmentgroup': 'True',
'hovertemplate': 'InvoiceId=%{x}
Total=%{marker.color}<extra></extra>',
'legendgroup': '',
'marker': {'color': array([13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 1
3.86,
13.86, 13.86, 17.91, 18.86, 21.86, 15.86, 13.86, 13.86, 13.86, 1
3.86,
13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 14.91, 2
1.86,
18.86, 15.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 1
3.86,
13.86, 13.86, 13.86, 13.86, 10.91, 23.86, 16.86, 11.94, 10.91, 1
6.86,

```
                                          13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 1
3.86,
                                          13.86, 13.86, 25.86, 13.86]),
                           'coloraxis': 'coloraxis',
                           'pattern': {'shape': ''}},
                'name': 'Invoices',
                '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': 'group',
                 'coloraxis': {'colorbar': {'title': {'text': 'Total'}},
                               'colorscale': [[0.0, '#0d0887'], [0.1111111111111111,
                                              '#46039f'], [0.2222222222222222,
                                              '#7201a8'], [0.3333333333333333,
                                              '#9c179e'], [0.4444444444444444,
                                              '#bd3786'], [0.5555555555555556,
                                              '#d8576b'], [0.6666666666666666,
                                              '#ed7953'], [0.7777777777777778,
                                              '#fb9f3a'], [0.8888888888888888,
                                              '#fdca26'], [1.0, '#f0f921']]},
                 'legend': {'tracegroupgap': 0},
                 'margin': {'t': 60},
                 'template': '...',
                 'title': {'text': 'Invoices with Total > $10'},
                 'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'InvoiceId'}},
```

```
                    '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
```

[{'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\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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON "invoice_items" (InvoiceId)\n\nCREATE INDEX IFK_InvoiceCustomerId ON "invoices" (CustomerId)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\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(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 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   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(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\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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 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\tON DELETE 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, TrackId),\r\n   FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\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 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 u
se the most relevant table(s). \n5. If the question has been asked and answered before, please repeat the a
nswer exactly as it was given before. \n'}, {'role': 'user', 'content': ' \n   Find all invoices since 20
10 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'}, {'rol
e': 'user', 'content': ' \n   List all invoices with a total exceeding $10:\n'}, {'role': 'assistant', 'c
ontent': '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.Customer
Id, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.Customer
Id\nGROUP BY c.CustomerId'}, {'role': 'user', 'content': ' \n   Find the top 5 customers who spent the m
ost money overall, \n   \n   Hint: order total can be found on invoices table, calculation using invoic
e_items detail table is unnecessary \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(i.Tota
l) AS TotalSpent\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.Customer
Id\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.C
ustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.Cust
omerId\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'}, {'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 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 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   Get the average invoice tot
al for each customer:\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, AVG(i.Total) AS AverageIn
voiceTotal\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=gemma: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\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\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_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 \"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 \"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\tON DELETE NO ACTION ON
UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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 D
ELETE 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   FOR
EIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistId) \r\n\t\tON DELETE 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 N
ULL,\r\n   CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\r\n   FOREIGN KEY (PlaylistId)
REFERENCES \"playlists\" (PlaylistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY
(TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\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 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 Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T01:59:33.665248335Z', 'message': {'role': 'assistant', 'content': 'SELECT i.InvoiceDate, SUM(i.Total) AS TotalAmount\nFROM "invoices" i\nWHERE i'}, 'done_reason': 'stop', 'done': True, 'total_duration': 93830005512, 'load_duration': 692516, 'prompt_eval_count': 2020, 'prompt_eval_duration': 86574671000, 'eval_count': 25, 'eval_duration': 6611595000}
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"\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(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 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   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(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\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   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\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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\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 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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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 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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\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=gemma: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

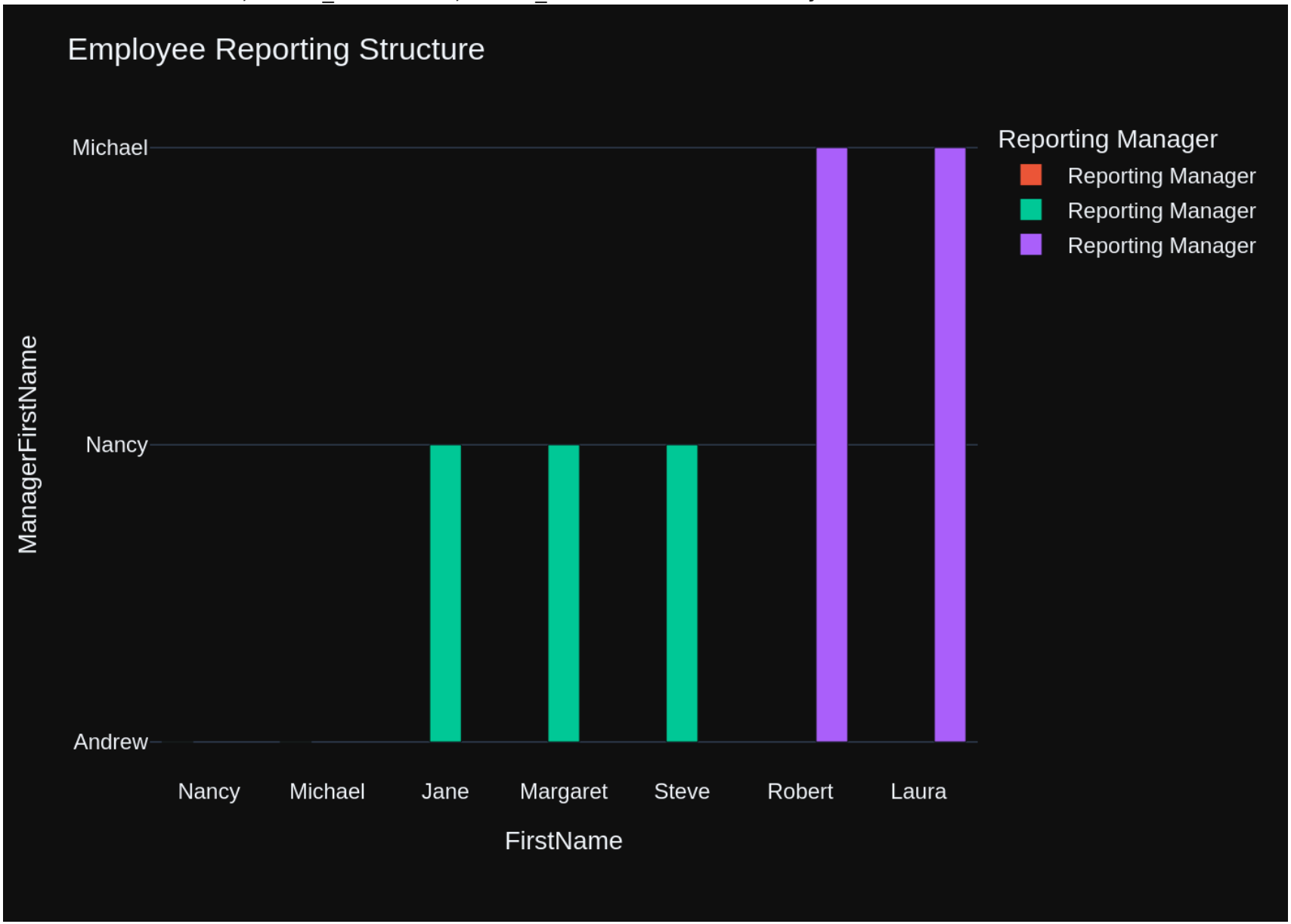
AR(40),\r\n   State NVARCHAR(40),\r\n   Country NVARCHAR(40),\r\n   PostalCode NVARCHAR(10),\r\n   Phon
e NVARCHAR(24),\r\n   Fax NVARCHAR(24),\r\n   Email NVARCHAR(60),\r\n   FOREIGN KEY (ReportsTo) REFERENC
ES \"employees\" (EmployeeId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE 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\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\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\tON DELETE NO ACTION ON
UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) REFERENC
ES \"genres\" (GenreId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (MediaTypeId) R
EFERENCES \"media_types\" (MediaTypeId) \r\n\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\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": "use
r", "content": " \n   Find the customer with the most invoices \n"}, {"role": "assistant", "content": "S
ELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.Cus
tomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1"}, {"role": "user", "co
ntent": "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 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   Get the total number of invoices for each customer\n"},
{"role": "assistant", "content": "SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customer
s\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}, {"role": "user", "conte
nt": " \n   Find all invoices since 2010 and the total amount invoiced:\n"}, {"role": "assistant", "conte
nt": "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 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.Custom
erId\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 \"customer
s\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId"}, {"role": "user", "conte
nt": " \n   Hint: album quantity is found in invoice_items, \n   \n   Find the top 5 customers who boug
ht 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 Tota
lAlbums DESC\nLIMIT 5"}, {"role": "user", "content": " \n   List all employees and their reporting manage
r'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

```
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T02:01:08.281916433Z', 'message': {'role': 'assistant',
'content': 'SELECT e.FirstName, e.LastName, mt.FirstName AS ManagerFirstName, mt.LastName AS ManagerLastNam
e\nFROM "employees" e\nLEFT JOIN "employees" mt ON e.ReportsTo = mt.EmployeeId'}, 'done_reason': 'stop', 'd
one': True, 'total_duration': 94553801488, 'load_duration': 986013, 'prompt_eval_count': 1874, 'prompt_eval
_duration': 82054732000, 'eval_count': 45, 'eval_duration': 11860587000}
SELECT e.FirstName, e.LastName, mt.FirstName AS ManagerFirstName, mt.LastName AS ManagerLastName
FROM "employees" e
LEFT JOIN "employees" mt ON e.ReportsTo = mt.EmployeeId
SELECT e.FirstName, e.LastName, mt.FirstName AS ManagerFirstName, mt.LastName AS ManagerLastName
FROM "employees" e
LEFT 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
Ollama parameters:
model=gemma: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 employees and their reporting manager's name (i
f any):\n'\n\nThe DataFrame was produced using this query: SELECT e.FirstName, e.LastName, mt.FirstName AS
ManagerFirstName, mt.LastName AS ManagerLastName\nFROM \"employees\" e\nLEFT JOIN \"employees\" mt ON e.Rep
ortsTo = mt.EmployeeId\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning
df.dtypes gives:\n FirstName           object\nLastName            object\nManagerFirstName    object\nMana
gerLastName     object\ndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly co
de 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 ex
planations -- just the code."}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T02:01:36.33601765Z', 'message': {'role': 'assistant',
'content': "```python\nfig = px.bar(df, x='FirstName', y='ManagerFirstName', color='ManagerLastName', barmo
de='group')\nfig.update_traces(name='Reporting Manager')\nfig.update_layout(title='Employee Reporting Struc
ture')\nfig.update_layout(legend_title='Reporting Manager')\nfig.show()\n```"}, 'done_reason': 'stop', 'don
```

e': True, 'total_duration': 28029488433, 'load_duration': 714612, 'prompt_eval_count': 173, 'prompt_eval_duration': 7621935000, 'eval_count': 77, 'eval_duration': 20318451000}



```
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',
                        'hovertemplate': 'ManagerLastName=Adams<br>FirstName=%{x}<br>ManagerFirstName=%{y}<extra></
         extra>',
                        'legendgroup': 'Adams',
                        'marker': {'color': '#EF553B', 'pattern': {'shape': ''}},
                        'name': 'Reporting Manager',
                        'offsetgroup': 'Adams',
                        'orientation': 'v',
                        'showlegend': True,
                        'textposition': 'auto',
                        'type': 'bar',
                        'x': array(['Nancy', 'Michael'], dtype=object),
                        'xaxis': 'x',
                        'y': array(['Andrew', 'Andrew'], dtype=object),
                        'yaxis': 'y'},
                       {'alignmentgroup': 'True',
                        'hovertemplate': ('ManagerLastName=Edwards<br>Fir' ... 'rFirstName=%{y}<extra></extra>'),
                        'legendgroup': 'Edwards',
                        'marker': {'color': '#00cc96', 'pattern': {'shape': ''}},
                        'name': 'Reporting Manager',
                        'offsetgroup': 'Edwards',
                        'orientation': 'v',
                        'showlegend': True,
                        'textposition': 'auto',
                        'type': 'bar',
                        'x': array(['Jane', 'Margaret', 'Steve'], dtype=object),
                        'xaxis': 'x',
                        'y': array(['Nancy', 'Nancy', 'Nancy'], dtype=object),
                        'yaxis': 'y'},
                       {'alignmentgroup': 'True',
```

```
                    'hovertemplate': ('ManagerLastName=Mitchell<br>Fi' ... 'rFirstName=%{y}<extra></extra>'),
                    'legendgroup': 'Mitchell',
                    'marker': {'color': '#ab63fa', 'pattern': {'shape': ''}},
                    'name': 'Reporting Manager',
                    'offsetgroup': 'Mitchell',
                    'orientation': 'v',
                    'showlegend': True,
                    'textposition': 'auto',
                    'type': 'bar',
                    'x': array(['Robert', 'Laura'], dtype=object),
                    'xaxis': 'x',
                    'y': array(['Michael', 'Michael'], dtype=object),
                    'yaxis': 'y'}],
         'layout': {'barmode': 'group',
                    'legend': {'title': {'text': 'Reporting Manager'}, 'tracegroupgap': 0},
                    'margin': {'t': 60},
                    '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': 'ManagerFirstName'}}}
          }))
```

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
```

[{'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\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON "invoices" (CustomerId)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON "invoice_items" (InvoiceId)\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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\n\nCREATE TABLE sqlite_stat1(tbl,idx,stat)\n\nCREATE INDEX IFK_CustomerSupportRepId ON "customers" (SupportRepId)\n\nCREATE TABLE "customers"\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(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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\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 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(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 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 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\nG

ROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n   Find the c
ustomer with the most invoices \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.Invoice
Id) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.Cust
omerId\nORDER BY TotalInvoices DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n   Find all invoices sinc
e 2010 and the total amount invoiced:\n'}, {'role': 'assistant', 'content': 'SELECT i.InvoiceDate, SUM(i.To
tal) 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': 'assistan
t', 'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "invoices" i\nGROUP BY i.BillingCo
untry'}, {'role': 'user', 'content': ' \n   Hint: album quantity is found in invoice_items, \n   \n   F
ind 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
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.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   Hint: album quantity is found in invoice_i
tems, \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 Tota
lAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n   List all invoices with a total exceeding $1
0:\n'}, {'role': 'assistant', 'content': 'SELECT * \nFROM "invoices" \nWHERE Total > 10.00'}, {'role': 'use
r', 'content': ' \n   Get the average invoice total for each customer:\n'}]
Ollama parameters:
model=gemma: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\tON DELETE NO ACTION ON UPDATE NO ACTION
\r\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON \"invoices\" (CustomerId)\n\nCREATE INDEX IFK_InvoiceLineInv
oiceId ON \"invoice_items\" (InvoiceId)\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\" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (Tr
ackId) REFERENCES \"tracks\" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX
IFK_InvoiceLineTrackId ON \"invoice_items\" (TrackId)\n\nCREATE TABLE sqlite_stat1(tbl,idx,stat)\n\nCREATE

INDEX IFK_CustomerSupportRepId ON \"customers\" (SupportRepId)\n\nCREATE TABLE \"customers\"\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(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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\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 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(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 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 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

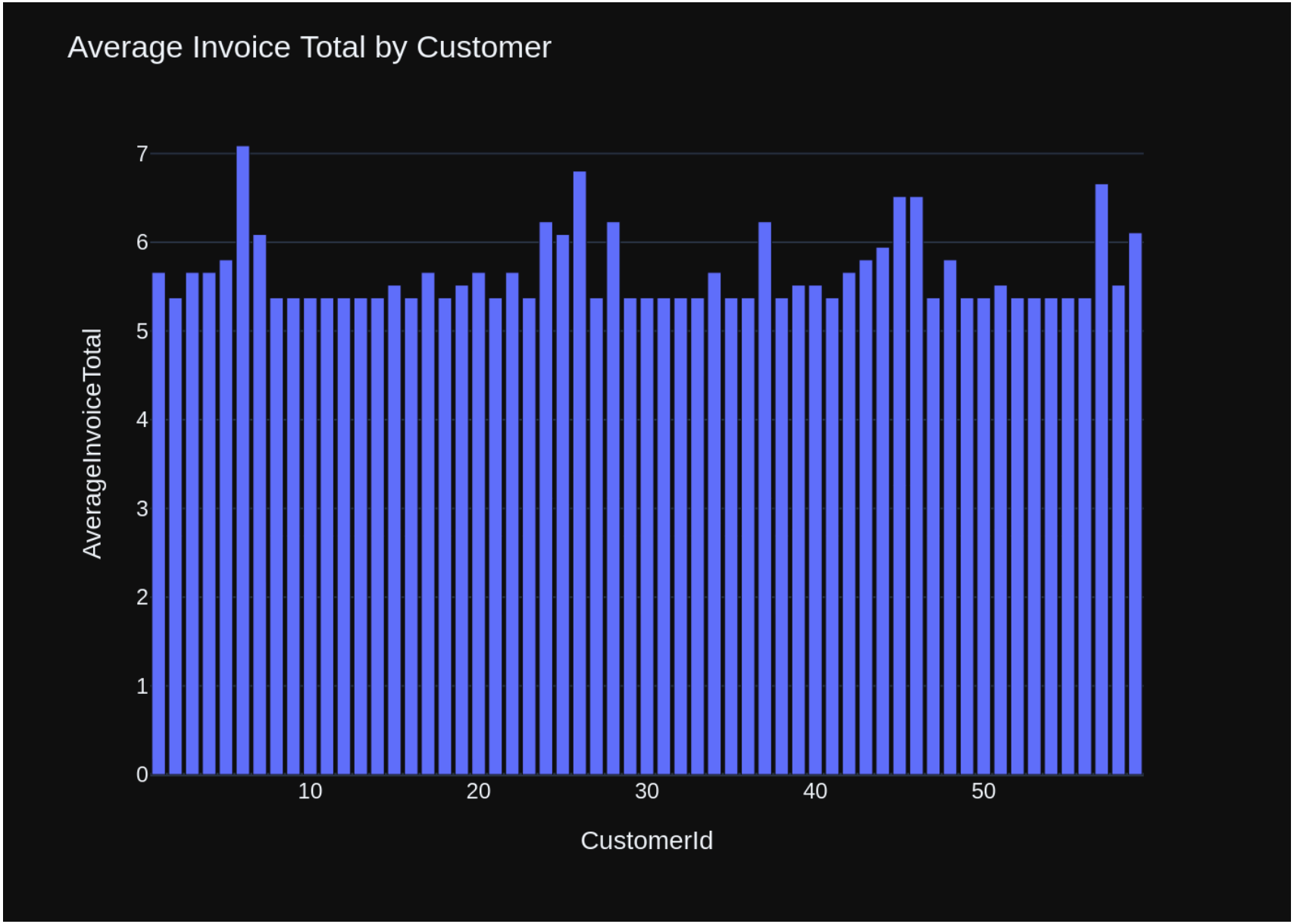
ll invoices): \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(ii.TrackId) AS TotalAlbums \nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1"}, {"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 invoices with a total exceeding $10:\n"}, {"role": "assistant", "content": "SELECT * \nFROM \"invoices\" \nWHERE Total > 10.00"}, {"role": "user", "content": " \n   Get the average invoice 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': 'gemma:latest', 'created_at': '2024-06-14T02:03:01.21559323Z', '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_durati
on': 84749457118, 'load_duration': 1019057, 'prompt_eval_count': 1704, 'prompt_eval_duration': 73083918000,
'eval_count': 42, 'eval_duration': 11028772000}
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=gemma: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
he DataFrame was produced using this query: SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\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:\n CustomerId
```

int64\nAverageInvoiceTotal   float64\ndtype: object"}, {"role": "user", "content": "Can you generate the P
ython plotly code to chart the results of the dataframe? Assume the data is in a pandas dataframe called 'd
f'. If there is only one value in the dataframe, use an Indicator. Respond with only Python code. Do not an
swer with any explanations -- just the code."}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T02:03:27.399533854Z', 'message': {'role': 'assistant',
'content': "```python\nimport plotly.express as px\n\nfig = px.bar(df, x='CustomerId', y='AverageInvoiceTot
al')\nfig.update_traces(hovertemplate='%{name}: %{y}')\nfig.update_layout(title='Average Invoice Total by C
ustomer')\nfig.show()\n```"}, 'done_reason': 'stop', 'done': True, 'total_duration': 26156124073, 'load_dur
ation': 684291, 'prompt_eval_count': 189, 'prompt_eval_duration': 8266657000, 'eval_count': 67, 'eval_durat
ion': 17794509000}


Average Invoice Total by Customer
AverageInvoiceTotal
0
1
2
3
4
5
6
7
10
20
30
40
50
CustomerId

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': '%{name}: %{y}',
              '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},
            'margin': {'t': 60},
            '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"\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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_TrackAlbumId ON "tracks" (AlbumId)\n\nCREATE INDEX IFK_TrackGenreId ON "tracks" (GenreId)\n\nCREATE INDEX IFK_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 "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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE 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, TrackId),\r\n   FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\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\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 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   \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

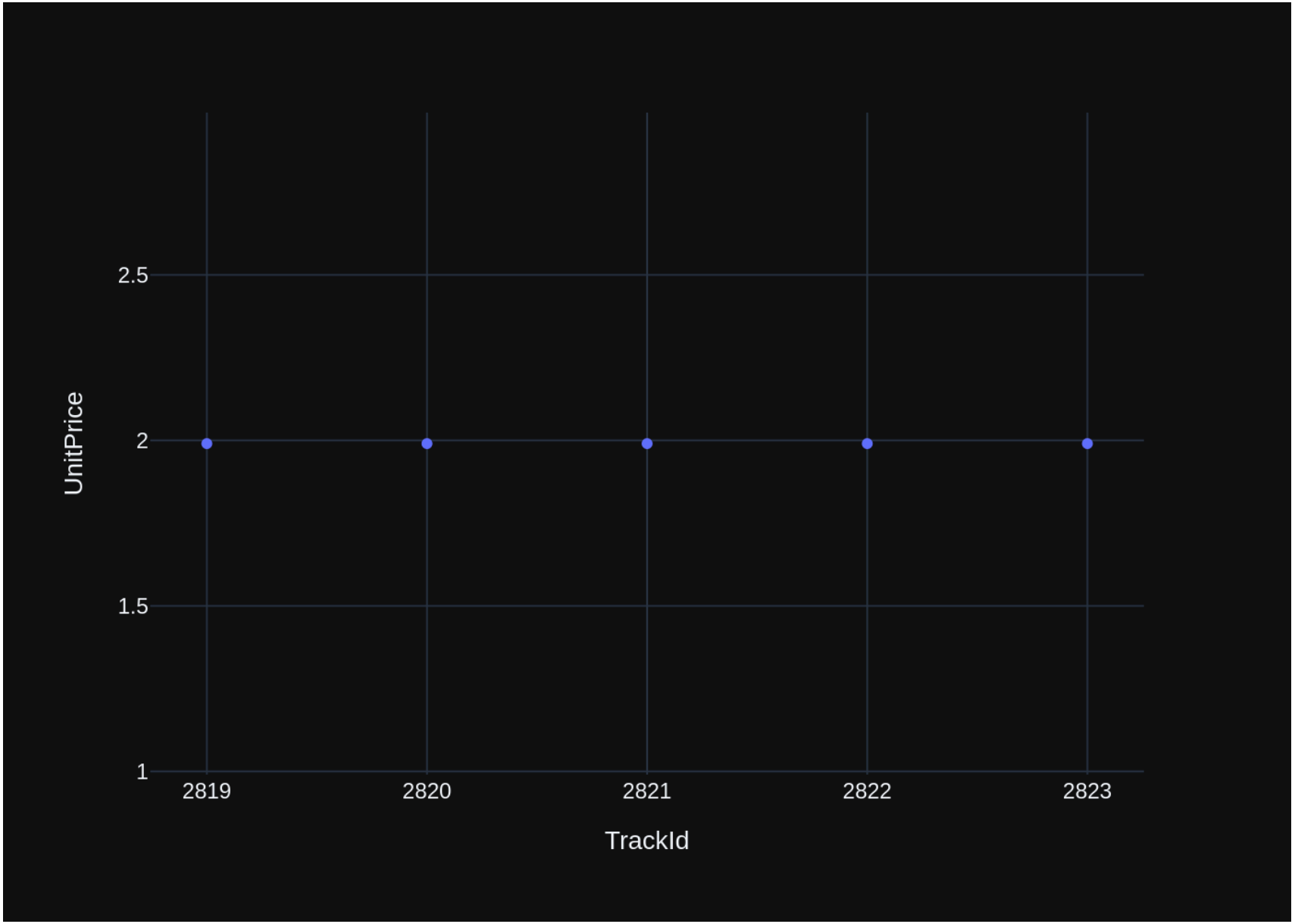
Find the top 5 customers who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n   Find the 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 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'}]
Ollama parameters:
model=gemma: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\"\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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_TrackAlbumId ON \"tracks\" (AlbumId)\n\nCREATE INDEX IFK_TrackGenreId ON \"tracks\" (GenreId)\n\nCREATE INDEX IFK_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\tON DELETE NO ACTIO
N ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\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, TrackI
d),\r\n   FOREIGN KEY (PlaylistId) REFERENCES \"playlists\" (PlaylistId) \r\n\t\tON DELETE NO ACTION ON UP
DATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\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\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"}, {"rol
e": "assistant", "content": "SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDura
tion\nFROM \"playlist_track\" pt\nJOIN \"playlists\" p ON pt.PlaylistId = p.PlaylistId\nJOIN \"tracks\" t O
N pt.TrackId = t.TrackId\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10"}, {"role": "use
r", "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 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", "conten
t": " \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': 'gemma:latest', 'created_at': '2024-06-14T02:04:55.733832374Z', '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': 88238382736, 'load_duration': 775002, 'prompt_eval_c
ount': 1799, 'prompt_eval_duration': 79514863000, 'eval_count': 31, 'eval_duration': 8092252000}
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=gemma: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 pric
e):\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 DataFra
me 'df': \nRunning df.dtypes gives:\n TrackId        int64\nName          object\nUnitPrice    float64\ndty
pe: 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 data
frame, use an Indicator. Respond with only Python code. Do not answer with any explanations -- just the cod
e."}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T02:05:24.063299204Z', 'message': {'role': 'assistant',
'content': "```python\nimport plotly.express as px\n\nfig = px.bar(df, x='Name', y='UnitPrice', title='Top
5 Most Expensive Tracks')\nfig.update_layout(title={'text': 'Top 5 Most Expensive Tracks', 'font': {'size':
18}})\nfig.update_traces(marker={'color': 'auto'})\nfig.show()\n```"}, 'done_reason': 'stop', 'done': True,
'total_duration': 28307812815, 'load_duration': 909458, 'prompt_eval_count': 185, 'prompt_eval_duration': 7
617213000, 'eval_count': 86, 'eval_duration': 20594635000}
```


2.5
2
1.5
1
UnitPrice
2819
2820
2821
2822
2823
TrackId

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': [{'hovertemplate': 'TrackId=%{x}<br>UnitPrice=%{y}<extra></extra>',
              'legendgroup': '',
              'marker': {'color': '#636efa', 'symbol': 'circle'},
              'mode': 'markers',
              'name': '',
              'orientation': 'v',
              'showlegend': False,
              'type': 'scatter',
              'x': array([2819, 2820, 2821, 2822, 2823]),
              'xaxis': 'x',
              'y': array([1.99, 1.99, 1.99, 1.99, 1.99]),
              'yaxis': 'y'}],
    'layout': {'legend': {'tracegroupgap': 0},
               'margin': {'t': 60},
               'template': '...',
               'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'TrackId'}},
               '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"\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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_TrackGenreId ON "tracks" (GenreId)\n\nCREATE TABLE "genres"\r\n(\r\n   GenreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   Name NVARCHAR(120)\r\n)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON "playlist_track" (TrackId)\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 INDEX IFK_TrackMediaTypeId ON "tracks" (MediaTypeId)\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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \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" (ArtistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_AlbumArtistId ON "albums" (ArtistId)\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 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\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\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': ' \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 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 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 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'}]
Ollama parameters:
model=gemma: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\"\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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_TrackGenreId ON \"tracks\" (GenreId)\n\nCREATE TABLE \"genres\"\r\n(\r\n   GenreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   Name NVARCHAR(120)\r\n)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON \"playlist_track\" (TrackId)\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 INDEX IFK_TrackMediaTypeId ON \"tracks\" (MediaTypeId)\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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\tON DELETE

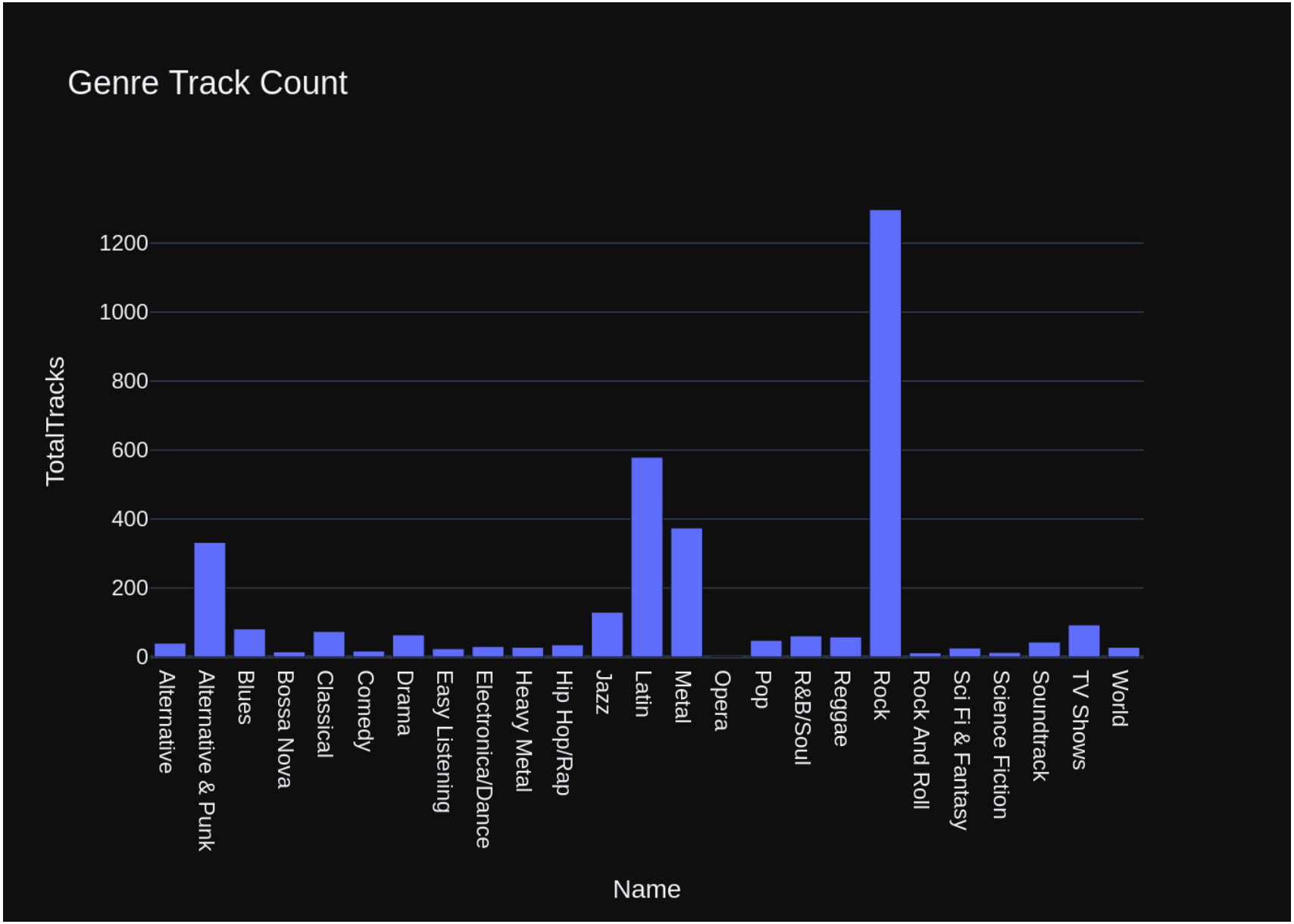
NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"albums\"\r\n(\r\n   AlbumId INTEGER PRIMARY KEY AUTOI
NCREMENT NOT NULL,\r\n   Title NVARCHAR(160) NOT NULL,\r\n   ArtistId INTEGER NOT NULL,\r\n   FOREIGN
KEY (ArtistId) REFERENCES \"artists\" (ArtistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCR
EATE INDEX IFK_AlbumArtistId ON \"albums\" (ArtistId)\n\n\n===Additional Context \n\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 invoi
ce_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': 'gemma:latest', 'created_at': '2024-06-14T02:06:45.183264874Z', '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': 81017516067, 'load_dur
ation': 705237, 'prompt_eval_count': 1677, 'prompt_eval_duration': 69399400000, 'eval_count': 44, 'eval_dur
ation': 10981089000}
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=gemma: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 genr
e:\n'\n\nThe DataFrame was produced using this query: SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM
\"genres\" g\nJOIN \"tracks\" t ON g.GenreId = t.GenreId\nGROUP BY g.Name\n\nThe following is information a
bout the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n Name           object\nTotalTracks
int64\ndtype: object"}, {"role": "user", "content": "Can you generate the Python plotly code to chart the r
esults of the dataframe? Assume the data is in a pandas dataframe called 'df'. If there is only one value i
n the dataframe, use an Indicator. Respond with only Python code. Do not answer with any explanations -- ju
st the code."}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T02:07:10.346988785Z', 'message': {'role': 'assistant',
'content': "```python\nimport plotly.express as px\n\nfig = px.bar(df, x='Name', y='TotalTracks', title='Ge
nre Track Count')\nfig.update_layout(title={'text': 'Genre Track Count', 'font': {'size': 18}})\nfig.show()
\n```"}, 'done_reason': 'stop', 'done': True, 'total_duration': 25137789684, 'load_duration': 667366, 'prom
pt_eval_count': 190, 'prompt_eval_duration': 8038423000, 'eval_count': 67, 'eval_duration': 17009697000}
```


Genre Track Count
TotalTracks
0
200
400
600
800
1000
1200
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
Name

```
Out[30]: ('SELECT g.Name, COUNT(t.GenreId) AS TotalTracks\nFROM "genres" g\nJOIN "tracks" t ON g.GenreId = t.GenreI
         d\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': {'font': {'size': 18}, 'text': 'Genre Track Count'},
                '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
```

[{'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 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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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_PlaylistTrackTrackId ON "playlist_track" (TrackId)\n\nCREATE INDEX IFK_TrackMediaTypeId ON "tracks" (MediaTypeId)\n\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\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 "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\tON DELETE 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, TrackId),\r\n   FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_AlbumArtistId ON "albums" (ArtistId)\n\nCREATE TABLE "playlists"\r\n(\r\n   PlaylistId 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 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   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'}, {'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 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.TrackI

d\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10'}, {'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 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 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   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 all genres that do not have any tracks associated with them:\n'}]
Ollama parameters:
model=gemma: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_TrackGenreId ON \"tracks\" (GenreId)\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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_PlaylistTrackTrackId ON \"playlist_track\" (TrackId)\n\nCREATE INDEX IFK_TrackMediaTypeId ON \"tracks\" (MediaTypeId)\n\nCREATE INDEX IFK_TrackAlbumId ON \"tracks\" (AlbumId)\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 \"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\tON DELETE 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, 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': 'gemma:latest', 'created_at': '2024-06-14T02:08:32.91344962Z', 'message': {'role': 'assistant', 'content': 'SELECT 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': 82453705461, 'load_duration': 877019, 'prompt_eval_count': 1678, 'prompt_eval_duration': 72441057000, 'eval_count': 36, 'eval_duration': 9373664000}
```
SELECT g.Name
FROM "genres" g
LEFT JOIN "tracks" t ON g.GenreId = t.GenreId
WHERE t.GenreId IS NULL
```
```
SELECT g.Name
FROM "genres" g
LEFT JOIN "tracks" t ON g.GenreId = t.GenreId
WHERE t.GenreId IS NULL
```
```
Empty DataFrame
Columns: [Name]
Index: []
```
Ollama parameters:
```
model=gemma: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.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:\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': 'gemma:latest', 'created_at': '2024-06-14T02:09:05.172576236Z', 'message': {'role': 'assistant', 'content': "```python\nimport plotly.graph_objs as go\n\nfig = go.Figure(go.Bar(x=df['Name'], y=0, name='No Tracks'))\n\nif df.shape[0] == 1:\n   fig.update_traces(marker={'size': 40})\n   fig.update_layout(title='Genres with no Tracks', showlegend=False)\n\nfig.show()\n```"}, 'done_reason': 'stop', 'done': True, 'total_duration': 32256737076, 'load_duration': 693864, 'prompt_eval_count': 176, 'prompt_eval_duration': 7513091000, 'eval_count': 93, 'eval_duration': 24652731000}

Out[31]:
```
('SELECT g.Name\nFROM "genres" g\nLEFT JOIN "tracks" t ON g.GenreId = t.GenreId\nWHERE t.GenreId IS NULL',
 Empty DataFrame
 Columns: [Name]
 Index: [],
 Figure({
     'data': [{'domain': {'x': [0.0, 1.0], 'y': [0.0, 1.0]},
               'hovertemplate': 'Name=%{label}<extra></extra>',
               'labels': array([], dtype=object),
               'legendgroup': '',
               'name': '',
               'showlegend': True,
               'type': 'pie'}],
     'layout': {'legend': {'tracegroupgap': 0}, 'margin': {'t': 60}, 'template': '...'}
 }))
```

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. \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\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   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(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\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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\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(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 DELETE 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, TrackId),\r\n   FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \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" (ArtistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON "customers" (SupportRepId)\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 "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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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_InvoiceCustomerId ON "invoices" (CustomerId)\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. Prepe

nd the query with a comment saying intermediate_sql \n3. If the provided context is insufficient, please ex
plain 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 "custom
ers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.Invoi
ceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'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 customer with the most invoices \n'}, {'role': 'assista
nt', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoic
es" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1'}, {'rol
e': 'user', 'content': ' \n   Find the top 5 customers who spent the most money overall, \n   \n   H
int: order total can be found on invoices table, calculation using invoice_items detail table is unnecessar
y \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': 'assi
stant', 'content': 'SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices" i\nJOIN "i
nvoice_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': 'assi
stant', 'content': 'SELECT c.Country, COUNT(*) AS TotalCustomers\nFROM "customers" c\nGROUP BY c.Country\nO
RDER BY TotalCustomers DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n   Get the total number of invoic
es for each customer\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT(i.InvoiceId) AS Tota
lInvoices\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 custome
r:\n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, AVG(i.Total) AS AverageInvoiceTotal\nFROM "c
ustomers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId'}, {'role': 'user', 'c
ontent': ' \n   List all customers who have not placed any orders:\n'}]
Ollama parameters:
model=gemma: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\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
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(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\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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   F
OREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\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\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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) RE
FERENCES \"tracks\" (TrackId) \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\" (ArtistId) \r\n\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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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_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 albu
ms in total quantity (across all invoices): \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, CO
UNT(ii.TrackId) AS TotalAlbums\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nJ
OIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nL
IMIT 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\nGRO
UP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": " \n   Find the cu
stomer with the most invoices \n"}, {"role": "assistant", "content": "SELECT c.CustomerId, COUNT(i.InvoiceI
d) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.C
ustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1"}, {"role": "user", "content": " \n   Find the top 5 cus
tomers who spent the most money overall, \n   \n   Hint: order total can be found on invoices table, ca
lculation 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.Custo
merId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5"}, {"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 i.CustomerId, COUN
T(ii.InvoiceLineId) AS TotalAlbums\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.Invo
iceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5"}, {"role": "user", "content": " \n   L
ist all customers from Canada and their email addresses:\n"}, {"role": "assistant", "content": "SELECT c.Em
ail, c.Country\nFROM \"customers\" c\nWHERE c.Country = 'Canada'"}, {"role": "user", "content": "what are t
he 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": "assi
stant", "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 ma
ny 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", "co
ntent": "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 cu
stomers who have not placed any orders:\n"}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T02:10:40.469355933Z', 'message': {'role': 'assistant',
'content': 'SELECT * FROM "customers" c\nWHERE NOT EXISTS (SELECT * FROM "invoices" i WHERE i.CustomerId =
c.CustomerId);'}, 'done_reason': 'stop', 'done': True, 'total_duration': 95220129135, 'load_duration': 1591
083, 'prompt_eval_count': 1980, 'prompt_eval_duration': 86920948000, 'eval_count': 29, 'eval_duration': 767
3668000}
SELECT * FROM "customers" c
WHERE NOT EXISTS (SELECT * FROM "invoices" i WHERE i.CustomerId = c.CustomerId);
Output from LLM: SELECT * FROM "customers" c
WHERE NOT EXISTS (SELECT * FROM "invoices" i WHERE i.CustomerId = c.CustomerId);

```
Extracted SQL: SELECT * FROM "customers" c
WHERE NOT EXISTS (SELECT * FROM "invoices" i WHERE i.CustomerId = c.CustomerId)
SELECT * FROM "customers" c
WHERE NOT EXISTS (SELECT * FROM "invoices" i WHERE i.CustomerId = c.CustomerId)
Empty DataFrame
Columns: [CustomerId, FirstName, LastName, Company, Address, City, State, Country, PostalCode, Phone, Fax,
Email, SupportRepId]
Index: []
Ollama parameters:
model=gemma: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 * FROM \"customers\" c\nWHERE NOT EXISTS (SELECT * FR
OM \"invoices\" i WHERE i.CustomerId = c.CustomerId)\n\nThe following is information about the resulting pa
ndas DataFrame 'df': \nRunning df.dtypes gives:\n CustomerId      object\nFirstName       object\nLastName
object\nCompany         object\nAddress         object\nCity            object\nState           object\nCou
ntry         object\nPostalCode      object\nPhone           object\nFax             object\nEmail
object\nSupportRepId    object\ndtype: object"}, {"role": "user", "content": "Can you generate the Python p
lotly 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 wi
th any explanations -- just the code."}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T02:11:09.745243459Z', 'message': {'role': 'assistant',
'content': "```python\nimport plotly.express as px\n\nfig = px.bar(df, x='CustomerName', height=400) if df.
shape[0] > 1 else px.indicator(label='No Orders Placed')\n\nfig.update_layout(title='Customers with No Orde
rs')\nfig.show()\n```"}, 'done_reason': 'stop', 'done': True, 'total_duration': 29273286570, 'load_duratio
n': 830486, 'prompt_eval_count': 217, 'prompt_eval_duration': 9696482000, 'eval_count': 72, 'eval_duratio
n': 19486120000}
```

Out[32]:
```
('SELECT * FROM "customers" c\nWHERE NOT EXISTS (SELECT * FROM "invoices" i WHERE i.CustomerId = c.Custome
rId)',
 Empty DataFrame
 Columns: [CustomerId, FirstName, LastName, Company, Address, City, State, Country, PostalCode, Phone, Fa
x, Email, SupportRepId]
 Index: [],
 Figure({
     'data': [{'domain': {'x': [0.0, 1.0], 'y': [0.0, 1.0]},
               'hovertemplate': 'CustomerId=%{label}<extra></extra>',
               'labels': array([], dtype=object),
               'legendgroup': '',
               'name': '',
               'showlegend': True,
               'type': 'pie'}],
     'layout': {'legend': {'tracegroupgap': 0}, 'margin': {'t': 60}, 'template': '...'}
 }))
```

In [33]:
```
question = """
    There are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums
    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"\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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 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\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 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 NULL,\r\n   CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\r\n   FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX 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 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   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   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 "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=gemma: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\"\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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 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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"artists\"\r\n(\r\n   ArtistId INTEGER PRIMAR

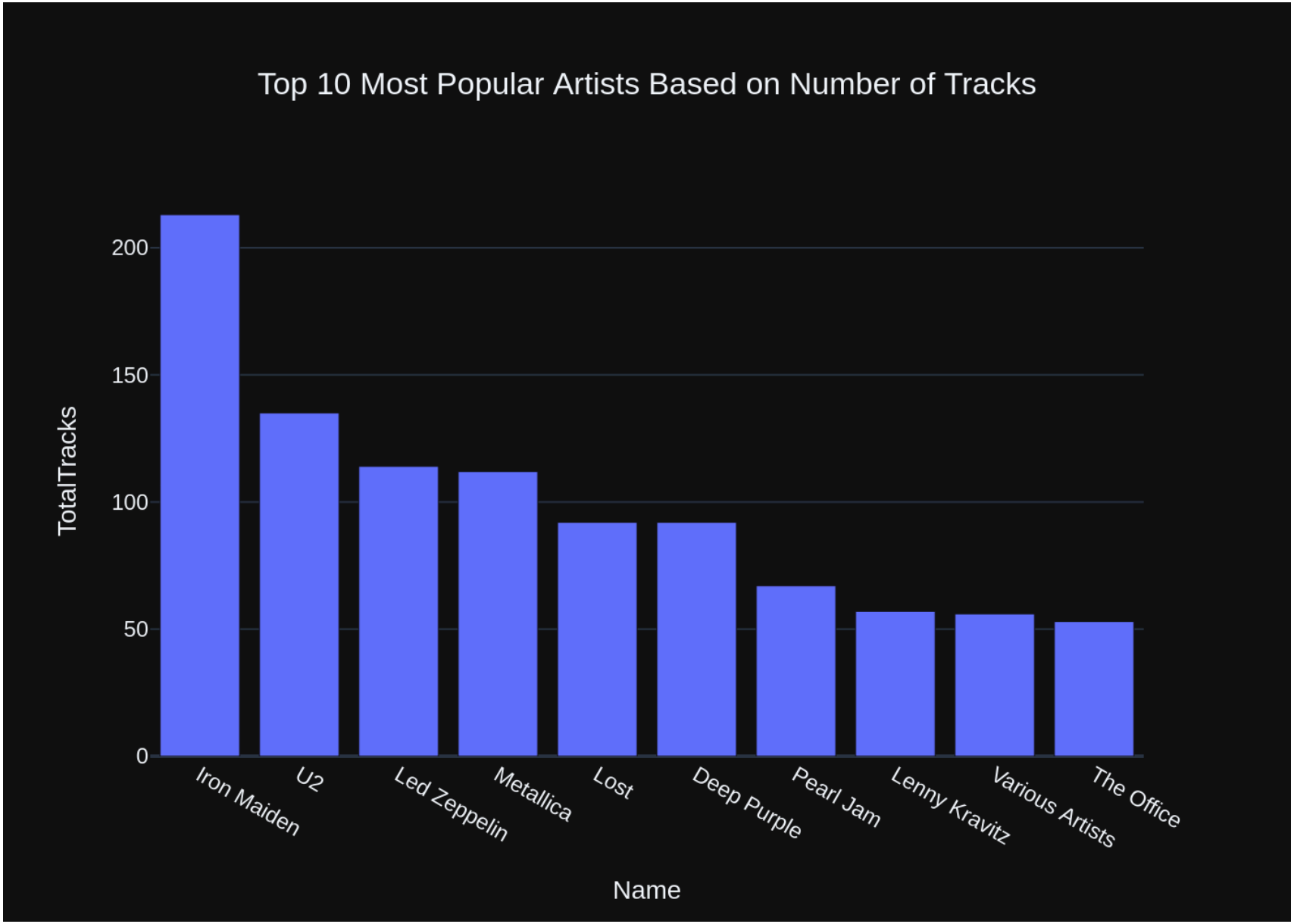
Y KEY AUTOINCREMENT NOT NULL,\r\n   Name NVARCHAR(120)\r\n)\n\nCREATE INDEX IFK_AlbumArtistId ON \"albums
\" (ArtistId)\n\nCREATE INDEX IFK_TrackAlbumId ON \"tracks\" (AlbumId)\n\nCREATE TABLE \"playlists\"\r\n(\r
\n   PlaylistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   Name NVARCHAR(120)\r\n)\n\nCREATE 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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (T
rackId) REFERENCES \"tracks\" (TrackId) \r\n\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\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': 'gemma:latest', 'created_at': '2024-06-14T02:12:51.433800663Z', 'message': {'role': 'assistant',
'content': 'SELECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM "artists" a\nJOIN "albums" al ON a.ArtistI
d = al.ArtistId\nJOIN "tracks" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nORDER BY TotalTracks DESC\nLIM
IT 10'}, 'done_reason': 'stop', 'done': True, 'total_duration': 101607256501, 'load_duration': 729726, 'pro
mpt_eval_count': 1785, 'prompt_eval_duration': 81559040000, 'eval_count': 71, 'eval_duration': 19396608000}
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=gemma: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 alb
ums 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: SE
LECT a.Name, COUNT(t.TrackId) AS TotalTracks\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.Art
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 object\nTotalTracks   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': 'gemma:latest', 'created_at': '2024-06-14T02:13:19.751282342Z', '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 Based on Number of Tracks')\nfig.update_layout(title_x=0.5)\nfig.show()\n```"}, 'done_reason': 'stop', 'done': True, 'total_duration': 28290948316, 'load_duration': 699620, 'prompt_eval_count': 255, 'prompt_eval_duration': 10902877000, 'eval_count': 65, 'eval_duration': 17293954000}


Top 10 Most Popular Artists Based on Number of Tracks
TotalTracks
200
150
100
50
0
Iron Maiden
U2
Led Zeppelin
Metallica
Lost
Deep Purple
Pearl Jam
Lenny Kravitz
Various Artists
The Office
Name

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 Based on Number of Tracks', 'x': 0.5},
                '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
```

[{'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"\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(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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\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\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(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(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 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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_EmployeeReportsTo ON "employees" (ReportsTo)\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\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 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 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 "cus

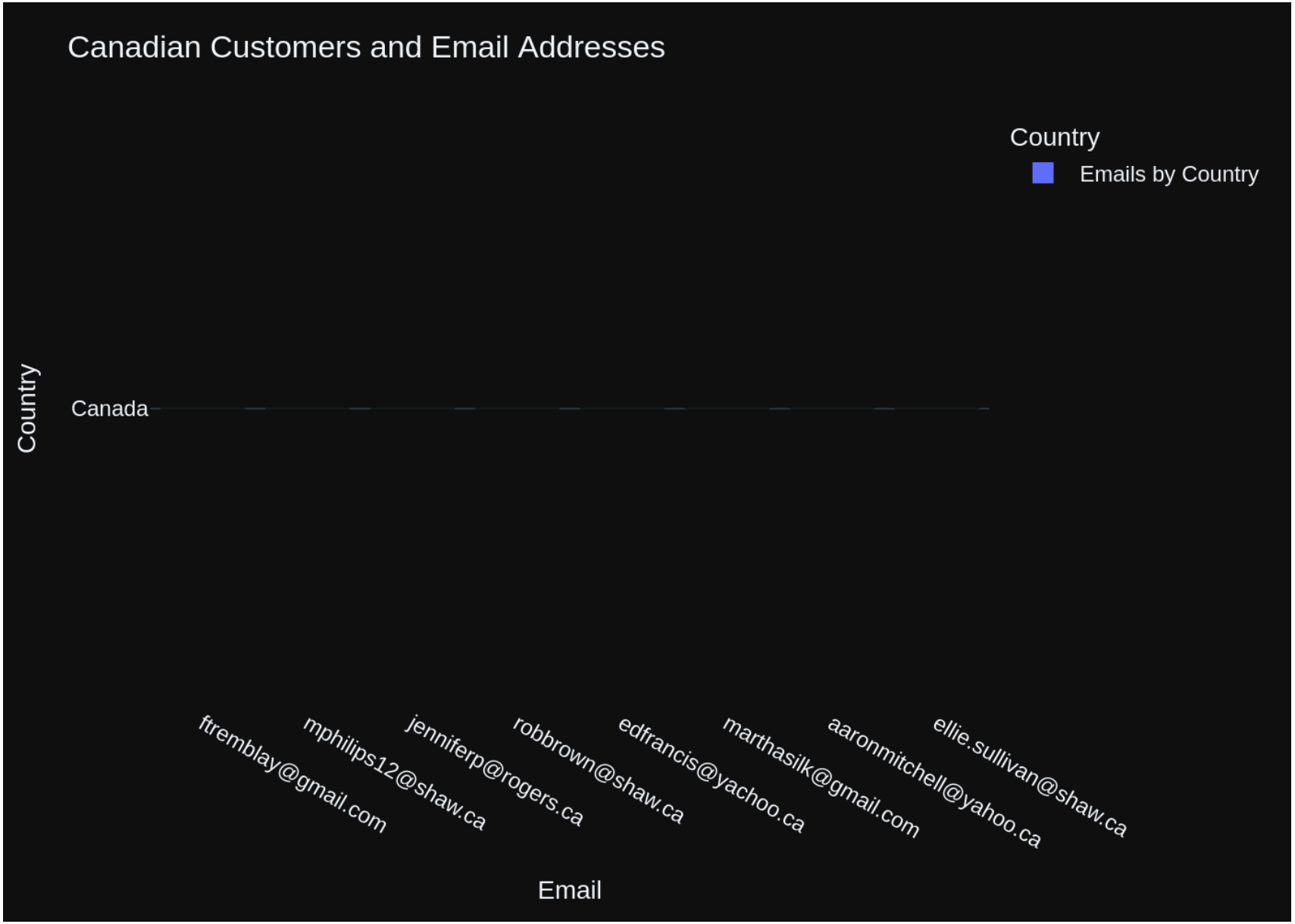
tomers" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n
Find the customer with the most invoices \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, COUNT
(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.CustomerId = 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 TotalSpen
t\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 "invoice
s" 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=gemma: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_CustomerSupportRepId ON \"customers\" (SupportRepId)\n\nC
REATE TABLE \"customers\"\r\n(\r\n   CustomerId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   FirstNa
me NVARCHAR(40) NOT NULL,\r\n   LastName NVARCHAR(20) NOT NULL,\r\n   Company NVARCHAR(80),\r\n   Addr
ess 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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"invoices\"\r\n(\r\n   InvoiceId INT
EGER 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\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\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\tON DELETE NO ACTION ON
UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackI
d) \r\n\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\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': 'gemma:latest', 'created_at': '2024-06-14T02:14:47.224838427Z', 'message': {'role': 'assistant',
'content': 'SELECT c.Email, c.Country\nFROM "customers" c\nWHERE c.Country = \'Canada\''}, 'done_reason':
'stop', 'done': True, 'total_duration': 87366830590, 'load_duration': 743502, 'prompt_eval_count': 1778, 'p
rompt_eval_duration': 80504508000, 'eval_count': 24, 'eval_duration': 6225483000}
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=gemma: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 addresse
s:\n'\n\nThe DataFrame was produced using this query: SELECT c.Email, c.Country\nFROM \"customers\" c\nWHER
E c.Country = 'Canada'\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning
df.dtypes gives:\n Email      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 d
ataframe called 'df'. If there is only one value in the dataframe, use an Indicator. Respond with only Pyth
on code. Do not answer with any explanations -- just the code."}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T02:15:23.335343171Z', 'message': {'role': 'assistant',
'content': "```python\nimport plotly.express as px\n\nfig = px.bar(df, x='Email', y='Country', color='Count
ry', barmode='group')\nfig.update_traces(name='Emails by Country')\nfig.update_layout(title='Canadian Custo
mers and Email Addresses')\n\nif df.shape[0] == 1:\n    fig = px.indicator(text=df['Email'][0], title='Sing
le Canadian Customer Email')\n\nfig.show()\n```"}, 'done_reason': 'stop', 'done': True, 'total_duration': 3
6085549445, 'load_duration': 645352, 'prompt_eval_count': 165, 'prompt_eval_duration': 7375360000, 'eval_co
unt': 106, 'eval_duration': 28620238000}
```


Canadian Customers and Email Addresses
Country
Emails by Country
Country
Canada
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
Email

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': 'Emails by Country',
              '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'], dtype=object),
              'yaxis': 'y'}],
    'layout': {'barmode': 'group',
               'legend': {'title': {'text': 'Country'}, 'tracegroupgap': 0},
               'margin': {'t': 60},
               'template': '...',
               'title': {'text': 'Canadian Customers and Email Addresses'},
               '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
```

[{'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\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON "invoices" (CustomerId)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON "invoice_items" (InvoiceId)\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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\n\nCREATE TABLE "customers"\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(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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON "customers" (SupportRepId)\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(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 DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_EmployeeReportsTo ON "employees" (ReportsTo)\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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\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

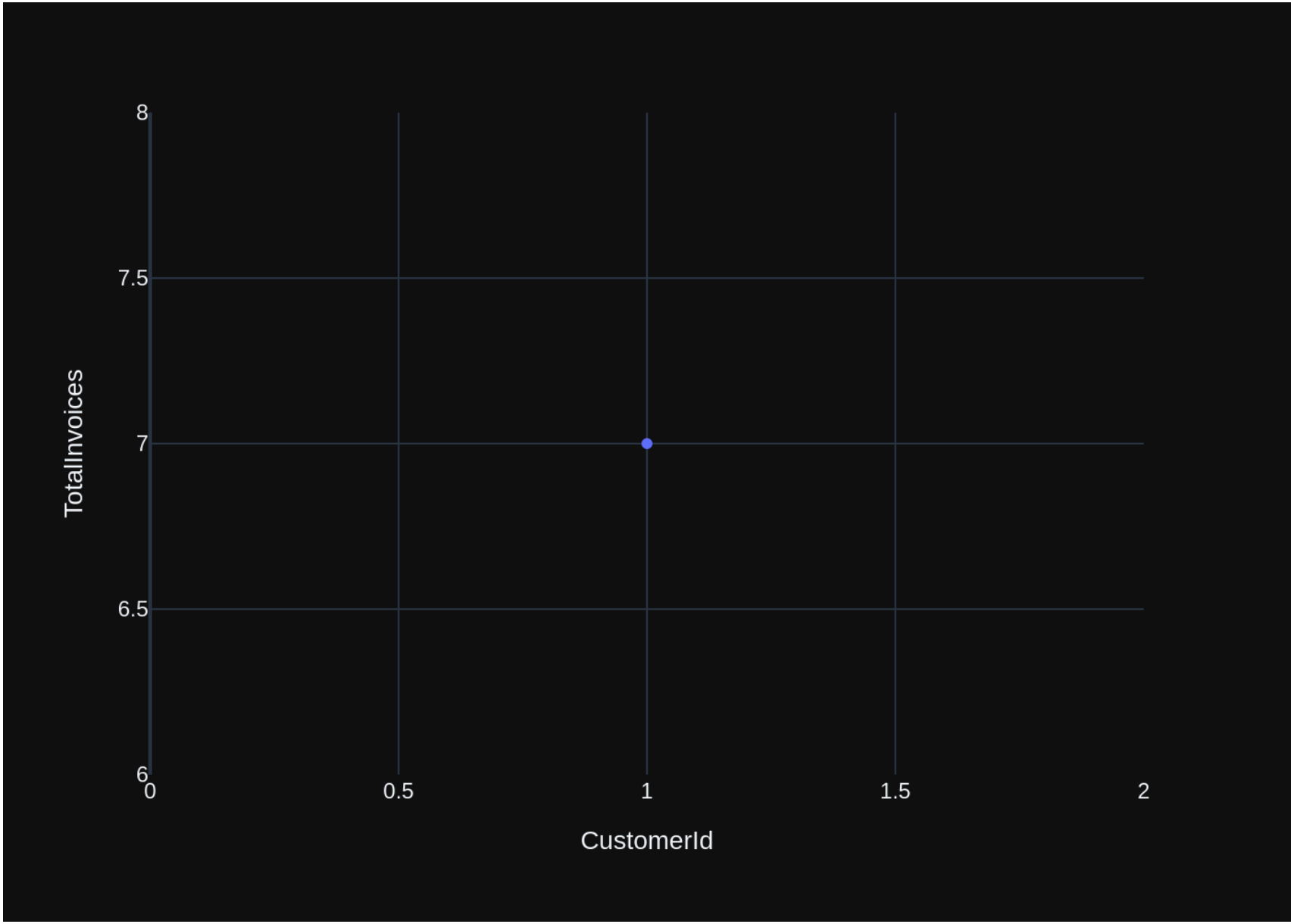
DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n   Find the top 5 customers who spent the most money ov
erall, \n   \n   Hint: order total can be found on invoices table, calculation using invoice_items deta
il table is unnecessary \n'}, {'role': 'assistant', 'content': 'SELECT c.CustomerId, SUM(i.Total) AS TotalS
pent\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 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.Custom
erId\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.CustomerI
d'}, {'role': 'user', 'content': ' \n   Get the average invoice total for each customer:\n'}, {'role': 'a
ssistant', '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 "invoi
ces" \nWHERE Total > 10.00'}, {'role': 'user', 'content': ' \n   Find the total number of invoices per co
untry:\n'}, {'role': 'assistant', 'content': 'SELECT i.BillingCountry, COUNT(*) AS TotalInvoices\nFROM "inv
oices" i\nGROUP BY i.BillingCountry'}, {'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 customer with the most invoices \n'}]
Ollama parameters:
model=gemma: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\tON DELETE NO ACTION ON UPDATE NO ACTION
\r\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON \"invoices\" (CustomerId)\n\nCREATE INDEX IFK_InvoiceLineInv

oiceId ON \"invoice_items\" (InvoiceId)\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\" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (Tr
ackId) REFERENCES \"tracks\" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX
IFK_InvoiceLineTrackId ON \"invoice_items\" (TrackId)\n\nCREATE TABLE \"customers\"\r\n(\r\n   CustomerId
INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   FirstName NVARCHAR(40) NOT NULL,\r\n   LastName NVARCH
AR(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\tON DELETE NO ACTION ON UPDATE NO ACTION
\r\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON \"customers\" (SupportRepId)\n\nCREATE TABLE \"employees
\"\r\n(\r\n   EmployeeId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   LastName NVARCHAR(20) NOT NUL
L,\r\n   FirstName NVARCHAR(20) NOT NULL,\r\n   Title NVARCHAR(30),\r\n   ReportsTo INTEGER,\r\n   Bir
thDate DATETIME,\r\n   HireDate DATETIME,\r\n   Address NVARCHAR(70),\r\n   City NVARCHAR(40),\r\n   St
ate 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 DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_EmployeeReportsTo ON
\"employees\" (ReportsTo)\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 NUL
L,\r\n   GenreId INTEGER,\r\n   Composer NVARCHAR(220),\r\n   Milliseconds INTEGER NOT NULL,\r\n   Byt
es INTEGER,\r\n   UnitPrice NUMERIC(10,2) NOT NULL,\r\n   FOREIGN KEY (AlbumId) REFERENCES \"albums\" (A
lbumId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) REFERENCES \"genres\"
(GenreId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (MediaTypeId) REFERENCES \"me
dia_types\" (MediaTypeId) \r\n\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 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   Fin
d 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\nGROU
P BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1"}, {"role": "user", "content": " \n   Find the t
op 5 customers who spent the most money overall, \n   \n   Hint: order total can be found on invoices t
able, calculation using invoice_items detail table is unnecessary \n"}, {"role": "assistant", "content": "S
ELECT 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": "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", "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"}, {"ro
le": "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   Get
the average invoice total for each customer:\n"}, {"role": "assistant", "content": "SELECT c.CustomerId, AV
G(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 $1
0:\n"}, {"role": "assistant", "content": "SELECT * \nFROM \"invoices\" \nWHERE Total > 10.00"}, {"role": "u
ser", "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   Find all invoices since 2010 and the total amount invoiced:\n"}, {"rol
e": "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 cus
tomer 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': 'gemma:latest', 'created_at': '2024-06-14T02:17:06.877157665Z', '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_reaso
n': 'stop', 'done': True, 'total_duration': 103402400835, 'load_duration': 875107, 'prompt_eval_count': 187
1, 'prompt_eval_duration': 87812031000, 'eval_count': 52, 'eval_duration': 14954264000}
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
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=gemma: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 Dat
aFrame was produced using this query: SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"cust
omers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoice
s DESC\nLIMIT 1\n\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dty
pes gives:\n CustomerId       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 pan
das 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': 'gemma:latest', 'created_at': '2024-06-14T02:17:32.010932629Z', 'message': {'role': 'assistant',
'content': "```python\nimport plotly.express as px\n\nfig = px.bar(df, x='CustomerId', y='TotalInvoices')\n
fig.update_traces(barmode='group')\nfig.update_layout(title='Customers with Most Invoices')\nfig.show()\n``
`"}, 'done_reason': 'stop', 'done': True, 'total_duration': 25108247555, 'load_duration': 754417, 'prompt_e
val_count': 197, 'prompt_eval_duration': 8885328000, 'eval_count': 62, 'eval_duration': 16131041000}
```


8
7.5
7
6.5
6
TotalInvoices
0
0.5
1
1.5
2
CustomerId

Out[35]:
```
('SELECT c.CustomerId, COUNT(i.InvoiceId) AS TotalInvoices\nFROM "customers" c\nJOIN "invoices" i ON c.Cus
tomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoices DESC\nLIMIT 1',
    CustomerId  TotalInvoices
 0           1             7,
 Figure({
     'data': [{'hovertemplate': 'CustomerId=%{x}<br>TotalInvoices=%{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([7]),
               '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': '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"\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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 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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \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" (ArtistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_AlbumArtistId ON "albums" (ArtistId)\n\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\t\tON DELETE NO ACTION ON UPDATE NO ACTION \r\n)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON "invoice_items" (InvoiceId)\n\nCREATE INDEX IFK_InvoiceCustomerId ON "invoices" (CustomerId)\n\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\n\nCREATE TABLE "artists"\r\n(\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 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 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, C

OUNT(ii.InvoiceLineId) AS TotalAlbums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.Invoi
ceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n   Fi
nd 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 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   The
re are 3 tables: artists, albums and tracks, where albums and artists are linked by ArtistId, albums and tr
acks 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\nJ
OIN "albums" al ON a.ArtistId = al.ArtistId\nJOIN "tracks" t ON al.AlbumId = t.AlbumId\nGROUP BY a.Name\nOR
DER 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 Tot
alAmount\nFROM "invoices" i\nWHERE i.InvoiceDate >= \'2010-01-01\'\nGROUP BY i.InvoiceDate'}, {'role': 'use
r', 'content': ' \n   Get the total number of invoices for each customer\n'}, {'role': 'assistant', 'cont
ent': '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 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
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'}, {'ro
le': 'user', 'content': ' \n   Find the customer who bought the most albums in total quantity (across al
l invoices): \n'}]
Ollama parameters:
model=gemma: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\"\r\n(\r\n   TrackId INTEGER PRIMARY KEY AUTOINCREM
ENT 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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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 TABLE \"invoic
e_items\"\r\n(\r\n   InvoiceLineId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   InvoiceId INTEGER N

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\tON DELETE NO ACTION
ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\t\tON DELETE NO ACTI
ON ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"albums\"\r\n(\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\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(\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\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(\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 TotalSpen
t 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"}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T02:19:14.752250163Z', '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 professionisti'}, 'done_reason': 'stop', 'done': True, 'total_duration': 102642838140, 'load_duration': 742021, 'prompt_eval_count': 1820, 'prompt_eval_duration': 82638798000, 'eval_count': 71, 'eval_duration': 19338827000}
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 professionisti
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 professionisti
Couldn't run sql: Execution failed on sql '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 professionisti': near "professionisti": syntax error
```

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"\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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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 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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX 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 "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\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON "invoices" (CustomerId)\n\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\n\nCREATE TABLE "artists"\r\n(\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 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   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 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

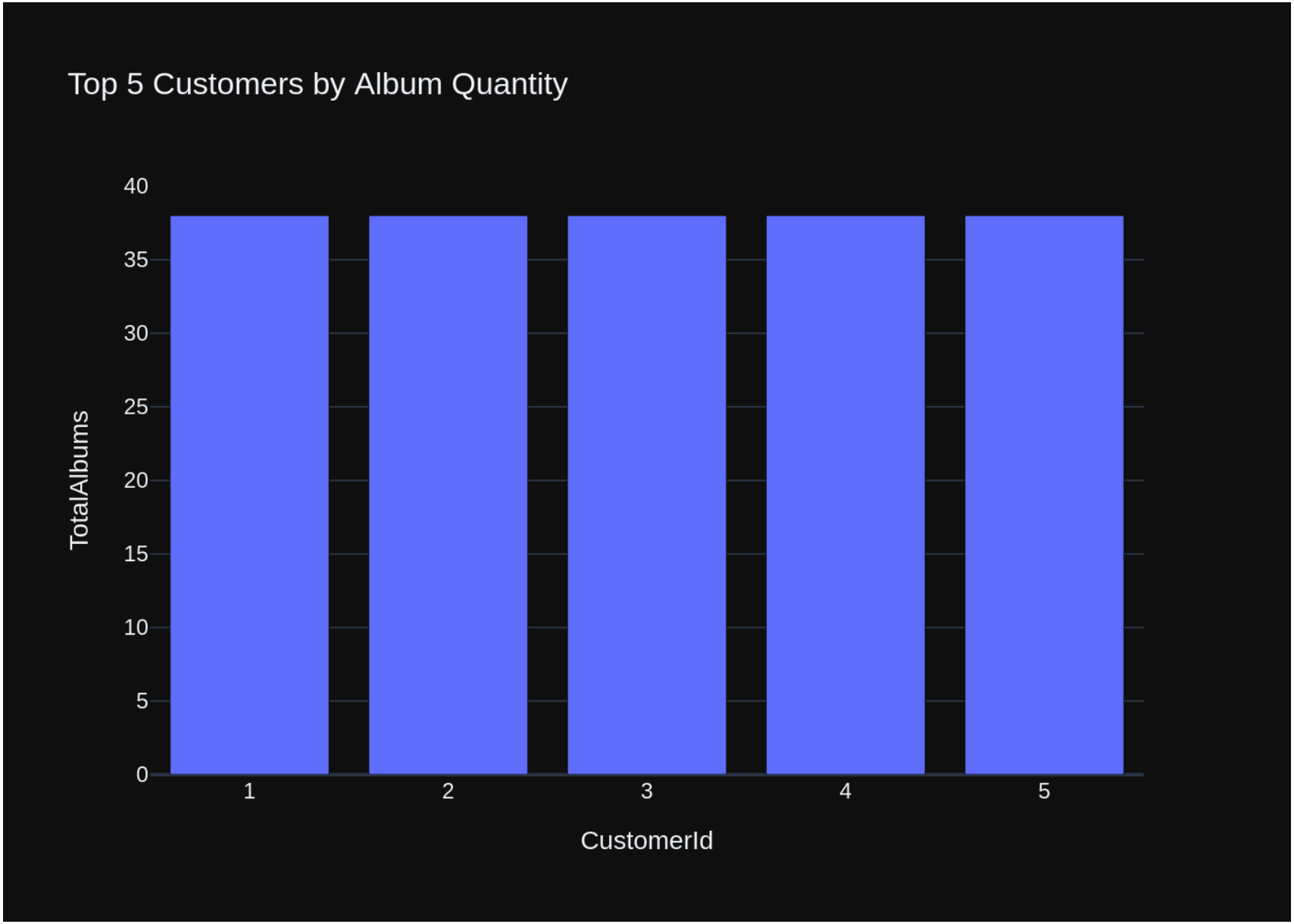
ustomers" c\nJOIN "invoices" i ON c.CustomerId = i.CustomerId\nJOIN "invoice_items" ii ON i.InvoiceId = ii.
InvoiceId\nGROUP BY c.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 1'}, {'role': 'user', 'content': ' \n
Find the top 5 customers who spent the most money overall, \n   \n   Hint: order total can be found on
invoices table, calculation using invoice_items detail table is unnecessary \n'}, {'role': 'assistant', 'co
ntent': 'SELECT c.CustomerId, SUM(i.Total) AS TotalSpent\nFROM "customers" c\nJOIN "invoices" i ON c.Custom
erId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalSpent DESC\nLIMIT 5'}, {'role': 'user', 'conten
t': ' \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.AlbumId\nGRO
UP BY a.Name\nORDER BY TotalTracks DESC\nLIMIT 10'}, {'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   Find the top 5 most expensiv
e tracks (based on unit price):\n'}, {'role': 'assistant', 'content': 'SELECT t.TrackId, t.Name, t.UnitPric
e\nFROM "tracks" t\nORDER BY t.UnitPrice DESC\nLIMIT 5'}, {'role': 'user', 'content': ' \n   Identify ar
tists 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\nW
HERE a.ArtistId IN (\n SELECT g2.GenreId\n FROM "genres" g2\n JOIN "tracks" t2 ON g2.GenreId = t2.GenreI
d\n GROUP BY g2.GenreId\n HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name'}, {'role': 'use
r', 'content': ' \n   Get the total number of invoices for each customer\n'}, {'role': 'assistant', 'cont
ent': '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 n
umber 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: a
lbum 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=gemma: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 \"invoice_items\"\r\n(\r\n   InvoiceLineId INTEGER PRIMARY K
EY AUTOINCREMENT NOT NULL,\r\n   InvoiceId INTEGER NOT NULL,\r\n   TrackId INTEGER NOT NULL,\r\n   Uni
tPrice NUMERIC(10,2) NOT NULL,\r\n   Quantity INTEGER NOT NULL,\r\n   FOREIGN KEY (InvoiceId) REFERENCE
S \"invoices\" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) RE
FERENCES \"tracks\" (TrackId) \r\n\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   Composer NVARCHAR(22
0),\r\n   Milliseconds INTEGER NOT NULL,\r\n   Bytes INTEGER,\r\n   UnitPrice NUMERIC(10,2) NOT NUL

L,\r\n   FOREIGN KEY (AlbumId) REFERENCES \"albums\" (AlbumId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO AC
TION,\r\n   FOREIGN KEY (GenreId) 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 ACTI
ON ON UPDATE NO ACTION\r\n)\n\nCREATE TABLE \"albums\"\r\n(\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\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(\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\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(\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": "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\n)\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 Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T02:20:52.284737579Z', 'message': {'role': 'assistant', 'content': 'SELECT i.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5'}, 'done_reason': 'stop', 'done': True, 'total_duration': 97494362385, 'load_duration': 819313, 'prompt_eval_count': 1762, 'prompt_eval_duration': 81348395000, 'eval_count': 56, 'eval_duration': 15501764000}
```
SELECT i.CustomerId, COUNT(ii.TrackId) AS TotalAlbums
FROM "invoices" i
JOIN "invoice_items" ii ON i.InvoiceId = ii.InvoiceId
GROUP BY i.CustomerId
ORDER BY TotalAlbums DESC
LIMIT 5
```
	CustomerId	TotalAlbums
0	1	38
1	2	38
2	3	38
3	4	38
4	5	38

Ollama parameters:

```
model=gemma: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    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'\n\nThe Data
Frame was produced using this query: SELECT i.CustomerId, COUNT(ii.TrackId) 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\nThe following is information about the resulting pandas DataFrame 'df': \nRunning df.dtype
s gives:\n CustomerId     int64\nTotalAlbums    int64\ndtype: object"}, {"role": "user", "content": "Can yo
u generate the Python plotly code to chart the results of the dataframe? Assume the data is in a pandas dat
aframe 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': 'gemma:latest', 'created_at': '2024-06-14T02:21:17.351065362Z', 'message': {'role': 'assistant',
'content': "```python\nimport plotly.express as px\n\nfig = px.bar(df, x='CustomerId', y='TotalAlbums', tit
le='Top 5 Customers by Album Quantity')\nfig.update_layout(barmode='stack')\nfig.show()\n```"}, 'done_reaso
n': 'stop', 'done': True, 'total_duration': 25046063588, 'load_duration': 650278, 'prompt_eval_count': 204,
'prompt_eval_duration': 8974645000, 'eval_count': 57, 'eval_duration': 15980925000}
```


Top 5 Customers by Album Quantity
TotalAlbums
40
35
30
25
20
15
10
5
0
1
2
3
4
5
CustomerId

Out[37]:
```
('SELECT i.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFROM "invoices" i\nJOIN "invoice_items" ii ON i.I
nvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORDER BY TotalAlbums DESC\nLIMIT 5',
    CustomerId  TotalAlbums
 0           1           38
 1           2           38
 2           3           38
 3           4           38
 4           5           38,
 Figure({
     'data': [{'alignmentgroup': 'True',
               'hovertemplate': 'CustomerId=%{x}<br>TotalAlbums=%{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]),
               'xaxis': 'x',
               'y': array([38, 38, 38, 38, 38]),
               'yaxis': 'y'}],
     'layout': {'barmode': 'stack',
                'legend': {'tracegroupgap': 0},
                'template': '...',
                'title': {'text': 'Top 5 Customers by Album Quantity'},
                'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'CustomerId'}},
                'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'TotalAlbums'}}}
 }))
```

```sql
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 unne
"""

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 "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\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" (InvoiceId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON "invoice_items" (InvoiceId)\n\nCREATE INDEX IFK_InvoiceCustomerId ON "invoices" (CustomerId)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON "invoice_items" (TrackId)\n\nCREATE TABLE "customers"\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(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\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(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 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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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 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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \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 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 r

epeat 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.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   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.TrackId) 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 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 parameters:
model=gemma: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\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\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_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\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\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\tON DELETE NO ACTION ON
UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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 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\tON DELETE NO ACTION O
N UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\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": "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   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   Hint: album quantity is found in inv
oice_items, \n   \n   Find the top 5 customers who bought the most albums in total quantity (across all i
nvoices):\n"}, {"role": "assistant", "content": "SELECT i.CustomerId, COUNT(ii.InvoiceLineId) AS TotalAlbum
s\nFROM \"invoices\" i\nJOIN \"invoice_items\" ii ON i.InvoiceId = ii.InvoiceId\nGROUP BY i.CustomerId\nORD
ER BY TotalAlbums DESC\nLIMIT 5"}, {"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 i.CustomerId, COUNT(ii.TrackId) AS TotalAlbums\nFR
OM \"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 \"cu
stomers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.CustomerId\nGROUP BY c.CustomerId\nORDER BY TotalInvoi
ces 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(i
i.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": "a
ssistant", "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.Customer
Id, COUNT(i.InvoiceId) AS TotalInvoices\nFROM \"customers\" c\nJOIN \"invoices\" i ON c.CustomerId = i.Cust
omerId\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 countri
es that customers come from?"}, {"role": "assistant", "content": "SELECT c.Country, COUNT(*) AS TotalCustom
ers\nFROM \"customers\" c\nGROUP BY c.Country\nORDER BY TotalCustomers DESC\nLIMIT 5"}, {"role": "user", "c
ontent": " \n   Find the top 5 customers who spent the most money overall, \n   \n   Hint: order tot
al can be found on invoices table, calculation using invoice_items detail table is unnecessary \n"}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T02:22:49.657933279Z', 'message': {'role': 'assistant',
'content': 'SELECT c.CustomerId, SUM(i.'}, 'done_reason': 'stop', 'done': True, 'total_duration': 921812399
05, 'load_duration': 793993, 'prompt_eval_count': 1989, 'prompt_eval_duration': 88878219000, 'eval_count':
10, 'eval_duration': 2669133000}
SELECT c.CustomerId, SUM(i.

```
SELECT c.CustomerId, SUM(i.
Couldn't run sql:  Execution failed on sql 'SELECT c.CustomerId, SUM(i.': incomplete input
```

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
```

[{'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"\r\n(\r\n   PlaylistId 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 NULL,\r\n   CONSTRAINT PK_PlaylistTrack PRIMARY KEY (PlaylistId, TrackId),\r\n   FOREIGN KEY (PlaylistId) REFERENCES "playlists" (PlaylistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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_TrackGenreId ON "tracks" (GenreId)\n\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\n\nCREATE INDEX IFK_TrackMediaTypeId ON "tracks" (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\tON 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\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 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   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\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 HAV

ING 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'}, {'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.TrackId) 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   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 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   Get all playlists containing at least 10 tracks and the total duration of those tracks:\n'}]
Ollama parameters:
model=gemma: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_PlaylistTrackTrackId ON \"playlist_track\" (TrackId)\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 \"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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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_TrackGenreId ON \"tracks\" (Gen

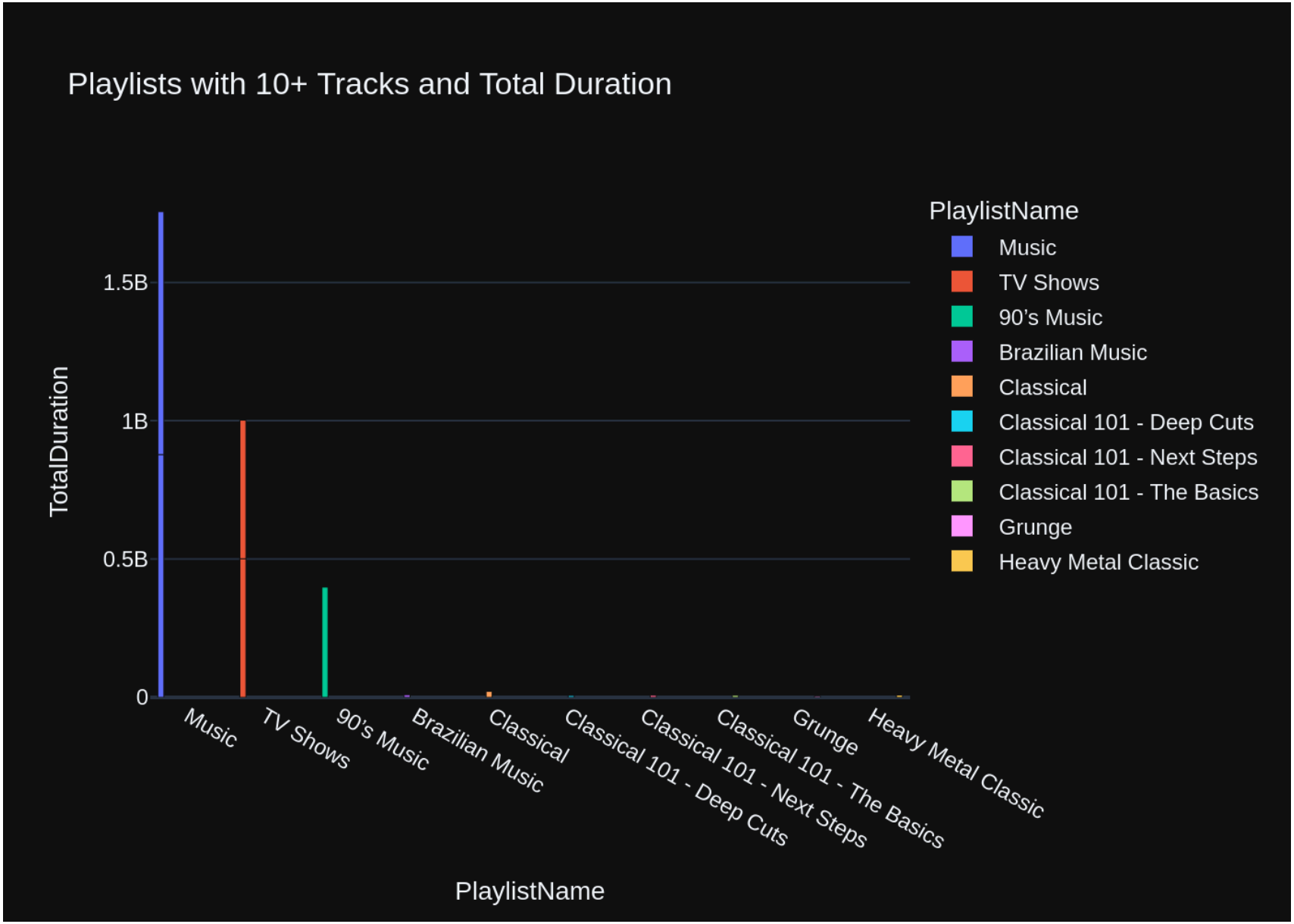
reId)\n\nCREATE INDEX IFK_TrackAlbumId ON \"tracks\" (AlbumId)\n\nCREATE INDEX IFK_TrackMediaTypeId ON \"tracks\" (MediaTypeId)\n\nCREATE INDEX IFK_AlbumArtistId ON \"albums\" (ArtistId)\n\nCREATE TABLE \"albums\"\r\n(\r\n   AlbumId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   Title NVARCHAR(160) NOT NULL,\r\n   ArtistId INTEGER NOT NULL,\r\n   FOREIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistId) \r\n\t\tON 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\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 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   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\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"}, {"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.TrackId) 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   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 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   Get all playlists containing at least 10 tracks and the total duration of those 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': 'gemma:latest', 'created_at': '2024-06-14T02:24:27.677322284Z', 'message': {'role': 'assistant',
'content': 'SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM "playl
ist_track" pt\nJOIN "playlists" p ON pt.PlaylistId = p.PlaylistId\nJOIN "tracks" t ON pt.TrackId = t.TrackI
d\nGROUP BY pt.PlaylistId, p.Name\nHAVING COUNT(pt.TrackId) >= 10'}, 'done_reason': 'stop', 'done': True,
'total_duration': 97977543119, 'load_duration': 816948, 'prompt_eval_count': 1682, 'prompt_eval_duration':
74260643000, 'eval_count': 88, 'eval_duration': 23089976000}
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=gemma: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\nPlaylistName   object\nTotalDuration   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': 'gemma:latest', 'created_at': '2024-06-14T02:25:08.679744571Z', 'message': {'role': 'assistant', 'content': '```python\nimport plotly.express as px\n\nfig = px.bar(df, x=\'PlaylistName\', y=\'TotalDuration\', color=\'PlaylistName\', barmode=\'group\',\n   title=\'Playlists with 10+ Tracks and Total Duration\')\n\nif df.shape[0] == 1:\n   fig = px.indicator(text=f\'Playlist with 10+ tracks: {df["PlaylistName"][0]}\', title=\'Playlist Summary\')\n\nfig.update_layout(hovermode=\'closest\')\nfig.show()\n```'}, 'done_reason': 'stop', 'done': True, 'total_duration': 40974968791, 'load_duration': 648772, 'prompt_eval_count': 224, 'prompt_eval_duration': 9569598000, 'eval_count': 120, 'eval_duration': 31315056000}


Playlists with 10+ Tracks and Total Duration
TotalDuration
1.5B
1B
0.5B
0
Music
TV Shows
90's Music
Brazilian Music
Classical
Classical 101 - Deep Cuts
Classical 101 - Next Steps
Classical 101 - The Basics
Grunge
Heavy Metal Classic
PlaylistName
PlaylistName
Music
TV Shows
90's Music
Brazilian Music
Classical
Classical 101 - Deep Cuts
Classical 101 - Next Steps
Classical 101 - The Basics
Grunge
Heavy Metal Classic

```
Out[39]: ('SELECT pt.PlaylistId, p.Name AS PlaylistName, SUM(t.Milliseconds) AS TotalDuration\nFROM "playlist_trac
         k" pt\nJOIN "playlists" p ON pt.PlaylistId = p.PlaylistId\nJOIN "tracks" t ON pt.TrackId = t.TrackId\nGROU
         P 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': 'PlaylistName=%{x}<br>TotalDuration=%{y}<extra></extra>',
                        'legendgroup': 'Music',
                        'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
                        'name': 'Music',
                        'offsetgroup': 'Music',
                        'orientation': 'v',
                        'showlegend': True,
                        'textposition': 'auto',
                        'type': 'bar',
                        'x': array(['Music', 'Music'], dtype=object),
                        'xaxis': 'x',
                        'y': array([877683083, 877683083]),
                        'yaxis': 'y'},
                       {'alignmentgroup': 'True',
                        'hovertemplate': 'PlaylistName=%{x}<br>TotalDuration=%{y}<extra></extra>',
                        'legendgroup': 'TV Shows',
                        'marker': {'color': '#EF553B', 'pattern': {'shape': ''}},
                        'name': 'TV Shows',
                        'offsetgroup': 'TV Shows',
                        'orientation': 'v',
                        'showlegend': True,
                        'textposition': 'auto',
                        'type': 'bar',
                        'x': array(['TV Shows', 'TV Shows'], dtype=object),
```

```
 'xaxis': 'x',
 'y': array([501094957, 501094957]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'PlaylistName=%{x}<br>TotalDuration=%{y}<extra></extra>',
 'legendgroup': '90’s Music',
 'marker': {'color': '#00cc96', 'pattern': {'shape': ''}},
 'name': '90’s Music',
 'offsetgroup': '90’s Music',
 'orientation': 'v',
 'showlegend': True,
 'textposition': 'auto',
 'type': 'bar',
 'x': array(['90’s Music'], dtype=object),
 'xaxis': 'x',
 'y': array([398705153]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'PlaylistName=%{x}<br>TotalDuration=%{y}<extra></extra>',
 'legendgroup': 'Brazilian Music',
 'marker': {'color': '#ab63fa', 'pattern': {'shape': ''}},
 'name': 'Brazilian Music',
 'offsetgroup': 'Brazilian Music',
 'orientation': 'v',
 'showlegend': True,
 'textposition': 'auto',
 'type': 'bar',
 'x': array(['Brazilian Music'], dtype=object),
 'xaxis': 'x',
 'y': array([9486559]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'PlaylistName=%{x}<br>TotalDuration=%{y}<extra></extra>',
 'legendgroup': 'Classical',
 'marker': {'color': '#FFA15A', 'pattern': {'shape': ''}},
 'name': 'Classical',
 'offsetgroup': 'Classical',
 'orientation': 'v',
 'showlegend': True,
 'textposition': 'auto',
 'type': 'bar',
 'x': array(['Classical'], dtype=object),
```

```
 'xaxis': 'x',
 'y': array([21770592]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'PlaylistName=%{x}<br>TotalDuration=%{y}<extra></extra>',
 'legendgroup': 'Classical 101 - Deep Cuts',
 'marker': {'color': '#19d3f3', 'pattern': {'shape': ''}},
 'name': 'Classical 101 - Deep Cuts',
 'offsetgroup': 'Classical 101 - Deep Cuts',
 'orientation': 'v',
 'showlegend': True,
 'textposition': 'auto',
 'type': 'bar',
 'x': array(['Classical 101 - Deep Cuts'], dtype=object),
 'xaxis': 'x',
 'y': array([6755730]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'PlaylistName=%{x}<br>TotalDuration=%{y}<extra></extra>',
 'legendgroup': 'Classical 101 - Next Steps',
 'marker': {'color': '#FF6692', 'pattern': {'shape': ''}},
 'name': 'Classical 101 - Next Steps',
 'offsetgroup': 'Classical 101 - Next Steps',
 'orientation': 'v',
 'showlegend': True,
 'textposition': 'auto',
 'type': 'bar',
 'x': array(['Classical 101 - Next Steps'], dtype=object),
 'xaxis': 'x',
 'y': array([7575051]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'PlaylistName=%{x}<br>TotalDuration=%{y}<extra></extra>',
 'legendgroup': 'Classical 101 - The Basics',
 'marker': {'color': '#B6E880', 'pattern': {'shape': ''}},
 'name': 'Classical 101 - The Basics',
 'offsetgroup': 'Classical 101 - The Basics',
 'orientation': 'v',
 'showlegend': True,
 'textposition': 'auto',
 'type': 'bar',
 'x': array(['Classical 101 - The Basics'], dtype=object),
```

```
                  'xaxis': 'x',
                  'y': array([7439811]),
                  'yaxis': 'y'},
                 {'alignmentgroup': 'True',
                  'hovertemplate': 'PlaylistName=%{x}<br>TotalDuration=%{y}<extra></extra>',
                  'legendgroup': 'Grunge',
                  'marker': {'color': '#FF97FF', 'pattern': {'shape': ''}},
                  'name': 'Grunge',
                  'offsetgroup': 'Grunge',
                  'orientation': 'v',
                  'showlegend': True,
                  'textposition': 'auto',
                  'type': 'bar',
                  'x': array(['Grunge'], dtype=object),
                  'xaxis': 'x',
                  'y': array([4122018]),
                  'yaxis': 'y'},
                 {'alignmentgroup': 'True',
                  'hovertemplate': 'PlaylistName=%{x}<br>TotalDuration=%{y}<extra></extra>',
                  'legendgroup': 'Heavy Metal Classic',
                  'marker': {'color': '#FECB52', 'pattern': {'shape': ''}},
                  'name': 'Heavy Metal Classic',
                  'offsetgroup': 'Heavy Metal Classic',
                  'orientation': 'v',
                  'showlegend': True,
                  'textposition': 'auto',
                  'type': 'bar',
                  'x': array(['Heavy Metal Classic'], dtype=object),
                  'xaxis': 'x',
                  'y': array([8206312]),
                  'yaxis': 'y'}],
     'layout': {'barmode': 'group',
                'hovermode': 'closest',
                'legend': {'title': {'text': 'PlaylistName'}, 'tracegroupgap': 0},
                'template': '...',
                'title': {'text': 'Playlists with 10+ Tracks and Total Duration'},
                'xaxis': {'anchor': 'y',
                          'categoryarray': [Music, TV Shows, 90’s Music, Brazilian
                                            Music, Classical, Classical 101 - Deep
                                            Cuts, Classical 101 - Next Steps,
                                            Classical 101 - The Basics, Grunge,
                                            Heavy Metal Classic],
```

```
                                'categoryorder': 'array',
                                '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"\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   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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_AlbumArtistId ON "albums" (ArtistId)\n\nCREATE INDEX IFK_TrackGenreId ON "tracks" (GenreId)\n\nCREATE INDEX IFK_TrackAlbumId ON "tracks" (AlbumId)\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\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)\n\nCREATE INDEX IFK_TrackMediaTypeId ON "tracks" (MediaTypeId)\n\nCREATE TABLE "genres"\r\n(\r\n   GenreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   Name NVARCHAR(120)\r\n)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON "playlist_track" (TrackId)\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 "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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES "tracks" (TrackId) \r\n\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 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   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'}, {'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 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   Hint: album quantity is found in invoice_items, \n   \n   Find the top 5 customers who bough

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   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.TrackId) 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 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 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', 'co
ntent': ' \n   Get all playlists containing at least 10 tracks and the total duration of those track
s:\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 "track
s" 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': 'assi
stant', 'content': 'SELECT t.TrackId, t.Name, t.UnitPrice\nFROM "tracks" t\nORDER BY t.UnitPrice DESC\nLIMI
T 5'}, {'role': 'user', 'content': ' \n   Identify artists who have albums with tracks appearing in mult
iple genres:\n\n\n'}]
Ollama parameters:
model=gemma: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\"\r\n(\r\n   TrackId INTEGER PRIMARY KEY AUTOINCREM
ENT 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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (GenreId) 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_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\"\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   FOR
EIGN KEY (ArtistId) REFERENCES \"artists\" (ArtistId) \r\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\r\n)
\n\nCREATE INDEX IFK_TrackMediaTypeId ON \"tracks\" (MediaTypeId)\n\nCREATE TABLE \"genres\"\r\n(\r\n   Ge

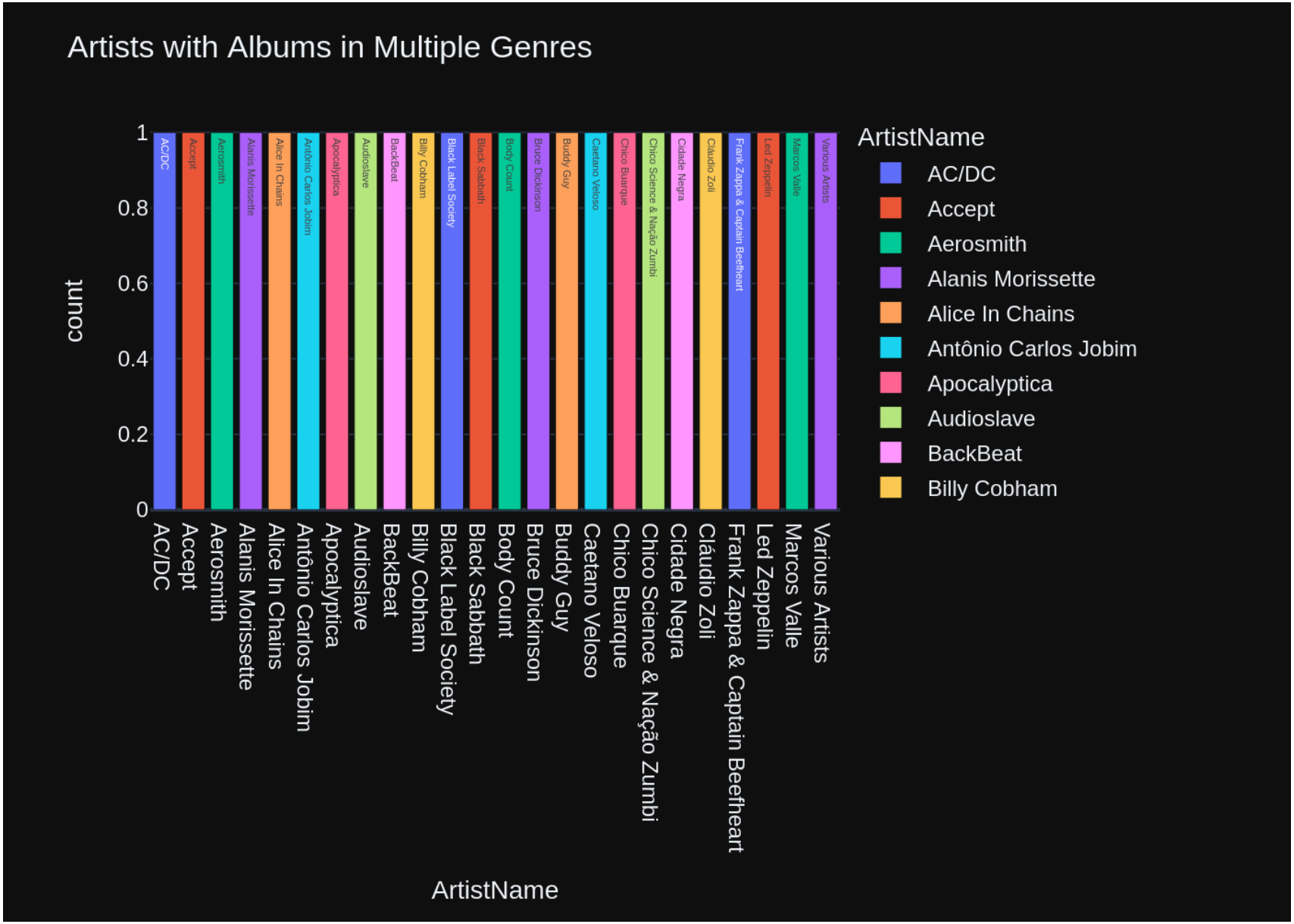
nreId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   Name NVARCHAR(120)\r\n)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON \"playlist_track\" (TrackId)\n\nCREATE TABLE \"artists\"\r\n(\r\n   ArtistId INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,\r\n   Name NVARCHAR(120)\r\n)\n\nCREATE TABLE \"playlist_track\"\r\n(\r\n   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\tON DELETE NO ACTION ON UPDATE NO ACTION,\r\n   FOREIGN KEY (TrackId) REFERENCES \"tracks\" (TrackId) \r\n\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 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   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"}, {"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 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   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.TrackId) 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 \"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.InvoiceLin

```
eId) 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 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\nG
ROUP 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.Trac
kId, t.Name, t.UnitPrice\nFROM \"tracks\" t\nORDER BY t.UnitPrice DESC\nLIMIT 5"}, {"role": "user", "conten
t": "  \n     Identify artists who have albums with tracks appearing in multiple genres:\n\n\n"}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T02:26:56.300187846Z', '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 ORDER B
Y ArtistName;'}, 'done_reason': 'stop', 'done': True, 'total_duration': 107490565969, 'load_duration': 7304
26, 'prompt_eval_count': 1732, 'prompt_eval_duration': 74809784000, 'eval_count': 121, 'eval_duration': 320
47567000}
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 ORDER BY ArtistName;
Output from LLM: 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 ORDER BY ArtistName;
Extracted SQL: 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 ORDER BY ArtistName
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 ORDER BY ArtistName
```

	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	23	Frank Zappa & Captain Beefheart
21	22	Led Zeppelin

```
22        24                    Marcos Valle
23        21                 Various Artists
Ollama parameters:
model=gemma: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\nThe DataFrame was produced using this query: SELECT a.ArtistId, a.Name AS Art
istName\nFROM \"artists\" a\nJOIN \"albums\" al ON a.ArtistId = al.ArtistId\nWHERE a.ArtistId IN (\n  SELEC
T 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 ORDER BY ArtistName\n\nThe following is in
formation about the resulting pandas DataFrame 'df': \nRunning df.dtypes gives:\n ArtistId      int64\nArt
istName    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 explana
tions -- just the code."}]
Ollama Response:
{'model': 'gemma:latest', 'created_at': '2024-06-14T02:27:28.028104225Z', 'message': {'role': 'assistant',
'content': "```python\nimport plotly.express as px\n\nfig = px.bar(df, x='ArtistName', color='ArtistName',
text='ArtistName')\nfig.update_traces(textposition='inside', texttemplate='%{text}')\nfig.update_layout(tit
le='Artists with Albums in Multiple Genres')\nfig.show()\n```"}, 'done_reason': 'stop', 'done': True, 'tota
l_duration': 31707521349, 'load_duration': 643906, 'prompt_eval_count': 267, 'prompt_eval_duration': 119721
85000, 'eval_count': 75, 'eval_duration': 19644386000}
```


Artists with Albums in Multiple Genres
count
1
0.8
0.6
0.4
0.2
0
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
Frank Zappa & Captain Beefheart
Led Zeppelin
Marcos Valle
Various Artists
ArtistName
ArtistName
AC/DC
Accept
Aerosmith
Alanis Morissette
Alice In Chains
Antônio Carlos Jobim
Apocalyptica
Audioslave
BackBeat
Billy Cobham

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.Ge
nreId\n GROUP BY g2.GenreId\n HAVING COUNT(g2.GenreId) > 1\n)\nGROUP BY a.ArtistId, a.Name ORDER BY Arti
stName',

	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	23	Frank Zappa & Captain Beefheart
21	22	Led Zeppelin
22	24	Marcos Valle
23	21	Various Artists,

```
Figure({
    'data': [{'alignmentgroup': 'True',
              'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
              'legendgroup': 'AC/DC',
              'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
              'name': 'AC/DC',
              'offsetgroup': 'AC/DC',
              'orientation': 'v',
              'showlegend': True,
              'text': array(['AC/DC'], dtype=object),
              'textposition': 'inside',
              'texttemplate': '%{text}',
              'type': 'bar',
```

```
 'x': array(['AC/DC'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Accept',
 'marker': {'color': '#EF553B', 'pattern': {'shape': ''}},
 'name': 'Accept',
 'offsetgroup': 'Accept',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Accept'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Accept'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Aerosmith',
 'marker': {'color': '#00cc96', 'pattern': {'shape': ''}},
 'name': 'Aerosmith',
 'offsetgroup': 'Aerosmith',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Aerosmith'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Aerosmith'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Alanis Morissette',
 'marker': {'color': '#ab63fa', 'pattern': {'shape': ''}},
 'name': 'Alanis Morissette',
 'offsetgroup': 'Alanis Morissette',
```

```
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Alanis Morissette'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Alanis Morissette'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Alice In Chains',
 'marker': {'color': '#FFA15A', 'pattern': {'shape': ''}},
 'name': 'Alice In Chains',
 'offsetgroup': 'Alice In Chains',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Alice In Chains'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Alice In Chains'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Antônio Carlos Jobim',
 'marker': {'color': '#19d3f3', 'pattern': {'shape': ''}},
 'name': 'Antônio Carlos Jobim',
 'offsetgroup': 'Antônio Carlos Jobim',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Antônio Carlos Jobim'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Antônio Carlos Jobim'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
```

```
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Apocalyptica',
 'marker': {'color': '#FF6692', 'pattern': {'shape': ''}},
 'name': 'Apocalyptica',
 'offsetgroup': 'Apocalyptica',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Apocalyptica'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Apocalyptica'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Audioslave',
 'marker': {'color': '#B6E880', 'pattern': {'shape': ''}},
 'name': 'Audioslave',
 'offsetgroup': 'Audioslave',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Audioslave'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Audioslave'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'BackBeat',
 'marker': {'color': '#FF97FF', 'pattern': {'shape': ''}},
 'name': 'BackBeat',
 'offsetgroup': 'BackBeat',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['BackBeat'], dtype=object),
 'textposition': 'inside',
```

```
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['BackBeat'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Billy Cobham',
 'marker': {'color': '#FECB52', 'pattern': {'shape': ''}},
 'name': 'Billy Cobham',
 'offsetgroup': 'Billy Cobham',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Billy Cobham'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Billy Cobham'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Black Label Society',
 'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
 'name': 'Black Label Society',
 'offsetgroup': 'Black Label Society',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Black Label Society'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Black Label Society'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Black Sabbath',
 'marker': {'color': '#EF553B', 'pattern': {'shape': ''}},
```

```
 'name': 'Black Sabbath',
 'offsetgroup': 'Black Sabbath',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Black Sabbath'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Black Sabbath'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Body Count',
 'marker': {'color': '#00cc96', 'pattern': {'shape': ''}},
 'name': 'Body Count',
 'offsetgroup': 'Body Count',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Body Count'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Body Count'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Bruce Dickinson',
 'marker': {'color': '#ab63fa', 'pattern': {'shape': ''}},
 'name': 'Bruce Dickinson',
 'offsetgroup': 'Bruce Dickinson',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Bruce Dickinson'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Bruce Dickinson'], dtype=object),
 'xaxis': 'x',
```

```
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Buddy Guy',
 'marker': {'color': '#FFA15A', 'pattern': {'shape': ''}},
 'name': 'Buddy Guy',
 'offsetgroup': 'Buddy Guy',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Buddy Guy'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Buddy Guy'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Caetano Veloso',
 'marker': {'color': '#19d3f3', 'pattern': {'shape': ''}},
 'name': 'Caetano Veloso',
 'offsetgroup': 'Caetano Veloso',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Caetano Veloso'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Caetano Veloso'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Chico Buarque',
 'marker': {'color': '#FF6692', 'pattern': {'shape': ''}},
 'name': 'Chico Buarque',
 'offsetgroup': 'Chico Buarque',
 'orientation': 'v',
 'showlegend': True,
```

```
 'text': array(['Chico Buarque'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Chico Buarque'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Chico Science & Nação Zumbi',
 'marker': {'color': '#B6E880', 'pattern': {'shape': ''}},
 'name': 'Chico Science & Nação Zumbi',
 'offsetgroup': 'Chico Science & Nação Zumbi',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Chico Science & Nação Zumbi'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Chico Science & Nação Zumbi'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Cidade Negra',
 'marker': {'color': '#FF97FF', 'pattern': {'shape': ''}},
 'name': 'Cidade Negra',
 'offsetgroup': 'Cidade Negra',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Cidade Negra'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Cidade Negra'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
```

```
 'legendgroup': 'Cláudio Zoli',
 'marker': {'color': '#FECB52', 'pattern': {'shape': ''}},
 'name': 'Cláudio Zoli',
 'offsetgroup': 'Cláudio Zoli',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Cláudio Zoli'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Cláudio Zoli'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Frank Zappa & Captain Beefheart',
 'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
 'name': 'Frank Zappa & Captain Beefheart',
 'offsetgroup': 'Frank Zappa & Captain Beefheart',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Frank Zappa & Captain Beefheart'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
 'x': array(['Frank Zappa & Captain Beefheart'], dtype=object),
 'xaxis': 'x',
 'y': array([1]),
 'yaxis': 'y'},
{'alignmentgroup': 'True',
 'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
 'legendgroup': 'Led Zeppelin',
 'marker': {'color': '#EF553B', 'pattern': {'shape': ''}},
 'name': 'Led Zeppelin',
 'offsetgroup': 'Led Zeppelin',
 'orientation': 'v',
 'showlegend': True,
 'text': array(['Led Zeppelin'], dtype=object),
 'textposition': 'inside',
 'texttemplate': '%{text}',
 'type': 'bar',
```

```
                    'x': array(['Led Zeppelin'], dtype=object),
                    'xaxis': 'x',
                    'y': array([1]),
                    'yaxis': 'y'},
                   {'alignmentgroup': 'True',
                    'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
                    'legendgroup': 'Marcos Valle',
                    'marker': {'color': '#00cc96', 'pattern': {'shape': ''}},
                    'name': 'Marcos Valle',
                    'offsetgroup': 'Marcos Valle',
                    'orientation': 'v',
                    'showlegend': True,
                    'text': array(['Marcos Valle'], dtype=object),
                    'textposition': 'inside',
                    'texttemplate': '%{text}',
                    'type': 'bar',
                    'x': array(['Marcos Valle'], dtype=object),
                    'xaxis': 'x',
                    'y': array([1]),
                    'yaxis': 'y'},
                   {'alignmentgroup': 'True',
                    'hovertemplate': 'ArtistName=%{text}<br>count=%{y}<extra></extra>',
                    'legendgroup': 'Various Artists',
                    'marker': {'color': '#ab63fa', 'pattern': {'shape': ''}},
                    'name': 'Various Artists',
                    'offsetgroup': 'Various Artists',
                    'orientation': 'v',
                    'showlegend': True,
                    'text': array(['Various Artists'], dtype=object),
                    'textposition': 'inside',
                    'texttemplate': '%{text}',
                    'type': 'bar',
                    'x': array(['Various Artists'], dtype=object),
                    'xaxis': 'x',
                    'y': array([1]),
                    'yaxis': 'y'}],
          'layout': {'barmode': 'relative',
                     'legend': {'title': {'text': 'ArtistName'}, 'tracegroupgap': 0},
                     'margin': {'t': 60},
                     'template': '...',
                     'title': {'text': 'Artists with Albums in Multiple Genres'},
                     'xaxis': {'anchor': 'y',
```

```
                            'categoryarray': [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,
                                              Frank Zappa & Captain Beefheart, Led
                                              Zeppelin, Marcos Valle, Various Artists],
                            'categoryorder': 'array',
                            'domain': [0.0, 1.0],
                            'title': {'text': 'ArtistName'}},
                  'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'count'}}}
 }))
```

Check completion time

In [ ]:

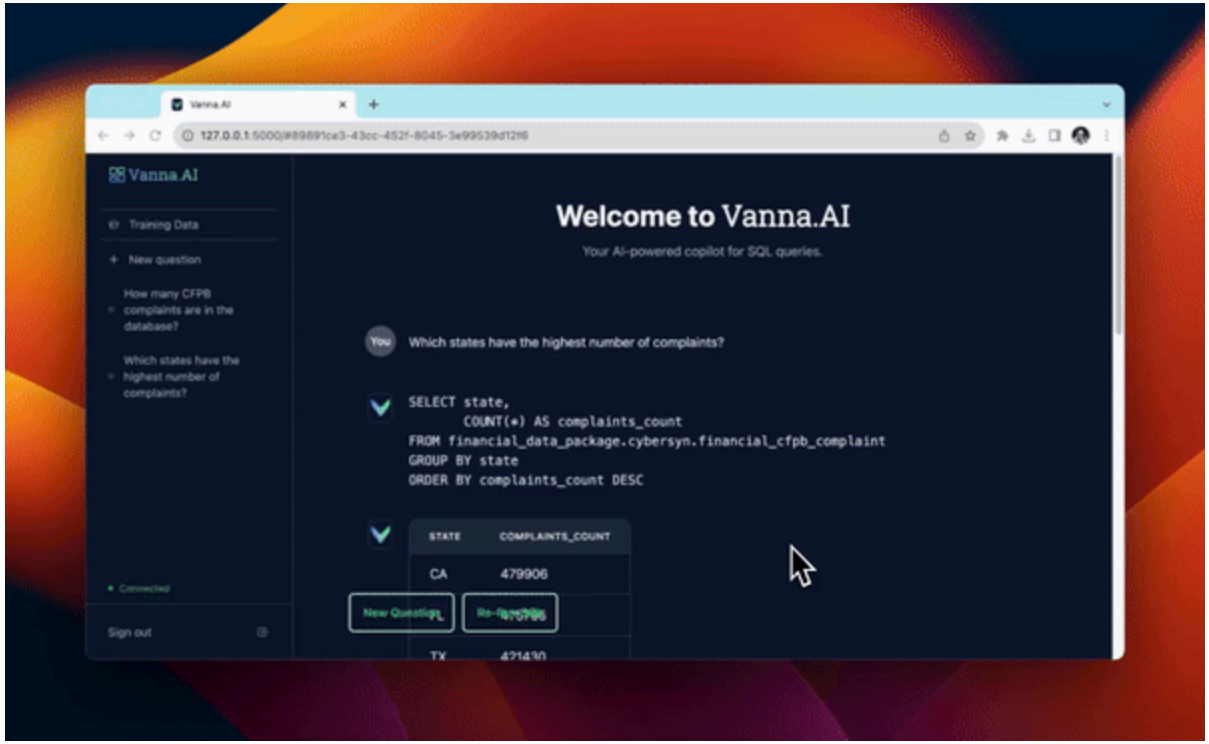
In [41]:
```
ts_stop = time()

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 'gemma' LLM took : 2713.56 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