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

!pip install 'vanna[chromadb]'

In [1]:
```
model_name = 'gpt-4'
file_db = "~/Downloads/chinook.sqlite"
```

In [2]:
```
from api_key_store import ApiKeyStore
s = ApiKeyStore()

openai_api_key = s.get_api_key(provider="OPENAI")
```

openai_api_key

In [3]:
```
from vanna.openai import OpenAI_Chat
from vanna.chromadb.chromadb_vector import ChromaDB_VectorStore
```

In [4]:
```
class MyVanna(ChromaDB_VectorStore, OpenAI_Chat):
    def __init__(self, config=None):
        ChromaDB_VectorStore.__init__(self, config=config)
        OpenAI_Chat.__init__(self, config=config)

config = {
    'api_key': openai_api_key,
    'model': model_name
}
vn = MyVanna(config=config)
```

Which database do you want to query?

- Postgres
- Microsoft SQL Server
- DuckDB
- Snowflake
- BigQuery
- [Selected] SQLite

- Other Database

  Use Vanna to generate queries for any SQL database

In [5]:
```
import os
import re
from time import time
```

In [6]:
```
# file_db = "./db/gpt3sql.sqlite"

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]:
```
clean_and_train = True  # False
```

In [9]:
```
hostname = os.uname().nodename
print("Hostname:", hostname)
```

```
Hostname: papa-game
```

In [10]:
```
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 [11]:
```
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 [12]:
```
if clean_and_train:
    remove_collections()
```

Training

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

In [13]:
```
# show training data
training_data = vn.get_training_data()
training_data
```

Out[13]:

id	question	content	training_data_type

In [14]:
```
df_ddl = vn.run_sql("SELECT type, sql FROM sqlite_master WHERE sql is not null")
```

In [15]:
```
df_ddl
```

Out[15]:

	type	sql
0	table	CREATE TABLE [Album]\n(\n [AlbumId] INTEGER...
1	table	CREATE TABLE [Artist]\n(\n [ArtistId] INTEG...
2	table	CREATE TABLE [Customer]\n(\n [CustomerId] I...
3	table	CREATE TABLE [Employee]\n(\n [EmployeeId] I...
4	table	CREATE TABLE [Genre]\n(\n [GenreId] INTEGER...
5	table	CREATE TABLE [Invoice]\n(\n [InvoiceId] INT...
6	table	CREATE TABLE [InvoiceLine]\n(\n [InvoiceLin...
7	table	CREATE TABLE [MediaType]\n(\n [MediaTypeId]...
8	table	CREATE TABLE [Playlist]\n(\n [PlaylistId] I...
9	table	CREATE TABLE [PlaylistTrack]\n(\n [Playlist...
10	table	CREATE TABLE [Track]\n(\n [TrackId] INTEGER...
11	index	CREATE INDEX [IFK_AlbumArtistId] ON [Album] ([...
12	index	CREATE INDEX [IFK_CustomerSupportRepId] ON [Cu...
13	index	CREATE INDEX [IFK_EmployeeReportsTo] ON [Emplo...
14	index	CREATE INDEX [IFK_InvoiceCustomerId] ON [Invoi...
15	index	CREATE INDEX [IFK_InvoiceLineInvoiceId] ON [In...
16	index	CREATE INDEX [IFK_InvoiceLineTrackId] ON [Invo...
17	index	CREATE INDEX [IFK_PlaylistTrackTrackId] ON [Pl...
18	index	CREATE INDEX [IFK_TrackAlbumId] ON [Track] ([A...
19	index	CREATE INDEX [IFK_TrackGenreId] ON [Track] ([G...
20	index	CREATE INDEX [IFK_TrackMediaTypeId] ON [Track]...

In [16]:

```python
if clean_and_train:
    for ddl in df_ddl['sql'].to_list():
        ddl = strip_brackets(ddl)
        vn.train(ddl=ddl)
    # Sometimes you may want to add documentation about your business terminology or definitions.
    vn.train(documentation="In the chinook database invoice means order")
```

```
Adding ddl: CREATE TABLE Album
(
    AlbumId INTEGER  NOT NULL,
    Title NVARCHAR(160)  NOT NULL,
    ArtistId INTEGER  NOT NULL,
    CONSTRAINT PK_Album PRIMARY KEY  (AlbumId),
    FOREIGN KEY (ArtistId) REFERENCES Artist (ArtistId)
                ON DELETE NO ACTION ON UPDATE NO ACTION
)
Adding ddl: CREATE TABLE Artist
(
    ArtistId INTEGER  NOT NULL,
    Name NVARCHAR(120),
    CONSTRAINT PK_Artist PRIMARY KEY  (ArtistId)
)
Adding ddl: CREATE TABLE Customer
(
    CustomerId INTEGER  NOT NULL,
    FirstName NVARCHAR(40)  NOT NULL,
    LastName NVARCHAR(20)  NOT NULL,
    Company NVARCHAR(80),
    Address NVARCHAR(70),
    City NVARCHAR(40),
    State NVARCHAR(40),
    Country NVARCHAR(40),
    PostalCode NVARCHAR(10),
    Phone NVARCHAR(24),
    Fax NVARCHAR(24),
    Email NVARCHAR(60)  NOT NULL,
    SupportRepId INTEGER,
    CONSTRAINT PK_Customer PRIMARY KEY  (CustomerId),
    FOREIGN KEY (SupportRepId) REFERENCES Employee (EmployeeId)
                ON DELETE NO ACTION ON UPDATE NO ACTION
)
Adding ddl: CREATE TABLE Employee
(
    EmployeeId INTEGER  NOT NULL,
    LastName NVARCHAR(20)  NOT NULL,
    FirstName NVARCHAR(20)  NOT NULL,
    Title NVARCHAR(30),
    ReportsTo INTEGER,
    BirthDate DATETIME,
    HireDate DATETIME,
    Address NVARCHAR(70),
    City NVARCHAR(40),
    State NVARCHAR(40),
    Country NVARCHAR(40),
    PostalCode NVARCHAR(10),
    Phone NVARCHAR(24),
    Fax NVARCHAR(24),
    Email NVARCHAR(60),
    CONSTRAINT PK_Employee PRIMARY KEY  (EmployeeId),
    FOREIGN KEY (ReportsTo) REFERENCES Employee (EmployeeId)
                ON DELETE NO ACTION ON UPDATE NO ACTION
)
Adding ddl: CREATE TABLE Genre
(
    GenreId INTEGER  NOT NULL,
    Name NVARCHAR(120),
    CONSTRAINT PK_Genre PRIMARY KEY  (GenreId)
)
Adding ddl: CREATE TABLE Invoice
(
    InvoiceId INTEGER  NOT NULL,
    CustomerId INTEGER  NOT NULL,
    InvoiceDate DATETIME  NOT NULL,
    BillingAddress NVARCHAR(70),
    BillingCity NVARCHAR(40),
    BillingState NVARCHAR(40),
    BillingCountry NVARCHAR(40),
    BillingPostalCode NVARCHAR(10),
    Total NUMERIC(10,2)  NOT NULL,
    CONSTRAINT PK_Invoice PRIMARY KEY  (InvoiceId),
    FOREIGN KEY (CustomerId) REFERENCES Customer (CustomerId)
                ON DELETE NO ACTION ON UPDATE NO ACTION
)
Adding ddl: CREATE TABLE InvoiceLine
(
    InvoiceLineId INTEGER  NOT NULL,
    InvoiceId INTEGER  NOT NULL,
    TrackId INTEGER  NOT NULL,
    UnitPrice NUMERIC(10,2)  NOT NULL,
    Quantity INTEGER  NOT NULL,
    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (InvoiceLineId),
    FOREIGN KEY (InvoiceId) REFERENCES Invoice (InvoiceId)
                ON DELETE NO ACTION ON UPDATE NO ACTION,
    FOREIGN KEY (TrackId) REFERENCES Track (TrackId)
                ON DELETE NO ACTION ON UPDATE NO ACTION
)
Adding ddl: CREATE TABLE MediaType
(
    MediaTypeId INTEGER  NOT NULL,
    Name NVARCHAR(120),
    CONSTRAINT PK_MediaType PRIMARY KEY  (MediaTypeId)
)
Adding ddl: CREATE TABLE Playlist
(
    PlaylistId INTEGER  NOT NULL,
    Name NVARCHAR(120),
    CONSTRAINT PK_Playlist PRIMARY KEY  (PlaylistId)
)
Adding ddl: CREATE TABLE PlaylistTrack
(
    PlaylistId INTEGER  NOT NULL,
    TrackId INTEGER  NOT NULL,
    CONSTRAINT PK_PlaylistTrack PRIMARY KEY  (PlaylistId, TrackId),
    FOREIGN KEY (PlaylistId) REFERENCES Playlist (PlaylistId)
                ON DELETE NO ACTION ON UPDATE NO ACTION,
    FOREIGN KEY (TrackId) REFERENCES Track (TrackId)
                ON DELETE NO ACTION ON UPDATE NO ACTION
)
Adding ddl: CREATE TABLE Track
(
    TrackId INTEGER  NOT NULL,
    Name NVARCHAR(200)  NOT NULL,
    AlbumId INTEGER,
    MediaTypeId INTEGER  NOT NULL,
    GenreId INTEGER,
    Composer NVARCHAR(220),
    Milliseconds INTEGER  NOT NULL,
    Bytes INTEGER,
    UnitPrice NUMERIC(10,2)  NOT NULL,
    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),
    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumId)
                ON DELETE NO ACTION ON UPDATE NO ACTION,
    FOREIGN KEY (GenreId) REFERENCES Genre (GenreId)
```

```
                ON DELETE NO ACTION ON UPDATE NO ACTION,
    FOREIGN KEY (MediaTypeId) REFERENCES MediaType (MediaTypeId)
                ON DELETE NO ACTION ON UPDATE NO ACTION
)
Adding ddl: CREATE INDEX IFK_AlbumArtistId ON Album (ArtistId)
Adding ddl: CREATE INDEX IFK_CustomerSupportRepId ON Customer (SupportRepId)
Adding ddl: CREATE INDEX IFK_EmployeeReportsTo ON Employee (ReportsTo)
Adding ddl: CREATE INDEX IFK_InvoiceCustomerId ON Invoice (CustomerId)
Adding ddl: CREATE INDEX IFK_InvoiceLineInvoiceId ON InvoiceLine (InvoiceId)
Adding ddl: CREATE INDEX IFK_InvoiceLineTrackId ON InvoiceLine (TrackId)
Adding ddl: CREATE INDEX IFK_PlaylistTrackTrackId ON PlaylistTrack (TrackId)
Adding ddl: CREATE INDEX IFK_TrackAlbumId ON Track (AlbumId)
Adding ddl: CREATE INDEX IFK_TrackGenreId ON Track (GenreId)
Adding ddl: CREATE INDEX IFK_TrackMediaTypeId ON Track (MediaTypeId)
Adding documentation....
```

In [ ]:

```

```

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 [17]:

```
ts_start = time()
```

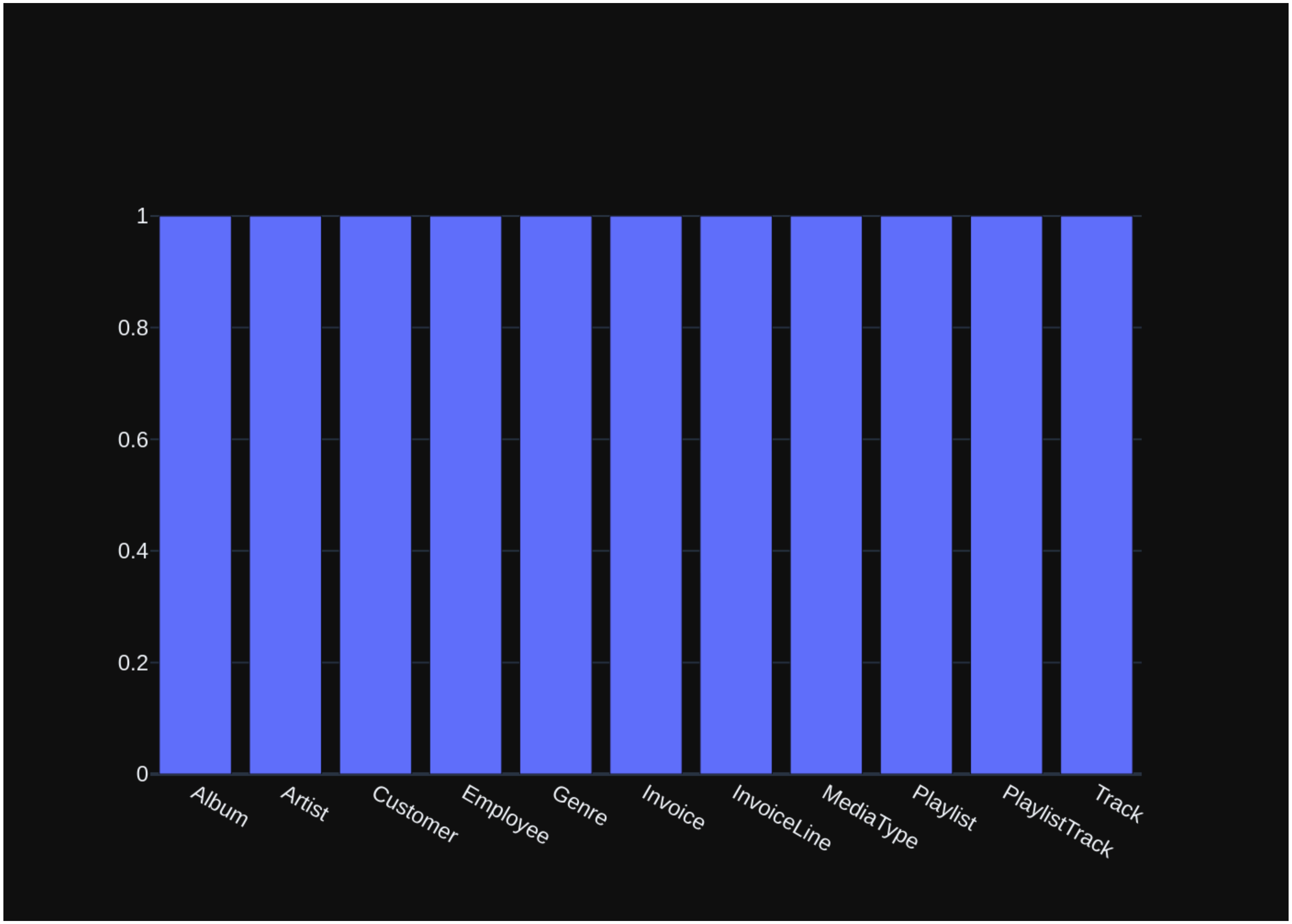
In [18]:

```
vn.ask(question="Show me a list of tables in the SQLite database")
```

```
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 fo
rmat instructions. \n===Tables \nCREATE TABLE Playlist\n(\n    PlaylistId INTEGER  NOT NULL,\n    Name NVAR
CHAR(120),\n    CONSTRAINT PK_Playlist PRIMARY KEY  (PlaylistId)\n)\n\nCREATE TABLE InvoiceLine\n(\n    Inv
oiceLineId INTEGER  NOT NULL,\n    InvoiceId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPr
ice NUMERIC(10,2)  NOT NULL,\n    Quantity INTEGER  NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY 
(InvoiceLineId),\n    FOREIGN KEY (InvoiceId) REFERENCES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON U
PDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE 
NO ACTION\n)\n\nCREATE TABLE PlaylistTrack\n(\n    PlaylistId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT 
NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY  (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) RE
FERENCES Playlist (PlaylistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) RE
FERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Track\n(\n    Tra
ckId INTEGER  NOT NULL,\n    Name NVARCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER 
NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Byte
s INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FO
REIGN KEY (AlbumId) REFERENCES Album (AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN 
KEY (GenreId) REFERENCES Genre (GenreId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (M
ediaTypeId) REFERENCES MediaType (MediaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE T
ABLE MediaType\n(\n    MediaTypeId INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK_MediaType
PRIMARY KEY  (MediaTypeId)\n)\n\nCREATE TABLE Artist\n(\n    ArtistId INTEGER  NOT NULL,\n    Name NVARCHAR
(120),\n    CONSTRAINT PK_Artist PRIMARY KEY  (ArtistId)\n)\n\nCREATE TABLE Album\n(\n    AlbumId INTEGER 
NOT NULL,\n    Title NVARCHAR(160)  NOT NULL,\n    ArtistId INTEGER  NOT NULL,\n    CONSTRAINT PK_Album PRI
MARY KEY  (AlbumId),\n    FOREIGN KEY (ArtistId) REFERENCES Artist (ArtistId) \n\t\tON DELETE NO ACTION ON 
UPDATE NO ACTION\n)\n\nCREATE TABLE Genre\n(\n    GenreId INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n   
CONSTRAINT PK_Genre PRIMARY KEY  (GenreId)\n)\n\nCREATE TABLE Invoice\n(\n    InvoiceId INTEGER  NOT NUL
L,\n    CustomerId INTEGER  NOT NULL,\n    InvoiceDate DATETIME  NOT NULL,\n    BillingAddress NVARCHAR(7
0),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n    B
illingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Invoice PRIMARY KEY 
(InvoiceId),\n    FOREIGN KEY (CustomerId) REFERENCES Customer (CustomerId) \n\t\tON DELETE NO ACTION ON UP
DATE NO ACTION\n)\n\nCREATE INDEX IFK_EmployeeReportsTo ON Employee (ReportsTo)\n\n\n===Additional Context 
\n\nIn the chinook database invoice means order\n\n===Response Guidelines \n1. If the provided context is s
ufficient, please generate a valid SQL query without any explanations for the question. \n2. If the provide
d context is almost sufficient but requires knowledge of a specific string in a particular column, please g
enerate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a com
ment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it can't be g
enerated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered befo
re, please repeat the answer exactly as it was given before. \n"}, {'role': 'user', 'content': 'Show me a l
ist of tables in the SQLite database'}]
Using model gpt-4 for 948.25 tokens (approx)
SELECT name FROM sqlite_master WHERE type='table';
SELECT name FROM sqlite_master WHERE type='table';
SELECT name FROM sqlite_master WHERE type='table';
```

```
             name
0           Album
1          Artist
2        Customer
3        Employee
4           Genre
5         Invoice
6     InvoiceLine
7       MediaType
8        Playlist
9   PlaylistTrack
10          Track
Using model gpt-4 for 168.0 tokens (approx)
```

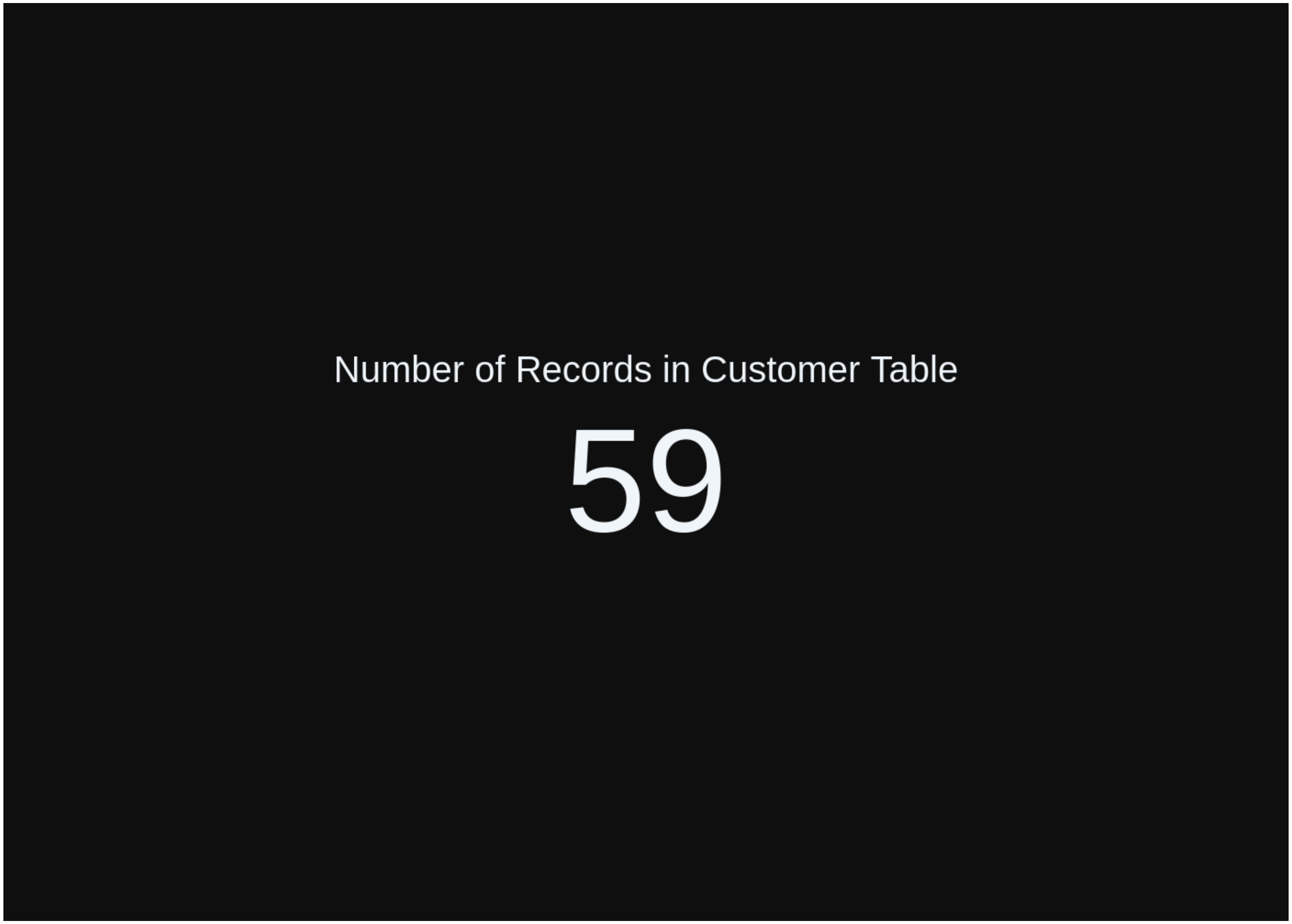

1
0.8
0.6
0.4
0.2
0
Album
Artist
Customer
Employee
Genre
Invoice
InvoiceLine
MediaType
Playlist
PlaylistTrack
Track

Out[18]:
```
("SELECT name FROM sqlite_master WHERE type='table';",
               name
 0            Album
 1           Artist
 2         Customer
 3         Employee
 4            Genre
 5          Invoice
 6      InvoiceLine
 7        MediaType
 8         Playlist
 9    PlaylistTrack
 10           Track,
 Figure({
     'data': [{'type': 'bar',
               'x': array(['Album', 'Artist', 'Customer', 'Employee', 'Genre', 'Invoice',
                           'InvoiceLine', 'MediaType', 'Playlist', 'PlaylistTrack', 'Track'],
                          dtype=object),
               'y': array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1])}],
     'layout': {'template': '...'}
 }))
```

In [19]:
```
vn.ask(question="How many records are in table called customer")
```

```
Number of requested results 10 is greater than number of elements in index 1, updating n_results = 1
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 fo
rmat instructions. \n===Tables \nCREATE TABLE Customer\n(\n    CustomerId INTEGER  NOT NULL,\n    FirstName
NVARCHAR(40)  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    Company NVARCHAR(80),\n    Address NVARC
HAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCH
AR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60)  NOT NULL,\n    SupportRepI
d INTEGER,\n    CONSTRAINT PK_Customer PRIMARY KEY  (CustomerId),\n    FOREIGN KEY (SupportRepId) REFERENCE
S Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Invoice\n(\n    In
voiceId INTEGER  NOT NULL,\n    CustomerId INTEGER  NOT NULL,\n    InvoiceDate DATETIME  NOT NULL,\n    Bil
lingAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountr
y NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK
_Invoice PRIMARY KEY  (InvoiceId),\n    FOREIGN KEY (CustomerId) REFERENCES Customer (CustomerId) \n\t\tON
DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE InvoiceLine\n(\n    InvoiceLineId INTEGER  NOT NUL
L,\n    InvoiceId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPrice NUMERIC(10,2)  NOT NUL
L,\n    Quantity INTEGER  NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (InvoiceLineId),\n    FOREI
GN KEY (InvoiceId) REFERENCES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREI
GN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABL
E Album\n(\n    AlbumId INTEGER  NOT NULL,\n    Title NVARCHAR(160)  NOT NULL,\n    ArtistId INTEGER  NOT N
ULL,\n    CONSTRAINT PK_Album PRIMARY KEY  (AlbumId),\n    FOREIGN KEY (ArtistId) REFERENCES Artist (Artist
Id) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON Invoice (Cust
omerId)\n\nCREATE TABLE Employee\n(\n    EmployeeId INTEGER  NOT NULL,\n    LastName NVARCHAR(20)  NOT NUL
L,\n    FirstName NVARCHAR(20)  NOT NULL,\n    Title NVARCHAR(30),\n    ReportsTo INTEGER,\n    BirthDate D
ATETIME,\n    HireDate DATETIME,\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(4
0),\n    Country NVARCHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(2
4),\n    Email NVARCHAR(60),\n    CONSTRAINT PK_Employee PRIMARY KEY  (EmployeeId),\n    FOREIGN KEY (Repor
tsTo) REFERENCES Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Tra
ck\n(\n    TrackId INTEGER  NOT NULL,\n    Name NVARCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTy
peId INTEGER  NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT N
ULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (Tra
ckId),\n    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTIO
N,\n    FOREIGN KEY (GenreId) REFERENCES Genre (GenreId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n
FOREIGN KEY (MediaTypeId) REFERENCES MediaType (MediaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION
\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON Customer (SupportRepId)\n\nCREATE TABLE Playlist\n(\n    Pl
aylistId INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK_Playlist PRIMARY KEY  (PlaylistId)
\n)\n\nCREATE TABLE PlaylistTrack\n(\n    PlaylistId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n
CONSTRAINT PK_PlaylistTrack PRIMARY KEY  (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES Pl
aylist (PlaylistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Tr
ack (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n\n===Additional Context \n\nIn the chinoo
k 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 almos
t sufficient but requires knowledge of a specific string in a particular column, please generate an interme
diate SQL query to find the distinct strings in that column. Prepend the query with a comment saying interm
ediate_sql \n3. If the provided context is insufficient, please explain why it can't be generated. \n4. Ple
ase 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': 'Show me a list of tables in t
he SQLite database'}, {'role': 'assistant', 'content': "SELECT name FROM sqlite_master WHERE type='tabl
e';"}, {'role': 'user', 'content': 'How many records are in table called customer'}]
Using model gpt-4 for 1174.75 tokens (approx)
SELECT COUNT(*) FROM Customer;
SELECT COUNT(*) FROM Customer;
SELECT COUNT(*) FROM Customer;
   COUNT(*)
0        59
Using model gpt-4 for 163.25 tokens (approx)
```


Number of Records in Customer Table

59

```
Out[19]: ('SELECT COUNT(*) FROM Customer;',
             COUNT(*)
          0        59,
          Figure({
              'data': [{'mode': 'number',
                        'title': {'text': 'Number of Records in Customer Table'},
                        'type': 'indicator',
                        'value': 59}],
              'layout': {'template': '...'}
          }))
```

```
In [20]: vn.ask(question="How many customers are there")
```

```
Number of requested results 10 is greater than number of elements in index 2, updating n_results = 2
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 fo
rmat instructions. \n===Tables \nCREATE TABLE Customer\n(\n    CustomerId INTEGER  NOT NULL,\n    FirstName
NVARCHAR(40)  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    Company NVARCHAR(80),\n    Address NVARC
HAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCH
AR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60)  NOT NULL,\n    SupportRepI
d INTEGER,\n    CONSTRAINT PK_Customer PRIMARY KEY  (CustomerId),\n    FOREIGN KEY (SupportRepId) REFERENCE
S Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_CustomerSuppor
tRepId ON Customer (SupportRepId)\n\nCREATE TABLE Invoice\n(\n    InvoiceId INTEGER  NOT NULL,\n    Custome
rId INTEGER  NOT NULL,\n    InvoiceDate DATETIME  NOT NULL,\n    BillingAddress NVARCHAR(70),\n    BillingC
ity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n    BillingPostalCode
NVARCHAR(10),\n    Total NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Invoice PRIMARY KEY  (InvoiceId),\n
FOREIGN KEY (CustomerId) REFERENCES Customer (CustomerId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)
\n\nCREATE INDEX IFK_InvoiceCustomerId ON Invoice (CustomerId)\n\nCREATE TABLE InvoiceLine\n(\n    InvoiceL
ineId INTEGER  NOT NULL,\n    InvoiceId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPrice N
UMERIC(10,2)  NOT NULL,\n    Quantity INTEGER  NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (Invoi
ceLineId),\n    FOREIGN KEY (InvoiceId) REFERENCES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON UPDATE
NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACT
ION\n)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON InvoiceLine (InvoiceId)\n\nCREATE TABLE Track\n(\n    Tr
ackId INTEGER  NOT NULL,\n    Name NVARCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER
NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Byte
s INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FO
REIGN KEY (AlbumId) REFERENCES Album (AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN
KEY (GenreId) REFERENCES Genre (GenreId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (M
ediaTypeId) REFERENCES MediaType (MediaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE I
NDEX IFK_InvoiceLineTrackId ON InvoiceLine (TrackId)\n\nCREATE TABLE Album\n(\n    AlbumId INTEGER  NOT NUL
L,\n    Title NVARCHAR(160)  NOT NULL,\n    ArtistId INTEGER  NOT NULL,\n    CONSTRAINT PK_Album PRIMARY KE
Y  (AlbumId),\n    FOREIGN KEY (ArtistId) REFERENCES Artist (ArtistId) \n\t\tON DELETE NO ACTION ON UPDATE
NO ACTION\n)\n\nCREATE TABLE Employee\n(\n    EmployeeId INTEGER  NOT NULL,\n    LastName NVARCHAR(20)  NOT
NULL,\n    FirstName NVARCHAR(20)  NOT NULL,\n    Title NVARCHAR(30),\n    ReportsTo INTEGER,\n    BirthDat
e DATETIME,\n    HireDate DATETIME,\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR
(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(2
4),\n    Email NVARCHAR(60),\n    CONSTRAINT PK_Employee PRIMARY KEY  (EmployeeId),\n    FOREIGN KEY (Repor
tsTo) REFERENCES Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n\n===Additional
Context \n\nIn the chinook database invoice means order\n\n===Response Guidelines \n1. If the provided cont
ext 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, p
lease generate an intermediate SQL query to find the distinct strings in that column. Prepend the query wit
h a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it ca
n't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answe
red before, please repeat the answer exactly as it was given before. \n"}, {'role': 'user', 'content': 'How
many records are in table called customer'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM Custome
r;'}, {'role': 'user', 'content': 'Show me a list of tables in the SQLite database'}, {'role': 'assistant',
'content': "SELECT name FROM sqlite_master WHERE type='table';"}, {'role': 'user', 'content': 'How many cus
tomers are there'}]
Using model gpt-4 for 1095.0 tokens (approx)
SELECT COUNT(*) FROM Customer;
SELECT COUNT(*) FROM Customer;
SELECT COUNT(*) FROM Customer;
   COUNT(*)
0        59
Using model gpt-4 for 159.0 tokens (approx)
```

```
Out[20]: ('SELECT COUNT(*) FROM Customer;',
             COUNT(*)
           0       59,
           Figure({
               'data': [{'mode': 'number', 'type': 'indicator', 'value': 59}], 'layout': {'template': '...'}
           }))
```

In [ ]:

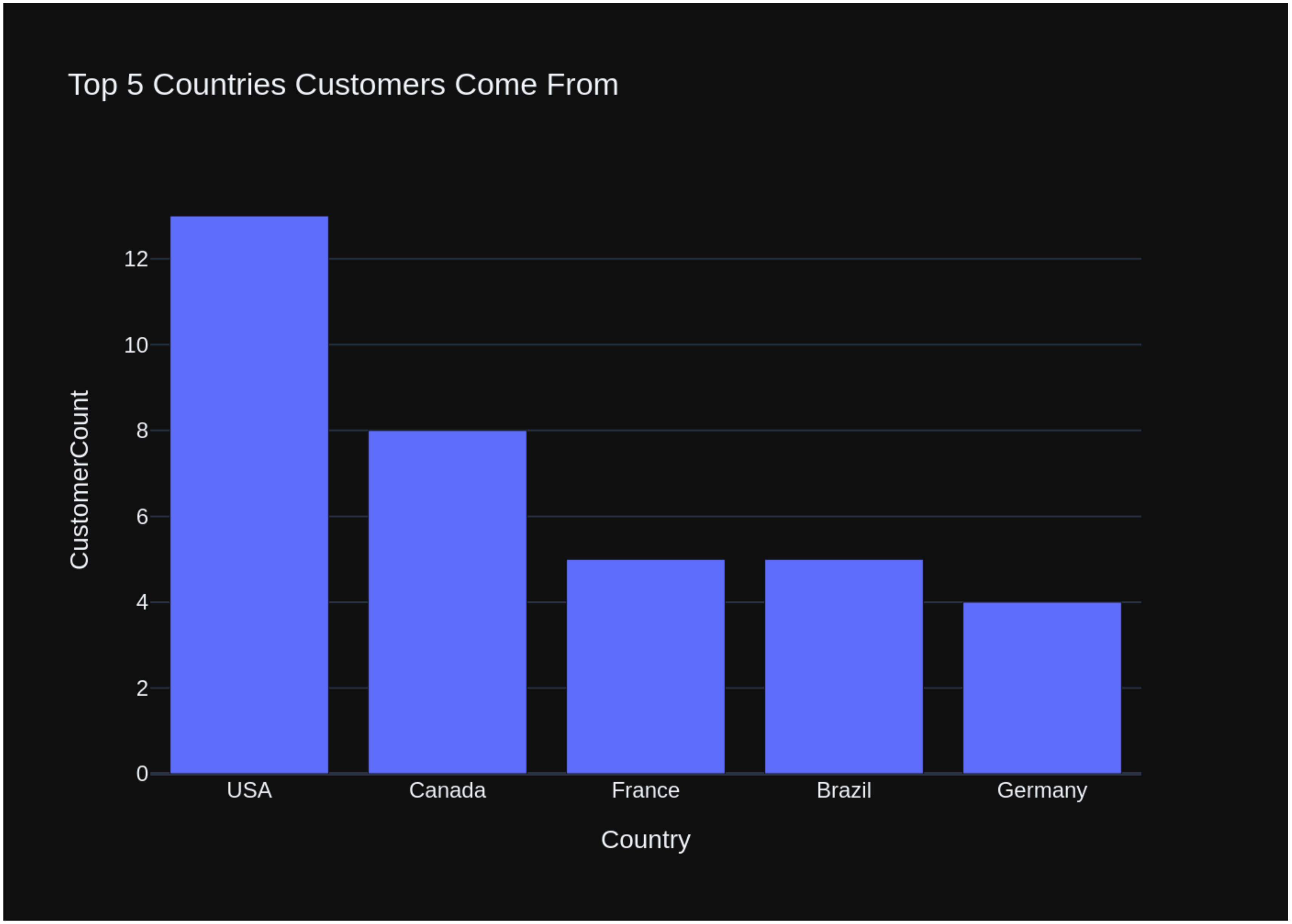
```
```

In [21]:

```
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 3, updating n_results = 3
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 fo
rmat instructions. \n===Tables \nCREATE TABLE Customer\n(\n    CustomerId INTEGER  NOT NULL,\n    FirstName
NVARCHAR(40)  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    Company NVARCHAR(80),\n    Address NVARC
HAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCH
AR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60)  NOT NULL,\n    SupportRepI
d INTEGER,\n    CONSTRAINT PK_Customer PRIMARY KEY  (CustomerId),\n    FOREIGN KEY (SupportRepId) REFERENCE
S Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Invoice\n(\n    In
voiceId INTEGER  NOT NULL,\n    CustomerId INTEGER  NOT NULL,\n    InvoiceDate DATETIME  NOT NULL,\n    Bil
lingAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountr
y NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK
_Invoice PRIMARY KEY  (InvoiceId),\n    FOREIGN KEY (CustomerId) REFERENCES Customer (CustomerId) \n\t\tON
DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE InvoiceLine\n(\n    InvoiceLineId INTEGER  NOT NUL
L,\n    InvoiceId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPrice NUMERIC(10,2)  NOT NUL
L,\n    Quantity INTEGER  NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (InvoiceLineId),\n    FOREI
GN KEY (InvoiceId) REFERENCES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREI
GN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABL
E Employee\n(\n    EmployeeId INTEGER  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    FirstName NVARC
HAR(20)  NOT NULL,\n    Title NVARCHAR(30),\n    ReportsTo INTEGER,\n    BirthDate DATETIME,\n    HireDate
DATETIME,\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR
(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(6
0),\n    CONSTRAINT PK_Employee PRIMARY KEY  (EmployeeId),\n    FOREIGN KEY (ReportsTo) REFERENCES Employee
(EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE MediaType\n(\n    MediaTypeId
INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK_MediaType PRIMARY KEY  (MediaTypeId)\n)\n\nC
REATE TABLE Playlist\n(\n    PlaylistId INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK_Play
list PRIMARY KEY  (PlaylistId)\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON Customer (SupportRepId)\n\nCR
EATE TABLE PlaylistTrack\n(\n    PlaylistId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    CONSTRA
INT PK_PlaylistTrack PRIMARY KEY  (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES Playlist
(PlaylistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (Tr
ackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Track\n(\n    TrackId INTEGER  NOT
NULL,\n    Name NVARCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER  NOT NULL,\n    Gen
reId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Bytes INTEGER,\n    Un
itPrice NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FOREIGN KEY (AlbumI
d) REFERENCES Album (AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REF
ERENCES Genre (GenreId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFER
ENCES MediaType (MediaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Album\n(\n
AlbumId INTEGER  NOT NULL,\n    Title NVARCHAR(160)  NOT NULL,\n    ArtistId INTEGER  NOT NULL,\n    CONSTR
AINT PK_Album PRIMARY KEY  (AlbumId),\n    FOREIGN KEY (ArtistId) REFERENCES Artist (ArtistId) \n\t\tON DEL
ETE NO ACTION ON UPDATE NO ACTION\n)\n\n\n===Additional Context \n\nIn the chinook database invoice means o
rder\n\n===Response Guidelines \n1. If the provided context is sufficient, please generate a valid SQL quer
y 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 th
e distinct strings in that column. Prepend the query with a comment saying intermediate_sql \n3. If the pro
vided 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 w
as given before. \n"}, {'role': 'user', 'content': 'How many customers are there'}, {'role': 'assistant',
'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'content': 'How many records are in table ca
lled customer'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'cont
ent': 'Show me a list of tables in the SQLite database'}, {'role': 'assistant', 'content': "SELECT name FRO
M sqlite_master WHERE type='table';"}, {'role': 'user', 'content': 'what are the top 5 countries that custo
mers come from?'}]
Using model gpt-4 for 1230.75 tokens (approx)
SELECT Country, COUNT(*) as CustomerCount
FROM Customer
GROUP BY Country
ORDER BY CustomerCount DESC
LIMIT 5;
SELECT Country, COUNT(*) as CustomerCount
FROM Customer
GROUP BY Country
ORDER BY CustomerCount DESC
LIMIT 5;
SELECT Country, COUNT(*) as CustomerCount
FROM Customer
GROUP BY Country
ORDER BY CustomerCount DESC
LIMIT 5;
   Country  CustomerCount
0      USA             13
1   Canada              8
2   France              5
3   Brazil              5
4  Germany              4
Using model gpt-4 for 192.75 tokens (approx)
```


Top 5 Countries Customers Come From
CustomerCount
0
2
4
6
8
10
12
USA
Canada
France
Brazil
Germany
Country

```
Out[21]: ('SELECT Country, COUNT(*) as CustomerCount\nFROM Customer\nGROUP BY Country\nORDER BY CustomerCount DESC
\nLIMIT 5;',
    Country  CustomerCount
 0      USA             13
 1   Canada              8
 2   France              5
 3   Brazil              5
 4  Germany              4,
 Figure({
     'data': [{'alignmentgroup': 'True',
               'hovertemplate': 'Country=%{x}<br>CustomerCount=%{y}<extra></extra>',
               'legendgroup': '',
               'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
               'name': '',
               'offsetgroup': '',
               'orientation': 'v',
               'showlegend': False,
               'textposition': 'auto',
               'type': 'bar',
               'x': array(['USA', 'Canada', 'France', 'Brazil', 'Germany'], dtype=object),
               'xaxis': 'x',
               'y': array([13,  8,  5,  5,  4]),
               'yaxis': 'y'}],
     'layout': {'barmode': 'relative',
                'legend': {'tracegroupgap': 0},
                'template': '...',
                'title': {'text': 'Top 5 Countries Customers Come From'},
                'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Country'}},
                'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'CustomerCount'}}}
 }))
```

More SQL questions

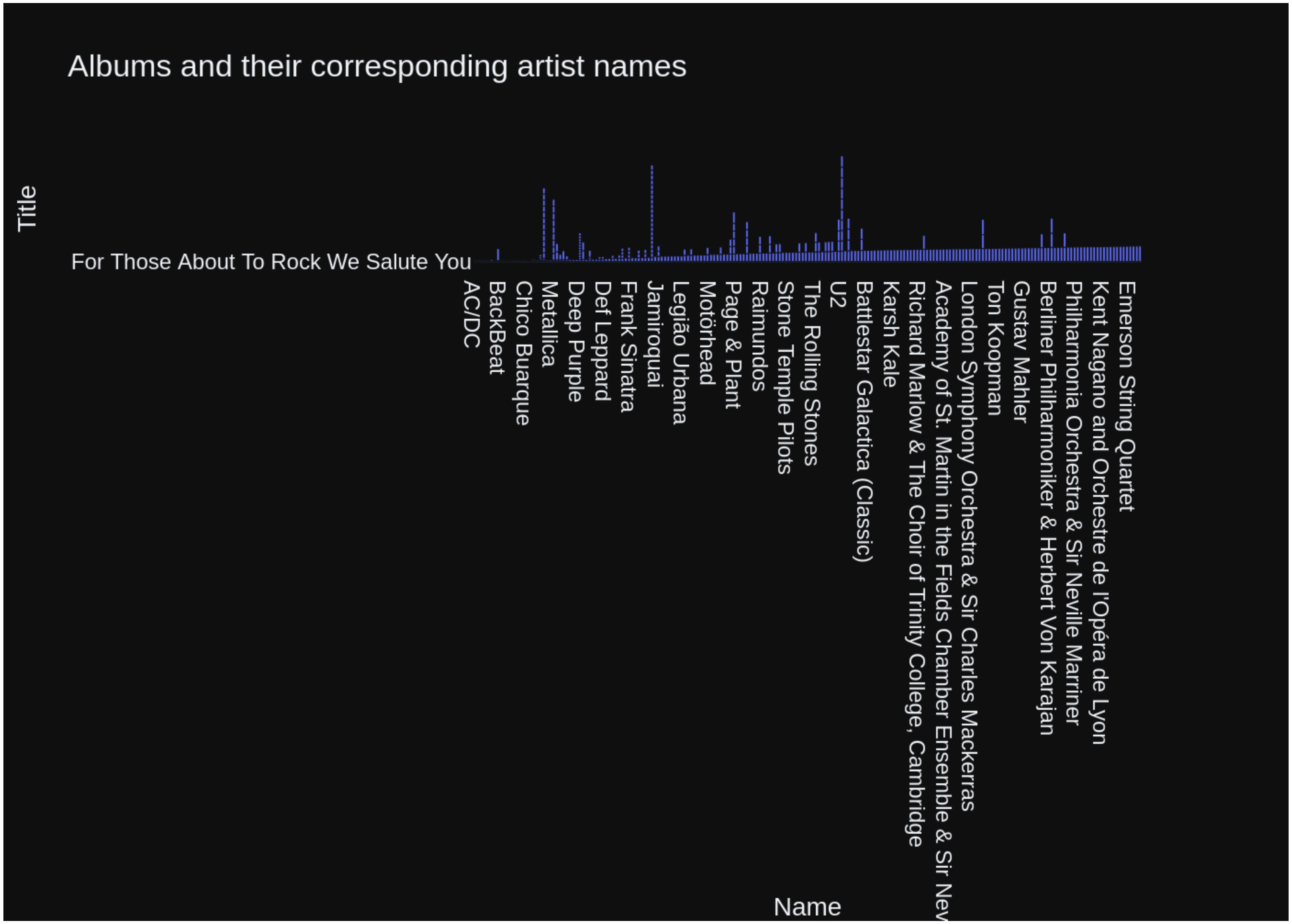
see `sample-sql-queries-sqlite-chinook.ipynb`

```
In [22]: 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 4, updating n_results = 4
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 fo
rmat instructions. \n===Tables \nCREATE INDEX IFK_AlbumArtistId ON Album (ArtistId)\n\nCREATE TABLE Album\n
(\n    AlbumId INTEGER  NOT NULL,\n    Title NVARCHAR(160)  NOT NULL,\n    ArtistId INTEGER  NOT NULL,\n
CONSTRAINT PK_Album PRIMARY KEY  (AlbumId),\n    FOREIGN KEY (ArtistId) REFERENCES Artist (ArtistId) \n\t\t
ON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Track\n(\n    TrackId INTEGER  NOT NULL,\n    Na
me NVARCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER  NOT NULL,\n    GenreId INTEGE
R,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUM
ERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FOREIGN KEY (AlbumId) REFERENCE
S Album (AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES Genr
e (GenreId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES MediaT
ype (MediaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_TrackAlbumId ON Trac
k (AlbumId)\n\nCREATE TABLE Artist\n(\n    ArtistId INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTR
AINT PK_Artist PRIMARY KEY  (ArtistId)\n)\n\nCREATE INDEX IFK_TrackGenreId ON Track (GenreId)\n\nCREATE IND
EX IFK_PlaylistTrackTrackId ON PlaylistTrack (TrackId)\n\nCREATE INDEX IFK_TrackMediaTypeId ON Track (Media
TypeId)\n\nCREATE TABLE Playlist\n(\n    PlaylistId INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTR
AINT PK_Playlist PRIMARY KEY  (PlaylistId)\n)\n\nCREATE TABLE PlaylistTrack\n(\n    PlaylistId INTEGER  NOT
NULL,\n    TrackId INTEGER  NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY  (PlaylistId, TrackI
d),\n    FOREIGN KEY (PlaylistId) REFERENCES Playlist (PlaylistId) \n\t\tON DELETE NO ACTION ON UPDATE NO A
CTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION
\n)\n\n\n===Additional Context \n\nIn the chinook database invoice means order\n\n===Response Guidelines \n
1. If the provided context is sufficient, please generate a valid SQL query without any explanations for th
e question. \n2. If the provided context is almost sufficient but requires knowledge of a specific string i
n a particular column, please generate an intermediate SQL query to find the distinct strings in that colum
n. Prepend the query with a comment saying intermediate_sql \n3. If the provided context is insufficient, p
lease explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question h
as been asked and answered before, please repeat the answer exactly as it was given before. \n"}, {'role':
'user', 'content': 'Show me a list of tables in the SQLite database'}, {'role': 'assistant', 'content': "SE
LECT name FROM sqlite_master WHERE type='table';"}, {'role': 'user', 'content': 'what are the top 5 countri
es that customers come from?'}, {'role': 'assistant', 'content': 'SELECT Country, COUNT(*) as CustomerCount
\nFROM Customer\nGROUP BY Country\nORDER BY CustomerCount DESC\nLIMIT 5;'}, {'role': 'user', 'content': 'Ho
w many records are in table called customer'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM Custo
mer;'}, {'role': 'user', 'content': 'How many customers are there'}, {'role': 'assistant', 'content': 'SELE
CT COUNT(*) FROM Customer;'}, {'role': 'user', 'content': '  \n    List all albums and their corresponding
artist names  \n'}]
Using model gpt-4 for 801.0 tokens (approx)
SELECT Album.Title, Artist.Name
FROM Album
JOIN Artist ON Album.ArtistId = Artist.ArtistId;
SELECT Album.Title, Artist.Name
FROM Album
JOIN Artist ON Album.ArtistId = Artist.ArtistId;
```

```
SELECT Album.Title, Artist.Name
FROM Album
JOIN Artist ON Album.ArtistId = Artist.ArtistId;
                                                  Title  \
0                  For Those About To Rock We Salute You
1                                      Balls to the Wall
2                                      Restless and Wild
3                                      Let There Be Rock
4                                               Big Ones
..                                                   ...
342                               Respighi:Pines of Rome
343  Schubert: The Late String Quartets & String Qu...
344                                Monteverdi: L'Orfeo
345                               Mozart: Chamber Music
346  Koyaanisqatsi (Soundtrack from the Motion Pict...

                                                   Name
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 2 columns]
Using model gpt-4 for 186.25 tokens (approx)
```


Albums and their corresponding artist names
Title
For Those About To Rock We Salute You
AC/DC
BackBeat
Chico Buarque
Metallica
Deep Purple
Def Leppard
Frank Sinatra
Jamiroquai
Legião Urbana
Motörhead
Page & Plant
Raimundos
Stone Temple Pilots
The Rolling Stones
U2
Battlestar Galactica (Classic)
Karsh Kale
Richard Marlow & The Choir of Trinity College, Cambridge
Academy of St. Martin in the Fields Chamber Ensemble & Sir Nev
London Symphony Orchestra & Sir Charles Mackerras
Ton Koopman
Gustav Mahler
Berliner Philharmoniker & Herbert Von Karajan
Philharmonia Orchestra & Sir Neville Marriner
Kent Nagano and Orchestre de l'Opéra de Lyon
Emerson String Quartet
Name

```
Out[22]:  ('SELECT Album.Title, Artist.Name\nFROM Album\nJOIN Artist ON Album.ArtistId = Artist.ArtistId;',
                                                       Title  \
           0                For Those About To Rock We Salute You
           1                                   Balls to the Wall
           2                                   Restless and Wild
           3                                   Let There Be Rock
           4                                            Big Ones
           ..                                                ...
           342                            Respighi:Pines of Rome
           343  Schubert: The Late String Quartets & String Qu...
           344                             Monteverdi: L'Orfeo
           345                            Mozart: Chamber Music
           346  Koyaanisqatsi (Soundtrack from the Motion Pict...

                                                        Name
           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 2 columns],
           Figure({
               'data': [{'alignmentgroup': 'True',
                         'hovertemplate': 'Name=%{x}<br>Title=%{y}<extra></extra>',
                         'legendgroup': '',
                         'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
                         'name': '',
                         'offsetgroup': '',
                         'orientation': 'v',
                         'showlegend': False,
                         'textposition': 'auto',
                         'type': 'bar',
                         'x': array(['AC/DC', 'Accept', 'Accept', ...,
                                     'C. Monteverdi, Nigel Rogers - Chiaroscuro; London Baroque; London Cornett & Sa
          ckbu',
                                     'Nash Ensemble', 'Philip Glass Ensemble'], dtype=object),
                         'xaxis': 'x',
                         'y': 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),
                         'yaxis': 'y'}],
               'layout': {'barmode': 'relative',
                          'legend': {'tracegroupgap': 0},
                          'template': '...',
                          'title': {'text': 'Albums and their corresponding artist names'},
                          'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Name'}},
                          'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'Title'}}}
           }))
```

In [23]:

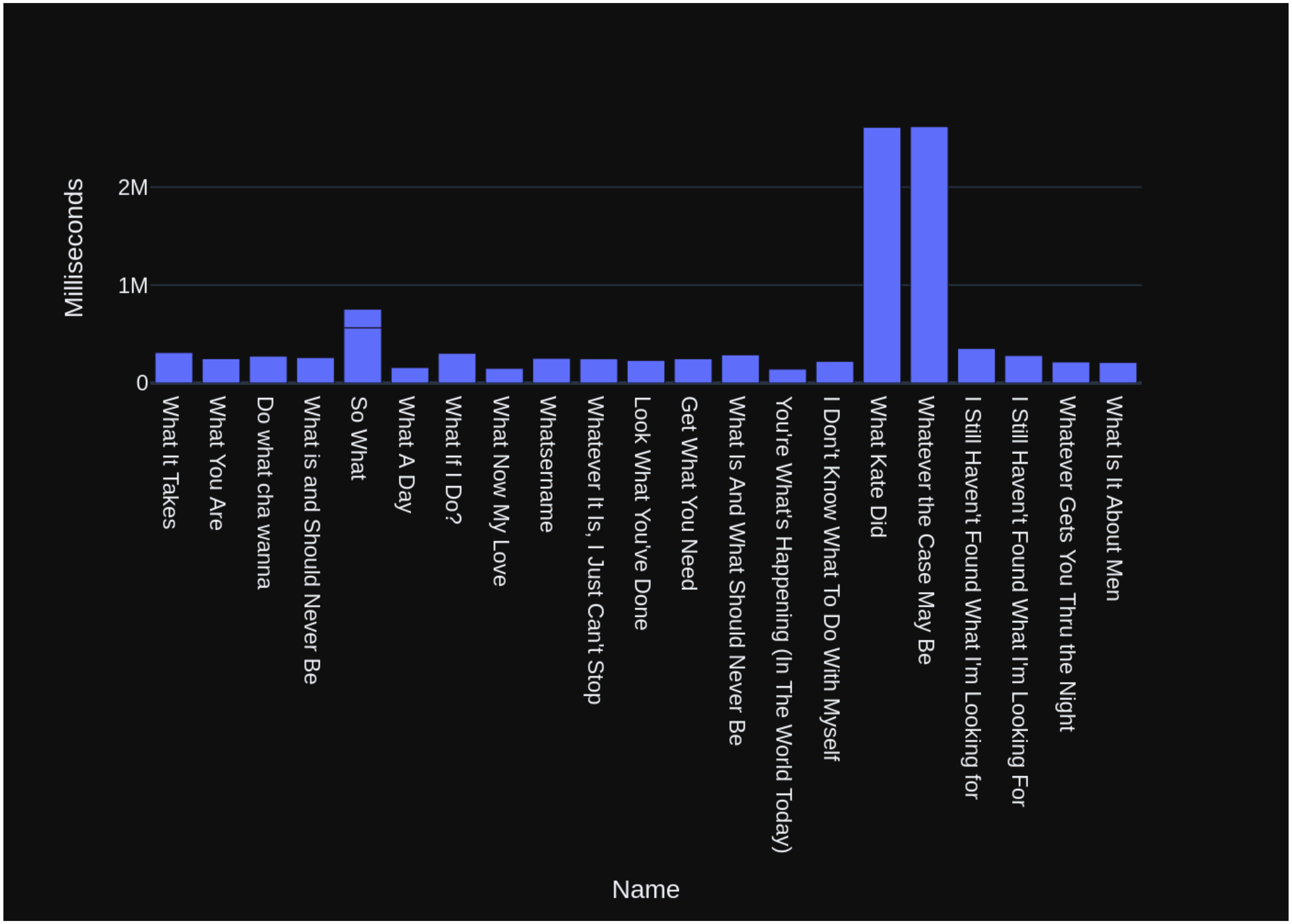
```
question = """
    Find all tracks with a name containing "What" (case-insensitive)
"""

vn.ask(question=question)
```

```
Number of requested results 10 is greater than number of elements in index 5, updating n_results = 5
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 fo
rmat instructions. \n===Tables \nCREATE INDEX IFK_TrackGenreId ON Track (GenreId)\n\nCREATE TABLE Track\n
(\n    TrackId INTEGER  NOT NULL,\n    Name NVARCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId
INTEGER  NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT NUL
L,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (Track
Id),\n    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n
FOREIGN KEY (GenreId) REFERENCES Genre (GenreId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIG
N KEY (MediaTypeId) REFERENCES MediaType (MediaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n
CREATE INDEX IFK_TrackAlbumId ON Track (AlbumId)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON PlaylistTrack
(TrackId)\n\nCREATE INDEX IFK_TrackMediaTypeId ON Track (MediaTypeId)\n\nCREATE INDEX IFK_InvoiceLineTrackI
d ON InvoiceLine (TrackId)\n\nCREATE TABLE PlaylistTrack\n(\n    PlaylistId INTEGER  NOT NULL,\n    TrackId
INTEGER  NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY  (PlaylistId, TrackId),\n    FOREIGN KEY (P
laylistId) REFERENCES Playlist (PlaylistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY
(TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_A
lbumArtistId ON Album (ArtistId)\n\nCREATE TABLE Album\n(\n    AlbumId INTEGER  NOT NULL,\n    Title NVARCH
AR(160)  NOT NULL,\n    ArtistId INTEGER  NOT NULL,\n    CONSTRAINT PK_Album PRIMARY KEY  (AlbumId),\n    F
OREIGN KEY (ArtistId) REFERENCES Artist (ArtistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREA
TE TABLE Playlist\n(\n    PlaylistId INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK_Playlis
t PRIMARY KEY  (PlaylistId)\n)\n\n\n===Additional Context \n\nIn the chinook database invoice means order\n
\n===Response Guidelines \n1. If the provided context is sufficient, please generate a valid SQL query with
out any explanations for the question. \n2. If the provided context is almost sufficient but requires knowl
edge of a specific string in a particular column, please generate an intermediate SQL query to find the dis
tinct 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 gi
ven before. \n"}, {'role': 'user', 'content': '  \n    List all albums and their corresponding artist names
\n'}, {'role': 'assistant', 'content': 'SELECT Album.Title, Artist.Name\nFROM Album\nJOIN Artist ON Album.A
rtistId = Artist.ArtistId;'}, {'role': 'user', 'content': 'Show me a list of tables in the SQLite databas
e'}, {'role': 'assistant', 'content': "SELECT name FROM sqlite_master WHERE type='table';"}, {'role': 'use
r', 'content': 'what are the top 5 countries that customers come from?'}, {'role': 'assistant', 'content':
'SELECT Country, COUNT(*) as CustomerCount\nFROM Customer\nGROUP BY Country\nORDER BY CustomerCount DESC\nL
IMIT 5;'}, {'role': 'user', 'content': 'How many customers are there'}, {'role': 'assistant', 'content': 'S
ELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'content': 'How many records are in table called custome
r'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'content': '  \n
Find all tracks with a name containing "What" (case-insensitive)\n'}]
Using model gpt-4 for 824.75 tokens (approx)
SELECT * FROM Track
WHERE Name LIKE '%What%' COLLATE NOCASE;
SELECT * FROM Track
WHERE Name LIKE '%What%' COLLATE NOCASE;
SELECT * FROM Track
```

```
WHERE Name LIKE '%What%' COLLATE NOCASE;
    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
Using model gpt-4 for 227.0 tokens (approx)
```


Milliseconds
2M
1M
0
What It Takes
What You Are
Do what cha wanna
What is and Should Never Be
So What
What A Day
What If I Do?
What Now My Love
Whatsername
Whatever It Is, I Just Can't Stop
Look What You've Done
Get What You Need
What Is And What Should Never Be
You're What's Happening (In The World Today)
I Don't Know What To Do With Myself
What Kate Did
Whatever the Case May Be
I Still Haven't Found What I'm Looking for
I Still Haven't Found What I'm Looking For
Whatever Gets You Thru the Night
What Is It About Men
Name

```
Out[23]:  ("SELECT * FROM Track\nWHERE Name LIKE '%What%' COLLATE NOCASE;",
               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   
```

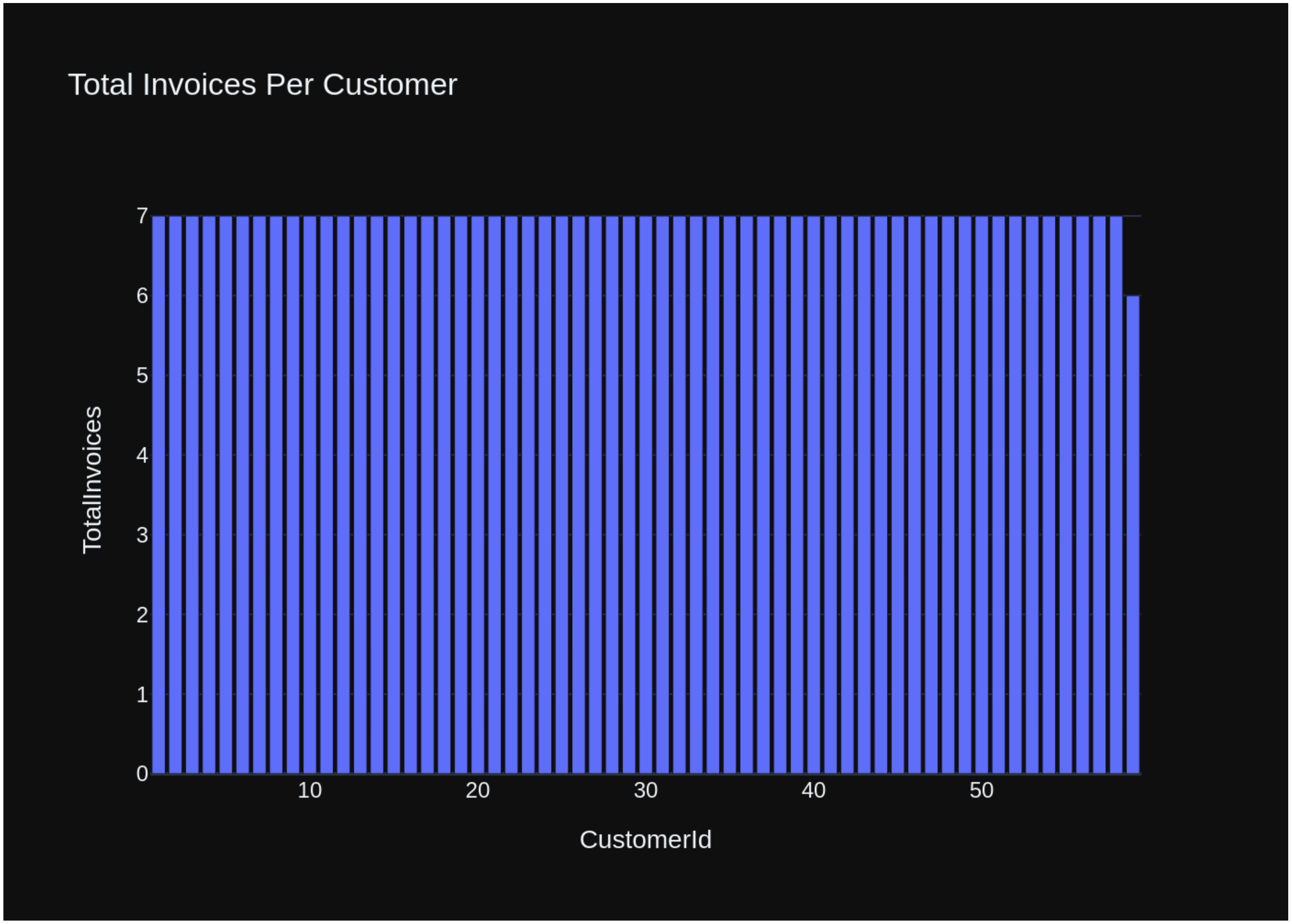
```
16           3       19                                                   None
17           3       19                                                   None
18           1        1      Bono/Clayton, Adam/Mullen Jr., Larry/The Edge
19           1        1                                                     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  ,
Figure({
    'data': [{'alignmentgroup': 'True',
              'hovertemplate': 'Name=%{x}<br>Milliseconds=%{y}<extra></extra>',
              'legendgroup': '',
              'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
              'name': '',
              'offsetgroup': '',
              'orientation': 'v',
              'showlegend': False,
              'textposition': 'auto',
              'type': 'bar',
              'x': array(['What It Takes', 'What You Are', 'Do what cha wanna',
                          'What is and Should Never Be', 'So What', 'What A Day', 'What If I Do?',
                          'What Now My Love', 'Whatsername', "Whatever It Is, I Just Can't Stop",
                          "Look What You've Done", 'Get What You Need',
                          'What Is And What Should Never Be',
                          "You're What's Happening (In The World Today)", 'So What',
                          "I Don't Know What To Do With Myself", 'What Kate Did',
                          'Whatever the Case May Be',
                          "I Still Haven't Found What I'm Looking for",
                          "I Still Haven't Found What I'm Looking For",
                          'Whatever Gets You Thru the Night', 'What Is It About Men'],
                         dtype=object),
              'xaxis': 'x',
              'y': array([ 310622,  249391,  274155,  260675,  564009,  158275,  302994,  149995,
                           252316,  247222,  230974,  247719,  287973,  142027,  189152,  221387,
                          2610250, 2616410,  353567,  280764,  215084,  209573]),
              'yaxis': 'y'}],
    'layout': {'barmode': 'relative',
               'legend': {'tracegroupgap': 0},
               'margin': {'t': 60},
               'template': '...',
               'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Name'}},
               'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'Milliseconds'}}}
}))
```

In [24]:
```
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 6, updating n_results = 6
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 fo
rmat instructions. \n===Tables \nCREATE INDEX IFK_InvoiceCustomerId ON Invoice (CustomerId)\n\nCREATE TABLE
Invoice\n(\n    InvoiceId INTEGER  NOT NULL,\n    CustomerId INTEGER  NOT NULL,\n    InvoiceDate DATETIME
NOT NULL,\n    BillingAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(4
0),\n    BillingCountry NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2)  NOT NU
LL,\n    CONSTRAINT PK_Invoice PRIMARY KEY  (InvoiceId),\n    FOREIGN KEY (CustomerId) REFERENCES Customer
(CustomerId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON I
nvoiceLine (InvoiceId)\n\nCREATE TABLE InvoiceLine\n(\n    InvoiceLineId INTEGER  NOT NULL,\n    InvoiceId
INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    Quantity IN
TEGER  NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (InvoiceLineId),\n    FOREIGN KEY (InvoiceId)
REFERENCES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) RE
FERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_InvoiceLineTr
ackId ON InvoiceLine (TrackId)\n\nCREATE TABLE Customer\n(\n    CustomerId INTEGER  NOT NULL,\n    FirstNam
e NVARCHAR(40)  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    Company NVARCHAR(80),\n    Address NVA
RCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVAR
CHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60)  NOT NULL,\n    SupportRe
pId INTEGER,\n    CONSTRAINT PK_Customer PRIMARY KEY  (CustomerId),\n    FOREIGN KEY (SupportRepId) REFEREN
CES Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_CustomerSupp
ortRepId ON Customer (SupportRepId)\n\nCREATE INDEX IFK_EmployeeReportsTo ON Employee (ReportsTo)\n\nCREATE
TABLE Employee\n(\n    EmployeeId INTEGER  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    FirstName N
VARCHAR(20)  NOT NULL,\n    Title NVARCHAR(30),\n    ReportsTo INTEGER,\n    BirthDate DATETIME,\n    HireD
ate DATETIME,\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVAR
CHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR
(60),\n    CONSTRAINT PK_Employee PRIMARY KEY  (EmployeeId),\n    FOREIGN KEY (ReportsTo) REFERENCES Employ
ee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Track\n(\n    TrackId INTE
GER  NOT NULL,\n    Name NVARCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER  NOT NUL
L,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Bytes INTEG
ER,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FOREIGN K
EY (AlbumId) REFERENCES Album (AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (Ge
nreId) REFERENCES Genre (GenreId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTyp
eId) REFERENCES MediaType (MediaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n\n===Additional
Context \n\nIn the chinook database invoice means order\n\n===Response Guidelines \n1. If the provided cont
ext 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, p
lease generate an intermediate SQL query to find the distinct strings in that column. Prepend the query wit
h a comment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it ca
n't be generated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answe
red before, please repeat the answer exactly as it was given before. \n"}, {'role': 'user', 'content': 'How
many records are in table called customer'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM Custome
r;'}, {'role': 'user', 'content': 'How many customers are there'}, {'role': 'assistant', 'content': 'SELECT
COUNT(*) FROM Customer;'}, {'role': 'user', 'content': 'what are the top 5 countries that customers come fr
om?'}, {'role': 'assistant', 'content': 'SELECT Country, COUNT(*) as CustomerCount\nFROM Customer\nGROUP BY
Country\nORDER BY CustomerCount DESC\nLIMIT 5;'}, {'role': 'user', 'content': '  \n    List all albums and
their corresponding artist names  \n'}, {'role': 'assistant', 'content': 'SELECT Album.Title, Artist.Name\n
FROM Album\nJOIN Artist ON Album.ArtistId = Artist.ArtistId;'}, {'role': 'user', 'content': '  \n    Find a
ll tracks with a name containing "What" (case-insensitive)\n'}, {'role': 'assistant', 'content': "SELECT *
FROM Track\nWHERE Name LIKE '%What%' COLLATE NOCASE;"}, {'role': 'user', 'content': 'Show me a list of tabl
es in the SQLite database'}, {'role': 'assistant', 'content': "SELECT name FROM sqlite_master WHERE type='t
able';"}, {'role': 'user', 'content': '  \n    Get the total number of invoices for each customer\n'}]
Using model gpt-4 for 1176.75 tokens (approx)
SELECT CustomerId, COUNT(InvoiceId) as TotalInvoices
FROM Invoice
GROUP BY CustomerId;
SELECT CustomerId, COUNT(InvoiceId) as TotalInvoices
FROM Invoice
GROUP BY CustomerId;
SELECT CustomerId, COUNT(InvoiceId) as TotalInvoices
FROM Invoice
GROUP BY 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
Using model gpt-4 for 187.5 tokens (approx)
```


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

```
Out[24]: ('SELECT CustomerId, COUNT(InvoiceId) as TotalInvoices\nFROM Invoice\nGROUP BY CustomerId;',
              CustomerId  TotalInvoices
          0            1              7
          1            2              7
          2            3              7
          3            4              7
          4            5              7
          5            6              7
          6            7              7
          7            8              7
          8            9              7
          9           10              7
          10          11              7
          11          12              7
          12          13              7
          13          14              7
          14          15              7
          15          16              7
          16          17              7
          17          18              7
          18          19              7
          19          20              7
          20          21              7
          21          22              7
          22          23              7
          23          24              7
          24          25              7
          25          26              7
          26          27              7
          27          28              7
          28          29              7
          29          30              7
          30          31              7
          31          32              7
          32          33              7
          33          34              7
          34          35              7
          35          36              7
          36          37              7
          37          38              7
          38          39              7
          39          40              7
```

```
40          41              7
41          42              7
42          43              7
43          44              7
44          45              7
45          46              7
46          47              7
47          48              7
48          49              7
49          50              7
50          51              7
51          52              7
52          53              7
53          54              7
54          55              7
55          56              7
56          57              7
57          58              7
58          59              6,
Figure({
    'data': [{'alignmentgroup': 'True',
              'hovertemplate': 'CustomerId=%{x}<br>TotalInvoices=%{y}<extra></extra>',
              'legendgroup': '',
              'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
              'name': '',
              'offsetgroup': '',
              'orientation': 'v',
              'showlegend': False,
              'textposition': 'auto',
              'type': 'bar',
              'x': array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
                          19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36,
                          37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54,
                          55, 56, 57, 58, 59]),
              'xaxis': 'x',
              'y': array([7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
                          7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
                          7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 6]),
              'yaxis': 'y'}],
    'layout': {'barmode': 'relative',
               'legend': {'tracegroupgap': 0},
               'template': '...',
               'title': {'text': 'Total Invoices Per Customer'},
               'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'CustomerId'}},
               'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'TotalInvoices'}}}
}))
```

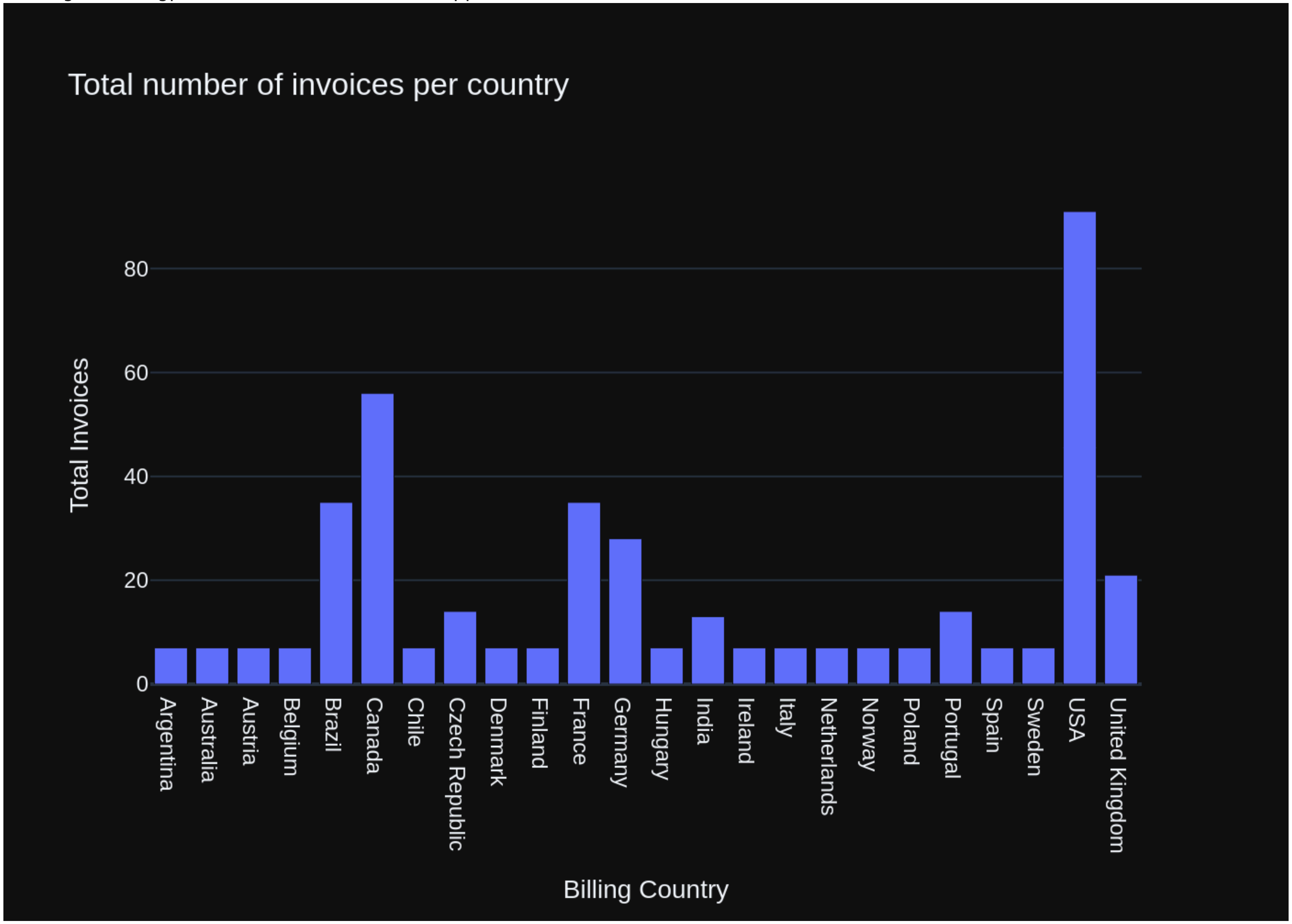
In [25]:

```
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 7, updating n_results = 7
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 fo
rmat instructions. \n===Tables \nCREATE TABLE Invoice\n(\n    InvoiceId INTEGER  NOT NULL,\n    CustomerId
INTEGER  NOT NULL,\n    InvoiceDate DATETIME  NOT NULL,\n    BillingAddress NVARCHAR(70),\n    BillingCity
NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n    BillingPostalCode NVAR
CHAR(10),\n    Total NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Invoice PRIMARY KEY  (InvoiceId),\n    FOR
EIGN KEY (CustomerId) REFERENCES Customer (CustomerId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n
CREATE INDEX IFK_InvoiceLineInvoiceId ON InvoiceLine (InvoiceId)\n\nCREATE INDEX IFK_InvoiceCustomerId ON I
nvoice (CustomerId)\n\nCREATE TABLE InvoiceLine\n(\n    InvoiceLineId INTEGER  NOT NULL,\n    InvoiceId INT
EGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    Quantity INTEG
ER  NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (InvoiceLineId),\n    FOREIGN KEY (InvoiceId) REF
ERENCES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFER
ENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_InvoiceLineTrack
Id ON InvoiceLine (TrackId)\n\nCREATE TABLE Customer\n(\n    CustomerId INTEGER  NOT NULL,\n    FirstName N
VARCHAR(40)  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    Company NVARCHAR(80),\n    Address NVARCH
AR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCHA
R(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60)  NOT NULL,\n    SupportRepId
INTEGER,\n    CONSTRAINT PK_Customer PRIMARY KEY  (CustomerId),\n    FOREIGN KEY (SupportRepId) REFERENCES
Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Employee\n(\n    Emp
loyeeId INTEGER  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    FirstName NVARCHAR(20)  NOT NULL,\n
Title NVARCHAR(30),\n    ReportsTo INTEGER,\n    BirthDate DATETIME,\n    HireDate DATETIME,\n    Address N
VARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NV
ARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60),\n    CONSTRAINT PK_Emp
loyee PRIMARY KEY  (EmployeeId),\n    FOREIGN KEY (ReportsTo) REFERENCES Employee (EmployeeId) \n\t\tON DEL
ETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Track\n(\n    TrackId INTEGER  NOT NULL,\n    Name NVA
RCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER  NOT NULL,\n    GenreId INTEGER,\n
Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,
2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FOREIGN KEY (AlbumId) REFERENCES Album
(AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES Genre (Genre
Id) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES MediaType (Med
iaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_EmployeeReportsTo ON Employe
e (ReportsTo)\n\nCREATE TABLE Album\n(\n    AlbumId INTEGER  NOT NULL,\n    Title NVARCHAR(160)  NOT NUL
L,\n    ArtistId INTEGER  NOT NULL,\n    CONSTRAINT PK_Album PRIMARY KEY  (AlbumId),\n    FOREIGN KEY (Arti
stId) REFERENCES Artist (ArtistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n\n===Additional Cont
ext \n\nIn the chinook 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
Get the total number of invoices for each customer\n'}, {'role': 'assistant', 'content': 'SELECT CustomerI
d, COUNT(InvoiceId) as TotalInvoices\nFROM Invoice\nGROUP BY CustomerId;'}, {'role': 'user', 'content': 'wh
at are the top 5 countries that customers come from?'}, {'role': 'assistant', 'content': 'SELECT Country, C
OUNT(*) as CustomerCount\nFROM Customer\nGROUP BY Country\nORDER BY CustomerCount DESC\nLIMIT 5;'}, {'rol
e': 'user', 'content': 'How many records are in table called customer'}, {'role': 'assistant', 'content':
'SELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'content': 'How many customers are there'}, {'role': 'a
ssistant', 'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'content': '  \n    List all albu
ms and their corresponding artist names  \n'}, {'role': 'assistant', 'content': 'SELECT Album.Title, Artis
t.Name\nFROM Album\nJOIN Artist ON Album.ArtistId = Artist.ArtistId;'}, {'role': 'user', 'content': '  \n
Find all tracks with a name containing "What" (case-insensitive)\n'}, {'role': 'assistant', 'content': "SEL
ECT * FROM Track\nWHERE Name LIKE '%What%' COLLATE NOCASE;"}, {'role': 'user', 'content': 'Show me a list o
f tables in the SQLite database'}, {'role': 'assistant', 'content': "SELECT name FROM sqlite_master WHERE t
ype='table';"}, {'role': 'user', 'content': '  \n    Find the total number of invoices per country:\n'}]
Using model gpt-4 for 1262.5 tokens (approx)
SELECT BillingCountry, COUNT(*) as TotalInvoices
FROM Invoice
GROUP BY BillingCountry;
SELECT BillingCountry, COUNT(*) as TotalInvoices
FROM Invoice
GROUP BY BillingCountry;
SELECT BillingCountry, COUNT(*) as TotalInvoices
FROM Invoice
GROUP BY 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
Using model gpt-4 for 187.5 tokens (approx)
```



```
Out[25]: ('SELECT BillingCountry, COUNT(*) as TotalInvoices\nFROM Invoice\nGROUP BY 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': 'Billing Country=%{x}<br>Total Invoices=%{y}<extra></extra>',
                        'legendgroup': '',
                        'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
                        'name': '',
                        'offsetgroup': '',
                        'orientation': 'v',
                        'showlegend': False,
                        'textposition': 'auto',
                        'type': 'bar',
                        'x': array(['Argentina', 'Australia', 'Austria', 'Belgium', 'Brazil', 'Canada',
                                    'Chile', 'Czech Republic', 'Denmark', 'Finland', 'France', 'Germany',
                                    'Hungary', 'India', 'Ireland', 'Italy', 'Netherlands', 'Norway',
                                    'Poland', 'Portugal', 'Spain', 'Sweden', 'USA', 'United Kingdom'],
                                   dtype=object),
                        'xaxis': 'x',
                        'y': array([ 7,  7,  7,  7, 35, 56,  7, 14,  7,  7, 35, 28,  7, 13,  7,  7,  7,  7,
                                     7, 14,  7,  7, 91, 21]),
                        'yaxis': 'y'}],
              'layout': {'barmode': 'relative',
                         'legend': {'tracegroupgap': 0},
                         'template': '...',
                         'title': {'text': 'Total number of invoices per country'},
                         'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Billing Country'}},
                         'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'Total Invoices'}}}
          }))
```

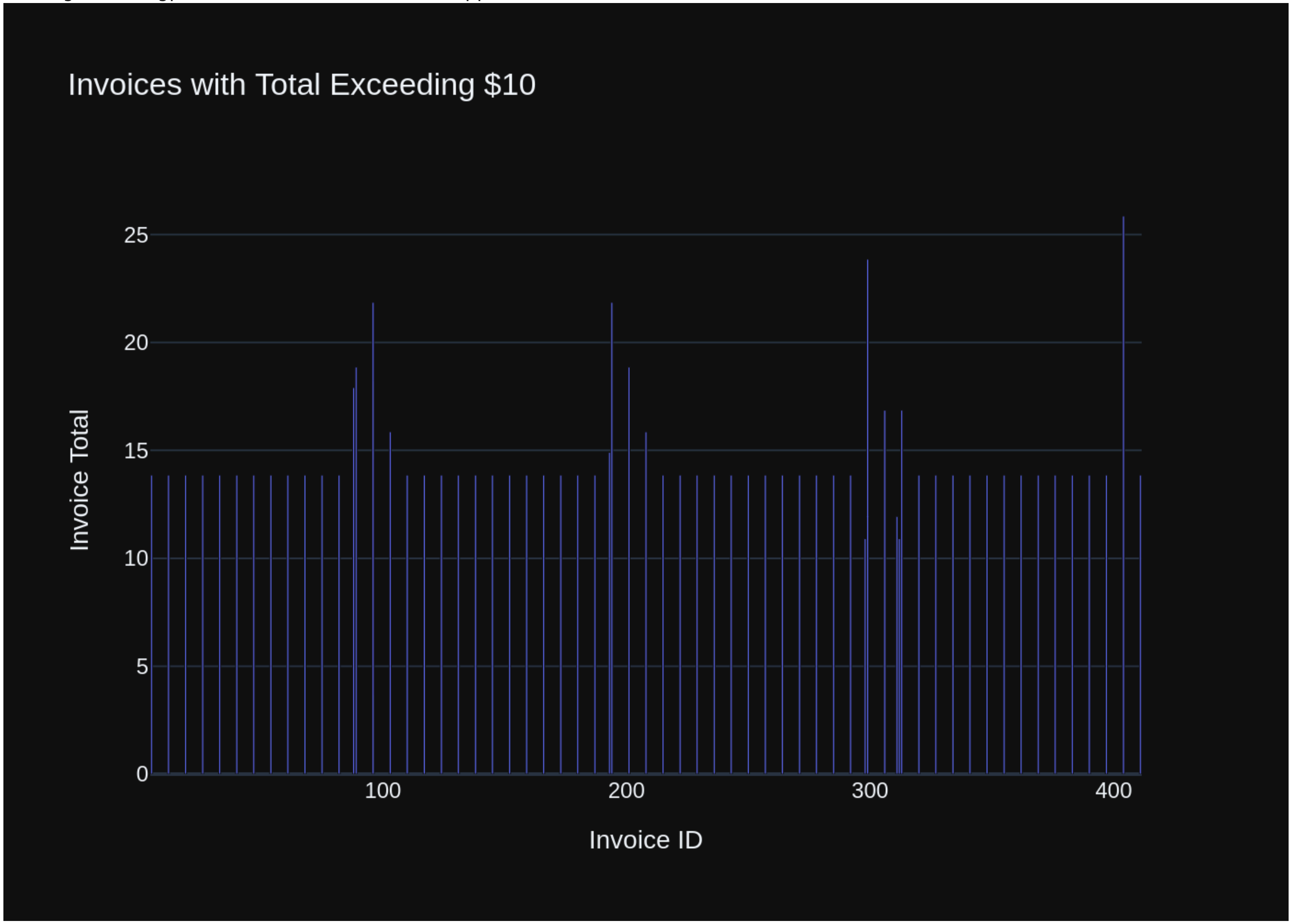
In [26]:

```
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 8, updating n_results = 8
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 fo
rmat instructions. \n===Tables \nCREATE TABLE InvoiceLine\n(\n    InvoiceLineId INTEGER  NOT NULL,\n    Inv
oiceId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    Quan
tity INTEGER  NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (InvoiceLineId),\n    FOREIGN KEY (Invo
iceId) REFERENCES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (Trac
kId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_Invoic
eLineInvoiceId ON InvoiceLine (InvoiceId)\n\nCREATE TABLE Invoice\n(\n    InvoiceId INTEGER  NOT NULL,\n
CustomerId INTEGER  NOT NULL,\n    InvoiceDate DATETIME  NOT NULL,\n    BillingAddress NVARCHAR(70),\n    B
illingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n    BillingPost
alCode NVARCHAR(10),\n    Total NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Invoice PRIMARY KEY  (InvoiceI
d),\n    FOREIGN KEY (CustomerId) REFERENCES Customer (CustomerId) \n\t\tON DELETE NO ACTION ON UPDATE NO A
CTION\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON Invoice (CustomerId)\n\nCREATE INDEX IFK_InvoiceLineTrack
Id ON InvoiceLine (TrackId)\n\nCREATE TABLE Track\n(\n    TrackId INTEGER  NOT NULL,\n    Name NVARCHAR(20
0)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER  NOT NULL,\n    GenreId INTEGER,\n    Composer
NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT N
ULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumId)
\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES Genre (GenreId) \n\t\t
ON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES MediaType (MediaTypeId)
\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_EmployeeReportsTo ON Employee (Reports
To)\n\nCREATE TABLE Customer\n(\n    CustomerId INTEGER  NOT NULL,\n    FirstName NVARCHAR(40)  NOT NULL,\n
LastName NVARCHAR(20)  NOT NULL,\n    Company NVARCHAR(80),\n    Address NVARCHAR(70),\n    City NVARCHAR(4
0),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(2
4),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60)  NOT NULL,\n    SupportRepId INTEGER,\n    CONSTRAINT PK
_Customer PRIMARY KEY  (CustomerId),\n    FOREIGN KEY (SupportRepId) REFERENCES Employee (EmployeeId) \n\t
\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Employee\n(\n    EmployeeId INTEGER  NOT NUL
L,\n    LastName NVARCHAR(20)  NOT NULL,\n    FirstName NVARCHAR(20)  NOT NULL,\n    Title NVARCHAR(30),\n
ReportsTo INTEGER,\n    BirthDate DATETIME,\n    HireDate DATETIME,\n    Address NVARCHAR(70),\n    City NV
ARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NV
ARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60),\n    CONSTRAINT PK_Employee PRIMARY KEY  (Emplo
yeeId),\n    FOREIGN KEY (ReportsTo) REFERENCES Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE N
O ACTION\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON Customer (SupportRepId)\n\n\n===Additional Context
\n\nIn the chinook database invoice means order\n\n===Response Guidelines \n1. If the provided context is s
ufficient, please generate a valid SQL query without any explanations for the question. \n2. If the provide
d context is almost sufficient but requires knowledge of a specific string in a particular column, please g
enerate an intermediate SQL query to find the distinct strings in that column. Prepend the query with a com
ment saying intermediate_sql \n3. If the provided context is insufficient, please explain why it can't be g
enerated. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered befo
re, 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 CustomerId, CO
UNT(InvoiceId) as TotalInvoices\nFROM Invoice\nGROUP BY CustomerId;'}, {'role': 'user', 'content': '  \n
Find the total number of invoices per country:\n'}, {'role': 'assistant', 'content': 'SELECT BillingCountr
y, COUNT(*) as TotalInvoices\nFROM Invoice\nGROUP BY BillingCountry;'}, {'role': 'user', 'content': 'How ma
ny records are in table called customer'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM Custome
r;'}, {'role': 'user', 'content': 'How many customers are there'}, {'role': 'assistant', 'content': 'SELECT
COUNT(*) FROM Customer;'}, {'role': 'user', 'content': 'what are the top 5 countries that customers come fr
om?'}, {'role': 'assistant', 'content': 'SELECT Country, COUNT(*) as CustomerCount\nFROM Customer\nGROUP BY
Country\nORDER BY CustomerCount DESC\nLIMIT 5;'}, {'role': 'user', 'content': '  \n    List all albums and
their corresponding artist names  \n'}, {'role': 'assistant', 'content': 'SELECT Album.Title, Artist.Name\n
FROM Album\nJOIN Artist ON Album.ArtistId = Artist.ArtistId;'}, {'role': 'user', 'content': '  \n    Find a
ll tracks with a name containing "What" (case-insensitive)\n'}, {'role': 'assistant', 'content': "SELECT *
FROM Track\nWHERE Name LIKE '%What%' COLLATE NOCASE;"}, {'role': 'user', 'content': 'Show me a list of tabl
es in the SQLite database'}, {'role': 'assistant', 'content': "SELECT name FROM sqlite_master WHERE type='t
able';"}, {'role': 'user', 'content': '  \n    List all invoices with a total exceeding $10:\n'}]
Using model gpt-4 for 1246.5 tokens (approx)
SELECT * FROM Invoice
WHERE Total > 10;
SELECT * FROM Invoice
WHERE Total > 10;
SELECT * FROM Invoice
WHERE Total > 10;
    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]
Using model gpt-4 for 228.25 tokens (approx)
```



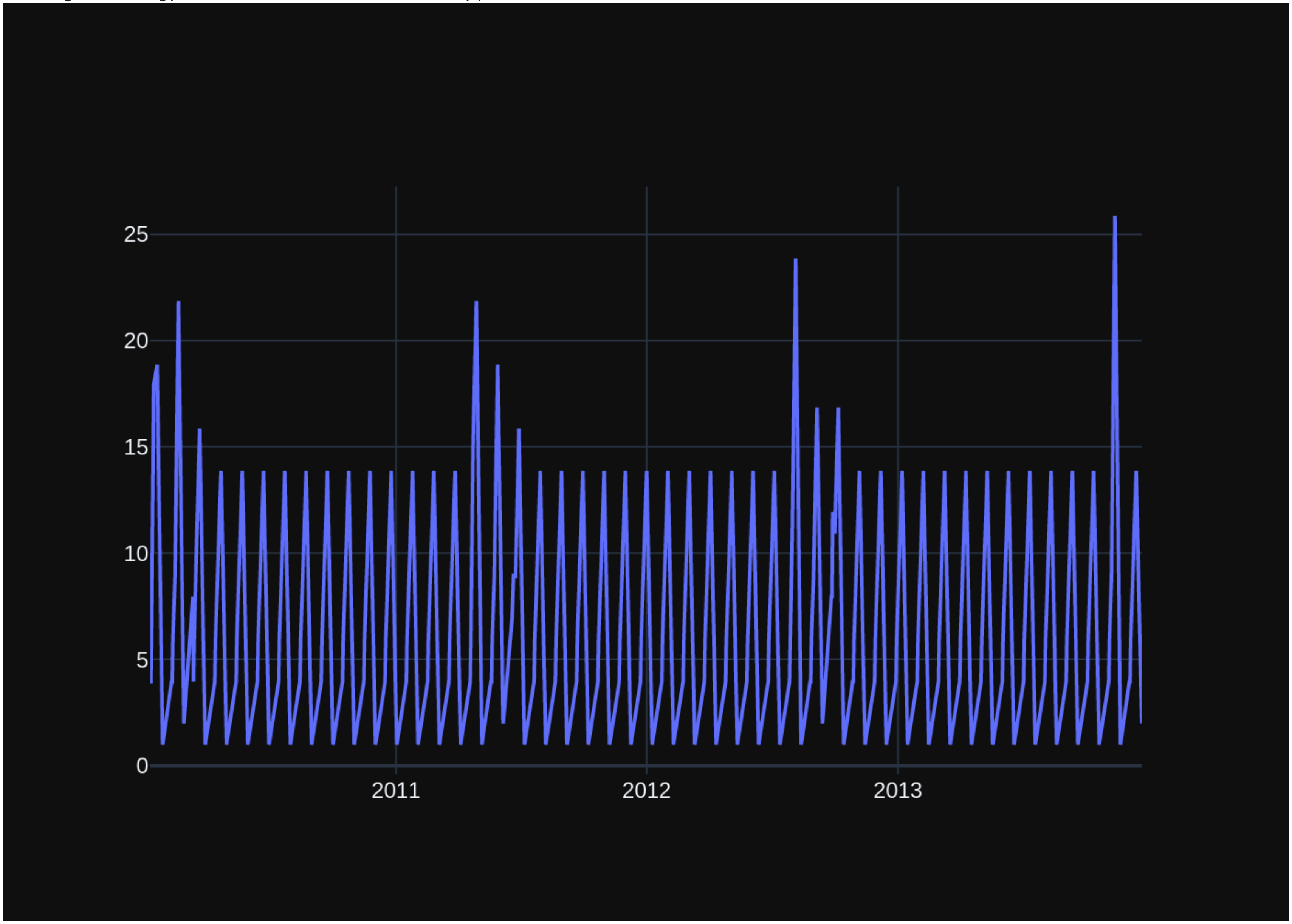
```
Out[26]: ('SELECT * FROM Invoice\nWHERE Total > 10;',
              InvoiceId  CustomerId          InvoiceDate             BillingAddress  \
          0           5          23  2009-01-11 00:00:00            69 Salem Street   
          1          12           2  2009-02-11 00:00:00    Theodor-Heuss-Straße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

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

          [64 rows x 9 columns],
          Figure({
              'data': [{'alignmentgroup': 'True',
                        'hovertemplate': 'Invoice ID=%{x}<br>Invoice Total=%{y}<extra></extra>',
                        'legendgroup': '',
                        'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
                        'name': '',
                        'offsetgroup': '',
                        'orientation': 'v',
                        'showlegend': False,
                        'textposition': 'auto',
                        'type': 'bar',
                        'x': array([  5,  12,  19,  26,  33,  40,  47,  54,  61,  68,  75,  82,  88,  89,
                                     96, 103, 110, 117, 124, 131, 138, 145, 152, 159, 166, 173, 180, 187,
                                    193, 194, 201, 208, 215, 222, 229, 236, 243, 250, 257, 264, 271, 278,
                        285, 292, 298, 299, 306, 311, 312, 313, 320, 327, 334, 341, 348, 355,
                        362, 369, 376, 383, 390, 397, 404, 411]),
                'xaxis': 'x',
                'y': array([13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86,
                            13.86, 13.86, 17.91, 18.86, 21.86, 15.86, 13.86, 13.86, 13.86, 13.86,
                            13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 14.91, 21.86,
                            18.86, 15.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86,
                            13.86, 13.86, 13.86, 13.86, 10.91, 23.86, 16.86, 11.94, 10.91, 16.86,
                            13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86, 13.86,
                            13.86, 13.86, 25.86, 13.86]),
                'yaxis': 'y'}],
     'layout': {'barmode': 'relative',
                'legend': {'tracegroupgap': 0},
                'template': '...',
                'title': {'text': 'Invoices with Total Exceeding $10'},
                'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Invoice ID'}},
                'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'Invoice Total'}}}
 }))
```

In [27]:
```python
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 9, updating n_results = 9
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 fo
rmat instructions. \n===Tables \nCREATE TABLE Invoice\n(\n    InvoiceId INTEGER  NOT NULL,\n    CustomerId
INTEGER  NOT NULL,\n    InvoiceDate DATETIME  NOT NULL,\n    BillingAddress NVARCHAR(70),\n    BillingCity
NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n    BillingPostalCode NVAR
CHAR(10),\n    Total NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Invoice PRIMARY KEY  (InvoiceId),\n    FOR
EIGN KEY (CustomerId) REFERENCES Customer (CustomerId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n
CREATE TABLE InvoiceLine\n(\n    InvoiceLineId INTEGER  NOT NULL,\n    InvoiceId INTEGER  NOT NULL,\n    Tr
ackId INTEGER  NOT NULL,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    Quantity INTEGER  NOT NULL,\n    CONS
TRAINT PK_InvoiceLine PRIMARY KEY  (InvoiceLineId),\n    FOREIGN KEY (InvoiceId) REFERENCES Invoice (Invoic
eId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackId)
\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON InvoiceLine (I
nvoiceId)\n\nCREATE INDEX IFK_InvoiceCustomerId ON Invoice (CustomerId)\n\nCREATE INDEX IFK_InvoiceLineTrac
kId ON InvoiceLine (TrackId)\n\nCREATE TABLE Customer\n(\n    CustomerId INTEGER  NOT NULL,\n    FirstName
NVARCHAR(40)  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    Company NVARCHAR(80),\n    Address NVARC
HAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCH
AR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60)  NOT NULL,\n    SupportRepI
d INTEGER,\n    CONSTRAINT PK_Customer PRIMARY KEY  (CustomerId),\n    FOREIGN KEY (SupportRepId) REFERENCE
S Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Employee\n(\n    E
mployeeId INTEGER  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    FirstName NVARCHAR(20)  NOT NULL,\n
Title NVARCHAR(30),\n    ReportsTo INTEGER,\n    BirthDate DATETIME,\n    HireDate DATETIME,\n    Address N
VARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NV
ARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60),\n    CONSTRAINT PK_Emp
loyee PRIMARY KEY  (EmployeeId),\n    FOREIGN KEY (ReportsTo) REFERENCES Employee (EmployeeId) \n\t\tON DEL
ETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Track\n(\n    TrackId INTEGER  NOT NULL,\n    Name NVA
RCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER  NOT NULL,\n    GenreId INTEGER,\n
Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,
2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FOREIGN KEY (AlbumId) REFERENCES Album
(AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES Genre (Genre
Id) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES MediaType (Med
iaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE PlaylistTrack\n(\n    PlaylistId
INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY  (PlaylistI
d, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES Playlist (PlaylistId) \n\t\tON DELETE NO ACTION ON UP
DATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE N
O ACTION\n)\n\nCREATE TABLE Album\n(\n    AlbumId INTEGER  NOT NULL,\n    Title NVARCHAR(160)  NOT NULL,\n
ArtistId INTEGER  NOT NULL,\n    CONSTRAINT PK_Album PRIMARY KEY  (AlbumId),\n    FOREIGN KEY (ArtistId) RE
FERENCES Artist (ArtistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n\n===Additional Context \n\n
In the chinook database invoice means order\n\n===Response Guidelines \n1. If the provided context is suffi
cient, please generate a valid SQL query without any explanations for the question. \n2. If the provided co
ntext is almost sufficient but requires knowledge of a specific string in a particular column, please gener
ate 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 genera
ted. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, p
lease 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 * FROM Invoice\nWHERE To
tal > 10;'}, {'role': 'user', 'content': '  \n    Find the total number of invoices per country:\n'}, {'rol
e': 'assistant', 'content': 'SELECT BillingCountry, COUNT(*) as TotalInvoices\nFROM Invoice\nGROUP BY Billi
ngCountry;'}, {'role': 'user', 'content': '  \n    Get the total number of invoices for each customer\n'},
{'role': 'assistant', 'content': 'SELECT CustomerId, COUNT(InvoiceId) as TotalInvoices\nFROM Invoice\nGROUP
BY CustomerId;'}, {'role': 'user', 'content': 'How many records are in table called customer'}, {'role': 'a
ssistant', 'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'content': 'How many customers ar
e there'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'content':
'what are the top 5 countries that customers come from?'}, {'role': 'assistant', 'content': 'SELECT Countr
y, COUNT(*) as CustomerCount\nFROM Customer\nGROUP BY Country\nORDER BY CustomerCount DESC\nLIMIT 5;'}, {'r
ole': 'user', 'content': '  \n    Find all tracks with a name containing "What" (case-insensitive)\n'}, {'r
ole': 'assistant', 'content': "SELECT * FROM Track\nWHERE Name LIKE '%What%' COLLATE NOCASE;"}, {'role': 'u
ser', 'content': '  \n    List all albums and their corresponding artist names  \n'}, {'role': 'assistant',
'content': 'SELECT Album.Title, Artist.Name\nFROM Album\nJOIN Artist ON Album.ArtistId = Artist.ArtistI
d;'}, {'role': 'user', 'content': 'Show me a list of tables in the SQLite database'}, {'role': 'assistant',
'content': "SELECT name FROM sqlite_master WHERE type='table';"}, {'role': 'user', 'content': '  \n    Find
all invoices since 2010 and the total amount invoiced:\n'}]
Using model gpt-4 for 1400.5 tokens (approx)
SELECT InvoiceDate, SUM(Total) as TotalAmount
FROM Invoice
WHERE InvoiceDate >= '2010-01-01'
GROUP BY InvoiceDate;
SELECT InvoiceDate, SUM(Total) as TotalAmount
FROM Invoice
WHERE InvoiceDate >= '2010-01-01'
GROUP BY InvoiceDate;
SELECT InvoiceDate, SUM(Total) as TotalAmount
FROM Invoice
WHERE InvoiceDate >= '2010-01-01'
GROUP BY InvoiceDate;
             InvoiceDate  TotalAmount
0    2010-01-08 00:00:00         3.96
1    2010-01-09 00:00:00         3.96
2    2010-01-10 00:00:00         6.94
3    2010-01-13 00:00:00        17.91
4    2010-01-18 00:00:00        18.86
..                   ...          ...
277  2013-12-05 00:00:00         3.96
278  2013-12-06 00:00:00         5.94
279  2013-12-09 00:00:00         8.91
```

```
280  2013-12-14 00:00:00        13.86
281  2013-12-22 00:00:00         1.99

[282 rows x 2 columns]
Using model gpt-4 for 196.75 tokens (approx)
```



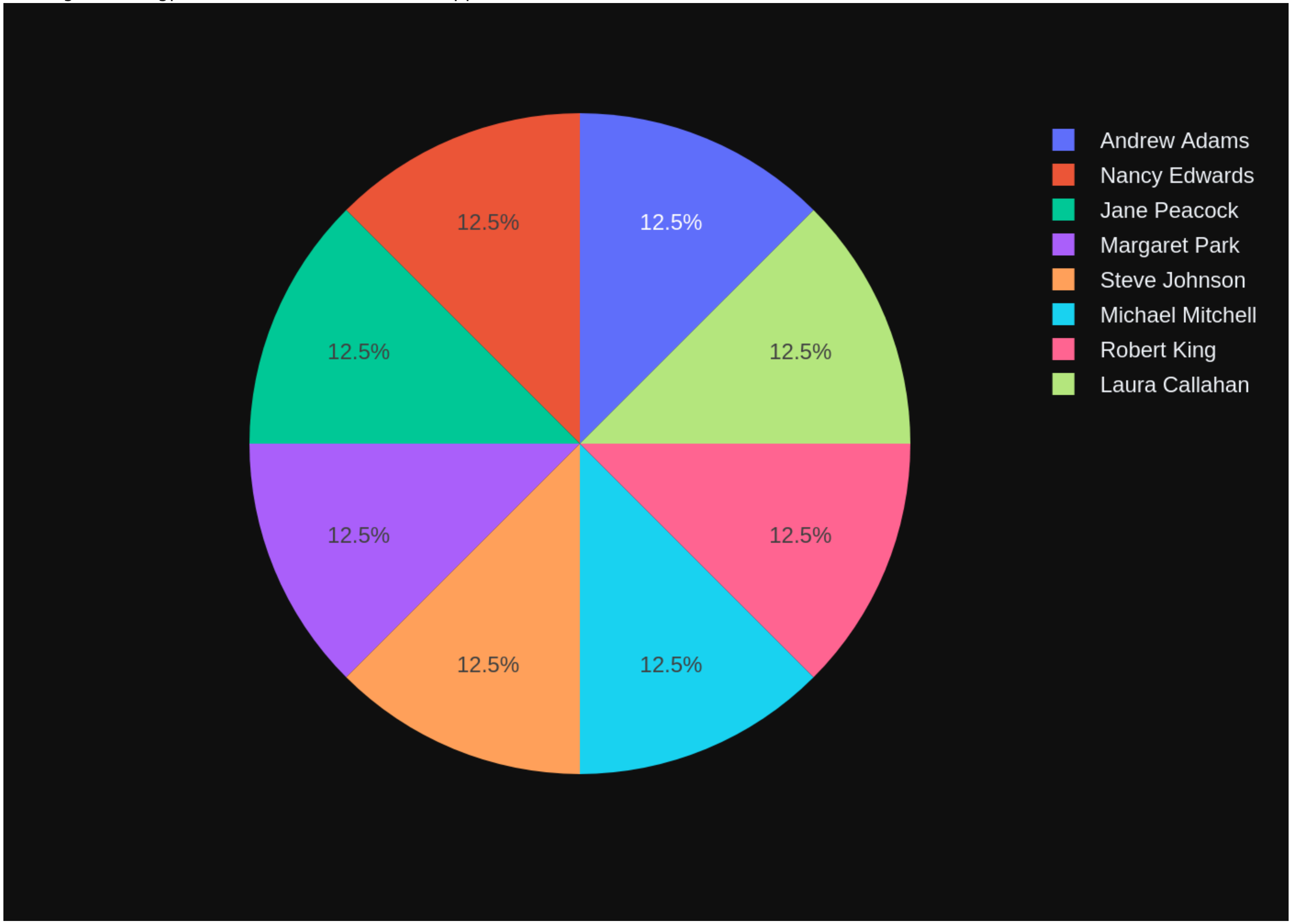
```
Out[27]: ("SELECT InvoiceDate, SUM(Total) as TotalAmount\nFROM Invoice\nWHERE InvoiceDate >= '2010-01-01'\nGROUP BY
         InvoiceDate;",
                       InvoiceDate  TotalAmount
          0    2010-01-08 00:00:00         3.96
          1    2010-01-09 00:00:00         3.96
          2    2010-01-10 00:00:00         6.94
          3    2010-01-13 00:00:00        17.91
          4    2010-01-18 00:00:00        18.86
          ..                   ...          ...
          277  2013-12-05 00:00:00         3.96
          278  2013-12-06 00:00:00         5.94
          279  2013-12-09 00:00:00         8.91
          280  2013-12-14 00:00:00        13.86
          281  2013-12-22 00:00:00         1.99

          [282 rows x 2 columns],
          Figure({
              'data': [{'type': 'scatter',
                        'x': array(['2010-01-08 00:00:00', '2010-01-09 00:00:00', '2010-01-10 00:00:00',
                                    ..., '2013-12-09 00:00:00', '2013-12-14 00:00:00',
                                    '2013-12-22 00:00:00'], dtype=object),
                        'y': array([ 3.96,  3.96,  6.94, ...,  8.91, 13.86,  1.99])}],
              'layout': {'template': '...'}
          }))
```

```
In [28]: 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 fo
rmat instructions. \n===Tables \nCREATE INDEX IFK_EmployeeReportsTo ON Employee (ReportsTo)\n\nCREATE TABLE
Employee\n(\n    EmployeeId INTEGER  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    FirstName NVARCHA
R(20)  NOT NULL,\n    Title NVARCHAR(30),\n    ReportsTo INTEGER,\n    BirthDate DATETIME,\n    HireDate DA
TETIME,\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(4
0),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(6
0),\n    CONSTRAINT PK_Employee PRIMARY KEY  (EmployeeId),\n    FOREIGN KEY (ReportsTo) REFERENCES Employee
(EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Customer\n(\n    CustomerId I
NTEGER  NOT NULL,\n    FirstName NVARCHAR(40)  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    Company
NVARCHAR(80),\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVAR
CHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR
(60)  NOT NULL,\n    SupportRepId INTEGER,\n    CONSTRAINT PK_Customer PRIMARY KEY  (CustomerId),\n    FORE
IGN KEY (SupportRepId) REFERENCES Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n
\nCREATE INDEX IFK_CustomerSupportRepId ON Customer (SupportRepId)\n\nCREATE TABLE Invoice\n(\n    InvoiceI
d INTEGER  NOT NULL,\n    CustomerId INTEGER  NOT NULL,\n    InvoiceDate DATETIME  NOT NULL,\n    BillingAd
dress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVAR
CHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Invoi
ce PRIMARY KEY  (InvoiceId),\n    FOREIGN KEY (CustomerId) REFERENCES Customer (CustomerId) \n\t\tON DELETE
NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE InvoiceLine\n(\n    InvoiceLineId INTEGER  NOT NULL,\n
InvoiceId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    Q
uantity INTEGER  NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (InvoiceLineId),\n    FOREIGN KEY (I
nvoiceId) REFERENCES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (T
rackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Track\n
(\n    TrackId INTEGER  NOT NULL,\n    Name NVARCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId
INTEGER  NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT NUL
L,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (Track
Id),\n    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n
FOREIGN KEY (GenreId) REFERENCES Genre (GenreId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIG
N KEY (MediaTypeId) REFERENCES MediaType (MediaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n
CREATE INDEX IFK_InvoiceCustomerId ON Invoice (CustomerId)\n\nCREATE TABLE Artist\n(\n    ArtistId INTEGER
NOT NULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK_Artist PRIMARY KEY  (ArtistId)\n)\n\nCREATE TABLE Pla
ylistTrack\n(\n    PlaylistId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    CONSTRAINT PK_Playlis
tTrack PRIMARY KEY  (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES Playlist (PlaylistId)
\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackId) \n\t\t
ON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n\n===Additional Context \n\nIn the chinook 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': 'what are the top 5 countries that customers come
from?'}, {'role': 'assistant', 'content': 'SELECT Country, COUNT(*) as CustomerCount\nFROM Customer\nGROUP
BY Country\nORDER BY CustomerCount DESC\nLIMIT 5;'}, {'role': 'user', 'content': '  \n    Find all invoices
since 2010 and the total amount invoiced:\n'}, {'role': 'assistant', 'content': "SELECT InvoiceDate, SUM(To
tal) as TotalAmount\nFROM Invoice\nWHERE InvoiceDate >= '2010-01-01'\nGROUP BY InvoiceDate;"}, {'role': 'us
er', 'content': '  \n    Get the total number of invoices for each customer\n'}, {'role': 'assistant', 'con
tent': 'SELECT CustomerId, COUNT(InvoiceId) as TotalInvoices\nFROM Invoice\nGROUP BY CustomerId;'}, {'rol
e': 'user', 'content': '  \n    List all albums and their corresponding artist names  \n'}, {'role': 'assis
tant', 'content': 'SELECT Album.Title, Artist.Name\nFROM Album\nJOIN Artist ON Album.ArtistId = Artist.Arti
stId;'}, {'role': 'user', 'content': '  \n    Find the total number of invoices per country:\n'}, {'role':
'assistant', 'content': 'SELECT BillingCountry, COUNT(*) as TotalInvoices\nFROM Invoice\nGROUP BY BillingCo
untry;'}, {'role': 'user', 'content': '  \n    List all invoices with a total exceeding $10:\n'}, {'role':
'assistant', 'content': 'SELECT * FROM Invoice\nWHERE Total > 10;'}, {'role': 'user', 'content': 'How many
records are in table called customer'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM Customer;'},
{'role': 'user', 'content': 'How many customers are there'}, {'role': 'assistant', 'content': 'SELECT COUNT
(*) FROM Customer;'}, {'role': 'user', 'content': '  \n    Find all tracks with a name containing "What" (c
ase-insensitive)\n'}, {'role': 'assistant', 'content': "SELECT * FROM Track\nWHERE Name LIKE '%What%' COLLA
TE NOCASE;"}, {'role': 'user', 'content': 'Show me a list of tables in the SQLite database'}, {'role': 'ass
istant', 'content': "SELECT name FROM sqlite_master WHERE type='table';"}, {'role': 'user', 'content': "
\n    List all employees and their reporting manager's name (if any):\n"}]
Using model gpt-4 for 1411.5 tokens (approx)
SELECT E1.FirstName || ' ' || E1.LastName AS EmployeeName,
       E2.FirstName || ' ' || E2.LastName AS ManagerName
FROM Employee E1
LEFT JOIN Employee E2 ON E1.ReportsTo = E2.EmployeeId;
SELECT E1.FirstName || ' ' || E1.LastName AS EmployeeName,
       E2.FirstName || ' ' || E2.LastName AS ManagerName
FROM Employee E1
LEFT JOIN Employee E2 ON E1.ReportsTo = E2.EmployeeId;
SELECT E1.FirstName || ' ' || E1.LastName AS EmployeeName,
       E2.FirstName || ' ' || E2.LastName AS ManagerName
FROM Employee E1
LEFT JOIN Employee E2 ON E1.ReportsTo = E2.EmployeeId;
       EmployeeName       ManagerName
0      Andrew Adams              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
Using model gpt-4 for 216.5 tokens (approx)
```



```
Out[28]: ("SELECT E1.FirstName || ' ' || E1.LastName AS EmployeeName, \n       E2.FirstName || ' ' || E2.LastName A
         S ManagerName\nFROM Employee E1 \nLEFT JOIN Employee E2 ON E1.ReportsTo = E2.EmployeeId;",
                EmployeeName       ManagerName
          0      Andrew Adams              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': [{'domain': {'x': [0.0, 1.0], 'y': [0.0, 1.0]},
                        'hovertemplate': 'EmployeeName=%{label}<extra></extra>',
                        'labels': array(['Andrew Adams', 'Nancy Edwards', 'Jane Peacock', 'Margaret Park',
                                         'Steve Johnson', 'Michael Mitchell', 'Robert King', 'Laura Callahan'],
                                        dtype=object),
                        'legendgroup': '',
                        'name': '',
                        'showlegend': True,
                        'type': 'pie'}],
              'layout': {'legend': {'tracegroupgap': 0}, 'margin': {'t': 60}, 'template': '...'}
          }))
```

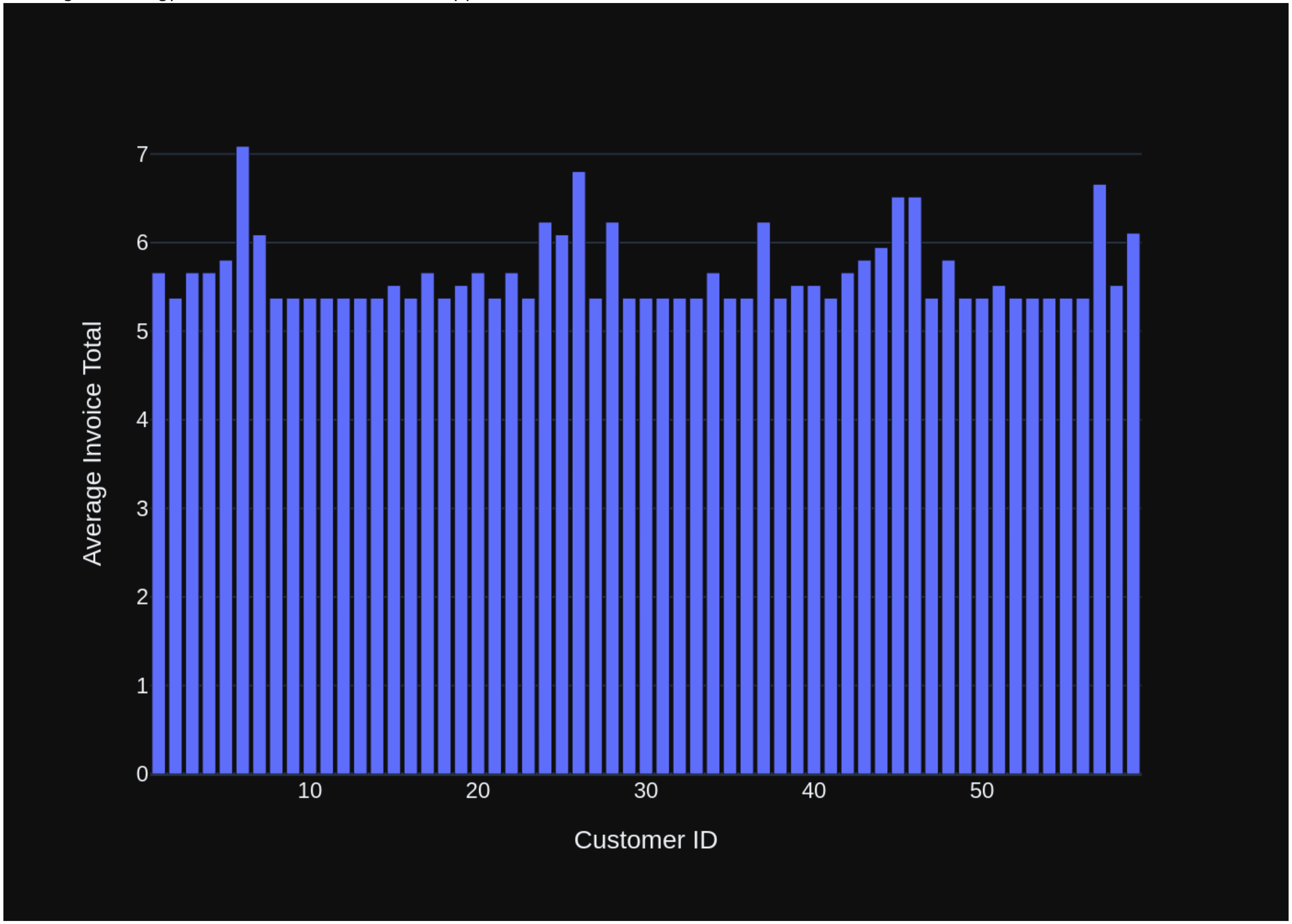
```
In [29]: question = """
             Get the average invoice total for each customer:
         """

         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 fo
rmat instructions. \n===Tables \nCREATE INDEX IFK_InvoiceCustomerId ON Invoice (CustomerId)\n\nCREATE INDEX
IFK_InvoiceLineInvoiceId ON InvoiceLine (InvoiceId)\n\nCREATE TABLE Invoice\n(\n    InvoiceId INTEGER  NOT
NULL,\n    CustomerId INTEGER  NOT NULL,\n    InvoiceDate DATETIME  NOT NULL,\n    BillingAddress NVARCHAR
(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n
BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Invoice PRIMARY KEY
(InvoiceId),\n    FOREIGN KEY (CustomerId) REFERENCES Customer (CustomerId) \n\t\tON DELETE NO ACTION ON UP
DATE NO ACTION\n)\n\nCREATE TABLE InvoiceLine\n(\n    InvoiceLineId INTEGER  NOT NULL,\n    InvoiceId INTEG
ER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    Quantity INTEGER
NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (InvoiceLineId),\n    FOREIGN KEY (InvoiceId) REFEREN
CES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCE
S Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_InvoiceLineTrackId O
N InvoiceLine (TrackId)\n\nCREATE INDEX IFK_CustomerSupportRepId ON Customer (SupportRepId)\n\nCREATE TABLE
Customer\n(\n    CustomerId INTEGER  NOT NULL,\n    FirstName NVARCHAR(40)  NOT NULL,\n    LastName NVARCHA
R(20)  NOT NULL,\n    Company NVARCHAR(80),\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State N
VARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVA
RCHAR(24),\n    Email NVARCHAR(60)  NOT NULL,\n    SupportRepId INTEGER,\n    CONSTRAINT PK_Customer PRIMAR
Y KEY  (CustomerId),\n    FOREIGN KEY (SupportRepId) REFERENCES Employee (EmployeeId) \n\t\tON DELETE NO AC
TION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_EmployeeReportsTo ON Employee (ReportsTo)\n\nCREATE TABLE T
rack\n(\n    TrackId INTEGER  NOT NULL,\n    Name NVARCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    Media
TypeId INTEGER  NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT
NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (Tr
ackId),\n    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTIO
N,\n    FOREIGN KEY (GenreId) REFERENCES Genre (GenreId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n
FOREIGN KEY (MediaTypeId) REFERENCES MediaType (MediaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION
\n)\n\nCREATE TABLE Employee\n(\n    EmployeeId INTEGER  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n
FirstName NVARCHAR(20)  NOT NULL,\n    Title NVARCHAR(30),\n    ReportsTo INTEGER,\n    BirthDate DATETIM
E,\n    HireDate DATETIME,\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n
Country NVARCHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Em
ail NVARCHAR(60),\n    CONSTRAINT PK_Employee PRIMARY KEY  (EmployeeId),\n    FOREIGN KEY (ReportsTo) REFER
ENCES Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n\n===Additional Context \n
\nIn the chinook database invoice means order\n\n===Response Guidelines \n1. If the provided context is suf
ficient, 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 CustomerId, CO
UNT(InvoiceId) as TotalInvoices\nFROM Invoice\nGROUP BY CustomerId;'}, {'role': 'user', 'content': '  \n
Find all invoices since 2010 and the total amount invoiced:\n'}, {'role': 'assistant', 'content': "SELECT I
nvoiceDate, SUM(Total) as TotalAmount\nFROM Invoice\nWHERE InvoiceDate >= '2010-01-01'\nGROUP BY InvoiceDat
e;"}, {'role': 'user', 'content': '  \n    Find the total number of invoices per country:\n'}, {'role': 'as
sistant', 'content': 'SELECT BillingCountry, COUNT(*) as TotalInvoices\nFROM Invoice\nGROUP BY BillingCount
ry;'}, {'role': 'user', 'content': '  \n    List all invoices with a total exceeding $10:\n'}, {'role': 'as
sistant', 'content': 'SELECT * FROM Invoice\nWHERE Total > 10;'}, {'role': 'user', 'content': 'How many cus
tomers are there'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'c
ontent': 'How many records are in table called customer'}, {'role': 'assistant', 'content': 'SELECT COUNT
(*) FROM Customer;'}, {'role': 'user', 'content': 'what are the top 5 countries that customers come fro
m?'}, {'role': 'assistant', 'content': 'SELECT Country, COUNT(*) as CustomerCount\nFROM Customer\nGROUP BY
Country\nORDER BY CustomerCount DESC\nLIMIT 5;'}, {'role': 'user', 'content': "  \n    List all employees a
nd their reporting manager's name (if any):\n"}, {'role': 'assistant', 'content': "SELECT E1.FirstName || '
' || E1.LastName AS EmployeeName, \n       E2.FirstName || ' ' || E2.LastName AS ManagerName\nFROM Employee
E1 \nLEFT JOIN Employee E2 ON E1.ReportsTo = E2.EmployeeId;"}, {'role': 'user', 'content': '  \n    Find al
l tracks with a name containing "What" (case-insensitive)\n'}, {'role': 'assistant', 'content': "SELECT * F
ROM Track\nWHERE Name LIKE '%What%' COLLATE NOCASE;"}, {'role': 'user', 'content': '  \n    List all albums
and their corresponding artist names  \n'}, {'role': 'assistant', 'content': 'SELECT Album.Title, Artist.Na
me\nFROM Album\nJOIN Artist ON Album.ArtistId = Artist.ArtistId;'}, {'role': 'user', 'content': '  \n    Ge
t the average invoice total for each customer:\n'}]
Using model gpt-4 for 1356.25 tokens (approx)
SELECT CustomerId, AVG(Total) as AverageInvoiceTotal
FROM Invoice
GROUP BY CustomerId;
SELECT CustomerId, AVG(Total) as AverageInvoiceTotal
FROM Invoice
GROUP BY CustomerId;
SELECT CustomerId, AVG(Total) as AverageInvoiceTotal
FROM Invoice
GROUP BY 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
Using model gpt-4 for 191.0 tokens (approx)
```



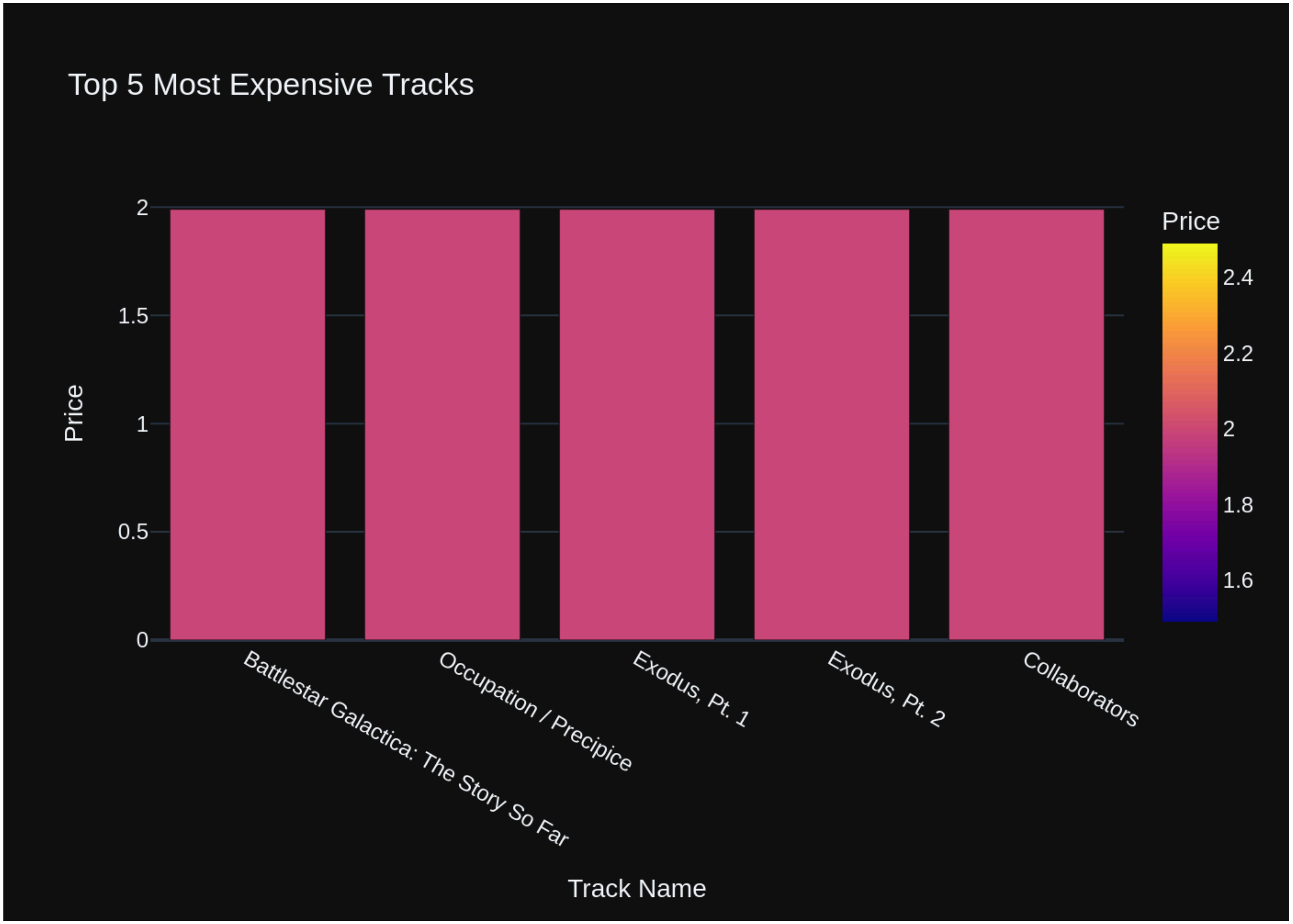
```
Out[29]: ('SELECT CustomerId, AVG(Total) as AverageInvoiceTotal\nFROM Invoice\nGROUP BY CustomerId;',
              CustomerId  AverageInvoiceTotal
          0            1             5.660000
          1            2             5.374286
          2            3             5.660000
          3            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': 'Customer ID=%{x}<br>Average Invoice Total=%{y}<extra></extra>',
              'legendgroup': '',
              'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
              'name': '',
              'offsetgroup': '',
              'orientation': 'v',
              'showlegend': False,
              'textposition': 'auto',
              'type': 'bar',
              'x': array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
                          19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36,
                          37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54,
                          55, 56, 57, 58, 59]),
              'xaxis': 'x',
              'y': array([5.66      , 5.37428571, 5.66      , 5.66      , 5.80285714, 7.08857143,
                          6.08857143, 5.37428571, 5.37428571, 5.37428571, 5.37428571, 5.37428571,
                          5.37428571, 5.37428571, 5.51714286, 5.37428571, 5.66      , 5.37428571,
                          5.51714286, 5.66      , 5.37428571, 5.66      , 5.37428571, 6.23142857,
                          6.08857143, 6.80285714, 5.37428571, 6.23142857, 5.37428571, 5.37428571,
                          5.37428571, 5.37428571, 5.37428571, 5.66      , 5.37428571, 5.37428571,
                          6.23142857, 5.37428571, 5.51714286, 5.51714286, 5.37428571, 5.66      ,
                          5.80285714, 5.94571429, 6.51714286, 6.51714286, 5.37428571, 5.80285714,
                          5.37428571, 5.37428571, 5.51714286, 5.37428571, 5.37428571, 5.37428571,
                          5.37428571, 5.37428571, 6.66      , 5.51714286, 6.10666667]),
              'yaxis': 'y'}],
    'layout': {'barmode': 'relative',
               'legend': {'tracegroupgap': 0},
               'margin': {'t': 60},
               'template': '...',
               'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Customer ID'}},
               'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'Average Invoice Tota
l'}}}
 }))
```

In [30]:
```python
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 fo
rmat instructions. \n===Tables \nCREATE TABLE Track\n(\n    TrackId INTEGER  NOT NULL,\n    Name NVARCHAR(2
00)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER  NOT NULL,\n    GenreId INTEGER,\n    Compose
r NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT
NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumI
d) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES Genre (GenreId) \n
\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES MediaType (MediaType
Id) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_TrackAlbumId ON Track (AlbumId)\n
\nCREATE INDEX IFK_TrackGenreId ON Track (GenreId)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON PlaylistTrac
k (TrackId)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON InvoiceLine (TrackId)\n\nCREATE INDEX IFK_TrackMediaT
ypeId ON Track (MediaTypeId)\n\nCREATE TABLE InvoiceLine\n(\n    InvoiceLineId INTEGER  NOT NULL,\n    Invo
iceId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    Quant
ity INTEGER  NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (InvoiceLineId),\n    FOREIGN KEY (Invoi
ceId) REFERENCES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (Track
Id) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE PlaylistTra
ck\n(\n    PlaylistId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    CONSTRAINT PK_PlaylistTrack P
RIMARY KEY  (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES Playlist (PlaylistId) \n\t\tON
DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE
NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_AlbumArtistId ON Album (ArtistId)\n\nCREATE TABLE Albu
m\n(\n    AlbumId INTEGER  NOT NULL,\n    Title NVARCHAR(160)  NOT NULL,\n    ArtistId INTEGER  NOT NULL,\n
CONSTRAINT PK_Album PRIMARY KEY  (AlbumId),\n    FOREIGN KEY (ArtistId) REFERENCES Artist (ArtistId) \n\t\t
ON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n\n\n===Additional Context \n\nIn the chinook 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 all tracks with a name containing
"What" (case-insensitive)\n'}, {'role': 'assistant', 'content': "SELECT * FROM Track\nWHERE Name LIKE '%Wha
t%' COLLATE NOCASE;"}, {'role': 'user', 'content': '  \n    List all invoices with a total exceeding $1
0:\n'}, {'role': 'assistant', 'content': 'SELECT * FROM Invoice\nWHERE Total > 10;'}, {'role': 'user', 'con
tent': '  \n    List all albums and their corresponding artist names  \n'}, {'role': 'assistant', 'conten
t': 'SELECT Album.Title, Artist.Name\nFROM Album\nJOIN Artist ON Album.ArtistId = Artist.ArtistId;'}, {'rol
e': 'user', 'content': '  \n    Find all invoices since 2010 and the total amount invoiced:\n'}, {'role':
'assistant', 'content': "SELECT InvoiceDate, SUM(Total) as TotalAmount\nFROM Invoice\nWHERE InvoiceDate >=
'2010-01-01'\nGROUP BY InvoiceDate;"}, {'role': 'user', 'content': '  \n    Get the average invoice total f
or each customer:\n'}, {'role': 'assistant', 'content': 'SELECT CustomerId, AVG(Total) as AverageInvoiceTot
al\nFROM Invoice\nGROUP BY CustomerId;'}, {'role': 'user', 'content': 'what are the top 5 countries that cu
stomers come from?'}, {'role': 'assistant', 'content': 'SELECT Country, COUNT(*) as CustomerCount\nFROM Cus
tomer\nGROUP BY Country\nORDER BY CustomerCount DESC\nLIMIT 5;'}, {'role': 'user', 'content': '  \n    Find
the total number of invoices per country:\n'}, {'role': 'assistant', 'content': 'SELECT BillingCountry, COU
NT(*) as TotalInvoices\nFROM Invoice\nGROUP BY BillingCountry;'}, {'role': 'user', 'content': '  \n    Get
the total number of invoices for each customer\n'}, {'role': 'assistant', 'content': 'SELECT CustomerId, CO
UNT(InvoiceId) as TotalInvoices\nFROM Invoice\nGROUP BY CustomerId;'}, {'role': 'user', 'content': 'Show me
a list of tables in the SQLite database'}, {'role': 'assistant', 'content': "SELECT name FROM sqlite_master
WHERE type='table';"}, {'role': 'user', 'content': 'How many customers are there'}, {'role': 'assistant',
'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'content': '  \n    Find the top 5 most expe
nsive tracks (based on unit price):\n'}]
Using model gpt-4 for 1093.25 tokens (approx)
SELECT * FROM Track
ORDER BY UnitPrice DESC
LIMIT 5;
SELECT * FROM Track
ORDER BY UnitPrice DESC
LIMIT 5;
SELECT * FROM Track
ORDER BY UnitPrice DESC
LIMIT 5;
   TrackId                                    Name  AlbumId  MediaTypeId  \
0     2819  Battlestar Galactica: The Story So Far      226            3
1     2820                    Occupation / Precipice      227            3
2     2821                           Exodus, Pt. 1      227            3
3     2822                           Exodus, Pt. 2      227            3
4     2823                           Collaborators      227            3

   GenreId Composer  Milliseconds       Bytes  UnitPrice
0       18     None       2622250   490750393       1.99
1       19     None       5286953  1054423946       1.99
2       19     None       2621708   475079441       1.99
3       19     None       2618000   466820021       1.99
4       19     None       2626626   483484911       1.99
Using model gpt-4 for 223.75 tokens (approx)
```


Top 5 Most Expensive Tracks
Price
2
1.5
1
0.5
0
Battlestar Galactica: The Story So Far
Occupation / Precipice
Exodus, Pt. 1
Exodus, Pt. 2
Collaborators
Track Name
Price
2.4
2.2
2
1.8
1.6

```
Out[30]:  ('SELECT * FROM Track\nORDER BY UnitPrice DESC\nLIMIT 5;',
              TrackId                                    Name  AlbumId  MediaTypeId  \
           0     2819  Battlestar Galactica: The Story So Far      226            3
           1     2820                   Occupation / Precipice      227            3
           2     2821                            Exodus, Pt. 1      227            3
           3     2822                            Exodus, Pt. 2      227            3
           4     2823                            Collaborators      227            3

              GenreId Composer  Milliseconds       Bytes  UnitPrice
           0       18     None       2622250   490750393       1.99
           1       19     None       5286953  1054423946       1.99
           2       19     None       2621708   475079441       1.99
           3       19     None       2618000   466820021       1.99
           4       19     None       2626626   483484911       1.99  ,
           Figure({
               'data': [{'alignmentgroup': 'True',
                         'hovertemplate': 'Track Name=%{x}<br>Price=%{marker.color}<extra></extra>',
                         'legendgroup': '',
                         'marker': {'color': array([1.99, 1.99, 1.99, 1.99, 1.99]),
                                    'coloraxis': 'coloraxis',
                                    'pattern': {'shape': ''}},
                         'name': '',
                         'offsetgroup': '',
                         'orientation': 'v',
                         'showlegend': False,
                         'textposition': 'auto',
                         'type': 'bar',
                         'x': array(['Battlestar Galactica: The Story So Far', 'Occupation / Precipice',
                                     'Exodus, Pt. 1', 'Exodus, Pt. 2', 'Collaborators'], dtype=object),
                         'xaxis': 'x',
                         'y': array([1.99, 1.99, 1.99, 1.99, 1.99]),
                         'yaxis': 'y'}],
               'layout': {'barmode': 'relative',
                          'coloraxis': {'colorbar': {'title': {'text': 'Price'}},
                                        '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},
                          'template': '...',
                          'title': {'text': 'Top 5 Most Expensive Tracks'},
                          'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Track Name'}},
                          'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'Price'}}}
           }))
```

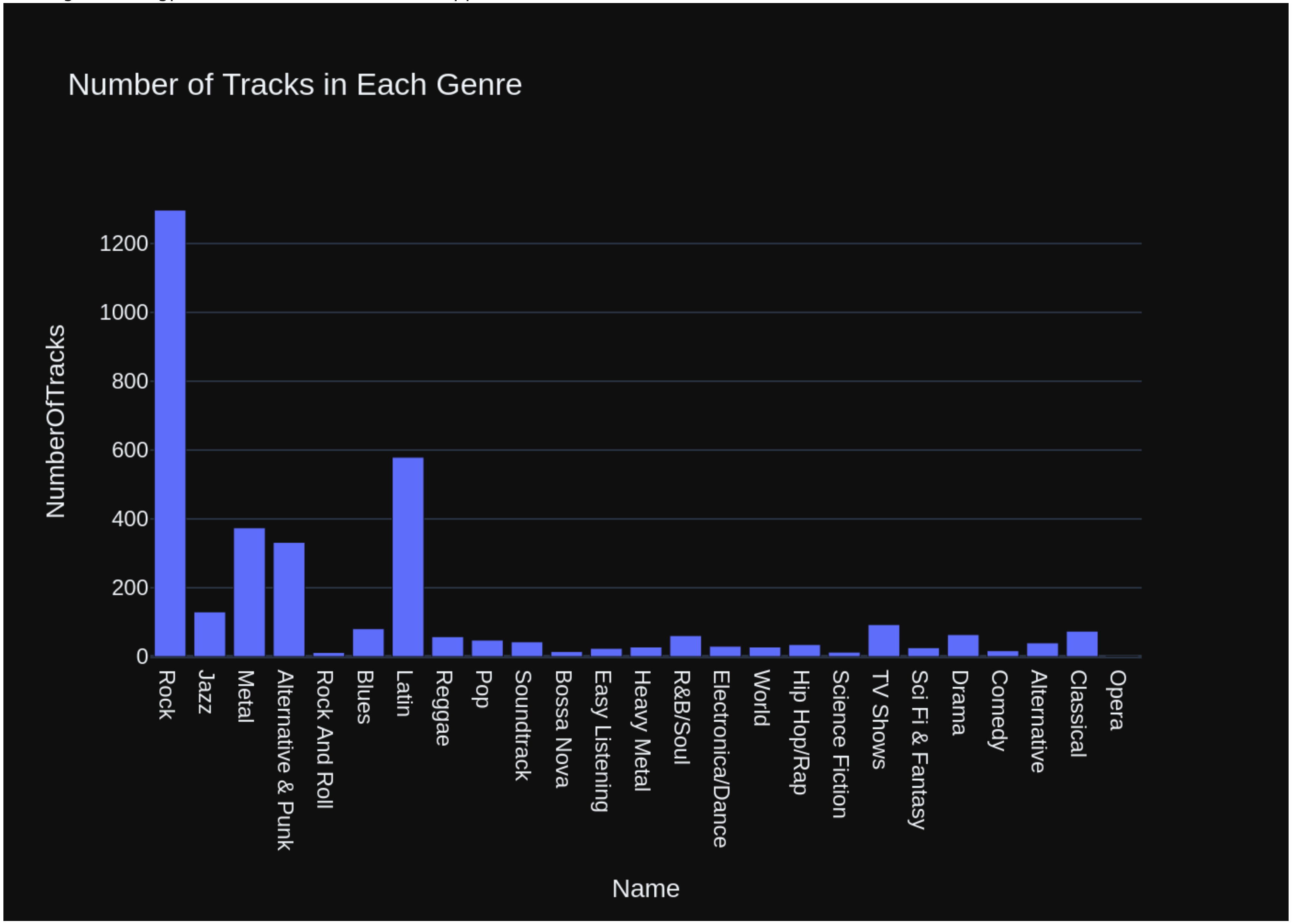
In [31]:

```
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 fo
rmat instructions. \n===Tables \nCREATE TABLE Track\n(\n    TrackId INTEGER  NOT NULL,\n    Name NVARCHAR(2
00)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER  NOT NULL,\n    GenreId INTEGER,\n    Compose
r NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT
NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumI
d) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES Genre (GenreId) \n
\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES MediaType (MediaType
Id) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_TrackGenreId ON Track (GenreId)\n
\nCREATE TABLE Genre\n(\n    GenreId INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK_Genre P
RIMARY KEY  (GenreId)\n)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON PlaylistTrack (TrackId)\n\nCREATE INDE
X IFK_TrackAlbumId ON Track (AlbumId)\n\nCREATE INDEX IFK_TrackMediaTypeId ON Track (MediaTypeId)\n\nCREATE
TABLE Album\n(\n    AlbumId INTEGER  NOT NULL,\n    Title NVARCHAR(160)  NOT NULL,\n    ArtistId INTEGER  N
OT NULL,\n    CONSTRAINT PK_Album PRIMARY KEY  (AlbumId),\n    FOREIGN KEY (ArtistId) REFERENCES Artist (Ar
tistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE PlaylistTrack\n(\n    PlaylistId I
NTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY  (PlaylistI
d, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES Playlist (PlaylistId) \n\t\tON DELETE NO ACTION ON UP
DATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE N
O ACTION\n)\n\nCREATE INDEX IFK_AlbumArtistId ON Album (ArtistId)\n\nCREATE TABLE Playlist\n(\n    Playlist
Id INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK_Playlist PRIMARY KEY  (PlaylistId)\n)\n\n
\n===Additional Context \n\nIn the chinook database invoice means order\n\n===Response Guidelines \n1. If t
he provided context is sufficient, please generate a valid SQL query without any explanations for the quest
ion. \n2. If the provided context is almost sufficient but requires knowledge of a specific string in a par
ticular column, please generate an intermediate SQL query to find the distinct strings in that column. Prep
end the query with a comment saying intermediate_sql \n3. If the provided context is insufficient, please e
xplain 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 * FROM Track\nORDER BY UnitPrice DESC\nLIMIT 5;'}, {'role': 'user', 'content': '  \n
List all albums and their corresponding artist names  \n'}, {'role': 'assistant', 'content': 'SELECT Album.
Title, Artist.Name\nFROM Album\nJOIN Artist ON Album.ArtistId = Artist.ArtistId;'}, {'role': 'user', 'conte
nt': '  \n    Find all tracks with a name containing "What" (case-insensitive)\n'}, {'role': 'assistant',
'content': "SELECT * FROM Track\nWHERE Name LIKE '%What%' COLLATE NOCASE;"}, {'role': 'user', 'content': 'w
hat are the top 5 countries that customers come from?'}, {'role': 'assistant', 'content': 'SELECT Country,
COUNT(*) as CustomerCount\nFROM Customer\nGROUP BY Country\nORDER BY CustomerCount DESC\nLIMIT 5;'}, {'rol
e': 'user', 'content': '  \n    Find the total number of invoices per country:\n'}, {'role': 'assistant',
'content': 'SELECT BillingCountry, COUNT(*) as TotalInvoices\nFROM Invoice\nGROUP BY BillingCountry;'}, {'r
ole': 'user', 'content': '  \n    List all invoices with a total exceeding $10:\n'}, {'role': 'assistant',
'content': 'SELECT * FROM Invoice\nWHERE Total > 10;'}, {'role': 'user', 'content': '  \n    Find all invoi
ces since 2010 and the total amount invoiced:\n'}, {'role': 'assistant', 'content': "SELECT InvoiceDate, SU
M(Total) as TotalAmount\nFROM Invoice\nWHERE InvoiceDate >= '2010-01-01'\nGROUP BY InvoiceDate;"}, {'role':
'user', 'content': 'Show me a list of tables in the SQLite database'}, {'role': 'assistant', 'content': "SE
LECT name FROM sqlite_master WHERE type='table';"}, {'role': 'user', 'content': '  \n    Get the total numb
er of invoices for each customer\n'}, {'role': 'assistant', 'content': 'SELECT CustomerId, COUNT(InvoiceId)
as TotalInvoices\nFROM Invoice\nGROUP BY CustomerId;'}, {'role': 'user', 'content': 'How many records are i
n table called customer'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'us
er', 'content': '  \n    List all genres and the number of tracks in each genre:\n'}]
Using model gpt-4 for 1026.0 tokens (approx)
SELECT Genre.Name, COUNT(Track.TrackId) as NumberOfTracks
FROM Genre
LEFT JOIN Track ON Genre.GenreId = Track.GenreId
GROUP BY Genre.GenreId;
SELECT Genre.Name, COUNT(Track.TrackId) as NumberOfTracks
FROM Genre
LEFT JOIN Track ON Genre.GenreId = Track.GenreId
GROUP BY Genre.GenreId;
SELECT Genre.Name, COUNT(Track.TrackId) as NumberOfTracks
FROM Genre
LEFT JOIN Track ON Genre.GenreId = Track.GenreId
GROUP BY Genre.GenreId;
                  Name  NumberOfTracks
0                 Rock            1297
1                 Jazz             130
2                Metal             374
3   Alternative & Punk             332
4        Rock And Roll              12
5                Blues              81
6                Latin             579
7               Reggae              58
8                  Pop              48
9           Soundtrack              43
10          Bossa Nova              15
11      Easy Listening              24
12         Heavy Metal              28
13            R&B/Soul              61
14   Electronica/Dance              30
15               World              28
16         Hip Hop/Rap              35
17     Science Fiction              13
18            TV Shows              93
19    Sci Fi & Fantasy              26
20               Drama              64
21              Comedy              17
22         Alternative              40
```

```
23           Classical              74
24               Opera               1
Using model gpt-4 for 203.5 tokens (approx)
```



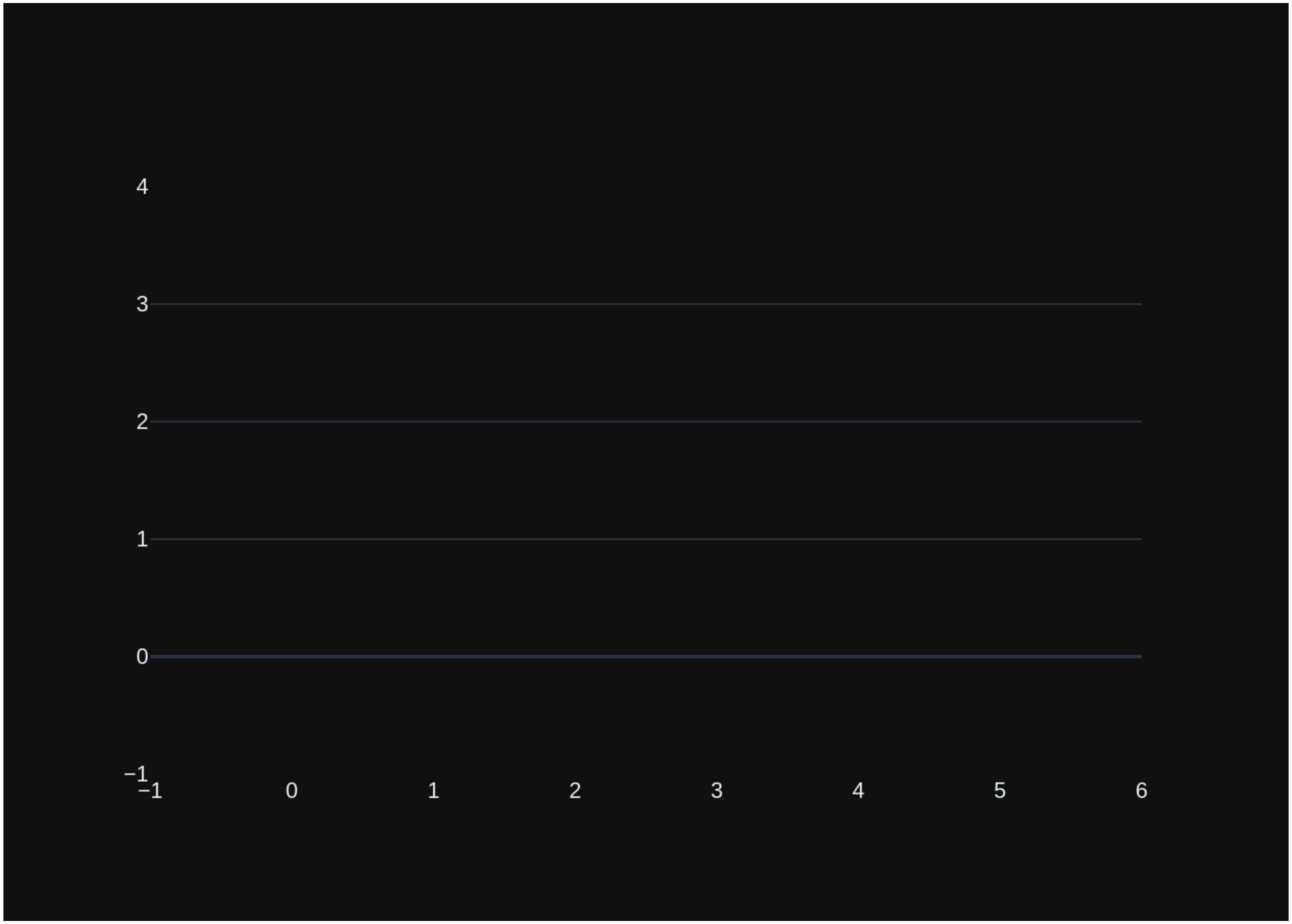
```
Out[31]: ('SELECT Genre.Name, COUNT(Track.TrackId) as NumberOfTracks\nFROM Genre\nLEFT JOIN Track ON Genre.GenreId
         = Track.GenreId\nGROUP BY Genre.GenreId;',
                            Name  NumberOfTracks
          0                 Rock            1297
          1                 Jazz             130
          2                Metal             374
          3   Alternative & Punk             332
          4        Rock And Roll              12
          5                Blues              81
          6                Latin             579
          7               Reggae              58
          8                  Pop              48
          9           Soundtrack              43
          10          Bossa Nova              15
          11      Easy Listening              24
          12         Heavy Metal              28
          13            R&B/Soul              61
          14   Electronica/Dance              30
          15               World              28
          16         Hip Hop/Rap              35
          17     Science Fiction              13
          18            TV Shows              93
          19    Sci Fi & Fantasy              26
          20               Drama              64
          21              Comedy              17
          22         Alternative              40
          23           Classical              74
          24               Opera               1,
          Figure({
              'data': [{'alignmentgroup': 'True',
                        'hovertemplate': 'Name=%{x}<br>NumberOfTracks=%{y}<extra></extra>',
                        'legendgroup': '',
                        'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
                        'name': '',
                        'offsetgroup': '',
                        'orientation': 'v',
                        'showlegend': False,
                        'textposition': 'auto',
                        'type': 'bar',
                        'x': array(['Rock', 'Jazz', 'Metal', 'Alternative & Punk', 'Rock And Roll', 'Blues',
                                    'Latin', 'Reggae', 'Pop', 'Soundtrack', 'Bossa Nova', 'Easy Listening',
                                    'Heavy Metal', 'R&B/Soul', 'Electronica/Dance', 'World', 'Hip Hop/Rap',
                                    'Science Fiction', 'TV Shows', 'Sci Fi & Fantasy', 'Drama', 'Comedy',
                                    'Alternative', 'Classical', 'Opera'], dtype=object),
                        'xaxis': 'x',
                        'y': array([1297,  130,  374,  332,   12,   81,  579,   58,   48,   43,   15,   24,
                                      28,   61,   30,   28,   35,   13,   93,   26,   64,   17,   40,   74,
                                       1]),
                        'yaxis': 'y'}],
              'layout': {'barmode': 'relative',
                         'legend': {'tracegroupgap': 0},
                         'template': '...',
                         'title': {'text': 'Number of Tracks in Each Genre'},
                         'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Name'}},
                         'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'NumberOfTracks'}}}
          }))
```

In [32]:

```python
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 fo
rmat instructions. \n===Tables \nCREATE INDEX IFK_TrackGenreId ON Track (GenreId)\n\nCREATE TABLE Track\n
(\n    TrackId INTEGER  NOT NULL,\n    Name NVARCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId
INTEGER  NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT NUL
L,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (Track
Id),\n    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n
FOREIGN KEY (GenreId) REFERENCES Genre (GenreId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIG
N KEY (MediaTypeId) REFERENCES MediaType (MediaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n
CREATE INDEX IFK_PlaylistTrackTrackId ON PlaylistTrack (TrackId)\n\nCREATE INDEX IFK_TrackAlbumId ON Track
(AlbumId)\n\nCREATE INDEX IFK_TrackMediaTypeId ON Track (MediaTypeId)\n\nCREATE TABLE Genre\n(\n    GenreId
INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK_Genre PRIMARY KEY  (GenreId)\n)\n\nCREATE TA
BLE Album\n(\n    AlbumId INTEGER  NOT NULL,\n    Title NVARCHAR(160)  NOT NULL,\n    ArtistId INTEGER  NOT
NULL,\n    CONSTRAINT PK_Album PRIMARY KEY  (AlbumId),\n    FOREIGN KEY (ArtistId) REFERENCES Artist (Artis
tId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_AlbumArtistId ON Album (ArtistId)
\n\nCREATE TABLE PlaylistTrack\n(\n    PlaylistId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    C
ONSTRAINT PK_PlaylistTrack PRIMARY KEY  (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES Pla
ylist (PlaylistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Tra
ck (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Artist\n(\n    ArtistId INTEG
ER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK_Artist PRIMARY KEY  (ArtistId)\n)\n\n\n===Additio
nal Context \n\nIn the chinook 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    List all genres and the number of tracks in each genre:\n'}, {'role': 'assistant', 'content': 'SELECT
Genre.Name, COUNT(Track.TrackId) as NumberOfTracks\nFROM Genre\nLEFT JOIN Track ON Genre.GenreId = Track.Ge
nreId\nGROUP BY Genre.GenreId;'}, {'role': 'user', 'content': '  \n    List all albums and their correspond
ing artist names  \n'}, {'role': 'assistant', 'content': 'SELECT Album.Title, Artist.Name\nFROM Album\nJOIN
Artist ON Album.ArtistId = Artist.ArtistId;'}, {'role': 'user', 'content': '  \n    Find all tracks with a
name containing "What" (case-insensitive)\n'}, {'role': 'assistant', 'content': "SELECT * FROM Track\nWHERE
Name LIKE '%What%' COLLATE NOCASE;"}, {'role': 'user', 'content': '  \n    Find the top 5 most expensive tr
acks (based on unit price):\n'}, {'role': 'assistant', 'content': 'SELECT * FROM Track\nORDER BY UnitPrice
DESC\nLIMIT 5;'}, {'role': 'user', 'content': 'Show me a list of tables in the SQLite database'}, {'role':
'assistant', 'content': "SELECT name FROM sqlite_master WHERE type='table';"}, {'role': 'user', 'content':
'  \n    Find all invoices since 2010 and the total amount invoiced:\n'}, {'role': 'assistant', 'content':
"SELECT InvoiceDate, SUM(Total) as TotalAmount\nFROM Invoice\nWHERE InvoiceDate >= '2010-01-01'\nGROUP BY I
nvoiceDate;"}, {'role': 'user', 'content': '  \n    List all invoices with a total exceeding $10:\n'}, {'ro
le': 'assistant', 'content': 'SELECT * FROM Invoice\nWHERE Total > 10;'}, {'role': 'user', 'content': 'what
are the top 5 countries that customers come from?'}, {'role': 'assistant', 'content': 'SELECT Country, COUN
T(*) as CustomerCount\nFROM Customer\nGROUP BY Country\nORDER BY CustomerCount DESC\nLIMIT 5;'}, {'role':
'user', 'content': "  \n    List all employees and their reporting manager's name (if any):\n"}, {'role':
'assistant', 'content': "SELECT E1.FirstName || ' ' || E1.LastName AS EmployeeName, \n       E2.FirstName |
| ' ' || E2.LastName AS ManagerName\nFROM Employee E1 \nLEFT JOIN Employee E2 ON E1.ReportsTo = E2.Employee
Id;"}, {'role': 'user', 'content': 'How many customers are there'}, {'role': 'assistant', 'content': 'SELEC
T COUNT(*) FROM Customer;'}, {'role': 'user', 'content': '  \n    Get all genres that do not have any track
s associated with them:\n'}]
Using model gpt-4 for 1067.0 tokens (approx)
SELECT Genre.Name
FROM Genre
LEFT JOIN Track ON Genre.GenreId = Track.GenreId
GROUP BY Genre.GenreId
HAVING COUNT(Track.TrackId) = 0;
SELECT Genre.Name
FROM Genre
LEFT JOIN Track ON Genre.GenreId = Track.GenreId
GROUP BY Genre.GenreId
HAVING COUNT(Track.TrackId) = 0;
SELECT Genre.Name
FROM Genre
LEFT JOIN Track ON Genre.GenreId = Track.GenreId
GROUP BY Genre.GenreId
HAVING COUNT(Track.TrackId) = 0;
Empty DataFrame
Columns: [Name]
Index: []
Using model gpt-4 for 195.0 tokens (approx)
```

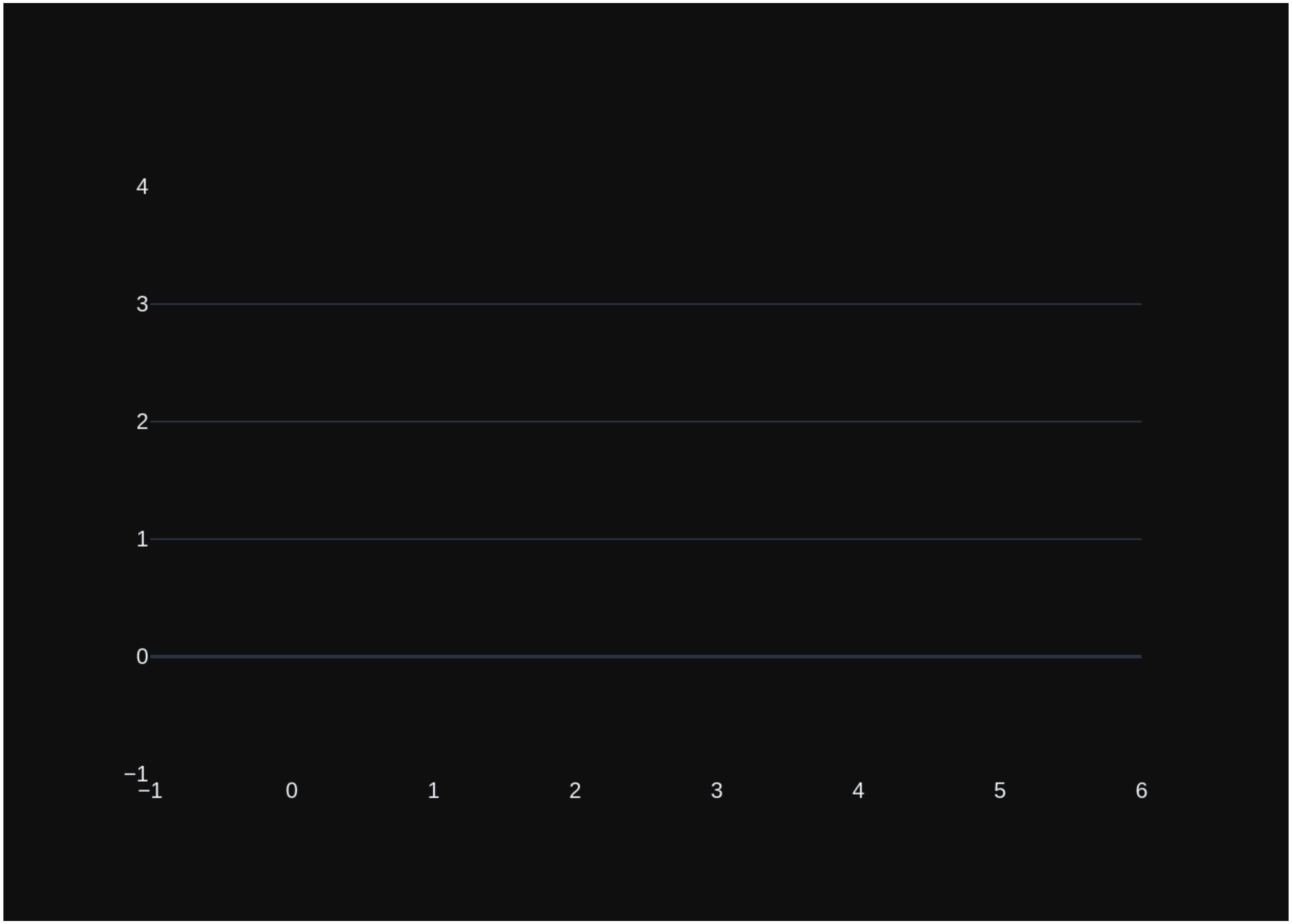

4
3
2
1
0
−1
−1
0
1
2
3
4
5
6

Out[32]:
```
('SELECT Genre.Name\nFROM Genre\nLEFT JOIN Track ON Genre.GenreId = Track.GenreId\nGROUP BY Genre.GenreId
\nHAVING COUNT(Track.TrackId) = 0;',
 Empty DataFrame
 Columns: [Name]
 Index: [],
 Figure({
     'data': [{'type': 'bar', 'x': array([], dtype=int64), 'y': array([], dtype=object)}], 'layout': {'tem
plate': '...'}
 }))
```

In [33]:
```
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 fo
rmat instructions. \n===Tables \nCREATE TABLE Customer\n(\n    CustomerId INTEGER  NOT NULL,\n    FirstName
NVARCHAR(40)  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    Company NVARCHAR(80),\n    Address NVARC
HAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCH
AR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60)  NOT NULL,\n    SupportRepI
d INTEGER,\n    CONSTRAINT PK_Customer PRIMARY KEY  (CustomerId),\n    FOREIGN KEY (SupportRepId) REFERENCE
S Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Invoice\n(\n    In
voiceId INTEGER  NOT NULL,\n    CustomerId INTEGER  NOT NULL,\n    InvoiceDate DATETIME  NOT NULL,\n    Bil
lingAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountr
y NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK
_Invoice PRIMARY KEY  (InvoiceId),\n    FOREIGN KEY (CustomerId) REFERENCES Customer (CustomerId) \n\t\tON
DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE InvoiceLine\n(\n    InvoiceLineId INTEGER  NOT NUL
L,\n    InvoiceId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPrice NUMERIC(10,2)  NOT NUL
L,\n    Quantity INTEGER  NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (InvoiceLineId),\n    FOREI
GN KEY (InvoiceId) REFERENCES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREI
GN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABL
E Employee\n(\n    EmployeeId INTEGER  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    FirstName NVARC
HAR(20)  NOT NULL,\n    Title NVARCHAR(30),\n    ReportsTo INTEGER,\n    BirthDate DATETIME,\n    HireDate
DATETIME,\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR
(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(6
0),\n    CONSTRAINT PK_Employee PRIMARY KEY  (EmployeeId),\n    FOREIGN KEY (ReportsTo) REFERENCES Employee
(EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_CustomerSupportRepId ON C
ustomer (SupportRepId)\n\nCREATE TABLE PlaylistTrack\n(\n    PlaylistId INTEGER  NOT NULL,\n    TrackId INT
EGER  NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY  (PlaylistId, TrackId),\n    FOREIGN KEY (Play
listId) REFERENCES Playlist (PlaylistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (T
rackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Album\n
(\n    AlbumId INTEGER  NOT NULL,\n    Title NVARCHAR(160)  NOT NULL,\n    ArtistId INTEGER  NOT NULL,\n
CONSTRAINT PK_Album PRIMARY KEY  (AlbumId),\n    FOREIGN KEY (ArtistId) REFERENCES Artist (ArtistId) \n\t\t
ON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON Invoice (CustomerId)\n
\nCREATE TABLE Track\n(\n    TrackId INTEGER  NOT NULL,\n    Name NVARCHAR(200)  NOT NULL,\n    AlbumId INT
EGER,\n    MediaTypeId INTEGER  NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseco
nds INTEGER  NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Track
PRIMARY KEY  (TrackId),\n    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumId) \n\t\tON DELETE NO ACTION ON
UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES Genre (GenreId) \n\t\tON DELETE NO ACTION ON UPDATE
NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES MediaType (MediaTypeId) \n\t\tON DELETE NO ACTION ON U
PDATE NO ACTION\n)\n\nCREATE TABLE Playlist\n(\n    PlaylistId INTEGER  NOT NULL,\n    Name NVARCHAR(12
0),\n    CONSTRAINT PK_Playlist PRIMARY KEY  (PlaylistId)\n)\n\n\n===Additional Context \n\nIn the chinook
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': 'How many customers are there'},
{'role': 'assistant', 'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'content': 'what are t
he top 5 countries that customers come from?'}, {'role': 'assistant', 'content': 'SELECT Country, COUNT(*)
as CustomerCount\nFROM Customer\nGROUP BY Country\nORDER BY CustomerCount DESC\nLIMIT 5;'}, {'role': 'use
r', 'content': 'How many records are in table called customer'}, {'role': 'assistant', 'content': 'SELECT C
OUNT(*) FROM Customer;'}, {'role': 'user', 'content': '  \n    Get the total number of invoices for each cu
stomer\n'}, {'role': 'assistant', 'content': 'SELECT CustomerId, COUNT(InvoiceId) as TotalInvoices\nFROM In
voice\nGROUP BY CustomerId;'}, {'role': 'user', 'content': '  \n    List all invoices with a total exceedin
g $10:\n'}, {'role': 'assistant', 'content': 'SELECT * FROM Invoice\nWHERE Total > 10;'}, {'role': 'user',
'content': '  \n    Get the average invoice total for each customer:\n'}, {'role': 'assistant', 'content':
'SELECT CustomerId, AVG(Total) as AverageInvoiceTotal\nFROM Invoice\nGROUP BY CustomerId;'}, {'role': 'use
r', 'content': '  \n    Find the total number of invoices per country:\n'}, {'role': 'assistant', 'conten
t': 'SELECT BillingCountry, COUNT(*) as TotalInvoices\nFROM Invoice\nGROUP BY BillingCountry;'}, {'role':
'user', 'content': '  \n    Find all invoices since 2010 and the total amount invoiced:\n'}, {'role': 'assi
stant', 'content': "SELECT InvoiceDate, SUM(Total) as TotalAmount\nFROM Invoice\nWHERE InvoiceDate >= '2010
-01-01'\nGROUP BY InvoiceDate;"}, {'role': 'user', 'content': "  \n    List all employees and their reporti
ng manager's name (if any):\n"}, {'role': 'assistant', 'content': "SELECT E1.FirstName || ' ' || E1.LastNam
e AS EmployeeName, \n       E2.FirstName || ' ' || E2.LastName AS ManagerName\nFROM Employee E1 \nLEFT JOIN
Employee E2 ON E1.ReportsTo = E2.EmployeeId;"}, {'role': 'user', 'content': '  \n    List all albums and th
eir corresponding artist names  \n'}, {'role': 'assistant', 'content': 'SELECT Album.Title, Artist.Name\nFR
OM Album\nJOIN Artist ON Album.ArtistId = Artist.ArtistId;'}, {'role': 'user', 'content': '  \n    List all
customers who have not placed any orders:\n'}]
Using model gpt-4 for 1505.75 tokens (approx)
SELECT *
FROM Customer
WHERE CustomerId NOT IN (SELECT DISTINCT CustomerId FROM Invoice);
SELECT *
FROM Customer
WHERE CustomerId NOT IN (SELECT DISTINCT CustomerId FROM Invoice);
SELECT *
FROM Customer
WHERE CustomerId NOT IN (SELECT DISTINCT CustomerId FROM Invoice);
Empty DataFrame
Columns: [CustomerId, FirstName, LastName, Company, Address, City, State, Country, PostalCode, Phone, Fax,
Email, SupportRepId]
Index: []
Using model gpt-4 for 251.5 tokens (approx)
```

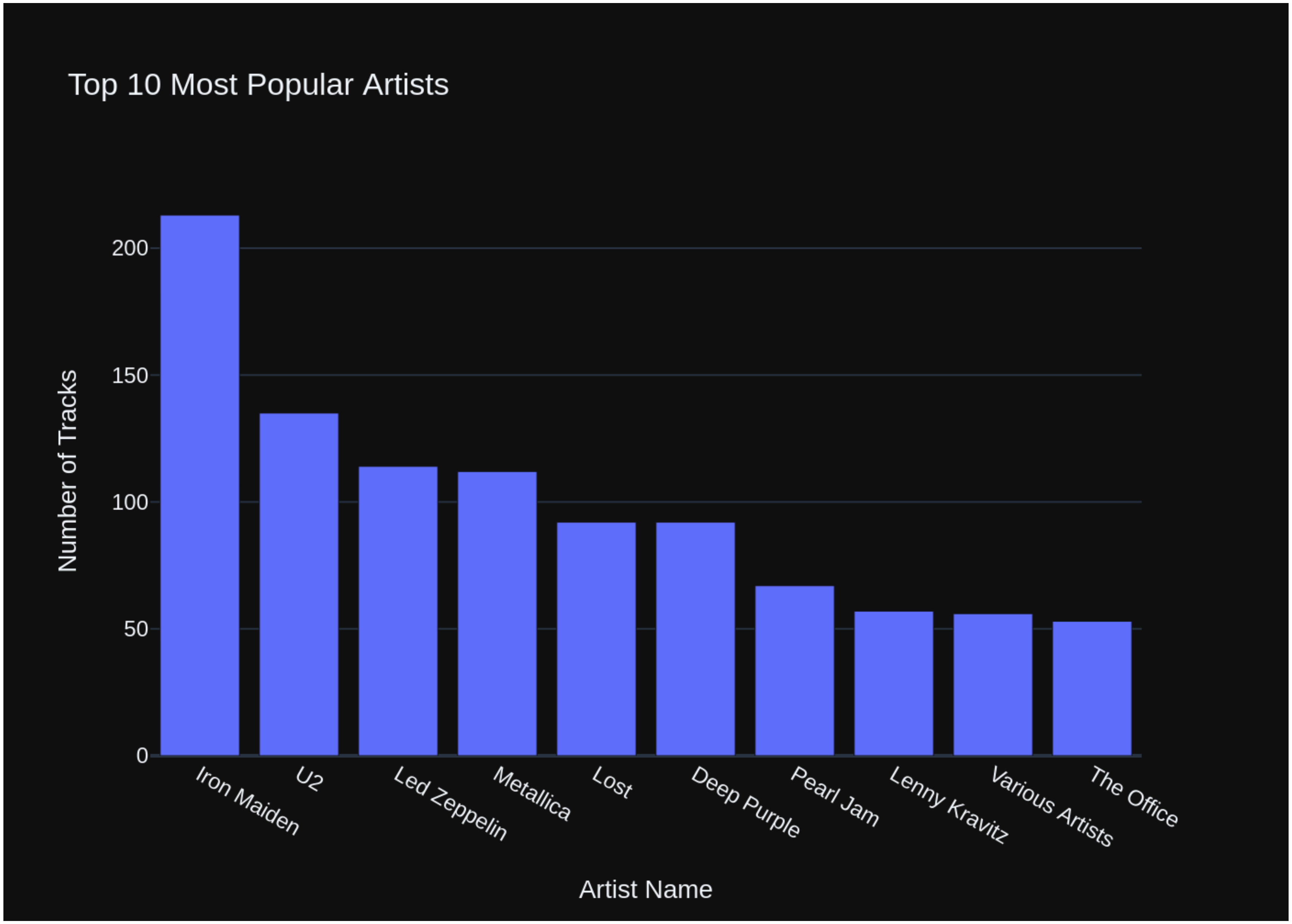

4
3
2
1
0
−1
−1
0
1
2
3
4
5
6

Out[33]:
```
('SELECT *\nFROM Customer\nWHERE CustomerId NOT IN (SELECT DISTINCT CustomerId FROM Invoice);',
 Empty DataFrame
 Columns: [CustomerId, FirstName, LastName, Company, Address, City, State, Country, PostalCode, Phone, Fa
x, Email, SupportRepId]
 Index: [],
 Figure({
     'data': [{'type': 'bar', 'x': array([], dtype=object), 'y': array([], dtype=object)}], 'layout': {'te
mplate': '...'}
 }))
```

In [34]:
```
question = """
    Get 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 fo
rmat instructions. \n===Tables \nCREATE TABLE Track\n(\n    TrackId INTEGER  NOT NULL,\n    Name NVARCHAR(2
00)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER  NOT NULL,\n    GenreId INTEGER,\n    Compose
r NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT
NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumI
d) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES Genre (GenreId) \n
\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES MediaType (MediaType
Id) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_AlbumArtistId ON Album (ArtistId)
\n\nCREATE INDEX IFK_TrackAlbumId ON Track (AlbumId)\n\nCREATE TABLE Artist\n(\n    ArtistId INTEGER  NOT N
ULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK_Artist PRIMARY KEY  (ArtistId)\n)\n\nCREATE INDEX IFK_Trac
kGenreId ON Track (GenreId)\n\nCREATE TABLE Album\n(\n    AlbumId INTEGER  NOT NULL,\n    Title NVARCHAR(16
0)  NOT NULL,\n    ArtistId INTEGER  NOT NULL,\n    CONSTRAINT PK_Album PRIMARY KEY  (AlbumId),\n    FOREIG
N KEY (ArtistId) REFERENCES Artist (ArtistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE IN
DEX IFK_PlaylistTrackTrackId ON PlaylistTrack (TrackId)\n\nCREATE INDEX IFK_TrackMediaTypeId ON Track (Medi
aTypeId)\n\nCREATE TABLE Playlist\n(\n    PlaylistId INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONST
RAINT PK_Playlist PRIMARY KEY  (PlaylistId)\n)\n\nCREATE TABLE PlaylistTrack\n(\n    PlaylistId INTEGER  NO
T NULL,\n    TrackId INTEGER  NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY  (PlaylistId, TrackI
d),\n    FOREIGN KEY (PlaylistId) REFERENCES Playlist (PlaylistId) \n\t\tON DELETE NO ACTION ON UPDATE NO A
CTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION
\n)\n\n\n===Additional Context \n\nIn the chinook database invoice means order\n\n===Response Guidelines \n
1. If the provided context is sufficient, please generate a valid SQL query without any explanations for th
e question. \n2. If the provided context is almost sufficient but requires knowledge of a specific string i
n a particular column, please generate an intermediate SQL query to find the distinct strings in that colum
n. Prepend the query with a comment saying intermediate_sql \n3. If the provided context is insufficient, p
lease explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the question h
as 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': 'assistan
t', 'content': 'SELECT Genre.Name, COUNT(Track.TrackId) as NumberOfTracks\nFROM Genre\nLEFT JOIN Track ON G
enre.GenreId = Track.GenreId\nGROUP BY Genre.GenreId;'}, {'role': 'user', 'content': '  \n    Find the top
5 most expensive tracks (based on unit price):\n'}, {'role': 'assistant', 'content': 'SELECT * FROM Track\n
ORDER BY UnitPrice DESC\nLIMIT 5;'}, {'role': 'user', 'content': '  \n    List all albums and their corresp
onding artist names  \n'}, {'role': 'assistant', 'content': 'SELECT Album.Title, Artist.Name\nFROM Album\nJ
OIN Artist ON Album.ArtistId = Artist.ArtistId;'}, {'role': 'user', 'content': 'what are the top 5 countrie
s that customers come from?'}, {'role': 'assistant', 'content': 'SELECT Country, COUNT(*) as CustomerCount
\nFROM Customer\nGROUP BY Country\nORDER BY CustomerCount DESC\nLIMIT 5;'}, {'role': 'user', 'content': '
\n    List all invoices with a total exceeding $10:\n'}, {'role': 'assistant', 'content': 'SELECT * FROM In
voice\nWHERE Total > 10;'}, {'role': 'user', 'content': '  \n    Find all tracks with a name containing "Wh
at" (case-insensitive)\n'}, {'role': 'assistant', 'content': "SELECT * FROM Track\nWHERE Name LIKE '%What%'
COLLATE NOCASE;"}, {'role': 'user', 'content': '  \n    Get the average invoice total for each custome
r:\n'}, {'role': 'assistant', 'content': 'SELECT CustomerId, AVG(Total) as AverageInvoiceTotal\nFROM Invoic
e\nGROUP BY CustomerId;'}, {'role': 'user', 'content': 'How many customers are there'}, {'role': 'assistan
t', 'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'content': 'How many records are in tabl
e called customer'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'user',
'content': '  \n    Find the total number of invoices per country:\n'}, {'role': 'assistant', 'content': 'S
ELECT BillingCountry, COUNT(*) as TotalInvoices\nFROM Invoice\nGROUP BY BillingCountry;'}, {'role': 'user',
'content': '  \n    Get the top 10 most popular artists (based on the number of tracks):\n'}]
Using model gpt-4 for 1025.75 tokens (approx)
SELECT Artist.Name, COUNT(Track.TrackId) as NumberOfTracks
FROM Artist
JOIN Album ON Artist.ArtistId = Album.ArtistId
JOIN Track ON Album.AlbumId = Track.AlbumId
GROUP BY Artist.ArtistId
ORDER BY NumberOfTracks DESC
LIMIT 10;
SELECT Artist.Name, COUNT(Track.TrackId) as NumberOfTracks
FROM Artist
JOIN Album ON Artist.ArtistId = Album.ArtistId
JOIN Track ON Album.AlbumId = Track.AlbumId
GROUP BY Artist.ArtistId
ORDER BY NumberOfTracks DESC
LIMIT 10;
SELECT Artist.Name, COUNT(Track.TrackId) as NumberOfTracks
FROM Artist
JOIN Album ON Artist.ArtistId = Album.ArtistId
JOIN Track ON Album.AlbumId = Track.AlbumId
GROUP BY Artist.ArtistId
ORDER BY NumberOfTracks DESC
LIMIT 10;
              Name  NumberOfTracks
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
Using model gpt-4 for 227.75 tokens (approx)
```


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

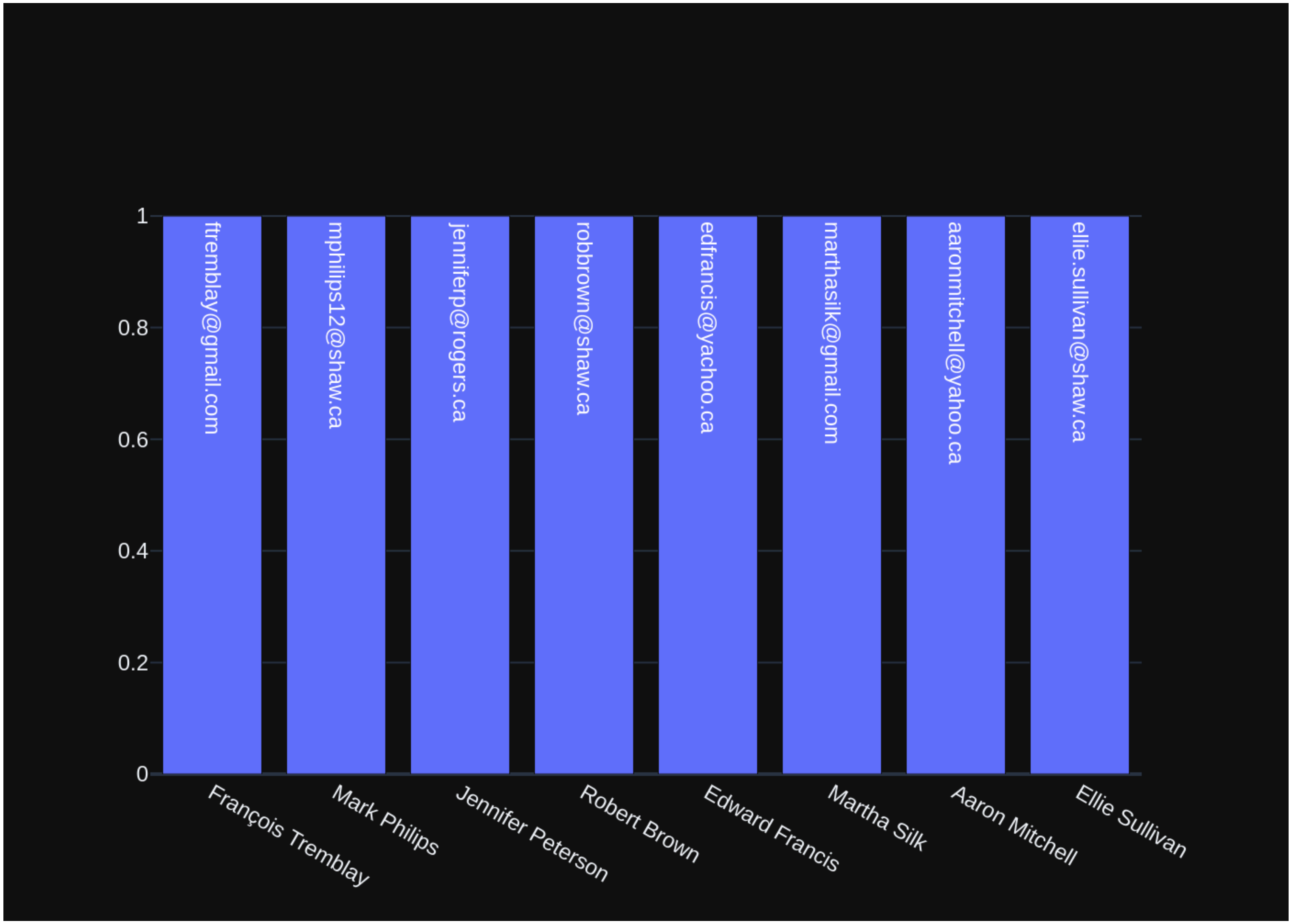
```
Out[34]: ('SELECT Artist.Name, COUNT(Track.TrackId) as NumberOfTracks\nFROM Artist\nJOIN Album ON Artist.ArtistId =
         Album.ArtistId\nJOIN Track ON Album.AlbumId = Track.AlbumId\nGROUP BY Artist.ArtistId\nORDER BY NumberOfTr
         acks DESC\nLIMIT 10;',
                       Name  NumberOfTracks
          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': 'Artist Name=%{x}<br>Number of Tracks=%{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'},
                         'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Artist Name'}},
                         'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'Number of Tracks'}}}
          }))
```

```
In [35]: 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 fo
rmat instructions. \n===Tables \nCREATE TABLE Customer\n(\n    CustomerId INTEGER  NOT NULL,\n    FirstName
NVARCHAR(40)  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    Company NVARCHAR(80),\n    Address NVARC
HAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCH
AR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60)  NOT NULL,\n    SupportRepI
d INTEGER,\n    CONSTRAINT PK_Customer PRIMARY KEY  (CustomerId),\n    FOREIGN KEY (SupportRepId) REFERENCE
S Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_CustomerSuppor
tRepId ON Customer (SupportRepId)\n\nCREATE TABLE Invoice\n(\n    InvoiceId INTEGER  NOT NULL,\n    Custome
rId INTEGER  NOT NULL,\n    InvoiceDate DATETIME  NOT NULL,\n    BillingAddress NVARCHAR(70),\n    BillingC
ity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n    BillingPostalCode
NVARCHAR(10),\n    Total NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Invoice PRIMARY KEY  (InvoiceId),\n
FOREIGN KEY (CustomerId) REFERENCES Customer (CustomerId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)
\n\nCREATE INDEX IFK_InvoiceCustomerId ON Invoice (CustomerId)\n\nCREATE TABLE Employee\n(\n    EmployeeId
INTEGER  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    FirstName NVARCHAR(20)  NOT NULL,\n    Title
NVARCHAR(30),\n    ReportsTo INTEGER,\n    BirthDate DATETIME,\n    HireDate DATETIME,\n    Address NVARCHA
R(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCHAR
(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60),\n    CONSTRAINT PK_Employee
PRIMARY KEY  (EmployeeId),\n    FOREIGN KEY (ReportsTo) REFERENCES Employee (EmployeeId) \n\t\tON DELETE NO
ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE InvoiceLine\n(\n    InvoiceLineId INTEGER  NOT NULL,\n    Inv
oiceId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    Quan
tity INTEGER  NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (InvoiceLineId),\n    FOREIGN KEY (Invo
iceId) REFERENCES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (Trac
kId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_Invoic
eLineTrackId ON InvoiceLine (TrackId)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON InvoiceLine (InvoiceId)\n
\nCREATE INDEX IFK_EmployeeReportsTo ON Employee (ReportsTo)\n\nCREATE TABLE PlaylistTrack\n(\n    Playlist
Id INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY  (Playli
stId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES Playlist (PlaylistId) \n\t\tON DELETE NO ACTION ON
UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE
NO ACTION\n)\n\n\n===Additional Context \n\nIn the chinook database invoice means order\n\n===Response Guid
elines \n1. If the provided context is sufficient, please generate a valid SQL query without any explanatio
ns 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 th
at column. Prepend the query with a comment saying intermediate_sql \n3. If the provided context is insuffi
cient, please explain why it can't be generated. \n4. Please use the most relevant table(s). \n5. If the qu
estion 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': 'assistan
t', 'content': 'SELECT Country, COUNT(*) as CustomerCount\nFROM Customer\nGROUP BY Country\nORDER BY Custom
erCount DESC\nLIMIT 5;'}, {'role': 'user', 'content': 'How many customers are there'}, {'role': 'assistan
t', 'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'content': '  \n    Get the total number
of invoices for each customer\n'}, {'role': 'assistant', 'content': 'SELECT CustomerId, COUNT(InvoiceId) as
TotalInvoices\nFROM Invoice\nGROUP BY CustomerId;'}, {'role': 'user', 'content': 'How many records are in t
able called customer'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'use
r', 'content': '  \n    Find the total number of invoices per country:\n'}, {'role': 'assistant', 'conten
t': 'SELECT BillingCountry, COUNT(*) as TotalInvoices\nFROM Invoice\nGROUP BY BillingCountry;'}, {'role':
'user', 'content': "  \n    List all employees and their reporting manager's name (if any):\n"}, {'role':
'assistant', 'content': "SELECT E1.FirstName || ' ' || E1.LastName AS EmployeeName, \n       E2.FirstName |
| ' ' || E2.LastName AS ManagerName\nFROM Employee E1 \nLEFT JOIN Employee E2 ON E1.ReportsTo = E2.Employee
Id;"}, {'role': 'user', 'content': '  \n    List all invoices with a total exceeding $10:\n'}, {'role': 'as
sistant', 'content': 'SELECT * FROM Invoice\nWHERE Total > 10;'}, {'role': 'user', 'content': '  \n    Get
the average invoice total for each customer:\n'}, {'role': 'assistant', 'content': 'SELECT CustomerId, AVG
(Total) as AverageInvoiceTotal\nFROM Invoice\nGROUP BY CustomerId;'}, {'role': 'user', 'content': '  \n
Find all invoices since 2010 and the total amount invoiced:\n'}, {'role': 'assistant', 'content': "SELECT I
nvoiceDate, SUM(Total) as TotalAmount\nFROM Invoice\nWHERE InvoiceDate >= '2010-01-01'\nGROUP BY InvoiceDat
e;"}, {'role': 'user', 'content': '  \n    Get the top 10 most popular artists (based on the number of trac
ks):\n'}, {'role': 'assistant', 'content': 'SELECT Artist.Name, COUNT(Track.TrackId) as NumberOfTracks\nFRO
M Artist\nJOIN Album ON Artist.ArtistId = Album.ArtistId\nJOIN Track ON Album.AlbumId = Track.AlbumId\nGROU
P BY Artist.ArtistId\nORDER BY NumberOfTracks DESC\nLIMIT 10;'}, {'role': 'user', 'content': '  \n     List
all customers from Canada and their email addresses:\n'}]
Using model gpt-4 for 1330.5 tokens (approx)
SELECT FirstName, LastName, Email
FROM Customer
WHERE Country = 'Canada';
SELECT FirstName, LastName, Email
FROM Customer
WHERE Country = 'Canada';
SELECT FirstName, LastName, Email
FROM Customer
WHERE Country = 'Canada';
  FirstName  LastName                   Email
0  François  Tremblay     ftremblay@gmail.com
1      Mark   Philips      mphilips12@shaw.ca
2  Jennifer  Peterson     jenniferp@rogers.ca
3    Robert     Brown        robbrown@shaw.ca
4    Edward   Francis     edfrancis@yachoo.ca
5    Martha      Silk    marthasilk@gmail.com
6     Aaron  Mitchell  aaronmitchell@yahoo.ca
7     Ellie  Sullivan  ellie.sullivan@shaw.ca
Using model gpt-4 for 190.0 tokens (approx)
```

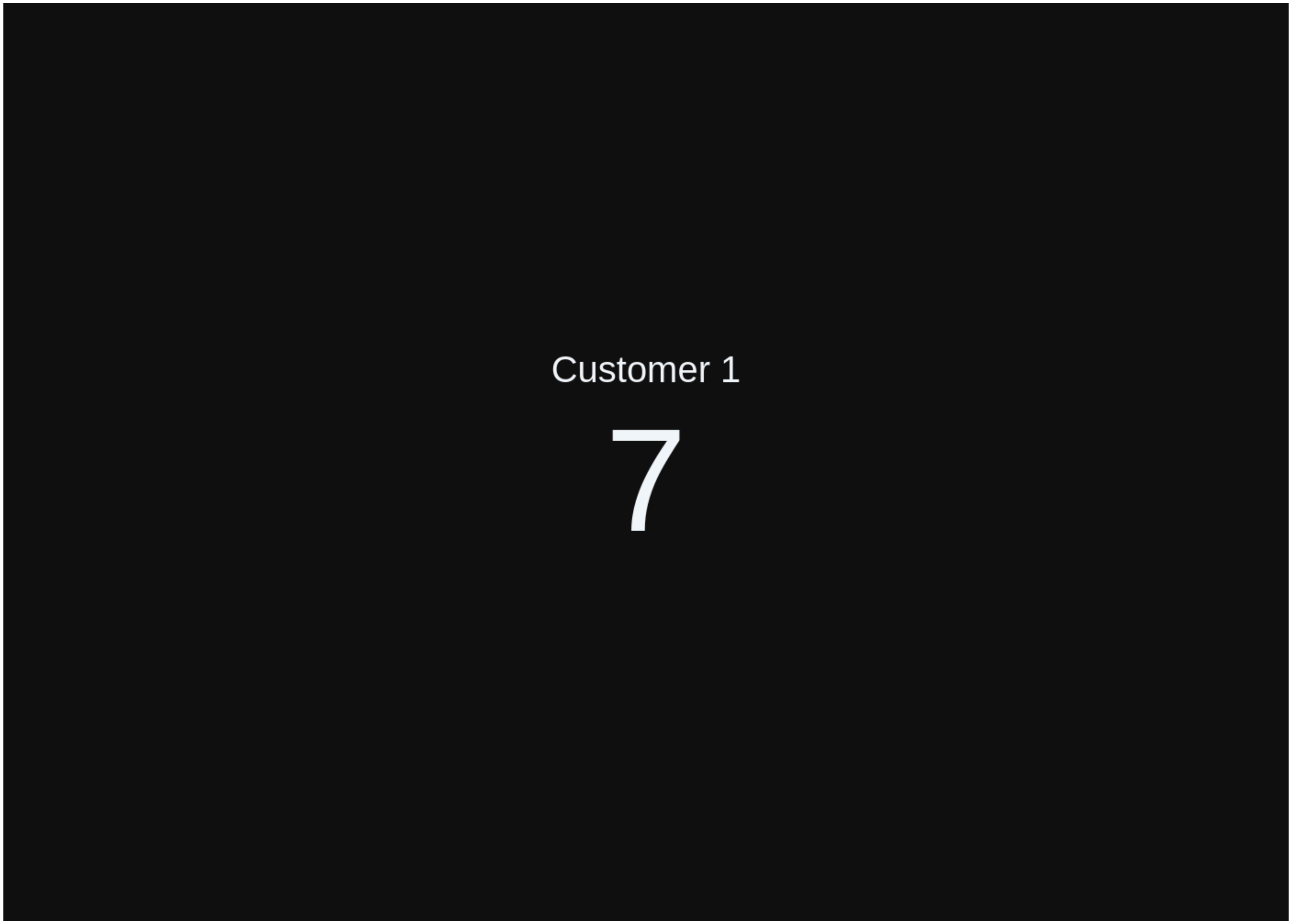

1
0.8
0.6
0.4
0.2
0
ftremblay@gmail.com
mphilips12@shaw.ca
jenniferp@rogers.ca
robbrown@shaw.ca
edfrancis@yachoo.ca
marthasilk@gmail.com
aaronmitchell@yahoo.ca
ellie.sullivan@shaw.ca
François Tremblay
Mark Philips
Jennifer Peterson
Robert Brown
Edward Francis
Martha Silk
Aaron Mitchell
Ellie Sullivan

```
Out[35]: ("SELECT FirstName, LastName, Email \nFROM Customer\nWHERE Country = 'Canada';",
            FirstName  LastName                 Email
          0  François  Tremblay     ftremblay@gmail.com
          1      Mark   Philips      mphilips12@shaw.ca
          2  Jennifer  Peterson     jenniferp@rogers.ca
          3    Robert     Brown        robbrown@shaw.ca
          4    Edward   Francis     edfrancis@yachoo.ca
          5    Martha      Silk    marthasilk@gmail.com
          6     Aaron  Mitchell  aaronmitchell@yahoo.ca
          7     Ellie  Sullivan  ellie.sullivan@shaw.ca,
          Figure({
              'data': [{'text': 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),
                        'textposition': 'auto',
                        'type': 'bar',
                        'x': array(['François Tremblay', 'Mark Philips', 'Jennifer Peterson',
                                    'Robert Brown', 'Edward Francis', 'Martha Silk', 'Aaron Mitchell',
                                    'Ellie Sullivan'], dtype=object),
                        'y': [1, 1, 1, 1, 1, 1, 1, 1]}],
              'layout': {'template': '...'}
          }))
```

```
In [36]: 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 fo
rmat instructions. \n===Tables \nCREATE INDEX IFK_InvoiceCustomerId ON Invoice (CustomerId)\n\nCREATE TABLE
Invoice\n(\n    InvoiceId INTEGER  NOT NULL,\n    CustomerId INTEGER  NOT NULL,\n    InvoiceDate DATETIME
NOT NULL,\n    BillingAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(4
0),\n    BillingCountry NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2)  NOT NU
LL,\n    CONSTRAINT PK_Invoice PRIMARY KEY  (InvoiceId),\n    FOREIGN KEY (CustomerId) REFERENCES Customer
(CustomerId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON I
nvoiceLine (InvoiceId)\n\nCREATE TABLE InvoiceLine\n(\n    InvoiceLineId INTEGER  NOT NULL,\n    InvoiceId
INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    Quantity IN
TEGER  NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (InvoiceLineId),\n    FOREIGN KEY (InvoiceId)
REFERENCES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) RE
FERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_InvoiceLineTr
ackId ON InvoiceLine (TrackId)\n\nCREATE TABLE Customer\n(\n    CustomerId INTEGER  NOT NULL,\n    FirstNam
e NVARCHAR(40)  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    Company NVARCHAR(80),\n    Address NVA
RCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVAR
CHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60)  NOT NULL,\n    SupportRe
pId INTEGER,\n    CONSTRAINT PK_Customer PRIMARY KEY  (CustomerId),\n    FOREIGN KEY (SupportRepId) REFEREN
CES Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_CustomerSupp
ortRepId ON Customer (SupportRepId)\n\nCREATE TABLE Employee\n(\n    EmployeeId INTEGER  NOT NULL,\n    Las
tName NVARCHAR(20)  NOT NULL,\n    FirstName NVARCHAR(20)  NOT NULL,\n    Title NVARCHAR(30),\n    ReportsT
o INTEGER,\n    BirthDate DATETIME,\n    HireDate DATETIME,\n    Address NVARCHAR(70),\n    City NVARCHAR(4
0),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(2
4),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60),\n    CONSTRAINT PK_Employee PRIMARY KEY  (EmployeeI
d),\n    FOREIGN KEY (ReportsTo) REFERENCES Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO AC
TION\n)\n\nCREATE INDEX IFK_EmployeeReportsTo ON Employee (ReportsTo)\n\nCREATE TABLE Track\n(\n    TrackId
INTEGER  NOT NULL,\n    Name NVARCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER  NOT N
ULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Bytes INT
EGER,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FOREIGN
KEY (AlbumId) REFERENCES Album (AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (G
enreId) REFERENCES Genre (GenreId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTy
peId) REFERENCES MediaType (MediaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n\n===Additiona
l Context \n\nIn the chinook database invoice means order\n\n===Response Guidelines \n1. If the provided co
ntext is sufficient, please generate a valid SQL query without any explanations for the question. \n2. If t
he 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    Get the total number of invoices for each customer\n'}, {'role': 'assistant', 'content': 'SELECT Cust
omerId, COUNT(InvoiceId) as TotalInvoices\nFROM Invoice\nGROUP BY CustomerId;'}, {'role': 'user', 'conten
t': '  \n    List all invoices with a total exceeding $10:\n'}, {'role': 'assistant', 'content': 'SELECT *
FROM Invoice\nWHERE Total > 10;'}, {'role': 'user', 'content': '  \n    Find the total number of invoices p
er country:\n'}, {'role': 'assistant', 'content': 'SELECT BillingCountry, COUNT(*) as TotalInvoices\nFROM I
nvoice\nGROUP BY BillingCountry;'}, {'role': 'user', 'content': '  \n    Find all invoices since 2010 and t
he total amount invoiced:\n'}, {'role': 'assistant', 'content': "SELECT InvoiceDate, SUM(Total) as TotalAmo
unt\nFROM Invoice\nWHERE InvoiceDate >= '2010-01-01'\nGROUP BY InvoiceDate;"}, {'role': 'user', 'content':
'  \n    Get the average invoice total for each customer:\n'}, {'role': 'assistant', 'content': 'SELECT Cus
tomerId, AVG(Total) as AverageInvoiceTotal\nFROM Invoice\nGROUP BY CustomerId;'}, {'role': 'user', 'conten
t': 'what are the top 5 countries that customers come from?'}, {'role': 'assistant', 'content': 'SELECT Cou
ntry, COUNT(*) as CustomerCount\nFROM Customer\nGROUP BY Country\nORDER BY CustomerCount DESC\nLIMIT 5;'},
{'role': 'user', 'content': '  \n    Find the top 5 most expensive tracks (based on unit price):\n'}, {'rol
e': 'assistant', 'content': 'SELECT * FROM Track\nORDER BY UnitPrice DESC\nLIMIT 5;'}, {'role': 'user', 'co
ntent': 'How many customers are there'}, {'role': 'assistant', 'content': 'SELECT COUNT(*) FROM Custome
r;'}, {'role': 'user', 'content': 'How many records are in table called customer'}, {'role': 'assistant',
'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'content': '  \n     List all customers from
Canada and their email addresses:\n'}, {'role': 'assistant', 'content': "SELECT FirstName, LastName, Email
\nFROM Customer\nWHERE Country = 'Canada';"}, {'role': 'user', 'content': '  \n     Find the customer with
the most invoices \n'}]
Using model gpt-4 for 1318.75 tokens (approx)
SELECT CustomerId, COUNT(*) as InvoiceCount
FROM Invoice
GROUP BY CustomerId
ORDER BY InvoiceCount DESC
LIMIT 1;
SELECT CustomerId, COUNT(*) as InvoiceCount
FROM Invoice
GROUP BY CustomerId
ORDER BY InvoiceCount DESC
LIMIT 1;
SELECT CustomerId, COUNT(*) as InvoiceCount
FROM Invoice
GROUP BY CustomerId
ORDER BY InvoiceCount DESC
LIMIT 1;
   CustomerId  InvoiceCount
0           1             7
Using model gpt-4 for 191.5 tokens (approx)
```



Customer 1

7

```
Out[36]: ('SELECT CustomerId, COUNT(*) as InvoiceCount\nFROM Invoice\nGROUP BY CustomerId\nORDER BY InvoiceCount DE
SC\nLIMIT 1;',
    CustomerId  InvoiceCount
 0           1             7,
 Figure({
     'data': [{'mode': 'number', 'title': {'text': 'Customer 1'}, 'type': 'indicator', 'value': 7}],
     'layout': {'template': '...'}
 }))
```

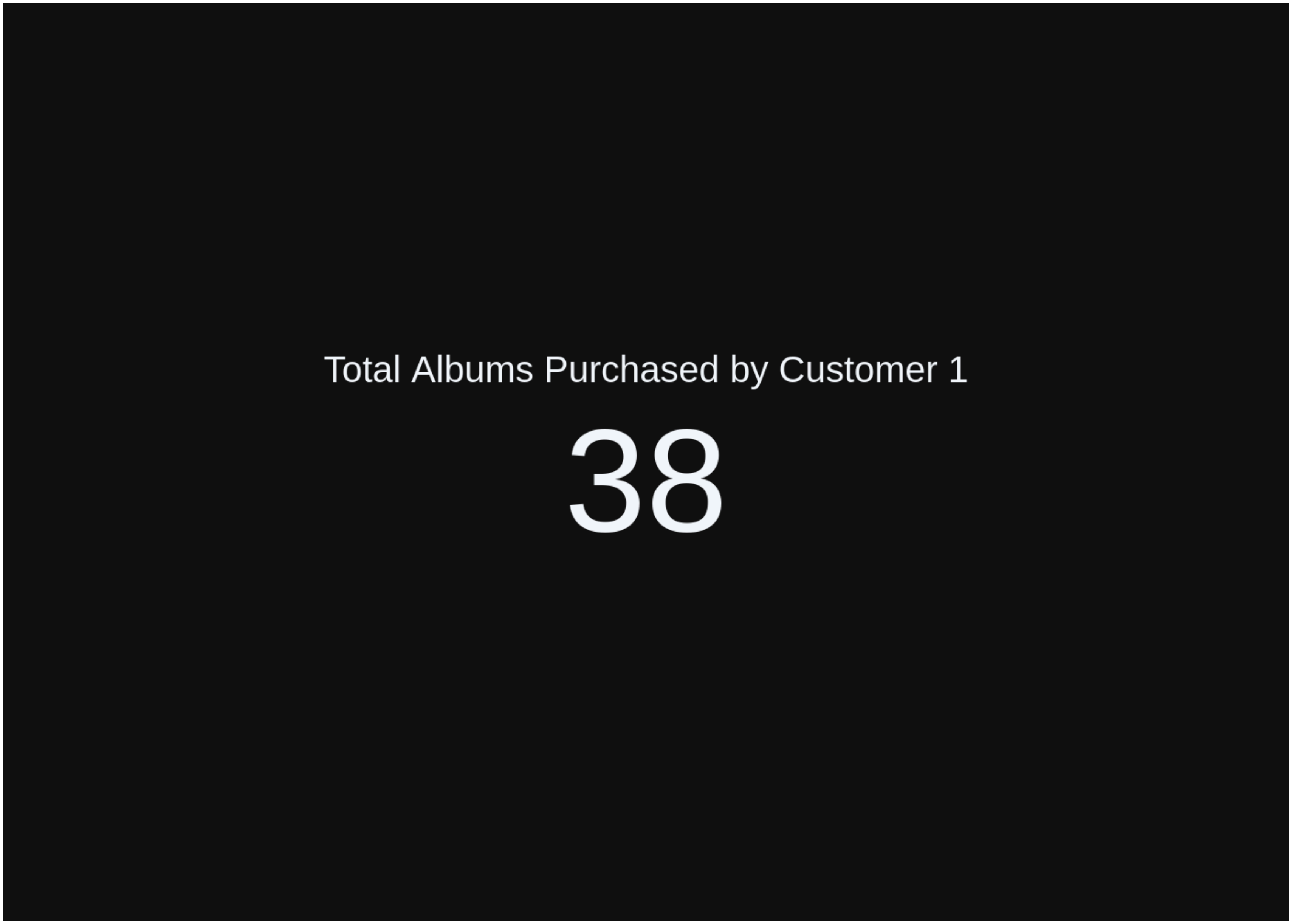
```
In [ ]:
```

Advanced SQL questions

```
In [37]: 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 fo
rmat instructions. \n===Tables \nCREATE TABLE Track\n(\n    TrackId INTEGER  NOT NULL,\n    Name NVARCHAR(2
00)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER  NOT NULL,\n    GenreId INTEGER,\n    Compose
r NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT
NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumI
d) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES Genre (GenreId) \n
\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES MediaType (MediaType
Id) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Album\n(\n    AlbumId INTEGER  NOT NUL
L,\n    Title NVARCHAR(160)  NOT NULL,\n    ArtistId INTEGER  NOT NULL,\n    CONSTRAINT PK_Album PRIMARY KE
Y  (AlbumId),\n    FOREIGN KEY (ArtistId) REFERENCES Artist (ArtistId) \n\t\tON DELETE NO ACTION ON UPDATE
NO ACTION\n)\n\nCREATE INDEX IFK_AlbumArtistId ON Album (ArtistId)\n\nCREATE TABLE InvoiceLine\n(\n    Invo
iceLineId INTEGER  NOT NULL,\n    InvoiceId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPri
ce NUMERIC(10,2)  NOT NULL,\n    Quantity INTEGER  NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (I
nvoiceLineId),\n    FOREIGN KEY (InvoiceId) REFERENCES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON UPD
ATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO
ACTION\n)\n\nCREATE INDEX IFK_InvoiceCustomerId ON Invoice (CustomerId)\n\nCREATE TABLE Invoice\n(\n    Inv
oiceId INTEGER  NOT NULL,\n    CustomerId INTEGER  NOT NULL,\n    InvoiceDate DATETIME  NOT NULL,\n    Bill
ingAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry
NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_I
nvoice PRIMARY KEY  (InvoiceId),\n    FOREIGN KEY (CustomerId) REFERENCES Customer (CustomerId) \n\t\tON DE
LETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_TrackAlbumId ON Track (AlbumId)\n\nCREATE INDEX I
FK_InvoiceLineInvoiceId ON InvoiceLine (InvoiceId)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON InvoiceLine (T
rackId)\n\nCREATE TABLE PlaylistTrack\n(\n    PlaylistId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NUL
L,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY  (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) REFER
ENCES Playlist (PlaylistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFER
ENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n\n===Additional Context \n\nIn th
e chinook database invoice means order\n\n===Response Guidelines \n1. If the provided context is sufficien
t, please generate a valid SQL query without any explanations for the question. \n2. If the provided contex
t is almost sufficient but requires knowledge of a specific string in a particular column, please generate
an intermediate SQL query to find the distinct strings in that column. Prepend the query with a comment say
ing intermediate_sql \n3. If the provided context is insufficient, please explain why it can't be generate
d. \n4. Please use the most relevant table(s). \n5. If the question has been asked and answered before, ple
ase repeat the answer exactly as it was given before. \n"}, {'role': 'user', 'content': '  \n     Find the
customer with the most invoices \n'}, {'role': 'assistant', 'content': 'SELECT CustomerId, COUNT(*) as Invo
iceCount\nFROM Invoice\nGROUP BY CustomerId\nORDER BY InvoiceCount DESC\nLIMIT 1;'}, {'role': 'user', 'cont
ent': '  \n    Get the total number of invoices for each customer\n'}, {'role': 'assistant', 'content': 'SE
LECT CustomerId, COUNT(InvoiceId) as TotalInvoices\nFROM Invoice\nGROUP BY CustomerId;'}, {'role': 'user',
'content': '  \n    Get the top 10 most popular artists (based on the number of tracks):\n'}, {'role': 'ass
istant', 'content': 'SELECT Artist.Name, COUNT(Track.TrackId) as NumberOfTracks\nFROM Artist\nJOIN Album ON
Artist.ArtistId = Album.ArtistId\nJOIN Track ON Album.AlbumId = Track.AlbumId\nGROUP BY Artist.ArtistId\nOR
DER BY NumberOfTracks DESC\nLIMIT 10;'}, {'role': 'user', 'content': '  \n    Find all invoices since 2010
and the total amount invoiced:\n'}, {'role': 'assistant', 'content': "SELECT InvoiceDate, SUM(Total) as Tot
alAmount\nFROM Invoice\nWHERE InvoiceDate >= '2010-01-01'\nGROUP BY InvoiceDate;"}, {'role': 'user', 'conte
nt': '  \n    List all invoices with a total exceeding $10:\n'}, {'role': 'assistant', 'content': 'SELECT *
FROM Invoice\nWHERE Total > 10;'}, {'role': 'user', 'content': '  \n    Find the top 5 most expensive track
s (based on unit price):\n'}, {'role': 'assistant', 'content': 'SELECT * FROM Track\nORDER BY UnitPrice DES
C\nLIMIT 5;'}, {'role': 'user', 'content': '  \n    Find the total number of invoices per country:\n'}, {'r
ole': 'assistant', 'content': 'SELECT BillingCountry, COUNT(*) as TotalInvoices\nFROM Invoice\nGROUP BY Bil
lingCountry;'}, {'role': 'user', 'content': '  \n    Get the average invoice total for each customer:\n'},
{'role': 'assistant', 'content': 'SELECT CustomerId, AVG(Total) as AverageInvoiceTotal\nFROM Invoice\nGROUP
BY CustomerId;'}, {'role': 'user', 'content': '  \n    List all albums and their corresponding artist names
\n'}, {'role': 'assistant', 'content': 'SELECT Album.Title, Artist.Name\nFROM Album\nJOIN Artist ON Album.A
rtistId = Artist.ArtistId;'}, {'role': 'user', 'content': '  \n    List all genres and the number of tracks
in each genre:\n'}, {'role': 'assistant', 'content': 'SELECT Genre.Name, COUNT(Track.TrackId) as NumberOfTr
acks\nFROM Genre\nLEFT JOIN Track ON Genre.GenreId = Track.GenreId\nGROUP BY Genre.GenreId;'}, {'role': 'us
er', 'content': '  \n     Find the customer who bought the most albums in total quantity (across all invoic
es): \n'}]
Using model gpt-4 for 1293.25 tokens (approx)
SELECT Invoice.CustomerId, SUM(InvoiceLine.Quantity) as TotalQuantity
FROM Invoice
JOIN InvoiceLine ON Invoice.InvoiceId = InvoiceLine.InvoiceId
GROUP BY Invoice.CustomerId
ORDER BY TotalQuantity DESC
LIMIT 1;
SELECT Invoice.CustomerId, SUM(InvoiceLine.Quantity) as TotalQuantity
FROM Invoice
JOIN InvoiceLine ON Invoice.InvoiceId = InvoiceLine.InvoiceId
GROUP BY Invoice.CustomerId
ORDER BY TotalQuantity DESC
LIMIT 1;
SELECT Invoice.CustomerId, SUM(InvoiceLine.Quantity) as TotalQuantity
FROM Invoice
JOIN InvoiceLine ON Invoice.InvoiceId = InvoiceLine.InvoiceId
GROUP BY Invoice.CustomerId
ORDER BY TotalQuantity DESC
LIMIT 1;
   CustomerId  TotalQuantity
0           1             38
Using model gpt-4 for 227.5 tokens (approx)
```

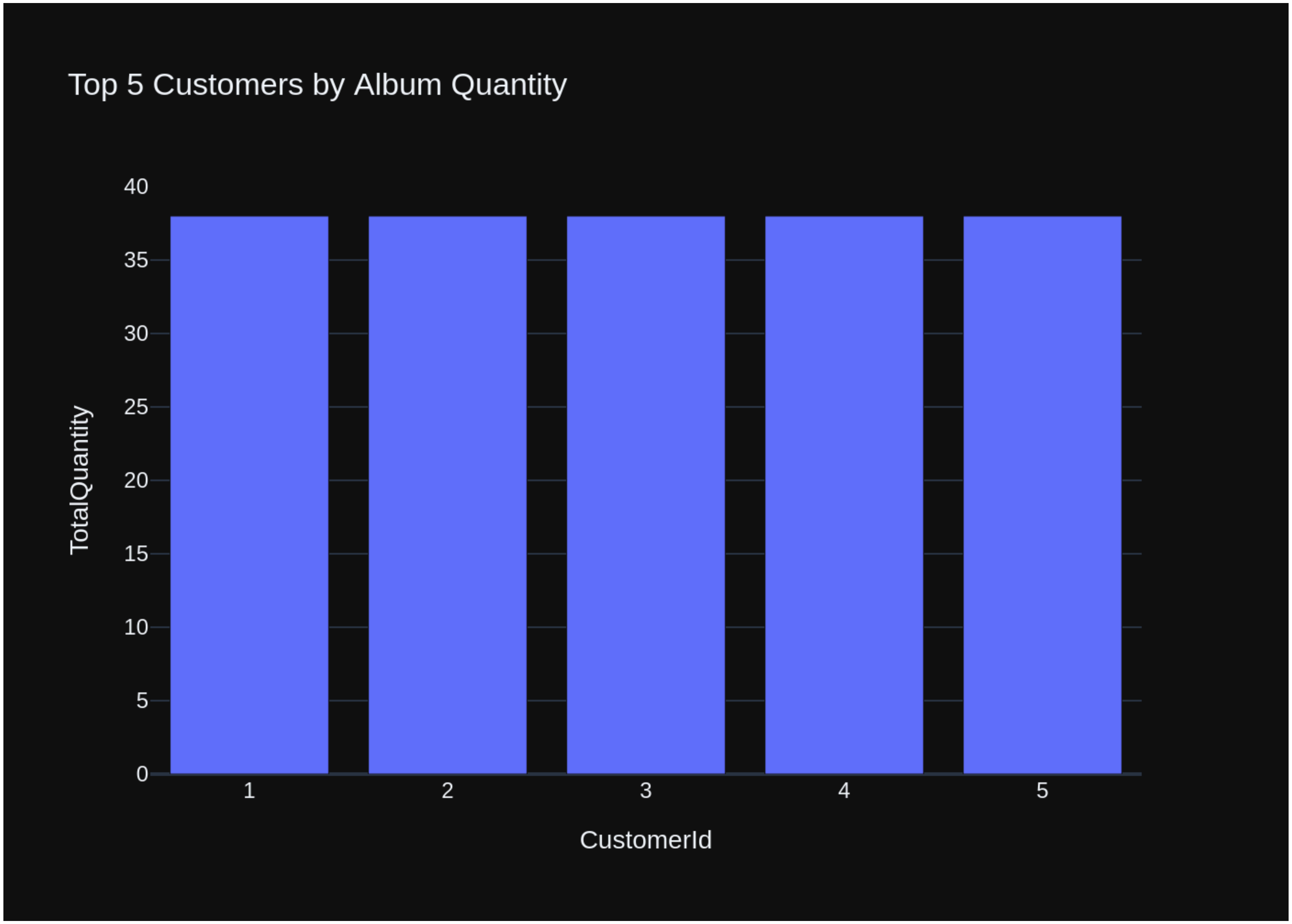

Total Albums Purchased by Customer 1
38

Out[37]:
```
('SELECT Invoice.CustomerId, SUM(InvoiceLine.Quantity) as TotalQuantity\nFROM Invoice\nJOIN InvoiceLine ON
Invoice.InvoiceId = InvoiceLine.InvoiceId\nGROUP BY Invoice.CustomerId\nORDER BY TotalQuantity DESC\nLIMIT
1;',
    CustomerId  TotalQuantity
 0           1             38,
 Figure({
     'data': [{'mode': 'number',
               'title': {'text': 'Total Albums Purchased by Customer 1'},
               'type': 'indicator',
               'value': 38}],
     'layout': {'template': '...'}
 }))
```

In [38]:
```python
question = """
     Find the top 5 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 fo
rmat instructions. \n===Tables \nCREATE TABLE Track\n(\n    TrackId INTEGER  NOT NULL,\n    Name NVARCHAR(2
00)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER  NOT NULL,\n    GenreId INTEGER,\n    Compose
r NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT
NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumI
d) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES Genre (GenreId) \n
\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES MediaType (MediaType
Id) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Album\n(\n    AlbumId INTEGER  NOT NUL
L,\n    Title NVARCHAR(160)  NOT NULL,\n    ArtistId INTEGER  NOT NULL,\n    CONSTRAINT PK_Album PRIMARY KE
Y  (AlbumId),\n    FOREIGN KEY (ArtistId) REFERENCES Artist (ArtistId) \n\t\tON DELETE NO ACTION ON UPDATE
NO ACTION\n)\n\nCREATE INDEX IFK_AlbumArtistId ON Album (ArtistId)\n\nCREATE TABLE InvoiceLine\n(\n    Invo
iceLineId INTEGER  NOT NULL,\n    InvoiceId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    UnitPri
ce NUMERIC(10,2)  NOT NULL,\n    Quantity INTEGER  NOT NULL,\n    CONSTRAINT PK_InvoiceLine PRIMARY KEY  (I
nvoiceLineId),\n    FOREIGN KEY (InvoiceId) REFERENCES Invoice (InvoiceId) \n\t\tON DELETE NO ACTION ON UPD
ATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO
ACTION\n)\n\nCREATE TABLE Invoice\n(\n    InvoiceId INTEGER  NOT NULL,\n    CustomerId INTEGER  NOT NULL,\n
InvoiceDate DATETIME  NOT NULL,\n    BillingAddress NVARCHAR(70),\n    BillingCity NVARCHAR(40),\n    Billi
ngState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n    BillingPostalCode NVARCHAR(10),\n    Total NUM
ERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Invoice PRIMARY KEY  (InvoiceId),\n    FOREIGN KEY (CustomerId) RE
FERENCES Customer (CustomerId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_Invoice
CustomerId ON Invoice (CustomerId)\n\nCREATE INDEX IFK_TrackAlbumId ON Track (AlbumId)\n\nCREATE INDEX IFK_
InvoiceLineTrackId ON InvoiceLine (TrackId)\n\nCREATE INDEX IFK_InvoiceLineInvoiceId ON InvoiceLine (Invoic
eId)\n\nCREATE TABLE Artist\n(\n    ArtistId INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK
_Artist PRIMARY KEY  (ArtistId)\n)\n\n\n===Additional Context \n\nIn the chinook database invoice means ord
er\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 k
nowledge 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 provi
ded context is insufficient, please explain why it can't be generated. \n4. Please use the most relevant ta
ble(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 t
otal quantity (across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT Invoice.CustomerId, SUM
(InvoiceLine.Quantity) as TotalQuantity\nFROM Invoice\nJOIN InvoiceLine ON Invoice.InvoiceId = InvoiceLine.
InvoiceId\nGROUP BY Invoice.CustomerId\nORDER BY TotalQuantity DESC\nLIMIT 1;'}, {'role': 'user', 'conten
t': '  \n    Get the top 10 most popular artists (based on the number of tracks):\n'}, {'role': 'assistan
t', 'content': 'SELECT Artist.Name, COUNT(Track.TrackId) as NumberOfTracks\nFROM Artist\nJOIN Album ON Arti
st.ArtistId = Album.ArtistId\nJOIN Track ON Album.AlbumId = Track.AlbumId\nGROUP BY Artist.ArtistId\nORDER
BY NumberOfTracks DESC\nLIMIT 10;'}, {'role': 'user', 'content': '  \n     Find the customer with the most
invoices \n'}, {'role': 'assistant', 'content': 'SELECT CustomerId, COUNT(*) as InvoiceCount\nFROM Invoice
\nGROUP BY CustomerId\nORDER BY InvoiceCount DESC\nLIMIT 1;'}, {'role': 'user', 'content': '  \n    Find th
e top 5 most expensive tracks (based on unit price):\n'}, {'role': 'assistant', 'content': 'SELECT * FROM T
rack\nORDER BY UnitPrice DESC\nLIMIT 5;'}, {'role': 'user', 'content': '  \n    List all invoices with a to
tal exceeding $10:\n'}, {'role': 'assistant', 'content': 'SELECT * FROM Invoice\nWHERE Total > 10;'}, {'rol
e': 'user', 'content': '  \n    Get the total number of invoices for each customer\n'}, {'role': 'assistan
t', 'content': 'SELECT CustomerId, COUNT(InvoiceId) as TotalInvoices\nFROM Invoice\nGROUP BY CustomerId;'},
{'role': 'user', 'content': '  \n    List all albums and their corresponding artist names  \n'}, {'role':
'assistant', 'content': 'SELECT Album.Title, Artist.Name\nFROM Album\nJOIN Artist ON Album.ArtistId = Artis
t.ArtistId;'}, {'role': 'user', 'content': 'what are the top 5 countries that customers come from?'}, {'rol
e': 'assistant', 'content': 'SELECT Country, COUNT(*) as CustomerCount\nFROM Customer\nGROUP BY Country\nOR
DER BY CustomerCount DESC\nLIMIT 5;'}, {'role': 'user', 'content': '  \n    Get the average invoice total f
or each customer:\n'}, {'role': 'assistant', 'content': 'SELECT CustomerId, AVG(Total) as AverageInvoiceTot
al\nFROM Invoice\nGROUP BY CustomerId;'}, {'role': 'user', 'content': '  \n    Find all invoices since 2010
and the total amount invoiced:\n'}, {'role': 'assistant', 'content': "SELECT InvoiceDate, SUM(Total) as Tot
alAmount\nFROM Invoice\nWHERE InvoiceDate >= '2010-01-01'\nGROUP BY InvoiceDate;"}, {'role': 'user', 'conte
nt': '  \n     Find the top 5 customer who bought the most albums in total quantity (across all invoice
s):\n'}]
Using model gpt-4 for 1266.0 tokens (approx)
SELECT Invoice.CustomerId, SUM(InvoiceLine.Quantity) as TotalQuantity
FROM Invoice
JOIN InvoiceLine ON Invoice.InvoiceId = InvoiceLine.InvoiceId
GROUP BY Invoice.CustomerId
ORDER BY TotalQuantity DESC
LIMIT 5;
SELECT Invoice.CustomerId, SUM(InvoiceLine.Quantity) as TotalQuantity
FROM Invoice
JOIN InvoiceLine ON Invoice.InvoiceId = InvoiceLine.InvoiceId
GROUP BY Invoice.CustomerId
ORDER BY TotalQuantity DESC
LIMIT 5;
SELECT Invoice.CustomerId, SUM(InvoiceLine.Quantity) as TotalQuantity
FROM Invoice
JOIN InvoiceLine ON Invoice.InvoiceId = InvoiceLine.InvoiceId
GROUP BY Invoice.CustomerId
ORDER BY TotalQuantity DESC
LIMIT 5;
   CustomerId  TotalQuantity
0           1             38
1           2             38
2           3             38
3           4             38
4           5             38
Using model gpt-4 for 228.75 tokens (approx)
```


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

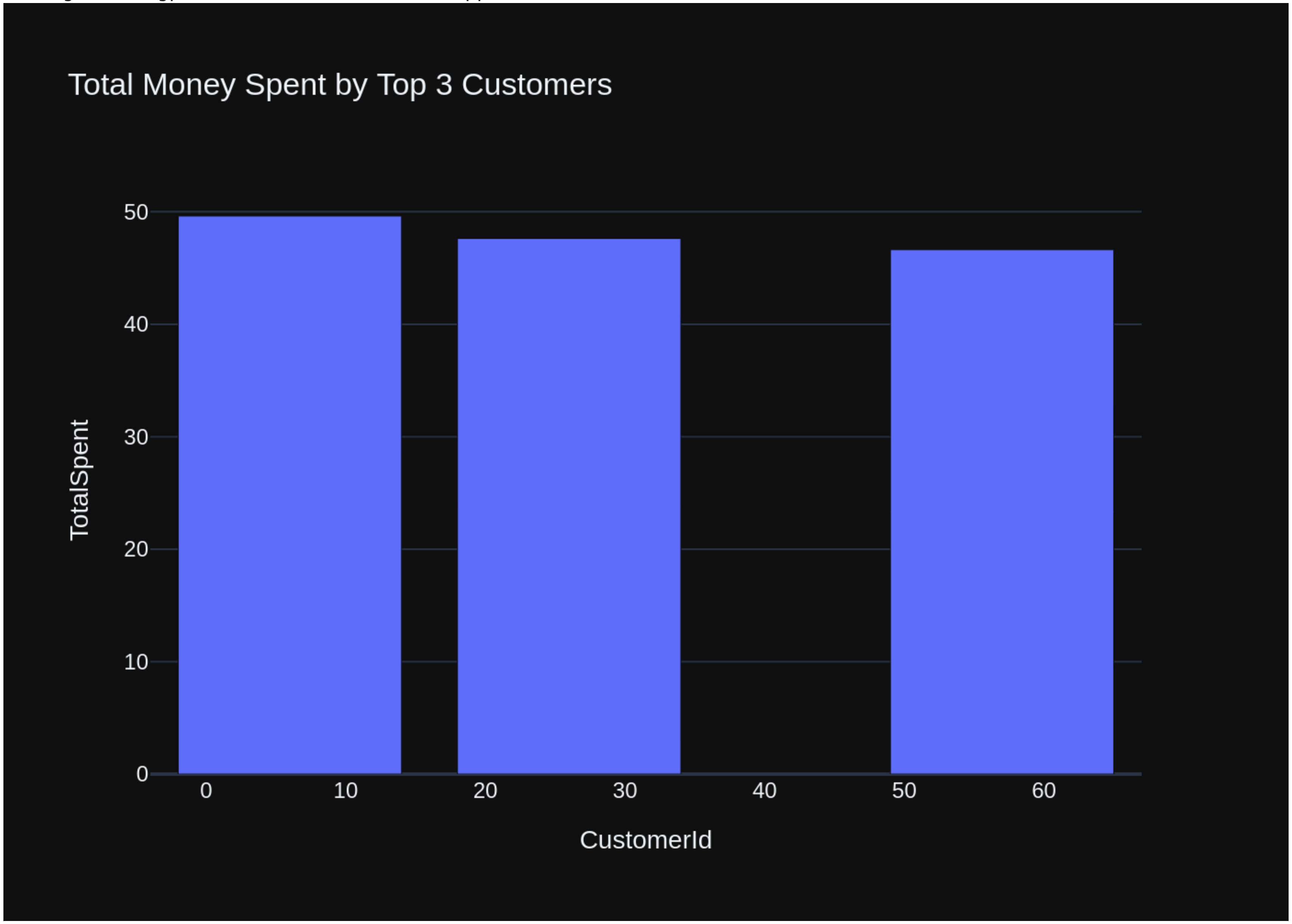
Out[38]:
```
('SELECT Invoice.CustomerId, SUM(InvoiceLine.Quantity) as TotalQuantity\nFROM Invoice\nJOIN InvoiceLine ON
Invoice.InvoiceId = InvoiceLine.InvoiceId\nGROUP BY Invoice.CustomerId\nORDER BY TotalQuantity DESC\nLIMIT
5;',
    CustomerId  TotalQuantity
 0           1             38
 1           2             38
 2           3             38
 3           4             38
 4           5             38,
 Figure({
     'data': [{'alignmentgroup': 'True',
               'hovertemplate': 'CustomerId=%{x}<br>TotalQuantity=%{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': 'relative',
                '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': 'TotalQuantity'}}}
 }))
```

In [39]:
```
question = """
     Find the top 3 customers who spent the most money overall:
"""

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 fo
rmat instructions. \n===Tables \nCREATE TABLE Invoice\n(\n    InvoiceId INTEGER  NOT NULL,\n    CustomerId
INTEGER  NOT NULL,\n    InvoiceDate DATETIME  NOT NULL,\n    BillingAddress NVARCHAR(70),\n    BillingCity
NVARCHAR(40),\n    BillingState NVARCHAR(40),\n    BillingCountry NVARCHAR(40),\n    BillingPostalCode NVAR
CHAR(10),\n    Total NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Invoice PRIMARY KEY  (InvoiceId),\n    FOR
EIGN KEY (CustomerId) REFERENCES Customer (CustomerId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n
CREATE TABLE InvoiceLine\n(\n    InvoiceLineId INTEGER  NOT NULL,\n    InvoiceId INTEGER  NOT NULL,\n    Tr
ackId INTEGER  NOT NULL,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    Quantity INTEGER  NOT NULL,\n    CONS
TRAINT PK_InvoiceLine PRIMARY KEY  (InvoiceLineId),\n    FOREIGN KEY (InvoiceId) REFERENCES Invoice (Invoic
eId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackId)
\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Customer\n(\n    CustomerId INTEGER  NOT N
ULL,\n    FirstName NVARCHAR(40)  NOT NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    Company NVARCHAR(8
0),\n    Address NVARCHAR(70),\n    City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(4
0),\n    PostalCode NVARCHAR(10),\n    Phone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60)
NOT NULL,\n    SupportRepId INTEGER,\n    CONSTRAINT PK_Customer PRIMARY KEY  (CustomerId),\n    FOREIGN KE
Y (SupportRepId) REFERENCES Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREAT
E INDEX IFK_CustomerSupportRepId ON Customer (SupportRepId)\n\nCREATE TABLE Track\n(\n    TrackId INTEGER
NOT NULL,\n    Name NVARCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER  NOT NULL,\n
GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Bytes INTEGER,\n
UnitPrice NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FOREIGN KEY (Album
Id) REFERENCES Album (AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) RE
FERENCES Genre (GenreId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFE
RENCES MediaType (MediaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_Invoice
CustomerId ON Invoice (CustomerId)\n\nCREATE INDEX IFK_EmployeeReportsTo ON Employee (ReportsTo)\n\nCREATE
TABLE PlaylistTrack\n(\n    PlaylistId INTEGER  NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    CONSTRAINT P
K_PlaylistTrack PRIMARY KEY  (PlaylistId, TrackId),\n    FOREIGN KEY (PlaylistId) REFERENCES Playlist (Play
listId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackI
d) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE Employee\n(\n    EmployeeId INTEGER  NO
T NULL,\n    LastName NVARCHAR(20)  NOT NULL,\n    FirstName NVARCHAR(20)  NOT NULL,\n    Title NVARCHAR(3
0),\n    ReportsTo INTEGER,\n    BirthDate DATETIME,\n    HireDate DATETIME,\n    Address NVARCHAR(70),\n
City NVARCHAR(40),\n    State NVARCHAR(40),\n    Country NVARCHAR(40),\n    PostalCode NVARCHAR(10),\n    P
hone NVARCHAR(24),\n    Fax NVARCHAR(24),\n    Email NVARCHAR(60),\n    CONSTRAINT PK_Employee PRIMARY KEY
(EmployeeId),\n    FOREIGN KEY (ReportsTo) REFERENCES Employee (EmployeeId) \n\t\tON DELETE NO ACTION ON UP
DATE NO ACTION\n)\n\nCREATE INDEX IFK_InvoiceLineTrackId ON InvoiceLine (TrackId)\n\n\n===Additional Contex
t \n\nIn the chinook 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 provid
ed 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 co
mment 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 bef
ore, please repeat the answer exactly as it was given before. \n"}, {'role': 'user', 'content': '  \n     F
ind the top 5 customer who bought the most albums in total quantity (across all invoices):\n'}, {'role': 'a
ssistant', 'content': 'SELECT Invoice.CustomerId, SUM(InvoiceLine.Quantity) as TotalQuantity\nFROM Invoice
\nJOIN InvoiceLine ON Invoice.InvoiceId = InvoiceLine.InvoiceId\nGROUP BY Invoice.CustomerId\nORDER BY Tota
lQuantity DESC\nLIMIT 5;'}, {'role': 'user', 'content': '  \n    Find the top 5 most expensive tracks (base
d on unit price):\n'}, {'role': 'assistant', 'content': 'SELECT * FROM Track\nORDER BY UnitPrice DESC\nLIMI
T 5;'}, {'role': 'user', 'content': '  \n     Find the customer who bought the most albums in total quantit
y (across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT Invoice.CustomerId, SUM(InvoiceLin
e.Quantity) as TotalQuantity\nFROM Invoice\nJOIN InvoiceLine ON Invoice.InvoiceId = InvoiceLine.InvoiceId\n
GROUP BY Invoice.CustomerId\nORDER BY TotalQuantity DESC\nLIMIT 1;'}, {'role': 'user', 'content': '  \n
Find the customer with the most invoices \n'}, {'role': 'assistant', 'content': 'SELECT CustomerId, COUNT
(*) as InvoiceCount\nFROM Invoice\nGROUP BY CustomerId\nORDER BY InvoiceCount DESC\nLIMIT 1;'}, {'role': 'u
ser', 'content': 'what are the top 5 countries that customers come from?'}, {'role': 'assistant', 'conten
t': 'SELECT Country, COUNT(*) as CustomerCount\nFROM Customer\nGROUP BY Country\nORDER BY CustomerCount DES
C\nLIMIT 5;'}, {'role': 'user', 'content': '  \n    Get the average invoice total for each customer:\n'},
{'role': 'assistant', 'content': 'SELECT CustomerId, AVG(Total) as AverageInvoiceTotal\nFROM Invoice\nGROUP
BY CustomerId;'}, {'role': 'user', 'content': '  \n    Get the top 10 most popular artists (based on the nu
mber of tracks):\n'}, {'role': 'assistant', 'content': 'SELECT Artist.Name, COUNT(Track.TrackId) as NumberO
fTracks\nFROM Artist\nJOIN Album ON Artist.ArtistId = Album.ArtistId\nJOIN Track ON Album.AlbumId = Track.A
lbumId\nGROUP BY Artist.ArtistId\nORDER BY NumberOfTracks DESC\nLIMIT 10;'}, {'role': 'user', 'content': '
\n    Get the total number of invoices for each customer\n'}, {'role': 'assistant', 'content': 'SELECT Cust
omerId, COUNT(InvoiceId) as TotalInvoices\nFROM Invoice\nGROUP BY CustomerId;'}, {'role': 'user', 'conten
t': '  \n    List all invoices with a total exceeding $10:\n'}, {'role': 'assistant', 'content': 'SELECT *
FROM Invoice\nWHERE Total > 10;'}, {'role': 'user', 'content': 'How many customers are there'}, {'role': 'a
ssistant', 'content': 'SELECT COUNT(*) FROM Customer;'}, {'role': 'user', 'content': '  \n     Find the top
3 customers who spent the most money overall:\n'}]
Using model gpt-4 for 1533.25 tokens (approx)
SELECT CustomerId, SUM(Total) as TotalSpent
FROM Invoice
GROUP BY CustomerId
ORDER BY TotalSpent DESC
LIMIT 3;
SELECT CustomerId, SUM(Total) as TotalSpent
FROM Invoice
GROUP BY CustomerId
ORDER BY TotalSpent DESC
LIMIT 3;
SELECT CustomerId, SUM(Total) as TotalSpent
FROM Invoice
GROUP BY CustomerId
ORDER BY TotalSpent DESC
LIMIT 3;
   CustomerId  TotalSpent
```

```
0            6       49.62
1           26       47.62
2           57       46.62
Using model gpt-4 for 195.25 tokens (approx)
```



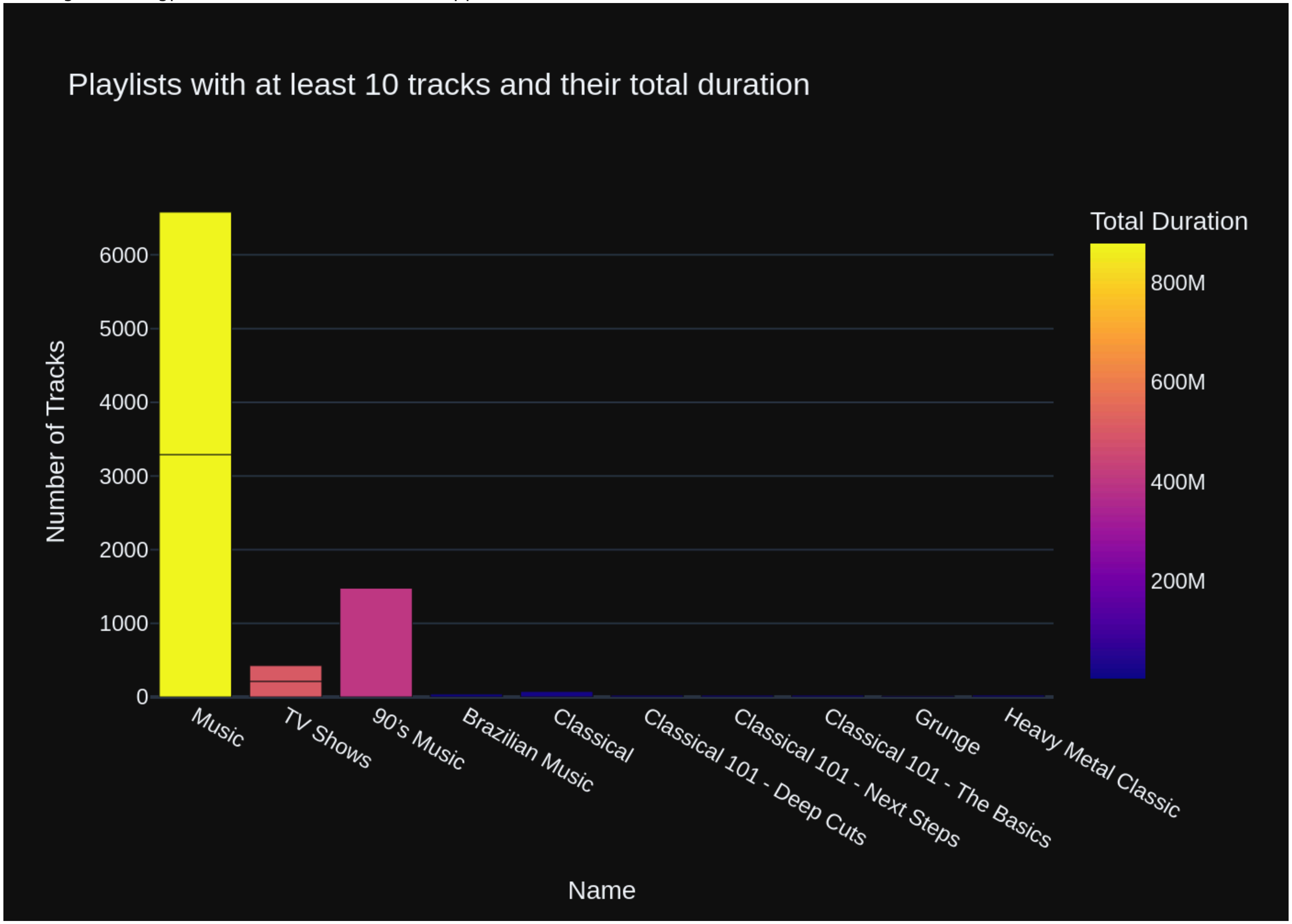
```
Out[39]: ('SELECT CustomerId, SUM(Total) as TotalSpent\nFROM Invoice\nGROUP BY CustomerId\nORDER BY TotalSpent DESC
         \nLIMIT 3;',
             CustomerId  TotalSpent
          0           6       49.62
          1          26       47.62
          2          57       46.62,
          Figure({
              'data': [{'alignmentgroup': 'True',
                        'hovertemplate': 'CustomerId=%{x}<br>TotalSpent=%{y}<extra></extra>',
                        'legendgroup': '',
                        'marker': {'color': '#636efa', 'pattern': {'shape': ''}},
                        'name': '',
                        'offsetgroup': '',
                        'orientation': 'v',
                        'showlegend': False,
                        'textposition': 'auto',
                        'type': 'bar',
                        'x': array([ 6, 26, 57]),
                        'xaxis': 'x',
                        'y': array([49.62, 47.62, 46.62]),
                        'yaxis': 'y'}],
              'layout': {'barmode': 'relative',
                         'legend': {'tracegroupgap': 0},
                         'template': '...',
                         'title': {'text': 'Total Money Spent by Top 3 Customers'},
                         'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'CustomerId'}},
                         'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'TotalSpent'}}}
          }))
```

```
In [40]: 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 fo
rmat instructions. \n===Tables \nCREATE INDEX IFK_PlaylistTrackTrackId ON PlaylistTrack (TrackId)\n\nCREATE
TABLE Playlist\n(\n    PlaylistId INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK_Playlist P
RIMARY KEY  (PlaylistId)\n)\n\nCREATE TABLE Track\n(\n    TrackId INTEGER  NOT NULL,\n    Name NVARCHAR(20
0)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId INTEGER  NOT NULL,\n    GenreId INTEGER,\n    Composer
NVARCHAR(220),\n    Milliseconds INTEGER  NOT NULL,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT N
ULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (TrackId),\n    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumId)
\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (GenreId) REFERENCES Genre (GenreId) \n\t\t
ON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIGN KEY (MediaTypeId) REFERENCES MediaType (MediaTypeId)
\n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE TABLE PlaylistTrack\n(\n    PlaylistId INTEGER
NOT NULL,\n    TrackId INTEGER  NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY  (PlaylistId, TrackI
d),\n    FOREIGN KEY (PlaylistId) REFERENCES Playlist (PlaylistId) \n\t\tON DELETE NO ACTION ON UPDATE NO A
CTION,\n    FOREIGN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION
\n)\n\nCREATE INDEX IFK_TrackGenreId ON Track (GenreId)\n\nCREATE INDEX IFK_TrackAlbumId ON Track (AlbumId)
\n\nCREATE INDEX IFK_TrackMediaTypeId ON Track (MediaTypeId)\n\nCREATE INDEX IFK_AlbumArtistId ON Album (Ar
tistId)\n\nCREATE TABLE Album\n(\n    AlbumId INTEGER  NOT NULL,\n    Title NVARCHAR(160)  NOT NULL,\n    A
rtistId INTEGER  NOT NULL,\n    CONSTRAINT PK_Album PRIMARY KEY  (AlbumId),\n    FOREIGN KEY (ArtistId) REF
ERENCES Artist (ArtistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_InvoiceLineT
rackId ON InvoiceLine (TrackId)\n\n\n===Additional Context \n\nIn the chinook 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': '  \n    List all genres and the number of tracks in each g
enre:\n'}, {'role': 'assistant', 'content': 'SELECT Genre.Name, COUNT(Track.TrackId) as NumberOfTracks\nFRO
M Genre\nLEFT JOIN Track ON Genre.GenreId = Track.GenreId\nGROUP BY Genre.GenreId;'}, {'role': 'user', 'con
tent': '  \n    Get the top 10 most popular artists (based on the number of tracks):\n'}, {'role': 'assista
nt', 'content': 'SELECT Artist.Name, COUNT(Track.TrackId) as NumberOfTracks\nFROM Artist\nJOIN Album ON Art
ist.ArtistId = Album.ArtistId\nJOIN Track ON Album.AlbumId = Track.AlbumId\nGROUP BY Artist.ArtistId\nORDER
BY NumberOfTracks DESC\nLIMIT 10;'}, {'role': 'user', 'content': '  \n     Find the top 5 customer who boug
ht the most albums in total quantity (across all invoices):\n'}, {'role': 'assistant', 'content': 'SELECT I
nvoice.CustomerId, SUM(InvoiceLine.Quantity) as TotalQuantity\nFROM Invoice\nJOIN InvoiceLine ON Invoice.In
voiceId = InvoiceLine.InvoiceId\nGROUP BY Invoice.CustomerId\nORDER BY TotalQuantity DESC\nLIMIT 5;'}, {'ro
le': 'user', 'content': '  \n     Find the customer who bought the most albums in total quantity (across al
l invoices): \n'}, {'role': 'assistant', 'content': 'SELECT Invoice.CustomerId, SUM(InvoiceLine.Quantity) a
s TotalQuantity\nFROM Invoice\nJOIN InvoiceLine ON Invoice.InvoiceId = InvoiceLine.InvoiceId\nGROUP BY Invo
ice.CustomerId\nORDER BY TotalQuantity DESC\nLIMIT 1;'}, {'role': 'user', 'content': '  \n    List all albu
ms and their corresponding artist names  \n'}, {'role': 'assistant', 'content': 'SELECT Album.Title, Artis
t.Name\nFROM Album\nJOIN Artist ON Album.ArtistId = Artist.ArtistId;'}, {'role': 'user', 'content': '  \n
Find the top 5 most expensive tracks (based on unit price):\n'}, {'role': 'assistant', 'content': 'SELECT *
FROM Track\nORDER BY UnitPrice DESC\nLIMIT 5;'}, {'role': 'user', 'content': '  \n    Find all tracks with
a name containing "What" (case-insensitive)\n'}, {'role': 'assistant', 'content': "SELECT * FROM Track\nWHE
RE Name LIKE '%What%' COLLATE NOCASE;"}, {'role': 'user', 'content': '  \n    Find all invoices since 2010
and the total amount invoiced:\n'}, {'role': 'assistant', 'content': "SELECT InvoiceDate, SUM(Total) as Tot
alAmount\nFROM Invoice\nWHERE InvoiceDate >= '2010-01-01'\nGROUP BY InvoiceDate;"}, {'role': 'user', 'conte
nt': 'Show me a list of tables in the SQLite database'}, {'role': 'assistant', 'content': "SELECT name FROM
sqlite_master WHERE type='table';"}, {'role': 'user', 'content': '  \n     Find the top 3 customers who spe
nt the most money overall:\n'}, {'role': 'assistant', 'content': 'SELECT CustomerId, SUM(Total) as TotalSpe
nt\nFROM Invoice\nGROUP BY CustomerId\nORDER BY TotalSpent DESC\nLIMIT 3;'}, {'role': 'user', 'content': '
\n     Get all playlists containing at least 10 tracks and the total duration of those tracks:\n'}]
Using model gpt-4 for 1188.5 tokens (approx)
SELECT Playlist.PlaylistId, Playlist.Name, COUNT(PlaylistTrack.TrackId) as NumberOfTracks, SUM(Track.Millis
econds) as TotalDuration
FROM Playlist
JOIN PlaylistTrack ON Playlist.PlaylistId = PlaylistTrack.PlaylistId
JOIN Track ON PlaylistTrack.TrackId = Track.TrackId
GROUP BY Playlist.PlaylistId
HAVING NumberOfTracks >= 10;
SELECT Playlist.PlaylistId, Playlist.Name, COUNT(PlaylistTrack.TrackId) as NumberOfTracks, SUM(Track.Millis
econds) as TotalDuration
FROM Playlist
JOIN PlaylistTrack ON Playlist.PlaylistId = PlaylistTrack.PlaylistId
JOIN Track ON PlaylistTrack.TrackId = Track.TrackId
GROUP BY Playlist.PlaylistId
HAVING NumberOfTracks >= 10;
SELECT Playlist.PlaylistId, Playlist.Name, COUNT(PlaylistTrack.TrackId) as NumberOfTracks, SUM(Track.Millis
econds) as TotalDuration
FROM Playlist
JOIN PlaylistTrack ON Playlist.PlaylistId = PlaylistTrack.PlaylistId
JOIN Track ON PlaylistTrack.TrackId = Track.TrackId
GROUP BY Playlist.PlaylistId
HAVING NumberOfTracks >= 10;
    PlaylistId                        Name  NumberOfTracks  TotalDuration
0            1                       Music            3290      877683083
1            3                    TV Shows             213      501094957
2            5                  90’s Music            1477      398705153
3            8                       Music            3290      877683083
4           10                    TV Shows             213      501094957
5           11             Brazilian Music              39        9486559
6           12                   Classical              75       21770592
7           13   Classical 101 - Deep Cuts              25        6755730
```

```
8           14  Classical 101 - Next Steps              25        7575051
9           15  Classical 101 - The Basics              25        7439811
10          16                      Grunge              15        4122018
11          17         Heavy Metal Classic              26        8206312
Using model gpt-4 for 270.0 tokens (approx)
```

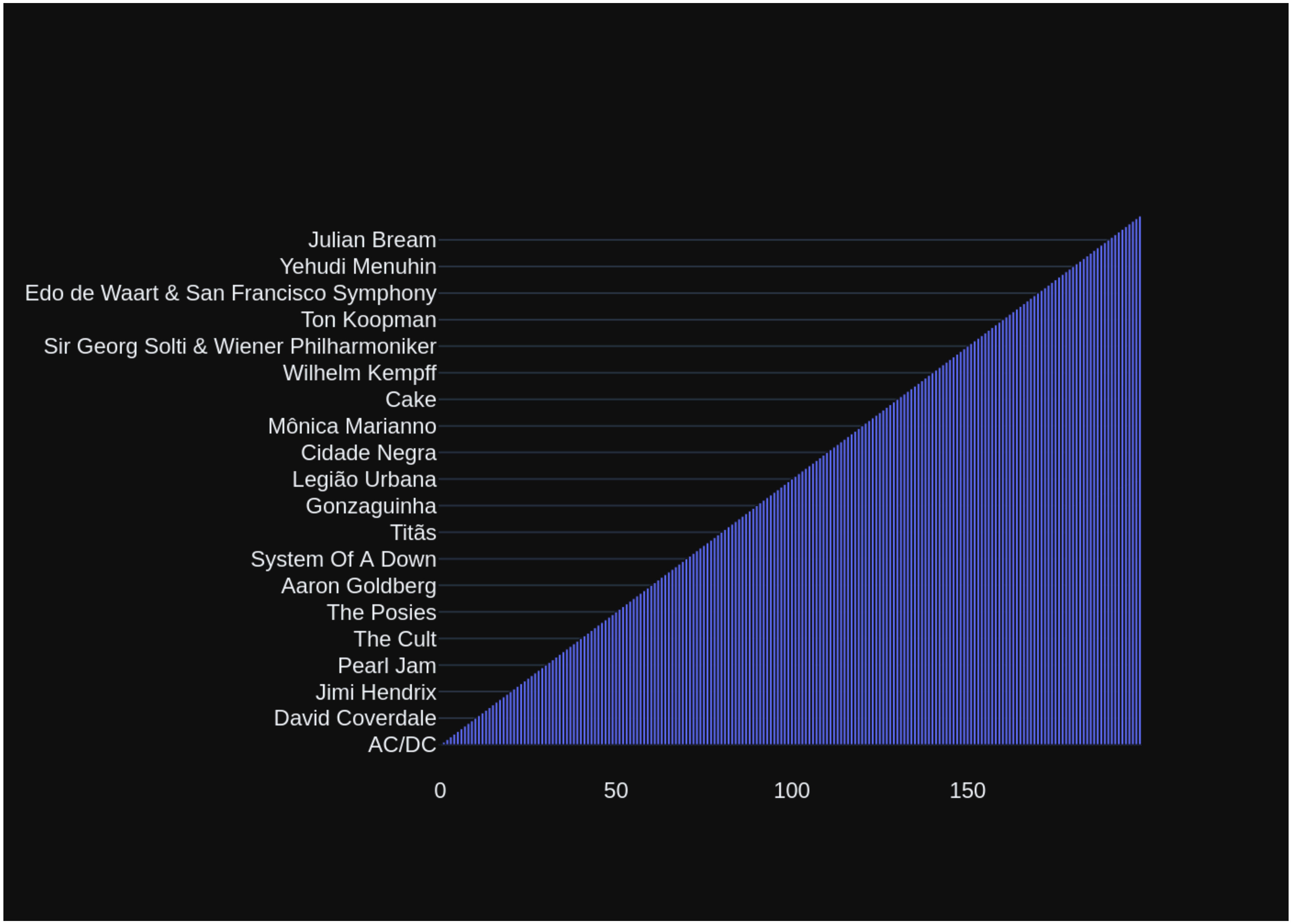


```
Out[40]: ('SELECT Playlist.PlaylistId, Playlist.Name, COUNT(PlaylistTrack.TrackId) as NumberOfTracks, SUM(Track.Mil
         liseconds) as TotalDuration\nFROM Playlist\nJOIN PlaylistTrack ON Playlist.PlaylistId = PlaylistTrack.Play
         listId\nJOIN Track ON PlaylistTrack.TrackId = Track.TrackId\nGROUP BY Playlist.PlaylistId\nHAVING NumberOf
         Tracks >= 10;',
              PlaylistId                        Name  NumberOfTracks  TotalDuration
          0            1                       Music            3290      877683083
          1            3                    TV Shows             213      501094957
          2            5                  90’s Music            1477      398705153
          3            8                       Music            3290      877683083
          4           10                    TV Shows             213      501094957
          5           11             Brazilian Music              39        9486559
          6           12                   Classical              75       21770592
          7           13   Classical 101 - Deep Cuts              25        6755730
          8           14  Classical 101 - Next Steps              25        7575051
          9           15  Classical 101 - The Basics              25        7439811
          10          16                      Grunge              15        4122018
          11          17         Heavy Metal Classic              26        8206312,
          Figure({
              'data': [{'alignmentgroup': 'True',
                        'hovertemplate': ('Name=%{x}<br>Number of Tracks=' ... '%{marker.color}<extra></extra>'),
                        'legendgroup': '',
                        'marker': {'color': array([877683083, 501094957, 398705153, 877683083, 501094957,   948655
         9,
                                                    21770592,   6755730,   7575051,   7439811,   4122018,   820631
         2]),
                                   'coloraxis': 'coloraxis',
                                   'pattern': {'shape': ''}},
                        'name': '',
                        'offsetgroup': '',
                        'orientation': 'v',
                        'showlegend': False,
                        'textposition': 'auto',
                        'type': 'bar',
                        'x': array(['Music', 'TV Shows', '90’s Music', 'Music', 'TV Shows',
                                    'Brazilian Music', 'Classical', 'Classical 101 - Deep Cuts',
                                    'Classical 101 - Next Steps', 'Classical 101 - The Basics', 'Grunge',
                                    'Heavy Metal Classic'], dtype=object),
                        'xaxis': 'x',
                        'y': array([3290,  213, 1477, 3290,  213,   39,   75,   25,   25,   25,   15,   26]),
                        'yaxis': 'y'}],
              'layout': {'barmode': 'relative',
                         'coloraxis': {'colorbar': {'title': {'text': 'Total Duration'}},
                                       '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},
                         'template': '...',
                         'title': {'text': 'Playlists with at least 10 tracks and their total duration'},
                         'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text': 'Name'}},
                         'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text': 'Number of Tracks'}}}
          }))
```

In [41]:
```python
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 fo
rmat instructions. \n===Tables \nCREATE INDEX IFK_AlbumArtistId ON Album (ArtistId)\n\nCREATE TABLE Track\n
(\n    TrackId INTEGER  NOT NULL,\n    Name NVARCHAR(200)  NOT NULL,\n    AlbumId INTEGER,\n    MediaTypeId
INTEGER  NOT NULL,\n    GenreId INTEGER,\n    Composer NVARCHAR(220),\n    Milliseconds INTEGER  NOT NUL
L,\n    Bytes INTEGER,\n    UnitPrice NUMERIC(10,2)  NOT NULL,\n    CONSTRAINT PK_Track PRIMARY KEY  (Track
Id),\n    FOREIGN KEY (AlbumId) REFERENCES Album (AlbumId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n
FOREIGN KEY (GenreId) REFERENCES Genre (GenreId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREIG
N KEY (MediaTypeId) REFERENCES MediaType (MediaTypeId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n
CREATE INDEX IFK_TrackGenreId ON Track (GenreId)\n\nCREATE INDEX IFK_TrackAlbumId ON Track (AlbumId)\n\nCRE
ATE TABLE Album\n(\n    AlbumId INTEGER  NOT NULL,\n    Title NVARCHAR(160)  NOT NULL,\n    ArtistId INTEGE
R  NOT NULL,\n    CONSTRAINT PK_Album PRIMARY KEY  (AlbumId),\n    FOREIGN KEY (ArtistId) REFERENCES Artist
(ArtistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\nCREATE INDEX IFK_TrackMediaTypeId ON Track
(MediaTypeId)\n\nCREATE INDEX IFK_PlaylistTrackTrackId ON PlaylistTrack (TrackId)\n\nCREATE TABLE Artist\n
(\n    ArtistId INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK_Artist PRIMARY KEY  (ArtistI
d)\n)\n\nCREATE TABLE Genre\n(\n    GenreId INTEGER  NOT NULL,\n    Name NVARCHAR(120),\n    CONSTRAINT PK_
Genre PRIMARY KEY  (GenreId)\n)\n\nCREATE TABLE PlaylistTrack\n(\n    PlaylistId INTEGER  NOT NULL,\n    Tr
ackId INTEGER  NOT NULL,\n    CONSTRAINT PK_PlaylistTrack PRIMARY KEY  (PlaylistId, TrackId),\n    FOREIGN
KEY (PlaylistId) REFERENCES Playlist (PlaylistId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION,\n    FOREI
GN KEY (TrackId) REFERENCES Track (TrackId) \n\t\tON DELETE NO ACTION ON UPDATE NO ACTION\n)\n\n\n===Additi
onal Context \n\nIn the chinook 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    Get the top 10 most popular artists (based on the number of tracks):\n'}, {'role': 'assistant', 'cont
ent': 'SELECT Artist.Name, COUNT(Track.TrackId) as NumberOfTracks\nFROM Artist\nJOIN Album ON Artist.Artist
Id = Album.ArtistId\nJOIN Track ON Album.AlbumId = Track.AlbumId\nGROUP BY Artist.ArtistId\nORDER BY Number
OfTracks DESC\nLIMIT 10;'}, {'role': 'user', 'content': '  \n    List all albums and their corresponding ar
tist names  \n'}, {'role': 'assistant', 'content': 'SELECT Album.Title, Artist.Name\nFROM Album\nJOIN Artis
t ON Album.ArtistId = Artist.ArtistId;'}, {'role': 'user', 'content': '  \n    List all genres and the numb
er of tracks in each genre:\n'}, {'role': 'assistant', 'content': 'SELECT Genre.Name, COUNT(Track.TrackId)
as NumberOfTracks\nFROM Genre\nLEFT JOIN Track ON Genre.GenreId = Track.GenreId\nGROUP BY Genre.GenreId;'},
{'role': 'user', 'content': '  \n     Get all playlists containing at least 10 tracks and the total duratio
n of those tracks:\n'}, {'role': 'assistant', 'content': 'SELECT Playlist.PlaylistId, Playlist.Name, COUNT
(PlaylistTrack.TrackId) as NumberOfTracks, SUM(Track.Milliseconds) as TotalDuration\nFROM Playlist\nJOIN Pl
aylistTrack ON Playlist.PlaylistId = PlaylistTrack.PlaylistId\nJOIN Track ON PlaylistTrack.TrackId = Track.
TrackId\nGROUP BY Playlist.PlaylistId\nHAVING NumberOfTracks >= 10;'}, {'role': 'user', 'content': '  \n
Find the top 5 customer who bought the most albums in total quantity (across all invoices):\n'}, {'role':
'assistant', 'content': 'SELECT Invoice.CustomerId, SUM(InvoiceLine.Quantity) as TotalQuantity\nFROM Invoic
e\nJOIN InvoiceLine ON Invoice.InvoiceId = InvoiceLine.InvoiceId\nGROUP BY Invoice.CustomerId\nORDER BY Tot
alQuantity DESC\nLIMIT 5;'}, {'role': 'user', 'content': '  \n     Find the customer who bought the most al
bums in total quantity (across all invoices): \n'}, {'role': 'assistant', 'content': 'SELECT Invoice.Custom
erId, SUM(InvoiceLine.Quantity) as TotalQuantity\nFROM Invoice\nJOIN InvoiceLine ON Invoice.InvoiceId = Inv
oiceLine.InvoiceId\nGROUP BY Invoice.CustomerId\nORDER BY TotalQuantity DESC\nLIMIT 1;'}, {'role': 'user',
'content': '  \n    Find the top 5 most expensive tracks (based on unit price):\n'}, {'role': 'assistant',
'content': 'SELECT * FROM Track\nORDER BY UnitPrice DESC\nLIMIT 5;'}, {'role': 'user', 'content': '  \n
Find all tracks with a name containing "What" (case-insensitive)\n'}, {'role': 'assistant', 'content': "SEL
ECT * FROM Track\nWHERE Name LIKE '%What%' COLLATE NOCASE;"}, {'role': 'user', 'content': 'Show me a list o
f tables in the SQLite database'}, {'role': 'assistant', 'content': "SELECT name FROM sqlite_master WHERE t
ype='table';"}, {'role': 'user', 'content': 'what are the top 5 countries that customers come from?'}, {'ro
le': 'assistant', 'content': 'SELECT Country, COUNT(*) as CustomerCount\nFROM Customer\nGROUP BY Country\nO
RDER BY CustomerCount DESC\nLIMIT 5;'}, {'role': 'user', 'content': '  \n     Identify artists who have alb
ums with tracks appearing in multiple genres:\n'}]
Using model gpt-4 for 1255.5 tokens (approx)
SELECT DISTINCT Artist.Name
FROM Artist
INNER JOIN Album ON Artist.ArtistId = Album.ArtistId
INNER JOIN Track ON Album.AlbumId = Track.AlbumId
WHERE Track.GenreId IN (
      SELECT GenreId
      FROM Track
      GROUP BY GenreId
      HAVING COUNT(DISTINCT AlbumId) > 1
);
SELECT DISTINCT Artist.Name
FROM Artist
INNER JOIN Album ON Artist.ArtistId = Album.ArtistId
INNER JOIN Track ON Album.AlbumId = Track.AlbumId
WHERE Track.GenreId IN (
      SELECT GenreId
      FROM Track
      GROUP BY GenreId
      HAVING COUNT(DISTINCT AlbumId) > 1
);
SELECT DISTINCT Artist.Name
FROM Artist
INNER JOIN Album ON Artist.ArtistId = Album.ArtistId
INNER JOIN Track ON Album.AlbumId = Track.AlbumId
WHERE Track.GenreId IN (
      SELECT GenreId
      FROM Track
      GROUP BY GenreId
      HAVING COUNT(DISTINCT AlbumId) > 1
);
                                                     Name
0                                                   AC/DC
1                                                  Accept
2                                               Aerosmith
3                                       Alanis Morissette
4                                         Alice In Chains
..                                                    ...
195                                          Gerald Moore
196   Mela Tenenbaum, Pro Musica Prague & Richard Kapp
197                                 Emerson String Quartet
198  C. Monteverdi, Nigel Rogers - Chiaroscuro; Lon...
199                                         Nash Ensemble

[200 rows x 1 columns]
Using model gpt-4 for 232.5 tokens (approx)
```


Julian Bream
Yehudi Menuhin
Edo de Waart & San Francisco Symphony
Ton Koopman
Sir Georg Solti & Wiener Philharmoniker
Wilhelm Kempff
Cake
Mônica Marianno
Cidade Negra
Legião Urbana
Gonzaguinha
Titãs
System Of A Down
Aaron Goldberg
The Posies
The Cult
Pearl Jam
Jimi Hendrix
David Coverdale
AC/DC
0
50
100
150

```
Out[41]: ('SELECT DISTINCT Artist.Name\nFROM Artist\nINNER JOIN Album ON Artist.ArtistId = Album.ArtistId\nINNER JO
         IN Track ON Album.AlbumId = Track.AlbumId\nWHERE Track.GenreId IN (\n      SELECT GenreId\n      FROM Trac
         k\n      GROUP BY GenreId\n      HAVING COUNT(DISTINCT AlbumId) > 1\n);',
                                                         Name
          0                                            AC/DC
          1                                           Accept
          2                                        Aerosmith
          3                                Alanis Morissette
          4                                  Alice In Chains
          ..                                             ...
          195                                   Gerald Moore
          196   Mela Tenenbaum, Pro Musica Prague & Richard Kapp
          197                          Emerson String Quartet
          198  C. Monteverdi, Nigel Rogers - Chiaroscuro; Lon...
          199                                  Nash Ensemble

          [200 rows x 1 columns],
          Figure({
              'data': [{'type': 'bar',
                        'x': array([  0,   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,  60,  61,  62,  63,  64,  65,  66,  67,  68,  69,
                                     70,  71,  72,  73,  74,  75,  76,  77,  78,  79,  80,  81,  82,  83,
                                     84,  85,  86,  87,  88,  89,  90,  91,  92,  93,  94,  95,  96,  97,
                                     98,  99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111,
                                    112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125,
                                    126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139,
                                    140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153,
                                    154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167,
                                    168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181,
                                    182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195,
                                    196, 197, 198, 199]),
                        'y': array(['AC/DC', 'Accept', 'Aerosmith', 'Alanis Morissette', 'Alice In Chains',
                                    'Audioslave', 'Led Zeppelin', 'Frank Zappa & Captain Beefheart',
                                    'Queen', 'Kiss', 'David Coverdale', 'Deep Purple', 'Santana',
                                    'Creedence Clearwater Revival', 'Def Leppard', 'Faith No More',
                                    'Foo Fighters', "Guns N' Roses", 'Iron Maiden', 'Jamiroquai',
                                    'Jimi Hendrix', 'Joe Satriani', 'Lenny Kravitz', 'Marillion',
                                    'Men At Work', 'Nirvana', 'O Terço', 'Ozzy Osbourne', 'Page & Plant',
                                    "Paul D'Ianno", 'Pearl Jam', 'Pink Floyd', 'R.E.M.', 'Raul Seixas',
                                    'Red Hot Chili Peppers', 'Rush', 'Skank', 'Soundgarden',
                                    'Stone Temple Pilots', 'Terry Bozzio, Tony Levin & Steve Stevens',
                                    'The Cult', 'The Doors', 'The Police', 'The Rolling Stones', 'The Who',
                                    'U2', 'Van Halen', 'Velvet Revolver', 'Dread Zeppelin', 'Scorpions',
                                    'The Posies', 'Antônio Carlos Jobim', 'Billy Cobham', 'Spyro Gyra',
                                    'Miles Davis', 'Gene Krupa', 'Dennis Chambers', 'Gilberto Gil',
                                    'Incognito', 'Aisha Duo', 'Aaron Goldberg', 'Apocalyptica',
                                    'Black Label Society', 'Black Sabbath', 'Bruce Dickinson', 'Metallica',
                                    'Godsmack', 'Judas Priest', 'Motörhead', 'Mötley Crüe',
                                    'System Of A Down', 'Body Count', 'Green Day', 'Os Mutantes', 'JET',
                                    'R.E.M. Feat. Kate Pearson', 'Raimundos', 'Smashing Pumpkins',
                                    'The Clash', 'The Tea Party', 'Titãs', 'Buddy Guy', 'Eric Clapton',
                                    'Stevie Ray Vaughan & Double Trouble', 'The Black Crowes',
                                    'Caetano Veloso', 'Chico Buarque', 'Chico Science & Nação Zumbi',
                                    'Cláudio Zoli', 'Marcos Valle', 'Gonzaguinha', 'Various Artists',
                                    'Ed Motta', 'Cássia Eller', 'Djavan', 'Elis Regina', 'Falamansa',
                                    'Funk Como Le Gusta', 'Jorge Ben', 'Jota Quest', 'Legião Urbana',
                                    'Lulu Santos', 'Marisa Monte', 'Milton Nascimento', 'Olodum',
                                    'Os Paralamas Do Sucesso', 'Tim Maia', 'Vinícius De Moraes',
                                    'Zeca Pagodinho', 'Luciana Souza/Romero Lubambo', 'Cidade Negra',
                                    'UB40', 'Amy Winehouse', 'Passengers', 'Philip Glass Ensemble',
                                    'James Brown', 'Marvin Gaye', 'O Rappa', 'Karsh Kale', 'João Suplicy',
                                    'Mônica Marianno', 'Habib Koité and Bamada', 'Planet Hemp',
                                    'House Of Pain', 'Battlestar Galactica', 'Heroes', 'Lost', 'The Office',
                                    'Aquaman', 'Battlestar Galactica (Classic)', 'Cake',
                                    'Temple of the Dog', 'Chris Cornell', 'Calexico',
                                    'Nicolaus Esterhazy Sinfonia', 'Alberto Turco & Nova Schola Gregoriana',
                                    'Richard Marlow & The Choir of Trinity College, Cambridge',
                                    'English Concert & Trevor Pinnock',
                                    'Anne-Sophie Mutter, Herbert Von Karajan & Wiener Philharmoniker',
                                    'Hilary Hahn, Jeffrey Kahane, Los Angeles Chamber Orchestra & Margaret Batjer',
                                    'Wilhelm Kempff', 'Yo-Yo Ma', 'Scholars Baroque Ensemble',
                                    'Academy of St. Martin in the Fields & Sir Neville Marriner',
                                    'Academy of St. Martin in the Fields Chamber Ensemble & Sir Neville Marriner',
                                    'Berliner Philharmoniker, Claudio Abbado & Sabine Meyer',
                                    'Royal Philharmonic Orchestra & Sir Thomas Beecham',
                                    'Orchestre Révolutionnaire et Romantique & John Eliot Gardiner',
                                    'Britten Sinfonia, Ivor Bolton & Lesley Garrett',
                                    'Chicago Symphony Chorus, Chicago Symphony Orchestra & Sir Georg Solti',
                                    'Sir Georg Solti & Wiener Philharmoniker',
                                    'Academy of St. Martin in the Fields, John Birch, Sir Neville Marriner & Sylvia
         McNair',
                                           'London Symphony Orchestra & Sir Charles Mackerras',
                                           'Barry Wordsworth & BBC Concert Orchestra',
                                           'Herbert Von Karajan, Mirella Freni & Wiener Philharmoniker',
                                           'Eugene Ormandy', 'Luciano Pavarotti',
                                           'Leonard Bernstein & New York Philharmonic',
                                           'Boston Symphony Orchestra & Seiji Ozawa',
                                           'Aaron Copland & London Symphony Orchestra', 'Ton Koopman',
                                           'Sergei Prokofiev & Yuri Temirkanov',
                                           'Chicago Symphony Orchestra & Fritz Reiner',
                                           'Orchestra of The Age of Enlightenment',
                                           'Emanuel Ax, Eugene Ormandy & Philadelphia Orchestra', 'James Levine',
                                           'Berliner Philharmoniker & Hans Rosbaud', 'Maurizio Pollini',
                                           'Gustav Mahler',
                                           'Felix Schmidt, London Symphony Orchestra & Rafael Frühbeck de Burgos',
                                           'Edo de Waart & San Francisco Symphony',
                                           'Antal Doráti & London Symphony Orchestra',
                                           'Choir Of Westminster Abbey & Simon Preston',
                                           'Michael Tilson Thomas & San Francisco Symphony',
                                           'Chor der Wiener Staatsoper, Herbert Von Karajan & Wiener Philharmoniker',
                                           "The King's Singers", 'Berliner Philharmoniker & Herbert Von Karajan',
                                           "Christopher O'Riley", 'Fretwork',
                                           'Otto Klemperer & Philharmonia Orchestra', 'Yehudi Menuhin',
                                           'Philharmonia Orchestra & Sir Neville Marriner',
                                           'Academy of St. Martin in the Fields, Sir Neville Marriner & Thurston Dart',
                                           'Les Arts Florissants & William Christie',
                                           'The 12 Cellists of The Berlin Philharmonic',
                                           'Adrian Leaper & Doreen de Feis',
                                           'Roger Norrington, London Classical Players',
                                           "Charles Dutoit & L'Orchestre Symphonique de Montréal",
                                           'Equale Brass Ensemble, John Eliot Gardiner & Munich Monteverdi Orchestra and C
hoir',
                                           "Kent Nagano and Orchestre de l'Opéra de Lyon", 'Julian Bream',
                                           'Martin Roscoe', 'Göteborgs Symfoniker & Neeme Järvi', 'Itzhak Perlman',
                                           'Michele Campanella', 'Gerald Moore',
                                           'Mela Tenenbaum, Pro Musica Prague & Richard Kapp',
                                           'Emerson String Quartet',
                                           'C. Monteverdi, Nigel Rogers - Chiaroscuro; London Baroque; London Cornett & Sa
ckbu',
                                           'Nash Ensemble'], dtype=object)}],
     'layout': {'template': '...'}
 }))
```

Check completion time

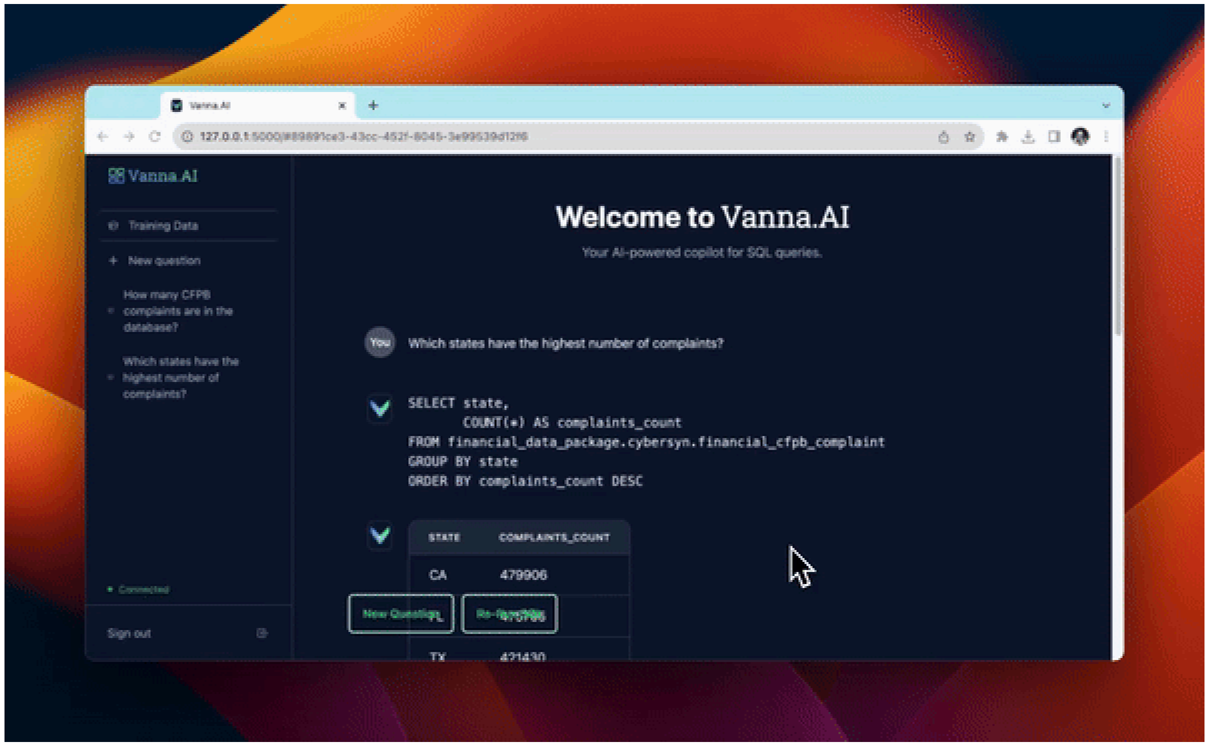
In [42]:
```python
ts_stop = time()

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

```
test running on 'papa-game' with 'gpt-4' LLM took : 167.69 sec
```

In [43]:
```python
from datetime import datetime
print(datetime.now())
```

```
2024-06-20 20:20:58.374285
```

Launch the User Interface



Welcome to Vanna.AI

Your AI-powered copilot for SQL queries.

How many CFPB complaints are in the database?

Which states have the highest number of complaints?

You Which states have the highest number of complaints?

```sql
SELECT state,
       COUNT(*) AS complaints_count
FROM financial_data_package.cybersyn.financial_cfpb_complaint
GROUP BY state
ORDER BY complaints_count DESC
```

STATE	COMPLAINTS_COUNT
CA	479906
FL	[illegible]
TX	421430

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