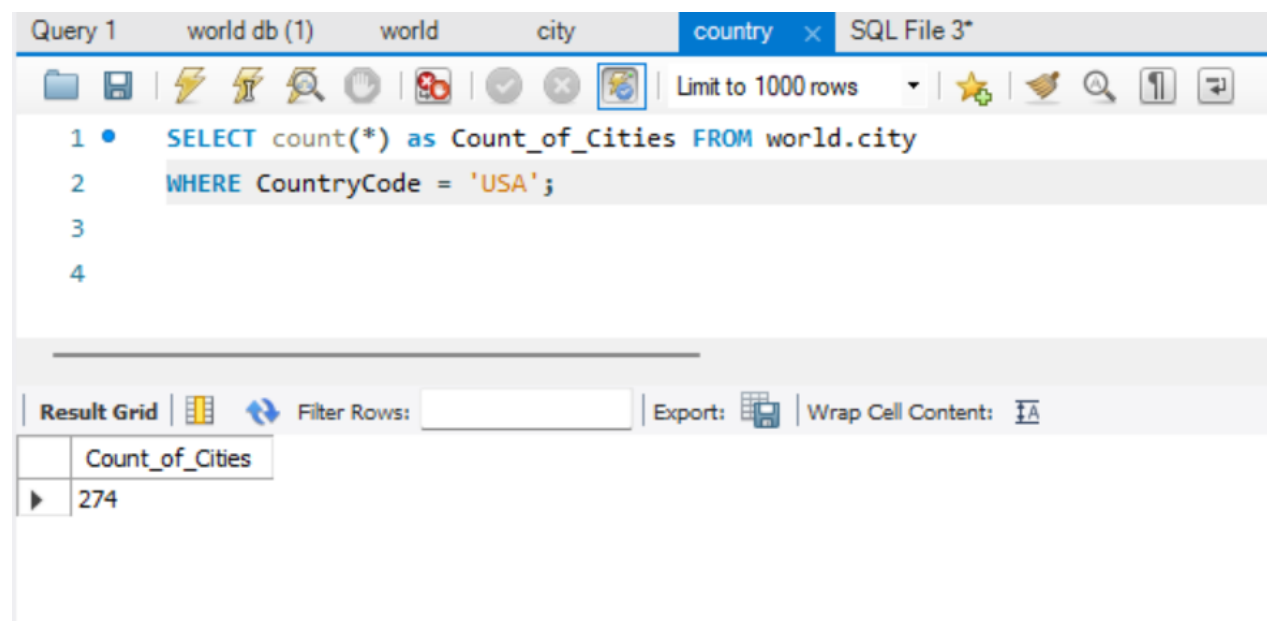


Day 4: Task 2: SQL Practical

- 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.

```
SELECT count(*) FROM world.city  
WHERE CountryCode = 'USA';
```



The screenshot shows a SQL IDE interface. The top pane displays the following SQL query:

```
1 • SELECT count(*) as Count_of_Cities FROM world.city  
2 WHERE CountryCode = 'USA';  
3  
4
```

The bottom pane shows the 'Result Grid' with the following data:

Count_of_Cities
274

The interface includes a toolbar with various icons for file operations, a 'Limit to 1000 rows' dropdown, and buttons for 'Export' and 'Wrap Cell Content'.

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.

```
SELECT
  Name AS Country,
  LifeExpectancy
FROM
  country
ORDER BY
  LifeExpectancy DESC
LIMIT 1
```

The screenshot shows a SQL IDE interface with a query editor and a result grid. The query editor contains the following SQL code:

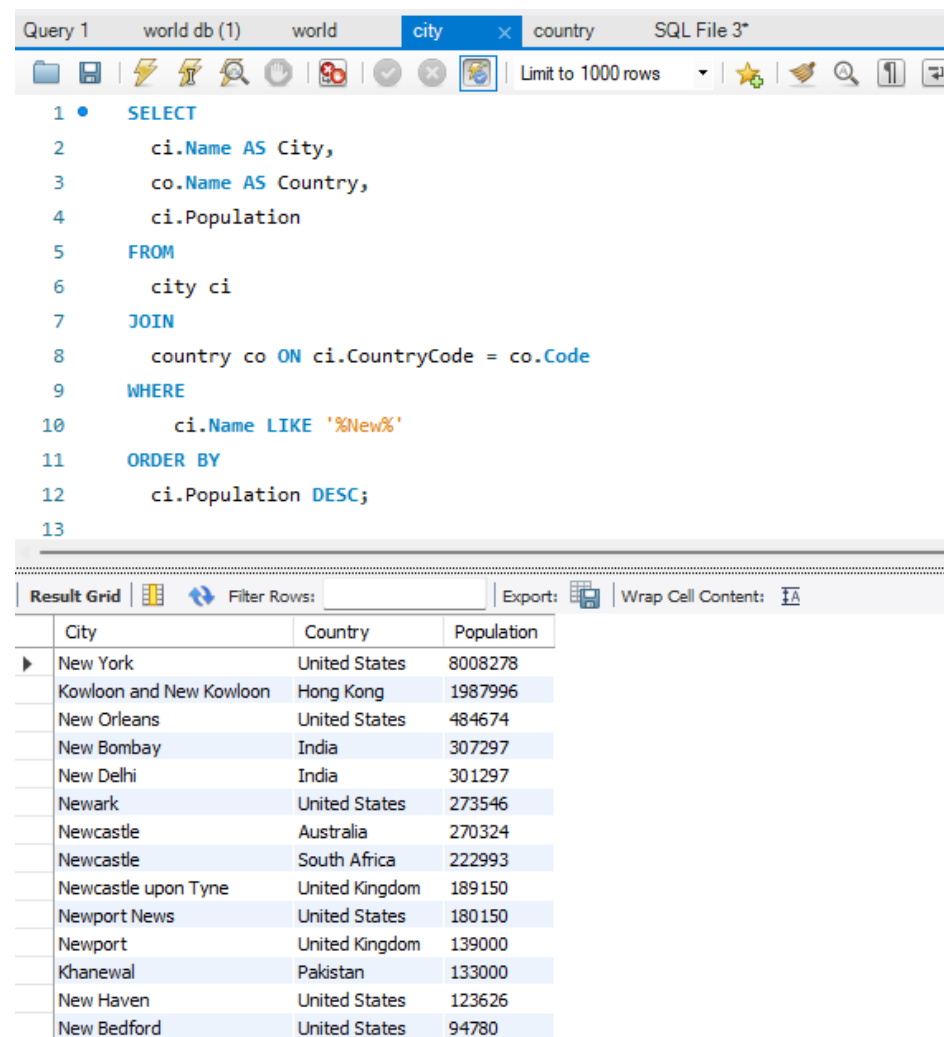
```
1 • SELECT
2     Name AS Country,
3     LifeExpectancy
4 FROM
5     country
6 ORDER BY
7     LifeExpectancy DESC
8 LIMIT 1
9
```

The result grid displays the following data:

Country	LifeExpectancy
Andorra	83.5

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.

```
SELECT
    ci.Name AS City,
    co.Name AS Country,
    ci.Population
FROM
    city ci
JOIN
    country co ON ci.CountryCode = co.Code
WHERE
    ci.Name LIKE '%New%'
ORDER BY
    ci.Population DESC;
```



The screenshot shows a SQL query editor with a toolbar and a results grid. The query is as follows:

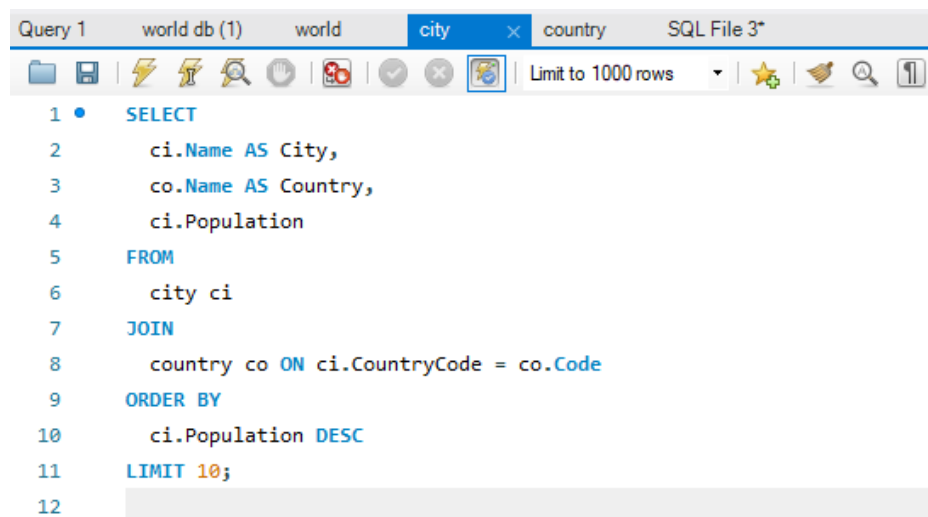
```
1 • SELECT
2     ci.Name AS City,
3     co.Name AS Country,
4     ci.Population
5 FROM
6     city ci
7 JOIN
8     country co ON ci.CountryCode = co.Code
9 WHERE
10    ci.Name LIKE '%New%'
11 ORDER BY
12    ci.Population DESC;
13
```

The results grid displays the following data:

City	Country	Population
New York	United States	8008278
Kowloon and New Kowloon	Hong Kong	1987996
New Orleans	United States	484674
New Bombay	India	307297
New Delhi	India	301297
Newark	United States	273546
Newcastle	Australia	270324
Newcastle	South Africa	222993
Newcastle upon Tyne	United Kingdom	189150
Newport News	United States	180150
Newport	United Kingdom	139000
Khanewal	Pakistan	133000
New Haven	United States	123626
New Bedford	United States	94780

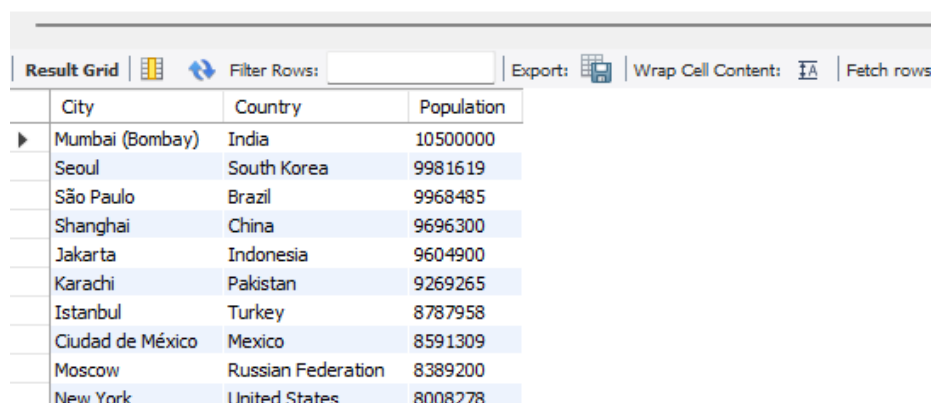
4. **Display Columns with Limit (First 10 Rows):** *Scenario:* You're tasked with providing a brief overview of the most populous cities in the world. To keep the report concise, you're instructed to list only the first 10 cities by population from the database.

```
SELECT
    ci.Name AS City,
    co.Name AS Country,
    ci.Population
FROM
    city ci
JOIN
    country co ON ci.CountryCode = co.Code
ORDER BY
    ci.Population DESC
LIMIT 10;
```



The screenshot shows a SQL IDE window with a tab labeled 'city'. The query editor contains the following SQL query:

```
1 • SELECT
2     ci.Name AS City,
3     co.Name AS Country,
4     ci.Population
5 FROM
6     city ci
7 JOIN
8     country co ON ci.CountryCode = co.Code
9 ORDER BY
10    ci.Population DESC
11 LIMIT 10;
12
```



The screenshot shows the 'Result Grid' tab in the SQL IDE. The query results are displayed in a table with the following data:

City	Country	Population
Mumbai (Bombay)	India	10500000
Seoul	South Korea	9981619
São Paulo	Brazil	9968485
Shanghai	China	9696300
Jakarta	Indonesia	9604900
Karachi	Pakistan	9269265
Istanbul	Turkey	8787958
Ciudad de México	Mexico	8591309
Moscow	Russian Federation	8389200
New York	United States	8008278

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.

```
SELECT
  ci.Name AS City,
  co.Name AS Country,
  ci.Population
FROM
  city ci
JOIN
  country co ON ci.CountryCode = co.Code
WHERE
  ci.Population > 2000000
ORDER BY
  ci.Population DESC;
```

The screenshot displays a SQL IDE interface. The top toolbar includes icons for file operations, execution, and a 'Limit to 1000 rows' dropdown. The query editor contains the following SQL code:

```
1 SELECT
2   ci.Name AS City,
3   co.Name AS Country,
4   ci.Population
5 FROM
6   city ci
7 JOIN
8   country co ON ci.CountryCode = co.Code
9 WHERE
10  ci.Population > 2000000
11 ORDER BY
12  ci.Population DESC;
13
```

Below the editor, the 'Result Grid' shows the query results:

City	Country	Population
Mumbai (Bombay)	India	10500000
Seoul	South Korea	9981619
São Paulo	Brazil	9968485
Shanghai	China	9696300
Jakarta	Indonesia	9604900
Karachi	Pakistan	9269265
Istanbul	Turkey	8787958
Ciudad de México	Mexico	8591309
Moscow	Russian Federation	8389200
New York	United States	8008278
Tokyo	Japan	7980230
Peking	China	7472000
London	United Kingdom	7285000

The 'Output' pane at the bottom shows the execution log:

#	Time	Action	Message
1	14:18:44	SELECT ci.Name AS City, co.Name AS Country, ci.Population FROM city ci JOIN country co ON ci.CountryCode = co.Code WHERE ci.Name LIKE 'Be%': ORDER BY ci.Name ASC LIMIT 0, 1000	51 row(s) returned
2	14:19:43	SELECT ci.Name AS City, co.Name AS Country, ci.Population FROM city ci JOIN country co ON ci.CountryCode = co.Code WHERE ci.Population > 2000000 ORDER BY ci.Population DESC LIMIT 0...	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.

```
SELECT
    ci.Name AS City,
    co.Name AS Country,
    ci.Population
FROM
    city ci
JOIN
    country co ON ci.CountryCode = co.Code
WHERE
    ci.Name LIKE 'Be%'
ORDER BY
    ci.Name ASC;
```

The screenshot displays a SQL IDE interface. The top pane shows the SQL query, and the bottom pane shows the results in a table format. The query is as follows:

```
1 SELECT
2     ci.Name AS City,
3     co.Name AS Country,
4     ci.Population
5 FROM
6     city ci
7 JOIN
8     country co ON ci.CountryCode = co.Code
9 WHERE
10    ci.Name LIKE 'Be%'
11 ORDER BY
12    ci.Name ASC;
```

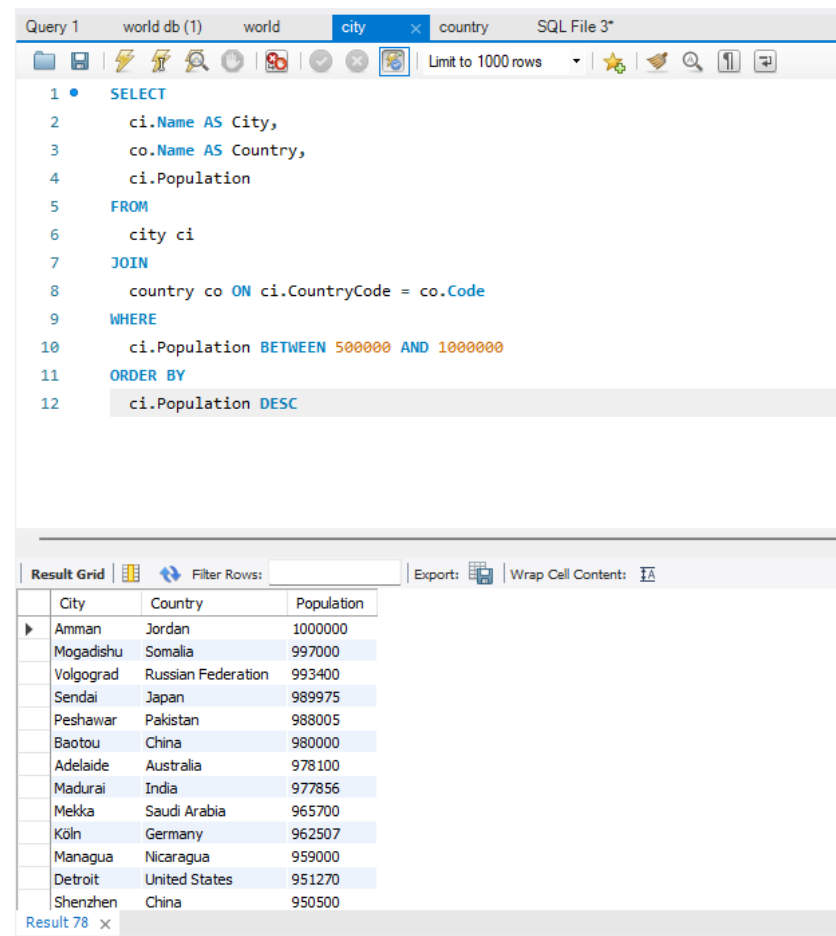
The results table shows the following data:

City	Country	Population
Beau Bassin-Rose Hill	Mauritius	100616
Beaumont	United States	113866
Beawar	India	105363
Béchar	Algeria	107311
Beerseba	Israel	163700
Bei'an	China	204899
Beihai	China	112673
Beipiao	China	194301
Berra	Mozambique	397368
Beirut	Lebanon	1100000
Béjaia	Algeria	117162
Bekasi	Indonesia	644300
Belém	Brazil	1186926

The bottom pane shows the output of the query, indicating that 51 row(s) were returned.

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.

```
SELECT
    ci.Name AS City,
    co.Name AS Country,
    ci.Population
FROM
    city ci
JOIN
    country co ON ci.CountryCode = co.Code
WHERE
    ci.Population BETWEEN 500000 AND 1000000
ORDER BY
    ci.Population ASC;
```



Query 1 world db (1) world city country SQL File 3*

Limit to 1000 rows

```
1 • SELECT
2     ci.Name AS City,
3     co.Name AS Country,
4     ci.Population
5 FROM
6     city ci
7 JOIN
8     country co ON ci.CountryCode = co.Code
9 WHERE
10    ci.Population BETWEEN 500000 AND 1000000
11 ORDER BY
12    ci.Population DESC
```

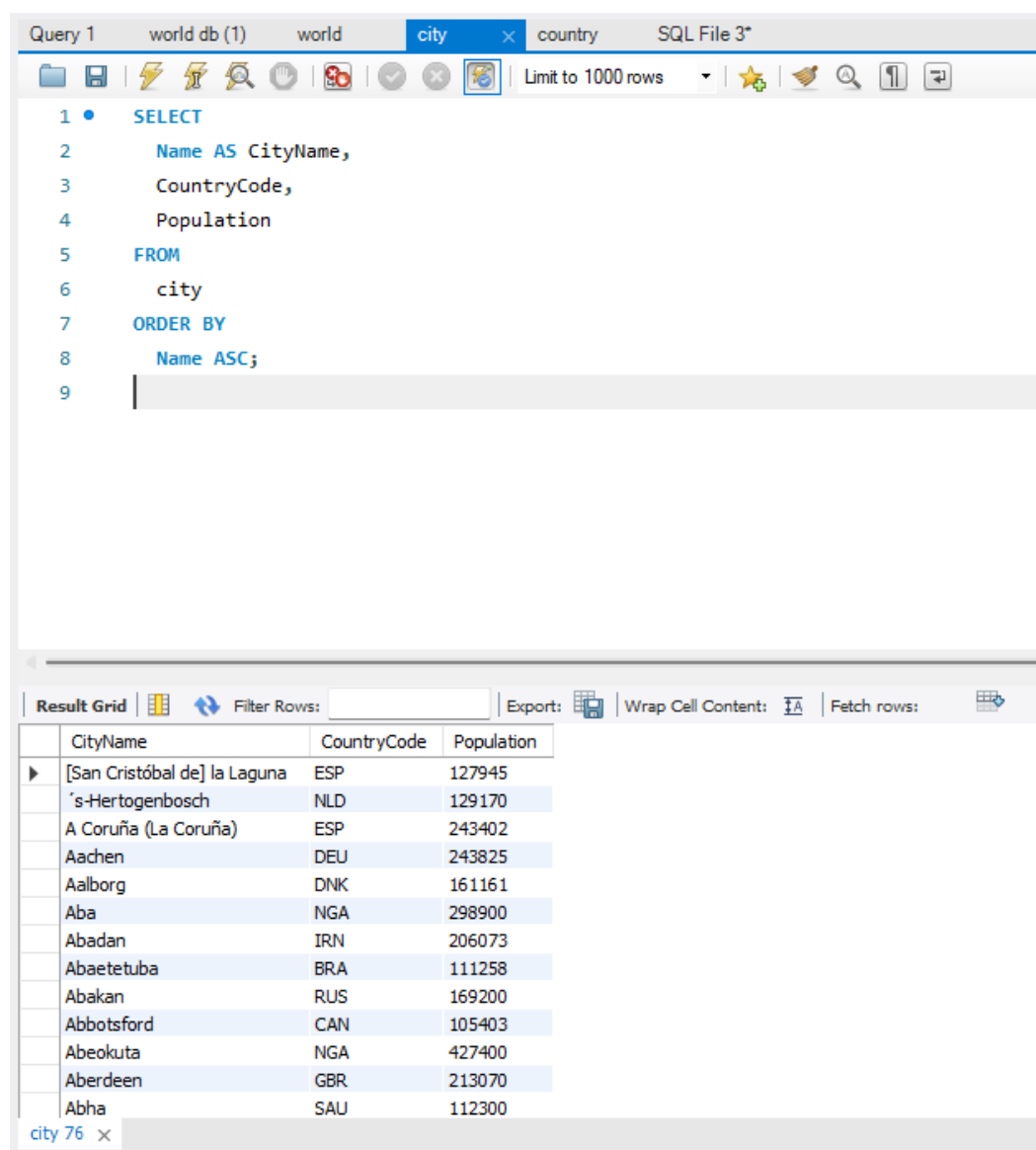
Result Grid Filter Rows: Export: Wrap Cell Content:

	City	Country	Population
▶	Amman	Jordan	1000000
	Mogadishu	Somalia	997000
	Volgograd	Russian Federation	993400
	Sendai	Japan	989975
	Peshawar	Pakistan	988005
	Baotou	China	980000
	Adelaide	Australia	978100
	Madurai	India	977856
	Mekka	Saudi Arabia	965700
	Köln	Germany	962507
	Managua	Nicaragua	959000
	Detroit	United States	951270
	Shenzhen	China	950500

Result 78 x

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.

```
SELECT
    Name AS CityName,
    CountryCode,
    Population
FROM
    city
ORDER BY
    Name ASC;
```

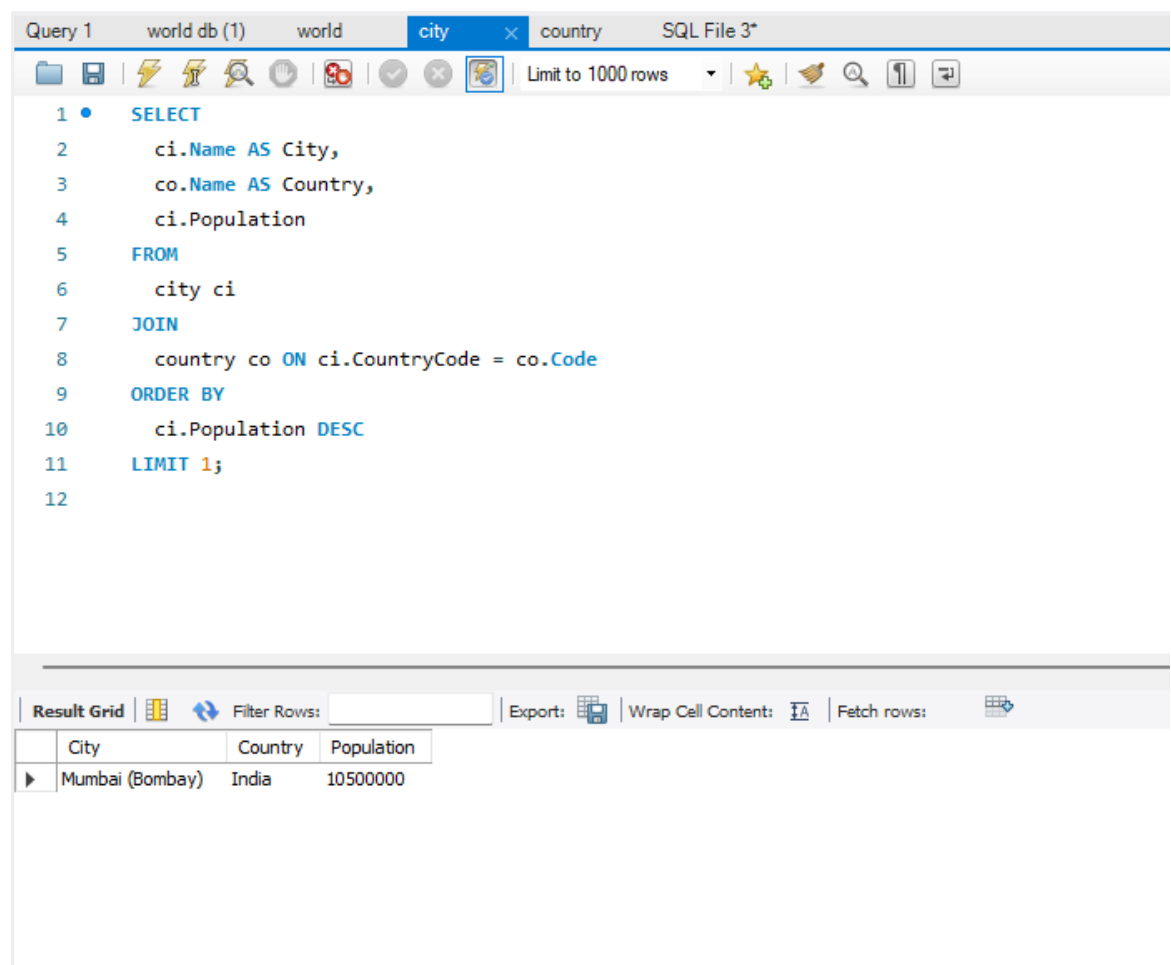


The screenshot shows a database query editor with a toolbar at the top. The query is displayed in a text area, and the results are shown in a table below. The table has four columns: CityName, CountryCode, and Population. The results are sorted alphabetically by CityName.

CityName	CountryCode	Population
[San Cristóbal de] la Laguna	ESP	127945
's-Hertogenbosch	NLD	129170
A Coruña (La Coruña)	ESP	243402
Aachen	DEU	243825
Aalborg	DNK	161161
Aba	NGA	298900
Abadan	IRN	206073
Abaetetuba	BRA	111258
Abakan	RUS	169200
Abbotsford	CAN	105403
Abeokuta	NGA	427400
Aberdeen	GBR	213070
Abha	SAU	112300

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.

```
SELECT
  ci.Name AS City,
  co.Name AS Country,
  ci.Population
FROM
  city ci
JOIN
  country co ON ci.CountryCode = co.Code
ORDER BY
  ci.Population DESC
LIMIT 1;
```



The screenshot displays a SQL query editor window with the following tabs: "Query 1", "world db (1)", "world", "city", "country", and "SQL File 3*". The "city" tab is active. The query editor shows the SQL query from the previous block. Below the query editor, the "Result Grid" is visible, showing the results of the query. The grid has columns for "City", "Country", and "Population". The first row shows "Mumbai (Bombay)" as the city, "India" as the country, and "10500000" as the population.

City	Country	Population
Mumbai (Bombay)	India	10500000

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.

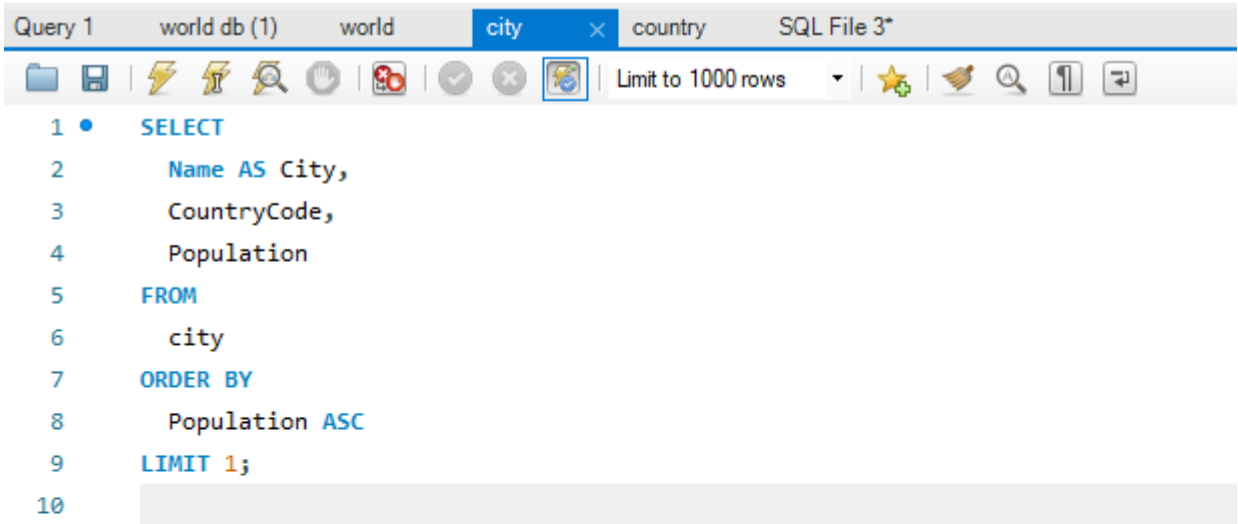
```
SELECT
  Name AS CityName,
  COUNT(*) AS OccurrenceCount
FROM
  city
GROUP BY
  Name
ORDER BY
  OccurrenceCount DESC;
```

The screenshot shows a database query editor with the following tabs: Query 1, world db (1), world, city, country, and SQL File 3*. The query editor displays the SQL query from the previous block. Below the query editor, the results are shown in a table with columns CityName and OccurrenceCount. The results are sorted by OccurrenceCount in descending order. The first row is San José with 4 occurrences. The remaining 15 rows have 3 occurrences each and are sorted alphabetically by CityName.

CityName	OccurrenceCount
San José	4
Córdoba	3
San Miguel	3
San Fernando	3
Hamilton	3
La Paz	3
Toledo	3
Cambridge	3
Springfield	3
Richmond	3
Valencia	3
León	3
Victoria	3

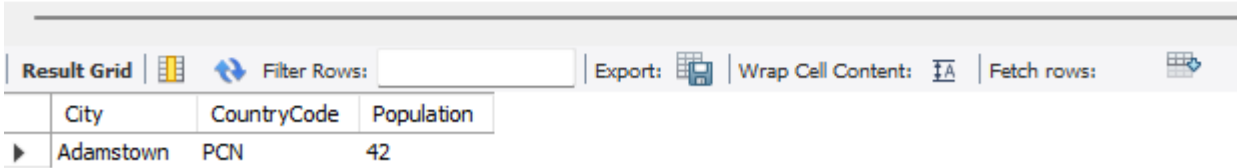
11. **City with the Lowest Population:** *Scenario:* A census bureau is conducting an analysis of urban population distribution. You're tasked with identifying the city with the lowest population from the database to provide a comprehensive overview of demographic trends.

```
SELECT
  Name AS City,
  CountryCode,
  Population
FROM
  city
ORDER BY
  Population ASC
LIMIT 1;
```



The screenshot shows a database query editor with a toolbar at the top containing icons for file operations, execution, and navigation. The query editor displays the following SQL query:

```
1 • SELECT
2     Name AS City,
3     CountryCode,
4     Population
5 FROM
6     city
7 ORDER BY
8     Population ASC
9 LIMIT 1;
10
```



The screenshot shows the result grid of the query. The toolbar at the top includes options for 'Result Grid', 'Filter Rows', 'Export', 'Wrap Cell Content', and 'Fetch rows'. The result grid displays the following data:

	City	CountryCode	Population
▶	Adamstown	PCN	42

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.

```
SELECT
  Name AS Country,
  Population
FROM
  country
ORDER BY
  Population DESC
LIMIT 1;
```

The screenshot shows a SQL query editor with the following tabs: Query 1, world db (1), world, city, country, and SQL File 3*. The query is as follows:

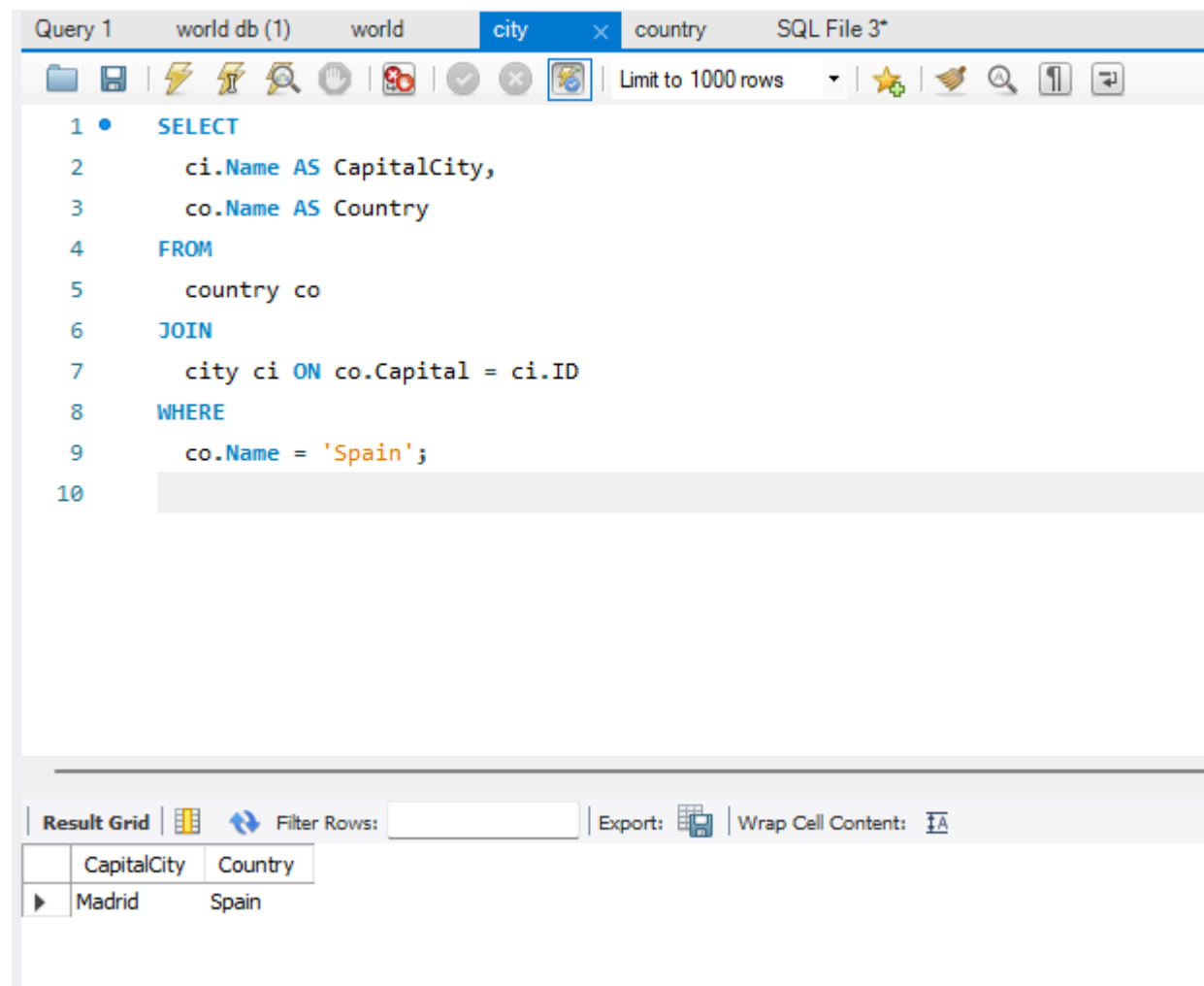
```
1 • SELECT
2   Name AS Country,
3   Population
4 FROM
5   country
6 ORDER BY
7   Population DESC
8 LIMIT 1;
9
```

The result grid shows the following data:

Country	Population
China	1277558000

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.

```
SELECT
  ci.Name AS CapitalCity,
  co.Name AS Country
FROM
  country co
JOIN
  city ci ON co.Capital = ci.ID
WHERE
  co.Name = 'Spain';
```



The screenshot shows a SQL IDE interface with a query editor and a results pane. The query editor displays the following SQL query:

```
1 • SELECT
2     ci.Name AS CapitalCity,
3     co.Name AS Country
4 FROM
5     country co
6 JOIN
7     city ci ON co.Capital = ci.ID
8 WHERE
9     co.Name = 'Spain';
10
```

The results pane shows the following data:

CapitalCity	Country
Madrid	Spain

14. **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.

```
SELECT
ci.Name AS City,
co.Name AS Country,
ci.District,
ci.Population
FROM
world.city ci
JOIN
world.country co
ON
ci.CountryCode = co.Code
WHERE
co.Continent = 'Europe'
ORDER BY
co.Name, ci.Name;
```

The screenshot displays a database query editor with the following SQL query:

```
1 SELECT
2   ci.Name AS City,
3   co.Name AS Country,
4   ci.District,
5   ci.Population
6 FROM
7   world.city ci
8 JOIN
9   world.country co
10 ON
11   ci.CountryCode = co.Code
12 WHERE
13   co.Continent = 'Europe'
14 ORDER BY
15   co.Name, ci.Name;
```

Below the query editor, the 'Result Grid' shows the first 10 rows of the query results:

City	Country	District	Population
Tirana	Albania	Tirana	270000
Andorra la Vella	Andorra	Andorra la Vella	21189
Graz	Austria	Steiermark	240967
Innsbruck	Austria	Tirol	111752
Klagenfurt	Austria	Kärnten	91141
Linz	Austria	North Austria	188022
Salzburg	Austria	Salzburg	144247
Wien	Austria	Wien	1608144
Baranoviči	Belarus	Brest	167000
Bobruisk	Belarus	Mogiljov	221000

The 'Output' pane shows the execution log for the query:

#	Time	Action	Message
18	12:57:17	SELECT ci.Name AS City, co.Name AS CountryName, ci.Population FROM world.city ci JOIN world.country co ON ci.CountryCode = co.Code ORDER BY ci.Population LIMIT 10 OFFSET 30	10 row(s) returned
19	12:57:30	SELECT ci.Name AS City, co.Name AS 'Country Name', ci.Population FROM world.city ci JOIN world.country co ON ci.CountryCode = co.Code ORDER BY ci.Population LIMIT 10 OFFSET 30	10 row(s) returned
20	12:57:40	SELECT ci.Name AS City, co.Name AS Country, ci.Population FROM world.city ci JOIN world.country co ON ci.CountryCode = co.Code ORDER BY ci.Population LIMIT 10 OFFSET 30	10 row(s) returned
21	13:00:24	* FROM world.city ci JOIN world.country co ON ci.CountryCode = co.Code LIMIT 0, 1000	1000 row(s) returned
22	13:01:42	* FROM world.city ci JOIN world.country co ON ci.CountryCode = co.Code WHERE co.Continent = 'Europe' LIMIT 0, 1000	841 row(s) returned
23	13:02:02	SELECT ci.Name AS City, co.Name AS Country, ci.District, ci.Population FROM world.city ci JOIN world.country co ON ci.CountryCode = co.Code WHERE co.Continent = 'Europe' LIMIT 0, 1000	841 row(s) returned
24	13:02:17	SELECT ci.Name AS City, co.Name AS Country, ci.District, ci.Population FROM world.city ci JOIN world.country co ON ci.CountryCode = co.Code WHERE co.Continent = 'Europe' ORDER BY co.Name, ci.Name	841 row(s) returned

15. **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.

```
SELECT
    Name,
    AVG(Population) AS AverageCountryPopulation
FROM
    country
Group By Name;
```

The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor contains the following SQL code:

```
1 • SELECT
2     Name,
3     AVG(Population) AS AverageCountryPopulation
4 FROM
5     country
6 Group By Name;
```

The results grid displays the following data:

Name	AverageCountryPopulation
Aruba	103000.0000
Afghanistan	22720000.0000
Angola	12878000.0000
Anguilla	8000.0000
Albania	3401200.0000
Andorra	78000.0000
Netherlands Antilles	217000.0000
United Arab Emirates	2441000.0000
Argentina	37032000.0000
Armenia	3520000.0000
American Samoa	68000.0000
Antarctica	0.0000
French Southern ter...	0.0000

The bottom of the screenshot shows the 'Action Output' tab with the following message:

```
1 12:34:44 SELECT Name, AVG(Population) AS AverageCountryPopulation FROM country Group By Name LIMIT 0, 1000
```

Message: 239 row(s) returned

```
SELECT
    c.Name AS Country,
    COUNT(ci.ID) AS NumberOfCities,
    AVG(ci.Population) AS AverageCityPopulation
FROM
    country c
JOIN
    city ci ON c.Code = ci.CountryCode
GROUP BY
    c.Code, c.Name
ORDER BY
    AverageCityPopulation DESC;
```


16. **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,
    city.Name AS CapitalCity,
    city.Population AS CapitalPopulation
FROM
    country
JOIN
    city ON country.Capital = city.ID
ORDER BY
    city.Population DESC;
```

The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor contains the following SQL code:

```
1 SELECT
2     country.Name AS Country,
3     city.Name AS CapitalCity,
4     city.Population AS CapitalPopulation
5 FROM
6     country
7 JOIN
8     city ON country.Capital = city.ID
9 ORDER BY
10    city.Population DESC;
```

The results grid displays the following data:

Country	CapitalCity	CapitalPopulation
South Korea	Seoul	9981619
Indonesia	Jakarta	9604900
Mexico	Ciudad de México	8591309
Russian Federation	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 Demo...	Kinshasa	5064000

The output pane shows the following message:

```
1 12:39:15 SELECT country.Name AS Country, city.Name AS CapitalCity, city.Population AS CapitalPopulation FROM country JOIN city ON country.Capital = city.ID ORDER BY city.Population DESC LIMIT 0, 1000 232 row(s) returned
```

17. **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,
  Population,
  SurfaceArea,
  ROUND(Population / SurfaceArea, 2) AS PopulationDensity
FROM
  country
ORDER BY
  PopulationDensity
LIMIT 10;
```

The screenshot shows a database query tool interface. The top pane displays the SQL query, and the bottom pane shows the results in a grid and an action log.

Query 1 world db (1) world city country SQL File 3*

Limit to 1000 rows

1 • SELECT
2 Name AS Country,
3 Population,
4 SurfaceArea,
5 ROUND(Population / SurfaceArea, 2) AS PopulationDensity
6 FROM
7 country
8 ORDER BY
9 PopulationDensity
10 LIMIT 10;
11
12

Result Grid Filter Rows: Export: Wrap Cell Contents: Fetch rows:

Country	Population	SurfaceArea	PopulationDensity
Antarctica	0	13120000.00	0.00
Bouvet Island	0	59.00	0.00
South Georgia and the South Sandwich Islands	0	3903.00	0.00
British Indian Ocean Territory	0	78.00	0.00
Heard Island and McDonald Islands	0	359.00	0.00
United States Minor Outlying Islands	0	16.00	0.00
French Southern territories	0	7780.00	0.00
Greenland	56000	2166090.00	0.03
Svalbard and Jan Mayen	3200	62422.00	0.05
Falkland Islands	2000	12173.00	0.16

Result 19 ×

Output

Action Output

#	Time	Action	Message
1	12:39:15	SELECT country.Name AS Country, city.Name AS CapitalCity, city.Population AS CapitalPopulation FROM country JOIN city ON country.Capital = city.ID ORDER BY city.Population DESC LIMIT 0, 1000	232 row(s) returned
2	12:41:51	SELECT * FROM country ORDER BY city.Population LIMIT 0, 1000	Error Code: 1054. Unknown column 'city.Population' in 'order clause'
3	12:41:59	SELECT * FROM country ORDER BY Population LIMIT 0, 1000	239 row(s) returned
4	12:42:31	SELECT * FROM country WHERE Population = 0 ORDER BY Population LIMIT 0, 1000	7 row(s) returned
5	12:43:30	SELECT Name AS Country, Population, SurfaceArea, ROUND(Population / SurfaceArea, 2) AS PopulationDensity FROM country ORDER BY PopulationDensity LIMIT 0, 1000	239 row(s) returned
6	12:44:15	SELECT Name AS Country, Population, SurfaceArea, ROUND(Population / SurfaceArea, 2) AS PopulationDensity FROM country ORDER BY PopulationDensity LIMIT 10	10 row(s) returned

18. **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
ci.Name AS City,
co.Name AS Country,
ci.Population AS CityPopulation,
co.GNP AS CountryGNP,
co.Population AS CountryPopulation,
ROUND((co.GNP / co.Population) * ci.Population / ci.Population, 2) AS GDPPerCapita
FROM
world.city ci
JOIN
country co ON ci.CountryCode = co.Code
WHERE
co.GNP IS NOT NULL AND co.Population > 0
AND ((co.GNP / co.Population) * ci.Population / ci.Population) >
(
SELECT
AVG(co.GNP / co.Population)
FROM
country co
WHERE
co.GNP IS NOT NULL AND co.Population > 0
)
ORDER BY
GDPPerCapita DESC;
```


19. **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.

```
SELECT
    ci.Name AS City,
    co.Name AS Country,
    ci.Population
FROM
    world.city ci
JOIN
    world.country co
ON
    ci.CountryCode = co.Code
ORDER BY
    ci.Population
LIMIT 10 OFFSET 30
```

