Just IT

B2W group

Apprenticeships | Training | Recruitment

Data Technician

Name:

Course Date:

Table of contents

Day 1: Task 1 3
Day 1: Task 2 4
Day 3: Task 1 5
Day 4: Task 1: Written 7
Day 4: Task 2: SQL Practical 13
Course Notes 37
Additional Information ..... **Error! Bookmark not defined.**

Day 1: Task 1

Please research and complete the below questions relating to key concepts of databases.

What is a primary key?	A primary key is a unique identifier for each record in a database table. It ensures that each record is distinct and cannot be duplicated.
How does this differ from a secondary key?	A secondary key (or alternate key) is any candidate key that is not selected as the primary key. It can still be used to access data but does not enforce uniqueness in the same way.
How are primary and foreign keys related?	A foreign key in one table is a reference to the primary key in another table. This establishes a relationship between the two tables, ensuring referential integrity.
Provide a real-world example of a one-to-one relationship	Each person has one passport, and each passport belongs to one person. The *Person* table and *Passport* table have a one-to-one relationship.
Provide a real-world example of a one-to-many relationship	A customer can place multiple orders, but each order is placed by only one customer. The *Customer* table is related to the *Orders* table in a one-to-many relationship.
Provide a real-world example of a many-to-many relationship	Students can enroll in many courses, and each course can have many students. This is typically modelled with a join table like *Enrolments* linking *Students* and *Courses*.

Day 1: Task 2

Please research and complete the below questions relating to key concepts of databases.

What is the difference between a relational and non-relational database?	A **relational database** stores data in tables with rows and columns, like a spreadsheet. It's great for structured data with clear relationships—think customer orders or employee records- like a customer list where each row has a name, email, and phone number. Example: **MySQL** is a relational database used by many companies for managing transactions or user accounts. A **non-relational database**, on the other hand, stores data in more flexible formats like documents, key-value pairs, graphs, or wide-columns. It's better for handling unstructured or changing data (that doesn't fit neatly into tables), like user profiles, chat messages, or product catalogues. Example: **MongoDB** is a non-relational (NoSQL) database used for things like user profiles on a social media site, where every user might have different sets of information.
What type of data would benefit off the non-relational model? **Why?**	Data that is unstructured, semi-structured, or constantly changing—like social media posts, multimedia files, sensor data, or user-generated content. Example: A product catalogue on an e-commerce site where each product has different attributes—like shoes with sizes, electronics with specs, or books with authors. Because non-relational databases are designed to be flexible, scalable, and can handle large volumes of data without needing a strict structure. They make it easier to adapt to changes in data format and are well-suited for big data and real-time applications. For Example, **a messaging app** like WhatsApp would benefit from a non-relational database to store messages, images, and user settings—all of which vary a lot.

Day 3: Task 1

Please research the below 'JOIN' types, explain what they are and provide an example of the types of data it would be used on.

Self-join	A **self-join** is a regular join but the table is joined with itself. **Use Case**: Used when rows in the same table need to be compared with each other. **Example**: In an `Employees` table, each employee has a `ManagerID` that references another employee in the same table.
Right join	Returns all rows from the **right** table and matched rows from the left table. Unmatched rows from the left table will have NULLs. **Use Case**: Useful when you want all records from the second table regardless of whether there's a match. **Example**: You want to list all `Products`, even if they don't have a `Sale`.
Full join	Returns all rows when there is a match in either left or right table. NULLs are filled in where there is no match. **Use Case**: Used when you want all records from both tables and to see what doesn't match. **Example**: Combining two client lists: one from online sales and one from in-store sales.

Inner join	Returns only the rows that have matching values in both tables. **Use Case**: Used to combine data where you only want results that exist in both tables. **Example**: Joining `Orders` and `Customers` to get only customers who placed orders.
Cross join	Also known as Cartesian Join- Returns the **Cartesian product** of both tables (every row of table A with every row of table B). **Use Case**: Used to generate combinations, like pairing every shirt size with every color. **Example**: Generate combinations of `ShirtSizes` and `Colors`.
Left join	Returns all rows from the **left** table and matched rows from the right table. Non-matching rows from the right will be NULL. **Use Case**: Used when you want to keep all records from the primary table, even if there's no match. **Example**: List all employees and their assigned projects (if any).

Day 4: Task 1: Written

In your groups, discuss and complete the below activity. You can either nominate one writer or split the elements between you. Everyone however must have the completed work below:

Imagine you have been hired by a small retail business that wants to streamline its operations by creating a new database system. This database will be used to manage inventory, sales, and customer information. The business is a small corner shop that sells a range of groceries and domestic products. It might help to picture your local convenience store and think of what they sell. They also have a loyalty program, which you will need to consider when deciding what tables to create.

Write a 500-word essay explaining the steps you would take to set up and create this database. Your essay should cover the following points:

1. ***Understanding the Business Requirements**:*
    a. *What kind of data will the database need to store?*
    b. *Who will be the users of the database, and what will they need to accomplish?*
2. ***Designing the Database Schema**:*
    a. *How would you structure the database tables to efficiently store inventory, sales, and customer information?*
    b. *What relationships between tables are necessary (e.g., how sales relate to inventory and customers)?*
3. ***Implementing the Database**:*
    a. *What SQL commands would you use to create the database and its tables?*
    b. *Provide examples of SQL statements for creating tables and defining relationships between them.*
4. ***Populating the Database**:*
    a. *How would you input initial data into the database? Give examples of SQL INSERT statements.*
5. ***Maintaining the Database**:*
    a. *What measures would you take to ensure the database remains accurate and up to date?*
    b. *How would you handle backups and data security?*

Your essay should include specific examples of SQL commands and explain why each step is necessary for creating a functional and efficient database for the retail business.

Please write your 500-word essay here

Setting Up a Database for a Small Retail Business:

1. Understanding the Business Requirements

To design an effective database, we first identify the core data the shop needs to manage:

Inventory Management

The system must track product entities with attributes such as `product_name`, `category`, `quantity_in_stock`, `unit_price`, and `supplier_id`. This enables inventory control and reorder management.

- *Example*: `Milk` – 10 units in stock, £1.50 per unit.

Transactional Data

Each sales transaction should be stored with a timestamp, foreign keys referencing sold products and customers, `quantity_sold`, and `total_amount`. This supports point-of-sale (POS) operations and sales analytics.

- *Example*: On `2025-01-15`, sale of 2 units of Milk and 1 unit of Bread.

Customer Records

Customer data includes `customer_id`, `name`, `email`, `phone_number`, and `loyalty_points`. Loyalty tracking is tied to purchase activity for reward allocation.

- *Example*: `Jane Doe`, `jane.doe@example.com`, 50 loyalty points.

System Users

- **Cashiers**: Perform `INSERT` operations for sales and update inventory.
- **Managers**: Require `SELECT`, `UPDATE`, and reporting access.
- **Automated Processes**: Handle triggers for stock reduction and loyalty point updates.

This analysis ensures the database will support daily operations and customer engagement efficiently. This requirement analysis forms the foundation for a normalized and scalable database schema.

---

2. Designing the Database Schema

In a relational database, each table represents a distinct entity with defined attributes. For this retail system, the following tables are required:

Products Table- *Stores all product-related data.*

- **Columns**: `product_id` (PK), `product_name`, `category`, `price`, `stock_level`
- **Example Data**:
  - (1, 'Milk', 'Dairy', 1.50, 10)
  - (2, 'Bread', 'Bakery', 1.00, 20)

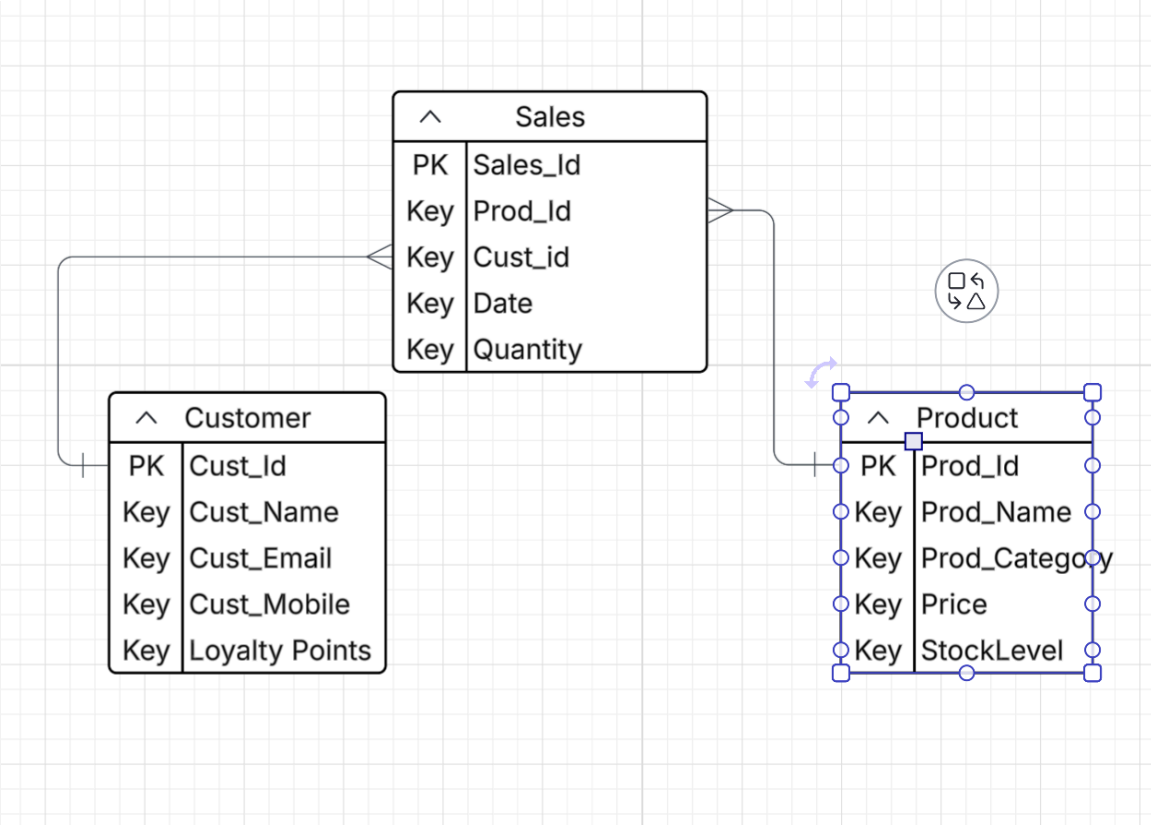
Customers Table- *Stores customer details and loyalty data.*

- **Columns**: `customer_id` (PK), `name`, `email`, `loyalty_points`
- **Example Data**:
  - (1, 'Jane Doe', 'jane.doe@example.com', 50)

Sales Table- *Records transactions made by customers.*

- **Columns**: `sale_id` (PK), `product_id` (FK), `customer_id` (FK), `sale_date`, `quantity`
- **Example Data**:
  - (1, 1, 1, '2025-01-15', 2)

LUCID CHART to show this Table Cardinality.



This schema supports efficient querying, normalization, and ensures referential integrity.

---

3. Implementing the Database

Create a New Database- In MySQL : allows you to define tables, fields, data types, and relationships between entities.

```
CREATE DATABASE RetailStore;

USE RetailStore;
```

Define Tables and Columns- Create the necessary tables with appropriate data types and constraints.

Products Table

```
CREATE TABLE Products (

    product_id INT AUTO_INCREMENT PRIMARY KEY,

    product_name VARCHAR(100),

    category VARCHAR(50),

    price DECIMAL(5,2),

    stock_level INT,
```

```
    supplier VARCHAR(100));
```

Customers Table

```
CREATE TABLE Customers (

    customer_id INT AUTO_INCREMENT PRIMARY KEY,

    name VARCHAR(100),

    email VARCHAR(100),

    loyalty_points INT DEFAULT 0);
```

Sales Table

```
CREATE TABLE Sales (

    sale_id INT AUTO_INCREMENT PRIMARY KEY,

    product_id INT,

    customer_id INT,

    sale_date DATE,

    quantity INT,

    FOREIGN KEY (product_id) REFERENCES
Products(product_id),

    FOREIGN KEY (customer_id) REFERENCES
Customers(customer_id));
```

---

4. Populating the Database

Once the database schema is created, data can be populated in two main ways:

Manual Data Entry Import from Excel ***or*** *use the RDBMS interface (such as Microsoft Access datasheets or MySQL Workbench forms) to input records directly into tables.*

```
INSERT INTO Products (product_name, category, price,
stock_level, supplier)

VALUES ('Milk', 'Dairy', 1.50, 10, 'Fresh Dairy Ltd.');
```

---

5. Maintaining the Database

a. **Accuracy and Updates**:

- Regular audits to verify stock levels.
- Scheduled updates after sales to decrease inventory and update loyalty points.
- Use of `UPDATE` and `DELETE` statements for data corrections.

b. **Backup and Security**:

- Automate daily backups using scripts or database tools.
- Restrict access using roles and permissions:

```
CREATE USER store_clerk IDENTIFIED BY 'securepass';

GRANT SELECT, INSERT, UPDATE ON Sales TO store_clerk;
```

Conclusion

By systematically analyzing requirements, designing an efficient schema, implementing it using SQL, and planning for future maintenance, a small retail business can create a powerful database system.

Day 4: Task 2: SQL Practical

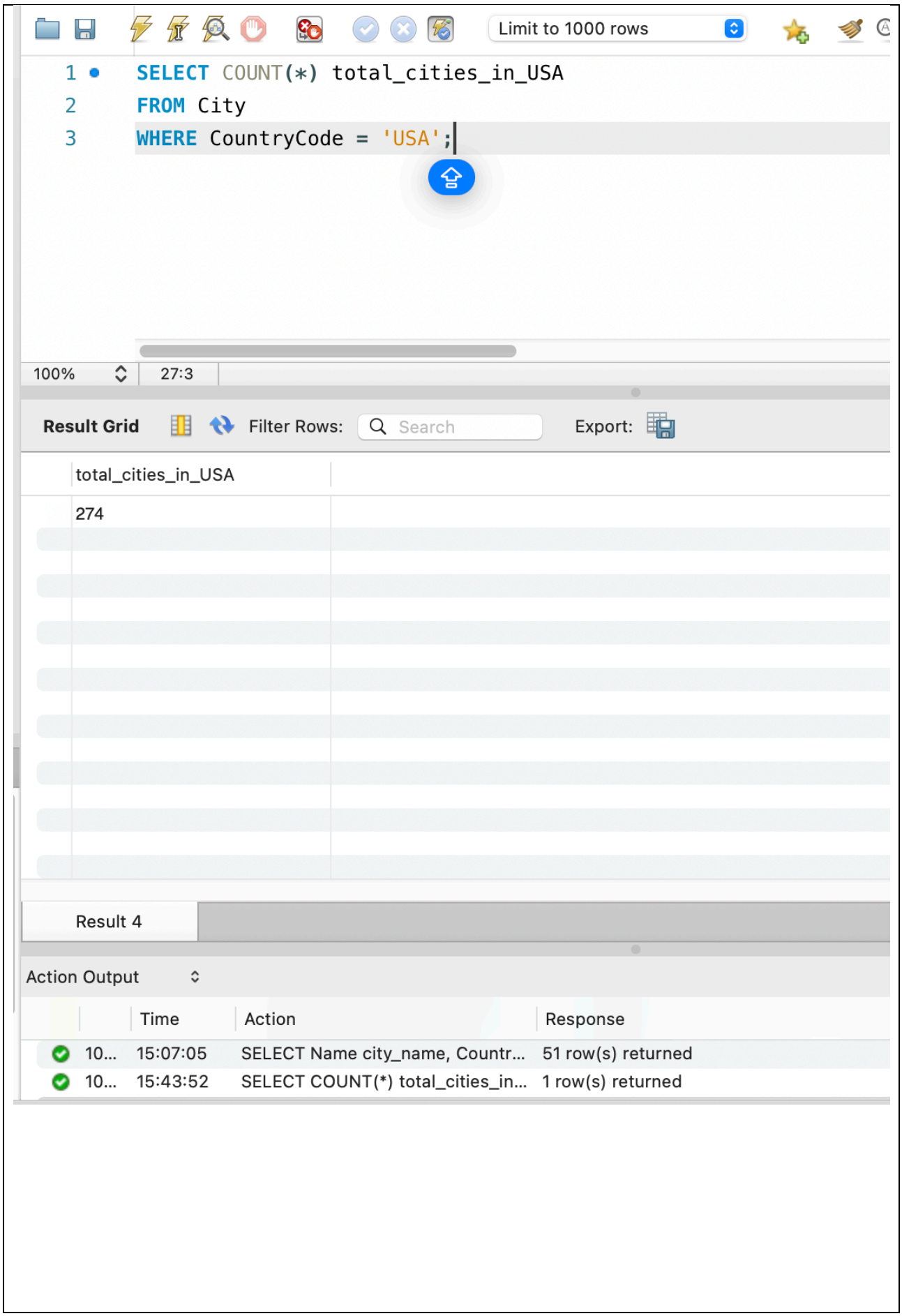
In your groups, work together to answer the below questions. It may be of benefit if one of you shares your screen with the group and as a team answer / take screen shots from there.

Setting up the database:

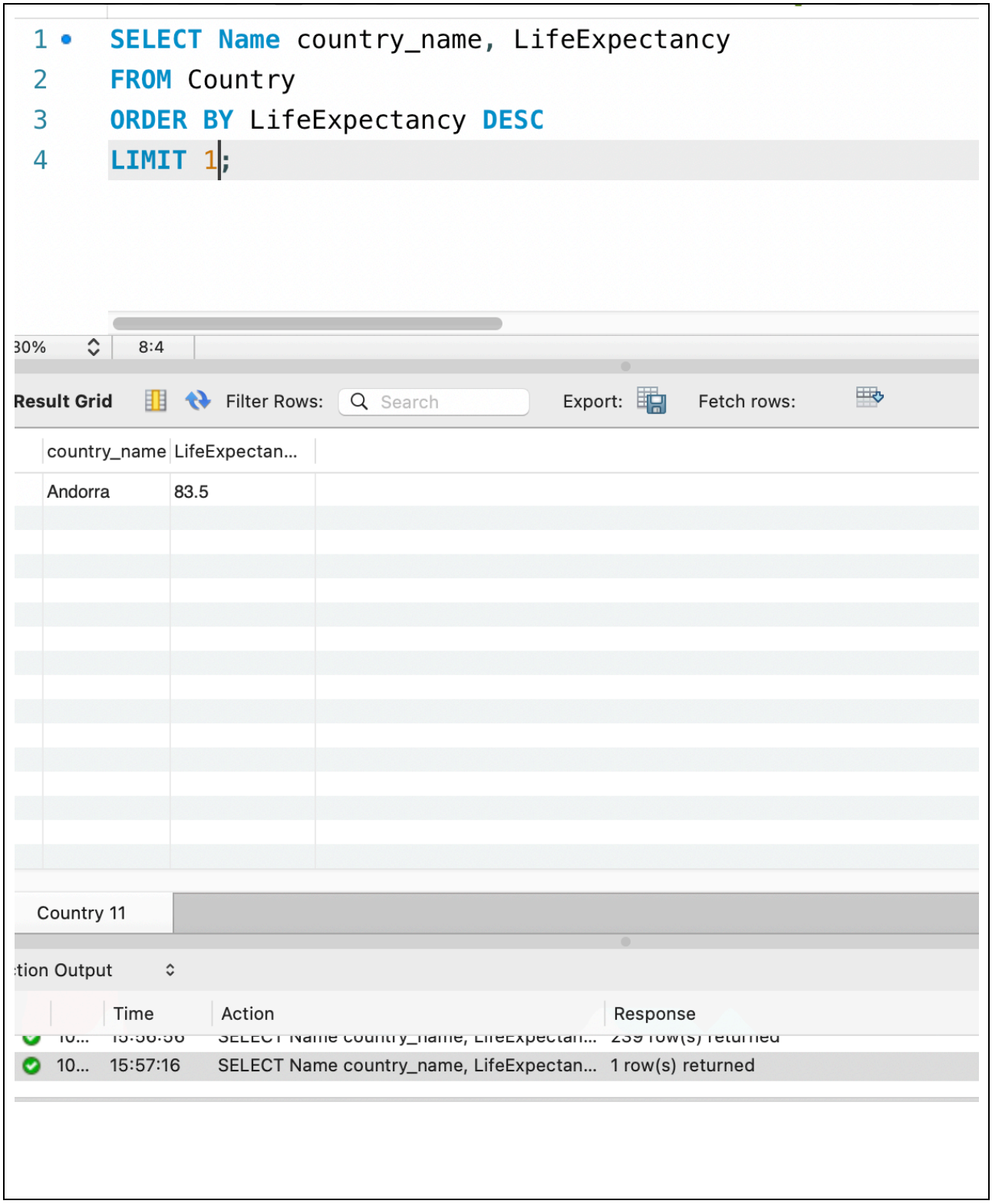
1. **Download world_db(1)**
2. **Follow each step to create your database**

For each question I would like to see both the syntax used and the output.

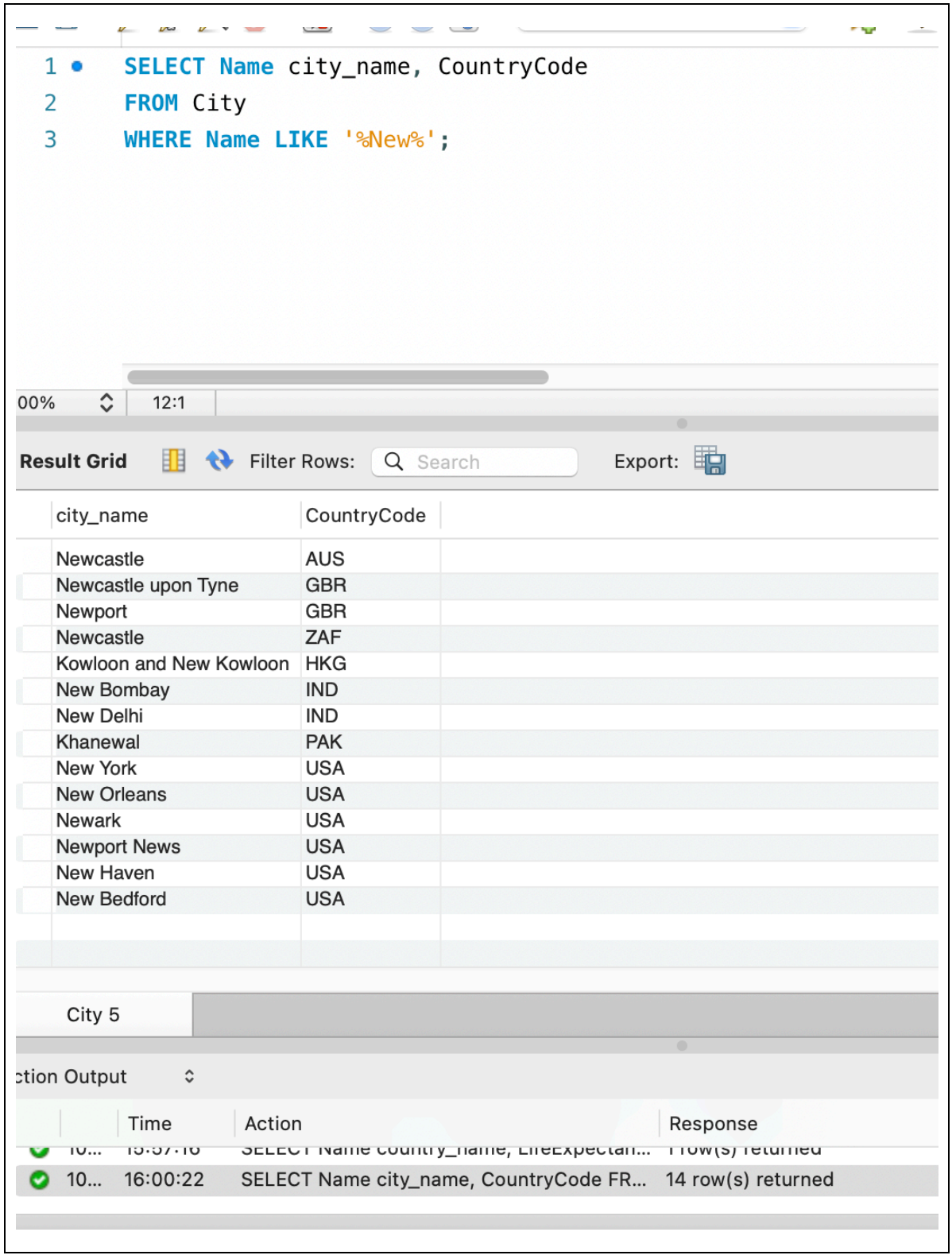
1. **Count Cities in USA:** *Scenario:* You've been tasked with conducting a demographic analysis of cities in the United States. Your first step is to determine the total number of cities within the country to provide a baseline for further analysis.


Limit to 1000 rows
1 SELECT COUNT(*) total_cities_in_USA
2 FROM City
3 WHERE CountryCode = 'USA';
100%
27:3
Result Grid
Filter Rows:
Search
Export:
total_cities_in_USA
274
Result 4
Action Output
Time
Action
Response
10... 15:07:05 SELECT Name city_name, Countr... 51 row(s) returned
10... 15:43:52 SELECT COUNT(*) total_cities_in... 1 row(s) returned

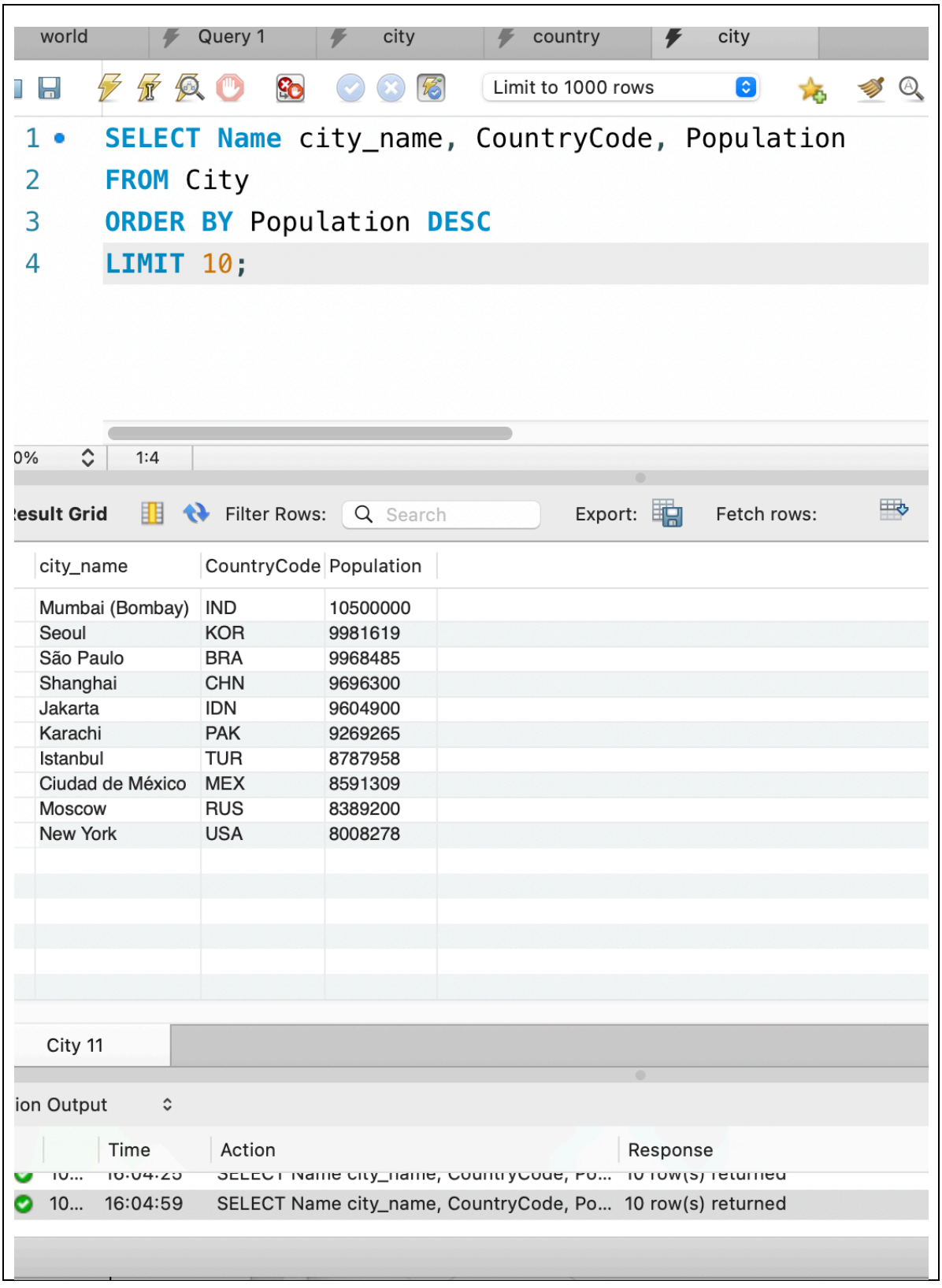
2. **Country with Highest Life Expectancy:** *Scenario:* As part of a global health initiative, you've been assigned to identify the country with the highest life expectancy. This information will be crucial for prioritising healthcare resources and interventions.



3. **"New Year Promotion: Featuring Cities with 'New :** *Scenario:* In anticipation of the upcoming New Year, your travel agency is gearing up for a special promotion featuring cities with names including the word 'New'. You're tasked with swiftly compiling a list of all cities from around the world. This curated selection will be essential in creating promotional materials and enticing travellers with exciting destinations to kick off the New Year in style.



4. **Display Columns with Limit (First 10 Rows):** *Scenario:* You're tasked with providing a brief overview of the most populous cities in the world. To keep the report concise, you're instructed to list only the first 10 cities by population from the database.



5. **Cities with Population Larger than 2,000,000:** *Scenario:* A real estate developer is interested in cities with substantial population sizes for potential investment opportunities. You're tasked with identifying cities from the database with populations exceeding 2 million to focus their research efforts.

```
1 •  SELECT Name city_name, CountryCode, Population
2    FROM City
3    WHERE Population > 2000000
4    ORDER BY Population DESC;
```

30% 26:4

Result Grid Filter Rows: Search Export:

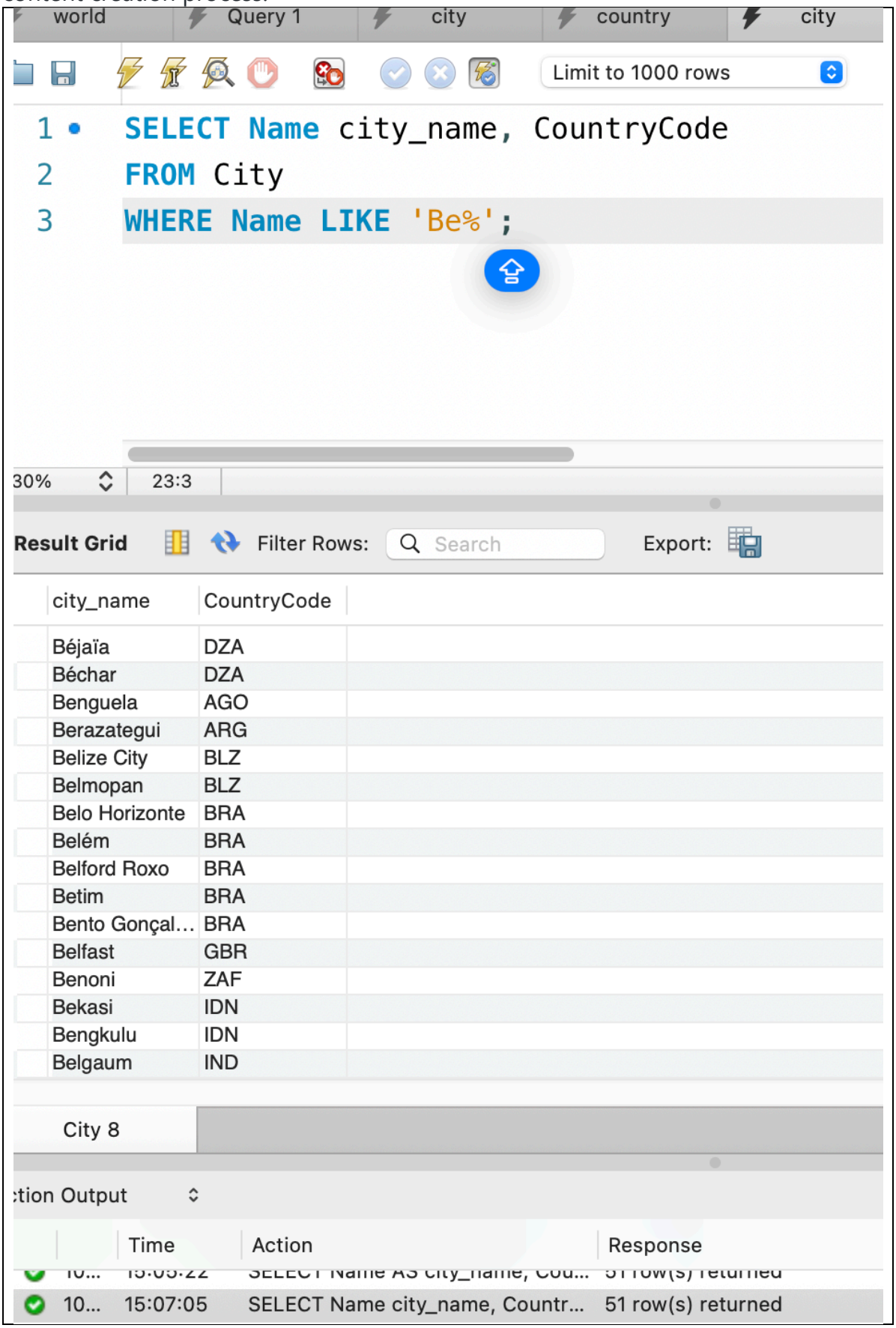
city_name	CountryCode	Population
Mumbai (Bombay)	IND	10500000
Seoul	KOR	9981619
São Paulo	BRA	9968485
Shanghai	CHN	9696300
Jakarta	IDN	9604900
Karachi	PAK	9269265
Istanbul	TUR	8787958
Ciudad de México	MEX	8591309
Moscow	RUS	8389200
New York	USA	8008278
Tokyo	JPN	7980230
Peking	CHN	7472000
London	GBR	7285000
Delhi	IND	7206704
Cairo	EGY	6789479
Teheran	IRN	6758845

City 12

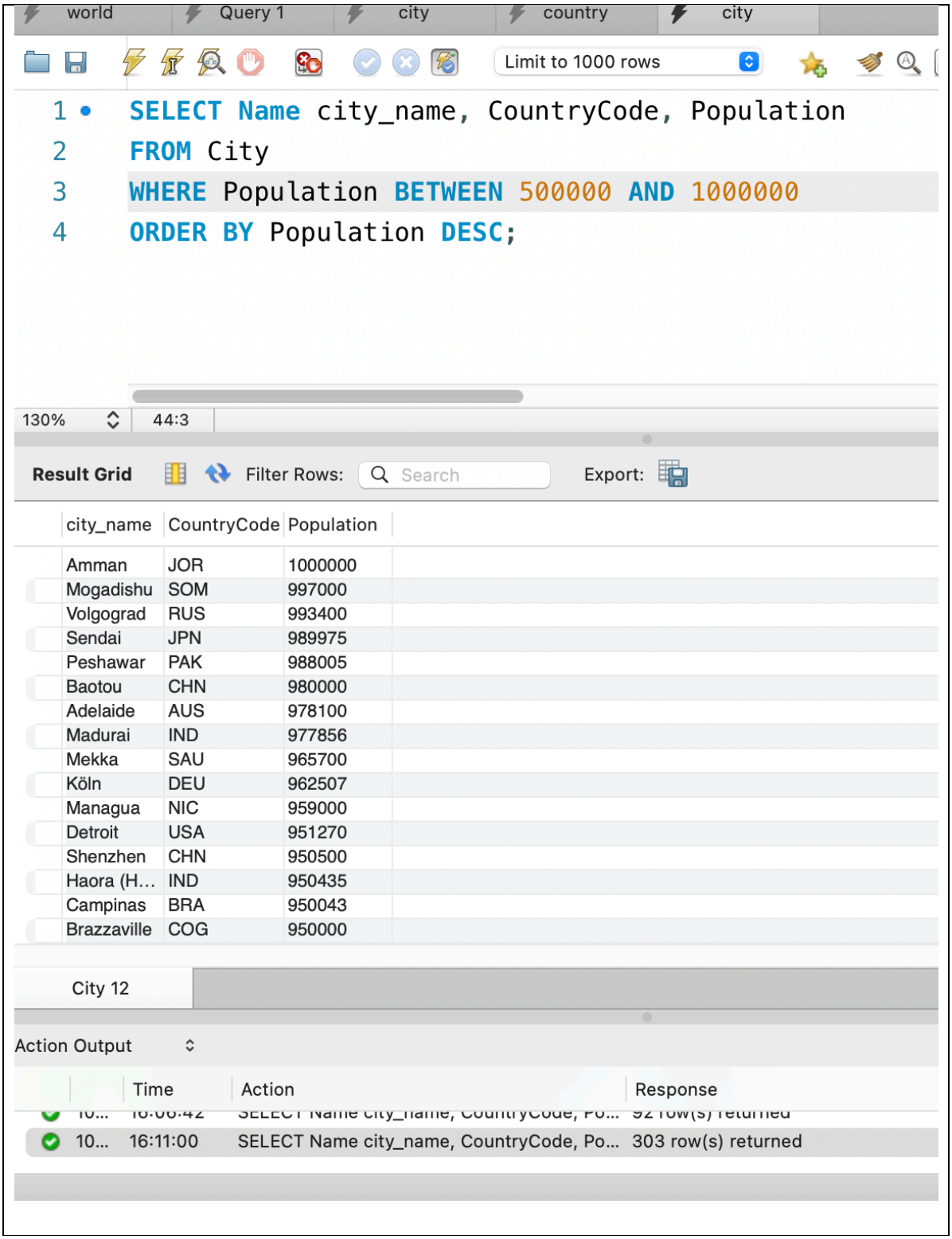
ction Output

	Time	Action	Response
10...	16:04:59	SELECT Name city_name, CountryCode, Po...	10 row(s) returned
10...	16:06:42	SELECT Name city_name, CountryCode, Po...	92 row(s) returned

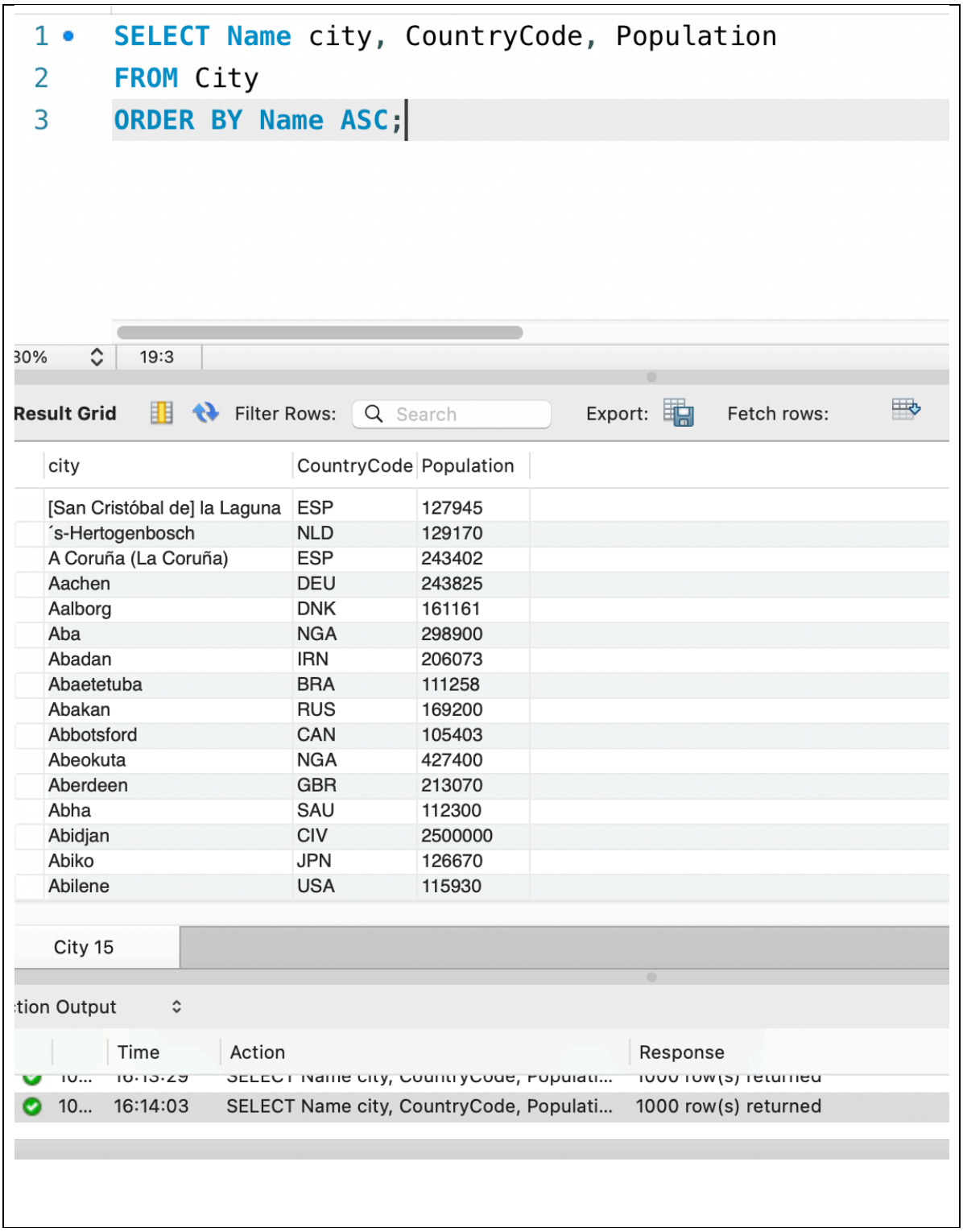
6. **Cities Beginning with 'Be' Prefix:** *Scenario:* A travel blogger is planning a series of articles featuring cities with unique names. You're tasked with compiling a list of cities from the database that start with the prefix 'Be' to assist in the blogger's content creation process.



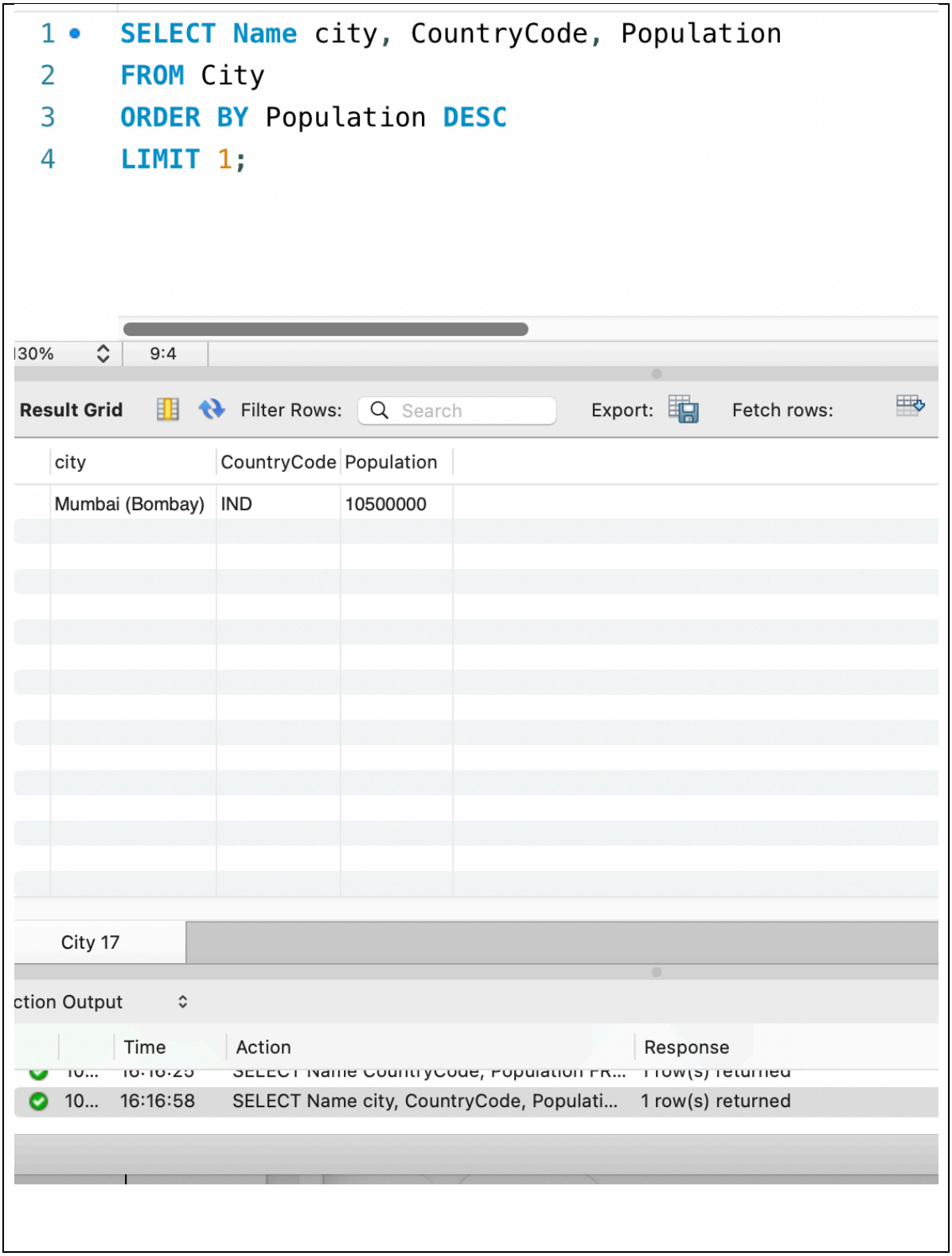
7. **Cities with Population Between 500,000-1,000,000:** *Scenario:* An urban planning committee needs to identify mid-sized cities suitable for infrastructure development projects. You're tasked with identifying cities with populations ranging between 500,000 and 1 million to inform their decision-making process.



8. **Display Cities Sorted by Name in Ascending Order:** *Scenario:* A geography teacher is preparing a lesson on alphabetical order using city names. You're tasked with providing a sorted list of cities from the database in ascending order by name to support the lesson plan.



9. **Most Populated City:** *Scenario:* A real estate investment firm is interested in cities with significant population densities for potential development projects. You're tasked with identifying the most populated city from the database to guide their investment decisions and strategic planning.



10. **City Name Frequency Analysis: Supporting Geography Education** *Scenario*: In a geography class, students are learning about the distribution of city names around the world. The teacher, in preparation for a lesson on city name frequencies, wants to provide students with a list of unique city names sorted alphabetically, along with their respective counts of occurrences in the database. You're tasked with this sorted list to support the geography teacher.

```
1 •  SELECT Name city, COUNT(*)
2    FROM City
3    GROUP BY Name
4    ORDER BY Name ASC;
```

30% 18:4

Result Grid Filter Rows: Search Export: Fetch rc

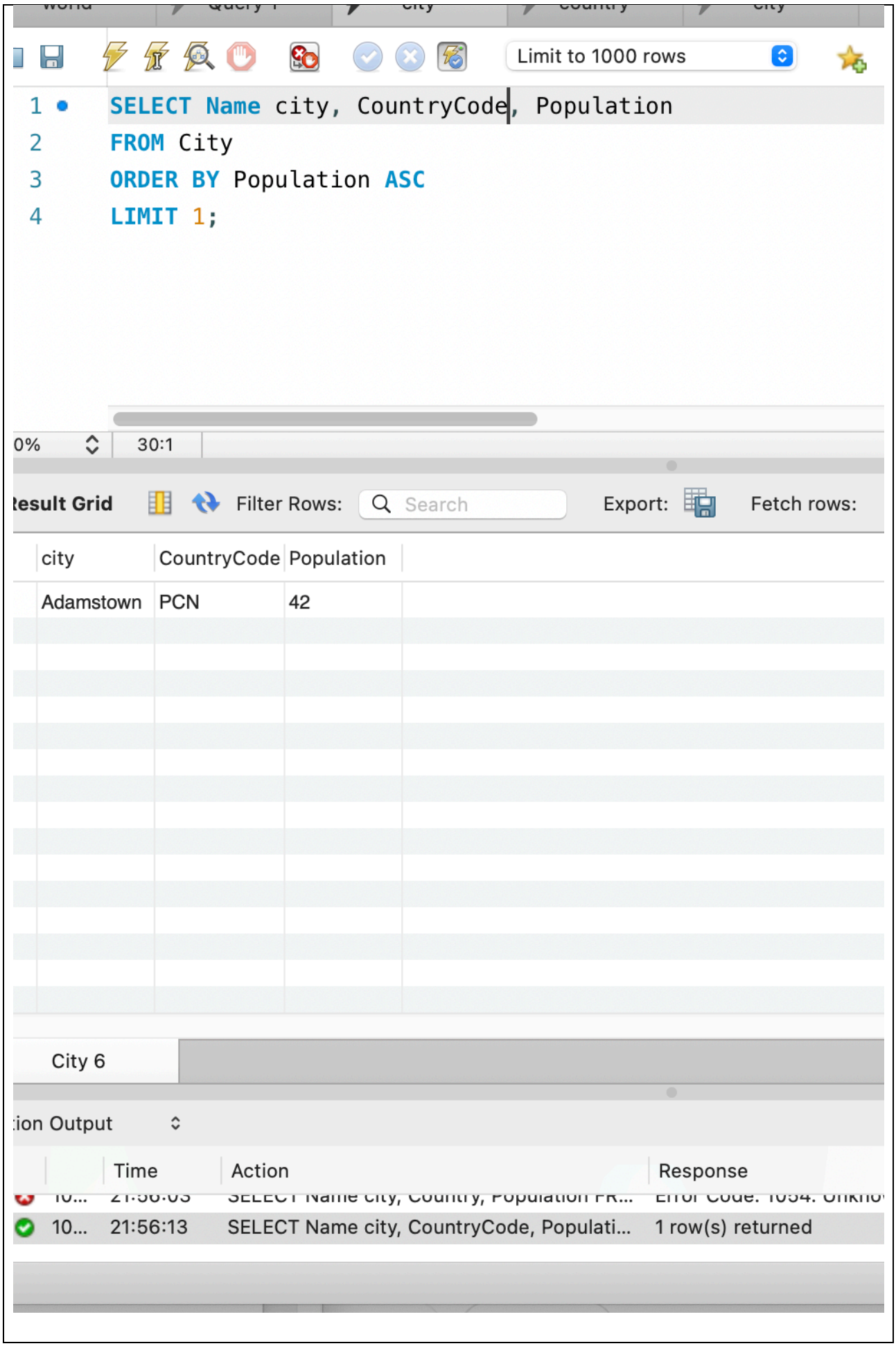
city	COUNT(*)
[San Cristóbal de] la Laguna	1
´s-Hertogenbosch	1
A Coruña (La Coruña)	1
Aachen	1
Aalborg	1
Aba	1
Abadan	1
Abaetetuba	1
Abakan	1
Abbotsford	1
Abeokuta	1
Aberdeen	1
Abha	1
Abidjan	1
Abiko	1
Abilene	1

Result 28

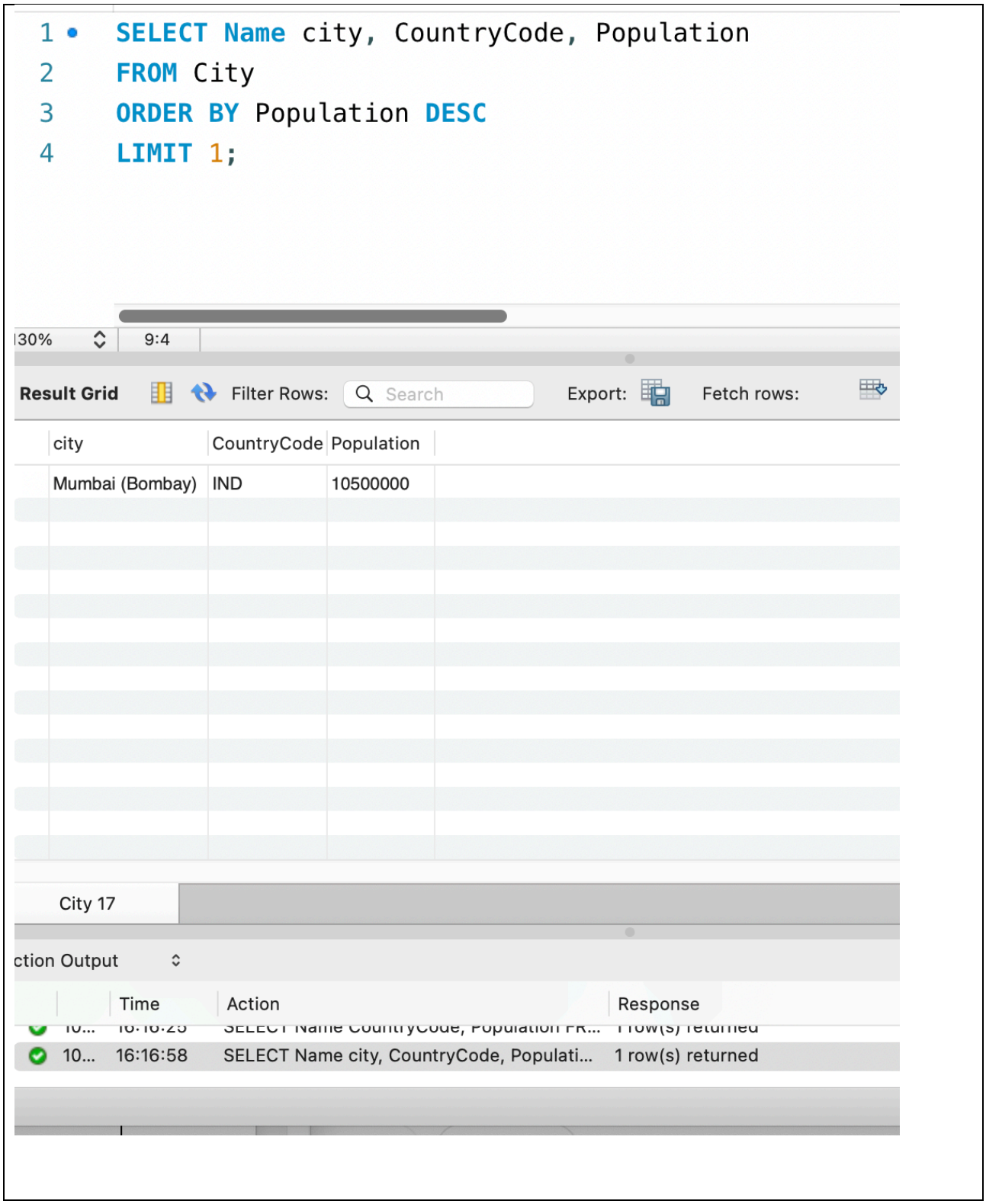
tion Output

	Time	Action	Response
10...	16:36:58	SELECT Name city, COUNT(*) FROM City G...	1000 row(s) returr

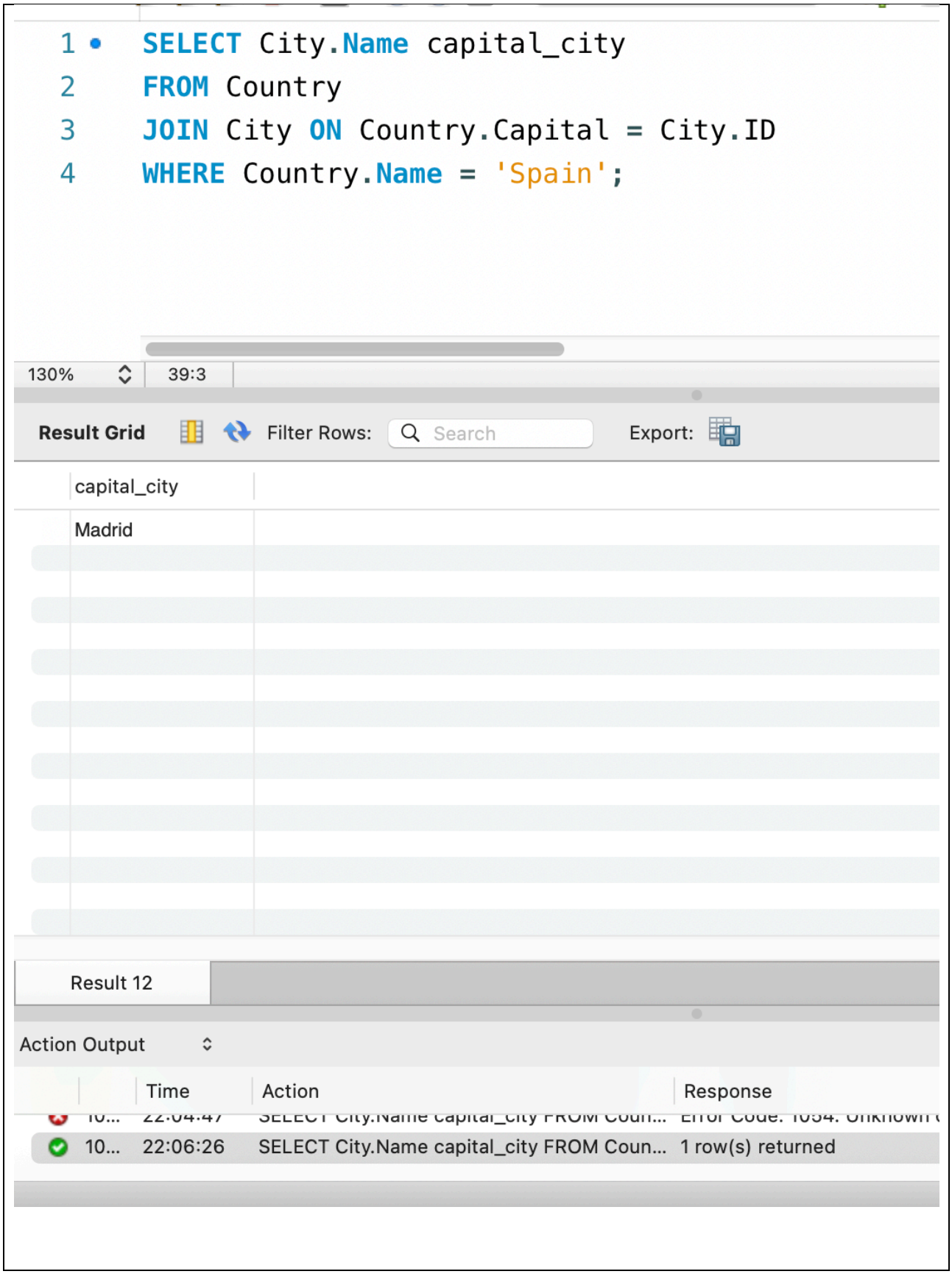
11. **City with the Lowest Population:** *Scenario:* A census bureau is conducting an analysis of urban population distribution. You're tasked with identifying the city with the lowest population from the database to provide a comprehensive overview of demographic trends.


Limit to 1000 rows
1 SELECT Name city, CountryCode, Population
2 FROM City
3 ORDER BY Population ASC
4 LIMIT 1;
0%
30:1
Result Grid
Filter Rows:
Search
Export:
Fetch rows:
city
CountryCode
Population
Adamstown
PCN
42
City 6
Time
Action
Response
10...
21:56:13
SELECT Name city, CountryCode, Populati...
1 row(s) returned

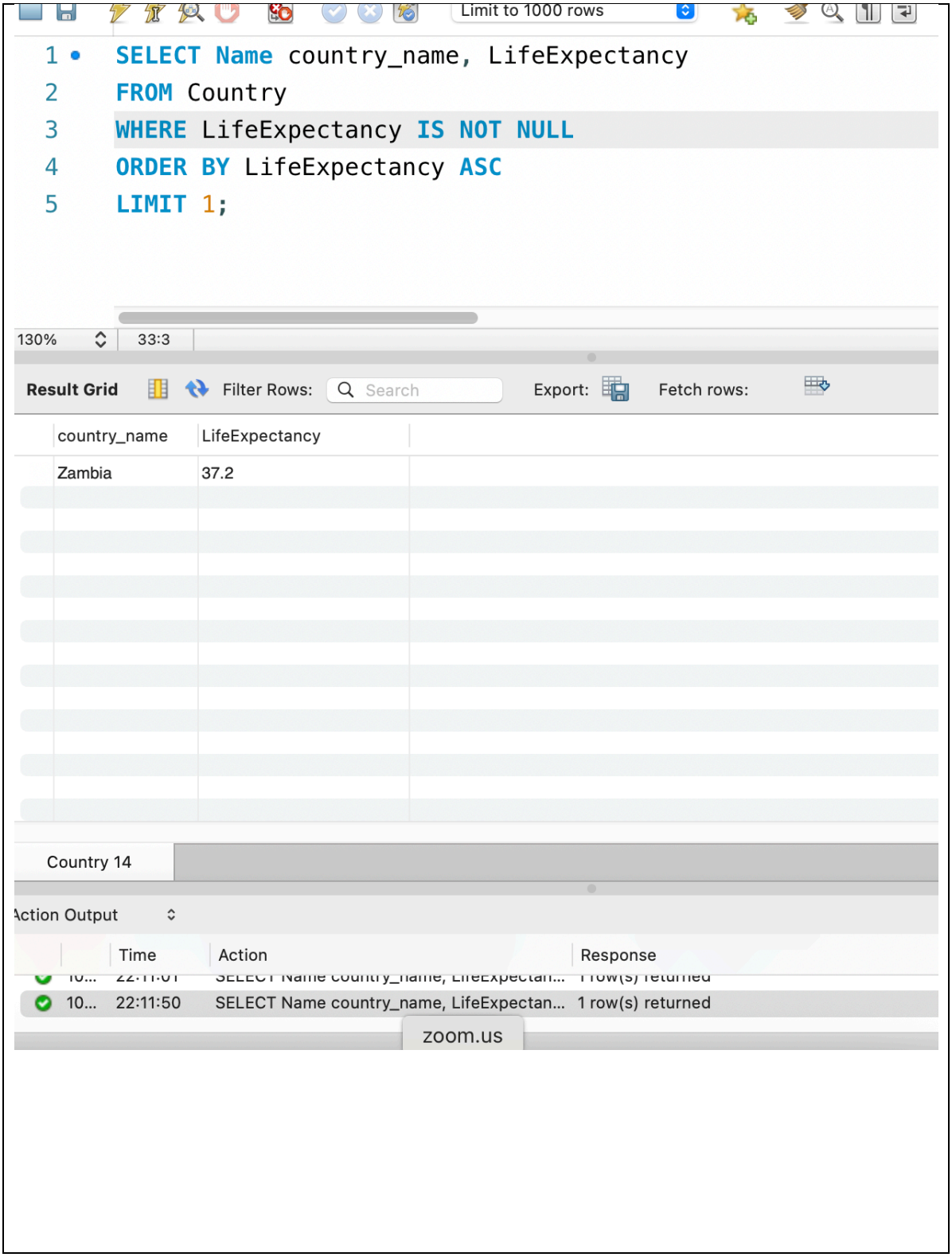
12. **Country with Largest Population:** *Scenario:* A global economic research institute requires data on countries with the largest populations for a comprehensive analysis. You're tasked with identifying the country with the highest population from the database to provide valuable insights into demographic trends.



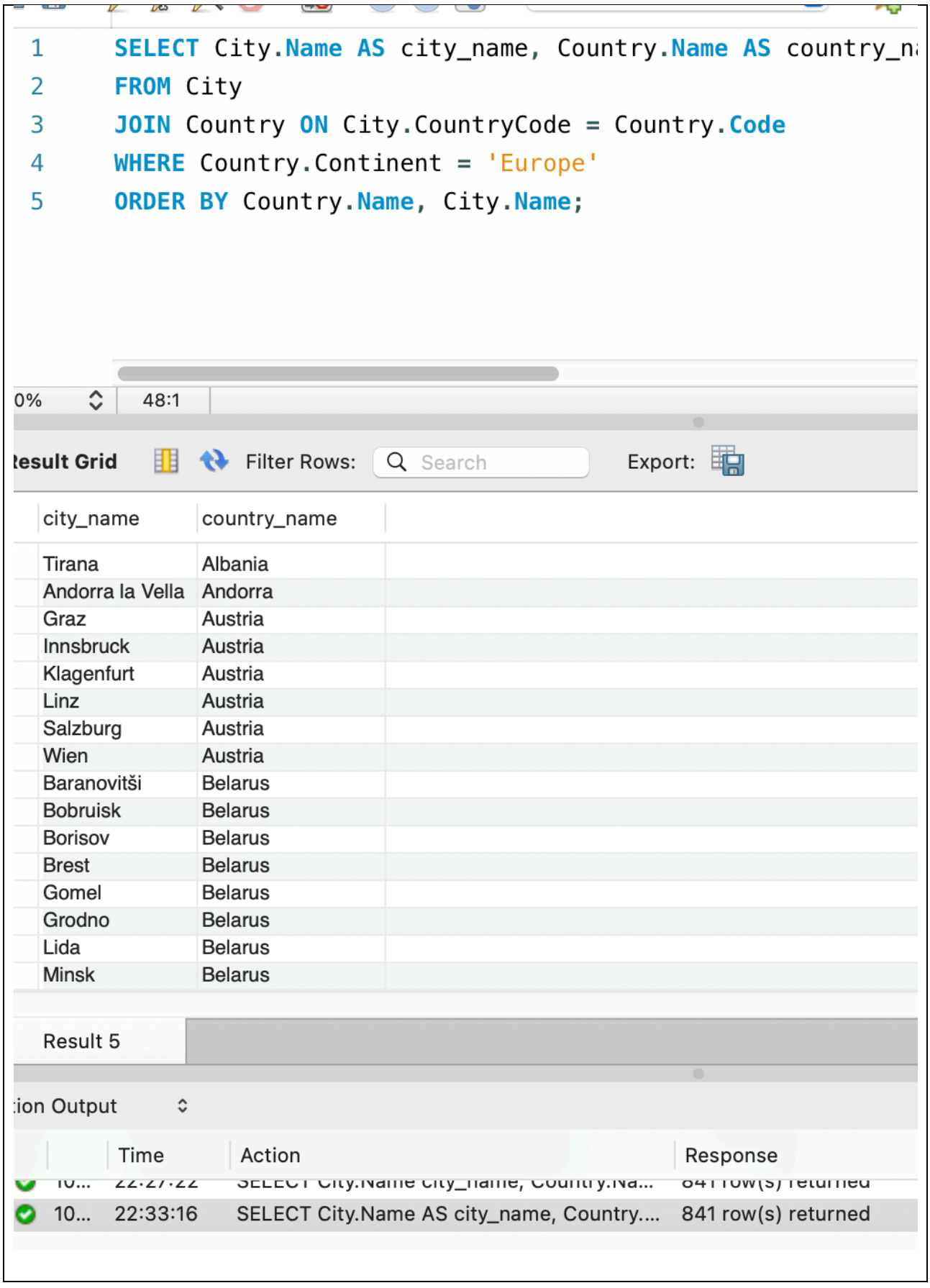
13. **Capital of Spain:** *Scenario:* A travel agency is organising tours across Europe and needs accurate information on capital cities. You're tasked with identifying the capital of Spain from the database to ensure itinerary accuracy and provide travellers with essential destination information.



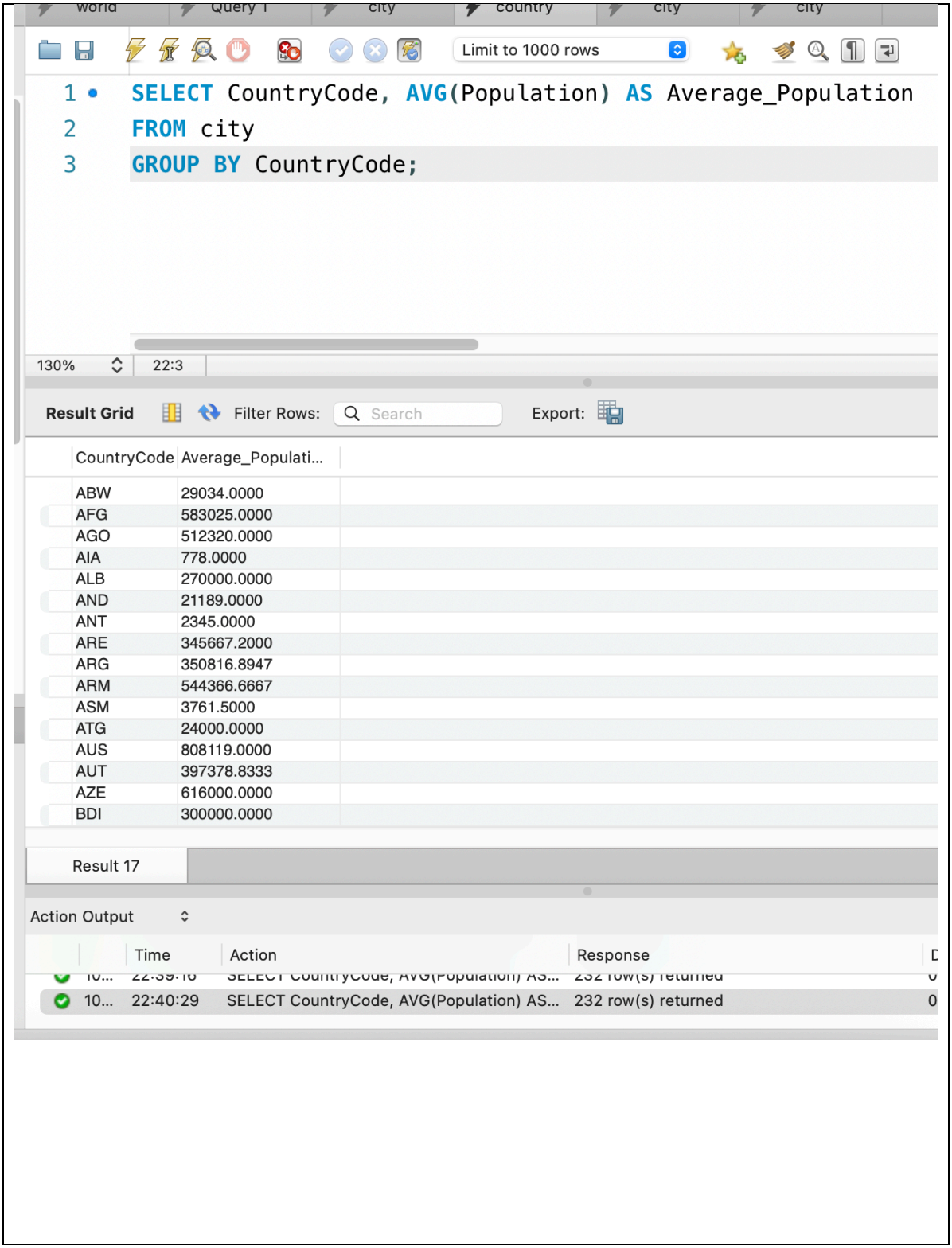
14. **Country with Shortest Life Expectancy:** *Scenario:* A healthcare foundation is conducting research on global health indicators. You're tasked with identifying the country with the shortest life expectancy from the database to inform their efforts in improving healthcare systems and policies.



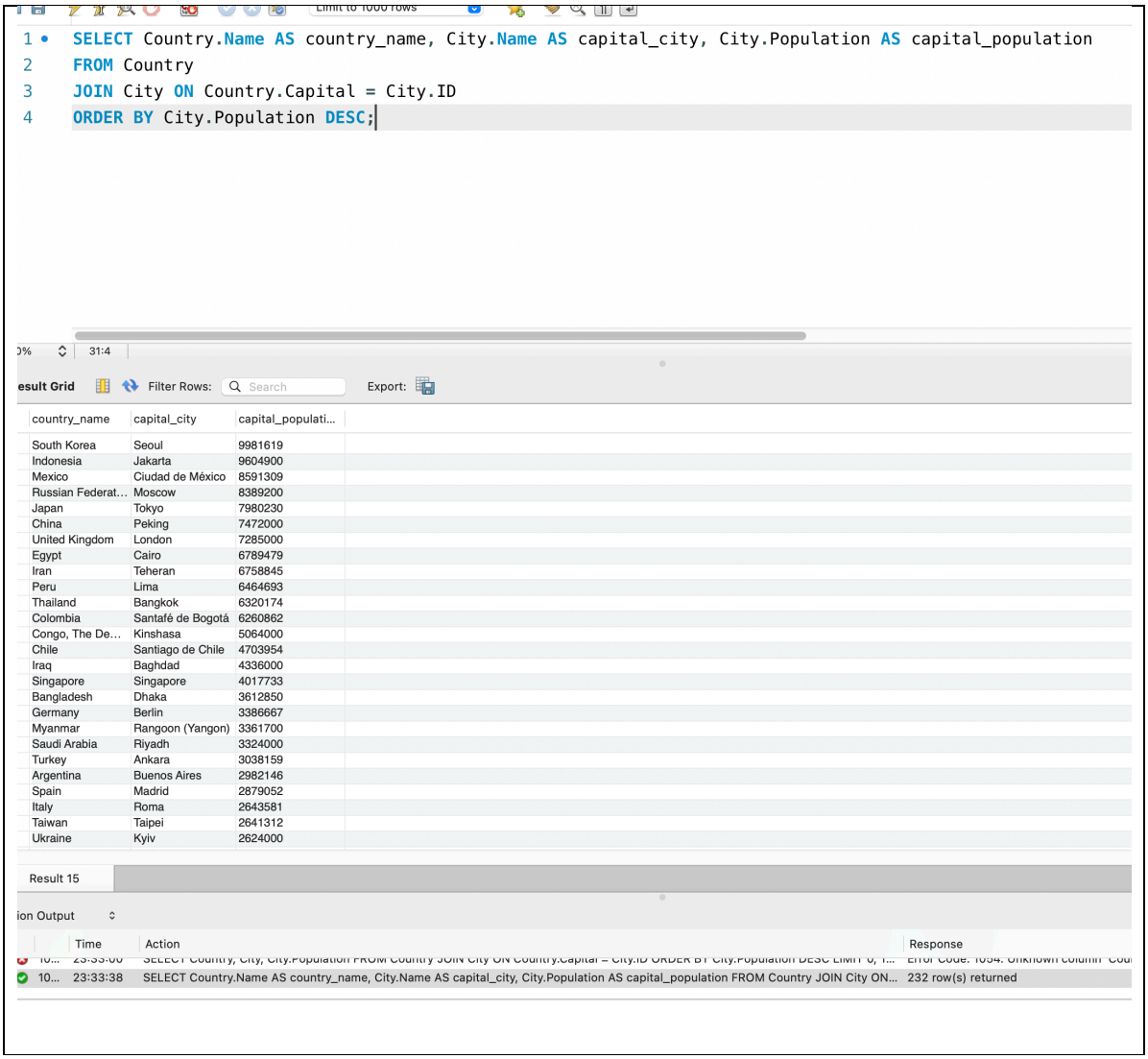
15. **Cities in Europe:** *Scenario:* A European cultural exchange program is seeking to connect students with cities across the continent. You're tasked with compiling a list of cities located in Europe from the database to facilitate program planning and student engagement.


1 SELECT City.Name AS city_name, Country.Name AS country_n
2 FROM City
3 JOIN Country ON City.CountryCode = Country.Code
4 WHERE Country.Continent = 'Europe'
5 ORDER BY Country.Name, City.Name;
0%
48:1
Result Grid
Filter Rows:
Search
Export:
city_name
country_name
Tirana
Albania
Andorra la Vella
Andorra
Graz
Austria
Innsbruck
Austria
Klagenfurt
Austria
Linz
Austria
Salzburg
Austria
Wien
Austria
Baranovitši
Belarus
Bobruisk
Belarus
Borisov
Belarus
Brest
Belarus
Gomel
Belarus
Grodno
Belarus
Lida
Belarus
Minsk
Belarus
Result 5
Time
Action
Response
10...
22:33:16
SELECT City.Name AS city_name, Country....
841 row(s) returned

16. **Average Population by Country:** *Scenario:* A demographic research team is conducting a comparative analysis of population distributions across countries. You're tasked with calculating the average population for each country from the database to provide valuable insights into global population trends.



17. **Capital Cities Population Comparison:** *Scenario:* A statistical analysis firm is examining population distributions between capital cities worldwide. You're tasked with comparing the populations of capital cities from different countries to identify trends and patterns in urban demographics.



```
SELECT Country.Name AS country_name, City.Name AS capital_city, City.Population AS capital_population
FROM Country
JOIN City ON Country.Capital = City.ID
ORDER BY City.Population DESC;
```

Result Grid — Filter Rows: Search — Export:

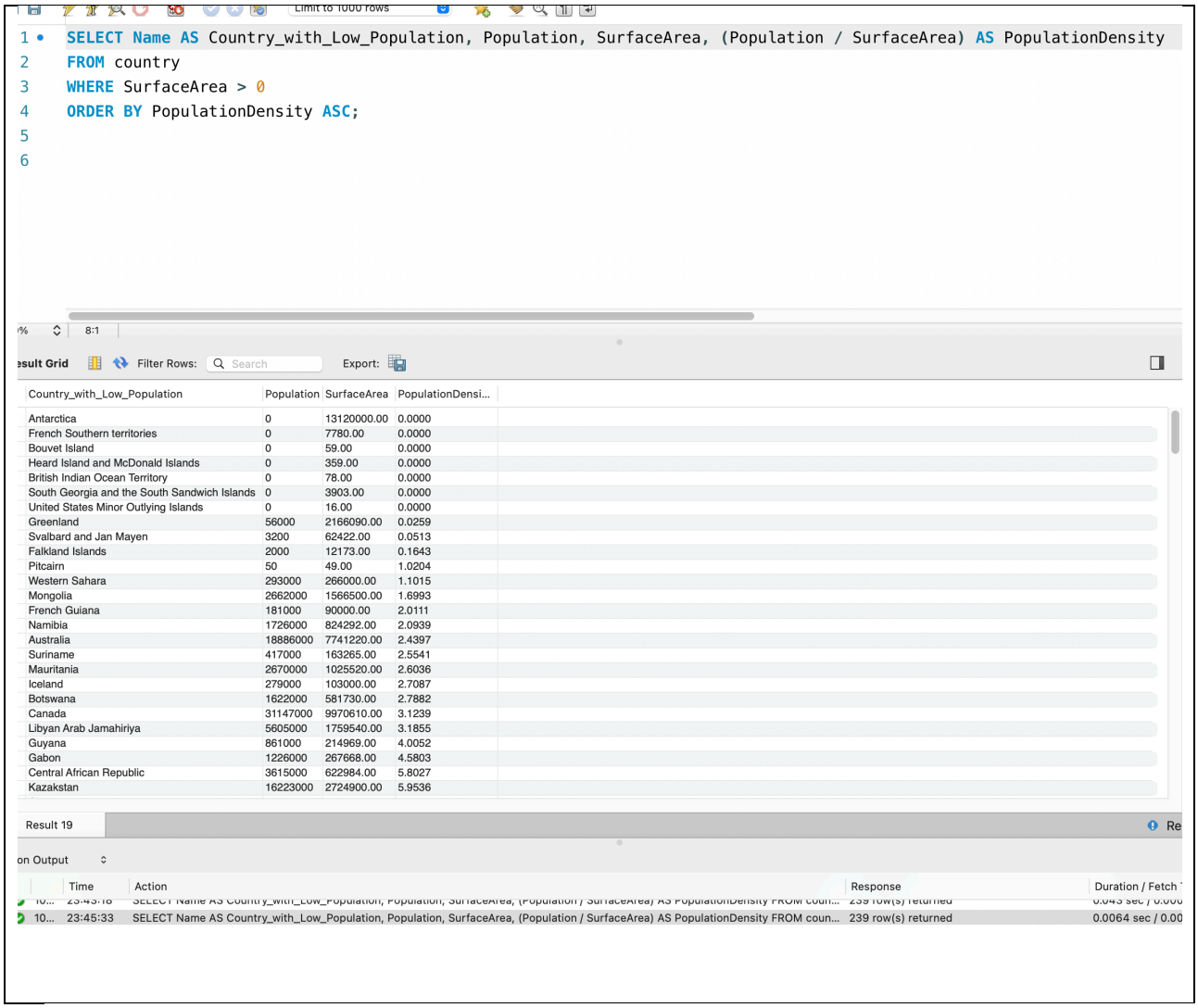
country_name	capital_city	capital_populati...
South Korea	Seoul	9981619
Indonesia	Jakarta	9604900
Mexico	Ciudad de México	8591309
Russian Federat...	Moscow	8389200
Japan	Tokyo	7980230
China	Peking	7472000
United Kingdom	London	7285000
Egypt	Cairo	6789479
Iran	Teheran	6758845
Peru	Lima	6464693
Thailand	Bangkok	6320174
Colombia	Santafé de Bogotá	6260862
Congo, The De...	Kinshasa	5064000
Chile	Santiago de Chile	4703954
Iraq	Baghdad	4336000
Singapore	Singapore	4017733
Bangladesh	Dhaka	3612850
Germany	Berlin	3386667
Myanmar	Rangoon (Yangon)	3361700
Saudi Arabia	Riyadh	3324000
Turkey	Ankara	3038159
Argentina	Buenos Aires	2982146
Spain	Madrid	2879052
Italy	Roma	2643581
Taiwan	Taipei	2641312
Ukraine	Kyiv	2624000

Result 15

Output

Time	Action	Response
10...	23:33:38	SELECT Country.Name AS country_name, City.Name AS capital_city, City.Population AS capital_population FROM Country JOIN City ON...	232 row(s) returned

18. **Countries with Low Population Density:** *Scenario:* An agricultural research institute is studying countries with low population densities for potential agricultural development projects. You're tasked with identifying countries with sparse populations from the database to support the institute's research efforts.



```
SELECT Name AS Country_with_Low_Population, Population, SurfaceArea, (Population / SurfaceArea) AS PopulationDensity
FROM country
WHERE SurfaceArea > 0
ORDER BY PopulationDensity ASC;
```

Country_with_Low_Population	Population	SurfaceArea	PopulationDensi...
Antarctica	0	13120000.00	0.0000
French Southern territories	0	7780.00	0.0000
Bouvet Island	0	59.00	0.0000
Heard Island and McDonald Islands	0	359.00	0.0000
British Indian Ocean Territory	0	78.00	0.0000
South Georgia and the South Sandwich Islands	0	3903.00	0.0000
United States Minor Outlying Islands	0	16.00	0.0000
Greenland	56000	2166090.00	0.0259
Svalbard and Jan Mayen	3200	62422.00	0.0513
Falkland Islands	2000	12173.00	0.1643
Pitcairn	50	49.00	1.0204
Western Sahara	293000	266000.00	1.1015
Mongolia	2662000	1566500.00	1.6993
French Guiana	181000	90000.00	2.0111
Namibia	1726000	824292.00	2.0939
Australia	18886000	7741220.00	2.4397
Suriname	417000	163265.00	2.5541
Mauritania	2670000	1025520.00	2.6036
Iceland	279000	103000.00	2.7087
Botswana	1622000	581730.00	2.7882
Canada	31147000	9970610.00	3.1239
Libyan Arab Jamahiriya	5605000	1759540.00	3.1855
Guyana	861000	214969.00	4.0052
Gabon	1226000	267668.00	4.5803
Central African Republic	3615000	622984.00	5.8027
Kazakstan	16223000	2724900.00	5.9536

Result 19

Time	Action	Response	Duration / Fetch
23:45:33	SELECT Name AS Country_with_Low_Population, Population, SurfaceArea, (Population / SurfaceArea) AS PopulationDensity FROM coun...	239 row(s) returned	0.0064 sec / 0.00

19. **Cities with High GDP per Capita:** *Scenario:* An economic consulting firm is analysing cities with high GDP per capita for investment opportunities. You're tasked with identifying cities with above-average GDP per capita from the database to assist the firm in identifying potential investment destinations.

```
SELECT City.Name AS city_name, Country.Name AS country_name, Country.GNP, Country.Population,
(Country.GNP / Country.Population) AS gdp_per_capita, City.Population AS city_population
FROM City
JOIN Country ON City.CountryCode = Country.Code
WHERE Country.Population > 0
  AND Country.GNP IS NOT NULL
  AND (Country.GNP / Country.Population) > (SELECT AVG(GNP / Population) FROM Country WHERE Population > 0 AND GNP IS NOT NULL)
ORDER BY gdp_per_capita DESC;
```

00% 30:8

Result Grid Filter Rows: Search Export: Fetch rows:

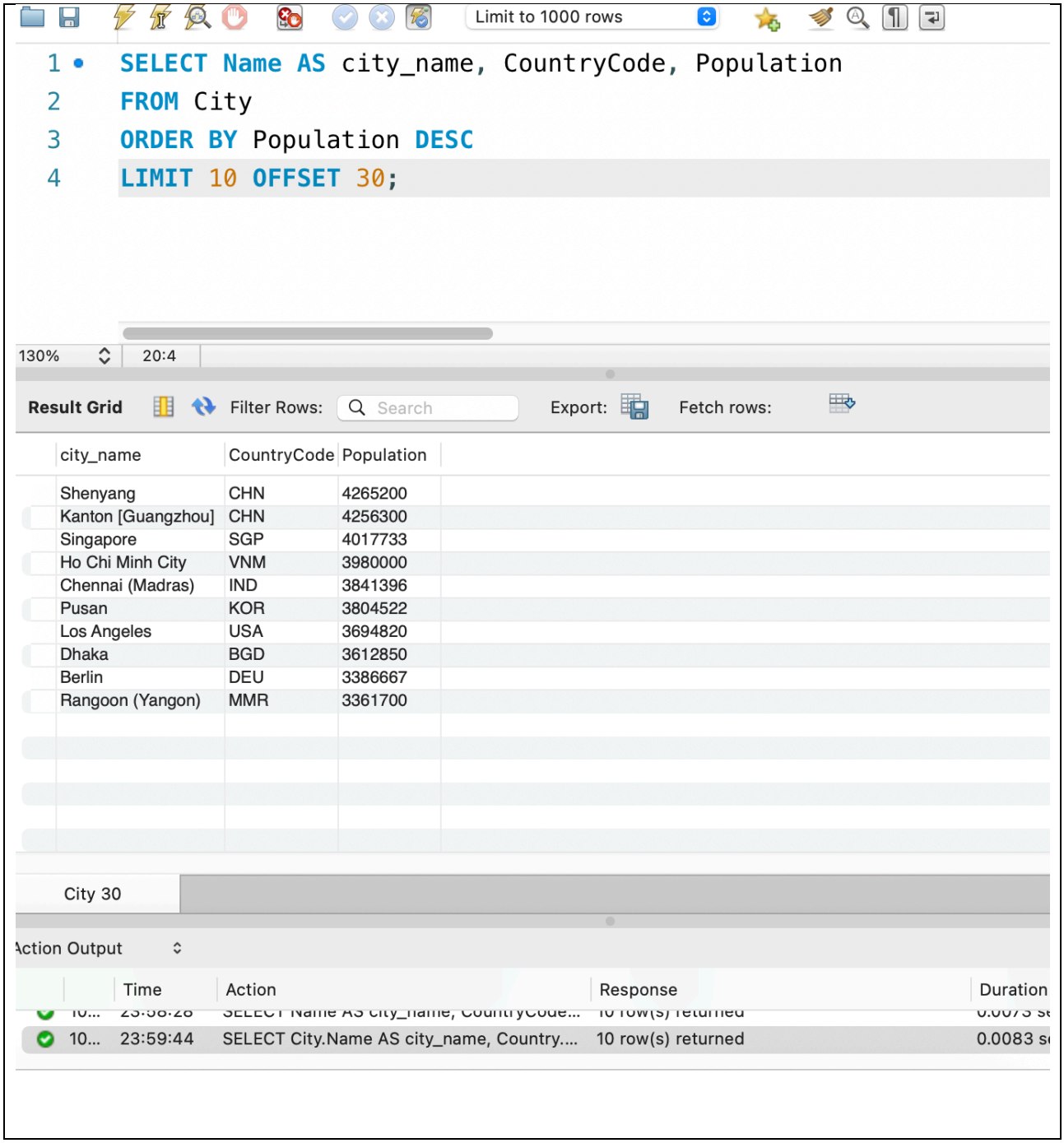
city_name	country_name	GNP	Population	gdp_per_capita	city_population
Luxembourg [Luxemburg/Lëtzebuerg]	Luxembourg	16321.00	435700	0.037459	80700
Zürich	Switzerland	264478.00	7160400	0.036936	336800
Geneve	Switzerland	264478.00	7160400	0.036936	173500
Basel	Switzerland	264478.00	7160400	0.036936	166700
Bern	Switzerland	264478.00	7160400	0.036936	122700
Lausanne	Switzerland	264478.00	7160400	0.036936	114500
Saint George	Bermuda	2328.00	65000	0.035815	1800
Hamilton	Bermuda	2328.00	65000	0.035815	1200
Bandar Seri Begawan	Brunei	11705.00	328000	0.035686	21484
Schaan	Liechtenstein	1119.00	32300	0.034644	5346
Vaduz	Liechtenstein	1119.00	32300	0.034644	5043
George Town	Cayman Islands	1263.00	38000	0.033237	19600
København	Denmark	174099.00	5330000	0.032664	495699
Århus	Denmark	174099.00	5330000	0.032664	284846
Odense	Denmark	174099.00	5330000	0.032664	183912
Aalborg	Denmark	174099.00	5330000	0.032664	161161
Frederiksberg	Denmark	174099.00	5330000	0.032664	90327
Oslo	Norway	145895.00	4478500	0.032577	508726
Bergen	Norway	145895.00	4478500	0.032577	230948
Trondheim	Norway	145895.00	4478500	0.032577	150166
Stavanger	Norway	145895.00	4478500	0.032577	108848
Bærum	Norway	145895.00	4478500	0.032577	101340
New York	United States	8510700.00	278357000	0.030575	8008278
Los Angeles	United States	8510700.00	278357000	0.030575	3694820
Chicago	United States	8510700.00	278357000	0.030575	2896016
Houston	United States	8510700.00	278357000	0.030575	1953631

Result 3

ction Output

	Time	Action	Response	Duration / Fet
10...	23:54:53	SELECT City.Name AS city_name, Country.Name AS country_name, Country.GNP, Country.Population, (Country.GNP / Country.Populatio...	1000 row(s) returned	0.0033 sec / 0

20. **Display Columns with Limit (Rows 31-40):** *Scenario:* A market research firm requires detailed information on cities beyond the top rankings for a comprehensive analysis. You're tasked with providing data on cities ranked between 31st and 40th by population to ensure a thorough understanding of urban demographics.



Course Notes

It is recommended to take notes from the course, use the space below to do so, or use the revision guide shared with the class:

We have included a range of additional links to further resources and information that you may find useful, these can be found within your revision guide.

END OF WORKBOOK

Please check through your work thoroughly before submitting and update the table of contents if required.

Please send your completed work booklet to your trainer.