



SQL Assignment Part 2

using the 'World database' script

Tamima Tarofdar

Data Technician Skills Bootcamp - Just IT (April 2024)



Task 1

- - - - X

Using count, get the number of cities in the USA

Answer: 3998

Within this code, the 'USE world' line specifies that I would like to use the 'world' database. This means that all queries from now on will be executed within this database.

The 'SELECT' statement is used to retrieve data from the database and the 'COUNT(DISTINCT Name)' function counts the number of unique values in the 'Name' column of the 'city' table.

The DISTINCT keyword ensures that each city name is counted only once and 'as 'No of cities in the USA'' operation is renaming the result of the count function to 'No of cities in the USA' so the viewer can read and understand the results better.

The screenshot shows the MySQL Workbench interface. The title bar says "MySQL Workbench". The main window has tabs for "Administration", "Schemas", "MySQL Assignment Part 1 (TT)", "create-databases", "world db", and "MySQL Assignment Part 2*". The "Schemas" tab is selected, showing the "world_database" schema with "Tables", "Views", "Stored Procedur...", and "Functions" listed. The "MySQL Assignment Part 1 (TT)" tab contains the following SQL code:

```
1 #Task_1
2 #Using count, get the number of cities in the USA
3 • USE world;
4 • SELECT COUNT(DISTINCT Name) as 'No of cities in the USA'
5 FROM city
6 :
```

The "Result Grid" below shows the output:

No of cities in the USA
3998



Task 2

- - - - X

Find out what the population and average life expectancy for people in Argentina (ARG) is.

Answers:

Population = 37032000

Life Expectancy = 75.1

Part 1 - Population

To tackle this task I decided to split it into two parts. In part 1 I used the 'SELECT' statement to select the 'population' column and the 'FROM' statement to specify that this data should be retrieved from the 'Country' table.

I then used the 'WHERE Code = 'ARG'' function to apply a condition so the results are filtered to only include rows where the 'Code' is equal to ARG, which is the country code for Argentina in this case.

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
8 #Task_2 - Find out what the population and average life expectancy for people in Argentina (ARG) is.
9 #Finding population
10 • USE world;
11 • SELECT population
12 FROM Country
13 WHERE Code = 'ARG';
14
```

The result grid below shows the output:

population
37032000



Part 2 - Life Expectancy

In part 2 I used a similar code to the previous one but instead used the ‘SELECT’ statement to select the ‘LifeExpectancy’ column.

I then used the same code from the previous part where the ‘FROM’ statement specifies that this data should be retrieved from the ‘Country’ table and the ‘WHERE Code = ‘ARG’ to view the results specific to Argentina.

The screenshot shows the MySQL Workbench interface. The SQL editor window contains the following code:

```
14
15 #Finding average life expectancy
16 • SELECT LifeExpectancy
17 FROM Country
18 WHERE Code = 'ARG';
19
20
21
```

The result grid below shows a single row of data:

LifeExpectancy
75.1



Task 3

- - - - X

Using ORDER BY, LIMIT, what country has the highest life expectancy?

Answer: Andorra

For this task, I used the asterisk (*) symbol with the 'SELECT' statement to select all the columns from the 'Country' table.

I used the 'ORDER BY' statement to organise results based on the 'LifeExpectancy' column. The 'DESC' statement was added for the column to be sorted in descending order so the countries with the highest life expectancy will be listed first in the results.

I then used the 'LIMIT 1' function for the query to return 1 row only and since the table result is now organised by life expectancy descending, this meant that only the row with the country with the highest life expectancy will be shown.

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
21  #Task_3
22  #Using ORDER BY, LIMIT, what country has the highest life expectancy?
23  •  SELECT *
24  FROM Country
25  ORDER BY LifeExpectancy DESC
26  LIMIT 1;
27
28
```

The results grid below shows the output of the query:

Code	Name	Continent	Region	SurfaceArea	IndepYear	Population	LifeExpectancy	GNP	GNPQd	LocalNa
AND	Andorra	Europe	Southern Europe	468.00	1278	78000	83.5	1630.00	HULL	Andorra



Task 4

- - - - X

Select 25 cities around the world that start with the letter 'F' in a single SQL query.

For Task 4 I have used the 'SELECT' statement to select the 'Name' column and the 'FROM' statement to specify this is to be retrieved from the 'City' table.

I used the 'WHERE' statement to apply a condition using the 'LIKE 'F%'' function to instruct the query to return only cities which begin with the letter 'F', followed by zero or more characters. The 'LIMIT 25' line is then used to limit the number of rows returned by the query to 25.

The screenshot shows the MySQL Workbench interface with the following details:

- Toolbar:** Standard MySQL Workbench icons for file operations, database management, and connection status.
- Left Panel (Schemas):** Shows the current schema is "world_database". Other schemas listed include "new_schema" and "sys".
- Central Area (Query Editor):** A code editor window titled "MySQL Assignment Part 1 (TT)". It contains the following SQL query:

```
29 #Task_4
30 #Select 25 cities around the world that start with the letter 'F' in a single SQL query
31 • SELECT Name
32 FROM City
33 WHERE Name LIKE 'F%'
34 LIMIT 25;
```
- Result Grid:** A table showing the results of the query. The column is labeled "Name" and contains 25 entries starting with 'F'.

Name
Fagatogo
Florencio Varela
Formosa
Francistown
Fortaleza
Feira de Santana
Franca
Florianópolis
Foz do Iguaçu
Ferraz de Vasconcelos
Francisco Morato
Franco da Rocha
Fuenlabrada
Faridabad
Firozabad
Farrukhabad-cum-Fa...
Faizabad
Fatehpur
Firenze
Foggia
Ferrara
Forlì
Fukuoka
- Right Panel (Panels):** A vertical panel on the right containing several tabs: "Result Grid" (selected), "Form Editor", "Field Types", "Query Stats", and "Execution Plan".



Task 5

- - - - X

Part a)

Create an SQL statement to display columns Id, Name, Population from the city table and limit results to first 10 rows only.

For part a) I used the 'SELECT' statement to select the 'ID', 'Name', and 'Population' columns and the 'FROM' statement to specify this is to be retrieved from the 'City' table. I then used the 'LIMIT 10' statement so that only the first 10 rows are shown in the results.

The screenshot shows the MySQL Workbench interface. The top navigation bar includes 'Local instance 3306', 'Administration', 'Schemas', and tabs for 'MySQL Assignment Part 1 (TT)', 'create-databases', 'world db', and 'MySQL Assignment Part 2*'. The 'Schemas' panel on the left lists 'new_schema', 'sys', and 'world_database'. The 'Tables' section is selected and shows 'Task_5a' with the following SQL code:

```
37 #Task_5a
38 #Create a SQL statement to display columns ID, Name, Population from the city table and limit results to first 10 rows only
39 • SELECT ID, Name, Population
40 FROM City
41 Limit 10;
42
43 #Task_5b
```

The 'Result Grid' below displays the query results:

ID	Name	Population
1	Kabul	1780000
2	Kandahar	237500
3	Herat	186800
4	Mazar-e-Sharif	127800
5	Amsterdam	731200
6	Rotterdam	593321
7	Haag	440900
8	Utrecht	234323
9	Eindhoven	201843
10	Tilburg	193238
NULL	NULL	NULL



Part b)

Create an SQL statement to display columns Id, Name, Population from the city table and limit results to rows 31-40.

For part b) I used the same code in part a) but changed the 'LIMIT 10' line to 'LIMIT 30, 10'. As rows in SQL are numbered from zero I had to consider that the 31st row has an offset of 30. Therefore 'LIMIT 30,10' specifies that the results should start from the 31st row (offset 30) and return 10 rows.

The screenshot shows the MySQL Workbench interface. The SQL editor pane contains the following code:

```
45 #Task_5b
46 # Create an SQL statement to display columns Id, Name, Population from the city table and limit results to rows 31-40.
47 • SELECT ID, Name, Population
48 FROM City
49 Limit 30, 10;
50
```

The result grid pane displays the following data:

ID	Name	Population
31	Heerlen	95052
32	Aalkmaar	92713
33	Willemstad	2345
34	Tirana	270000
35	Alger	2168000
36	Oran	609823
37	Constantine	443727
38	Annaba	222518
39	Batna	183377
40	Sétif	179055
HULL	HULL	HULL



Task 6

- - - - X

Create an SQL statement to find only those cities from city table whose population is larger than 2000000.

For this task, I used the asterisk (*) symbol with the 'SELECT' statement to select all the columns from the 'City' table.

I then used the 'WHERE Population > 2000000' clause to apply a condition so the results are filtered to only include rows where cities with populations exceeding 2,000,000 are shown in the results.

The screenshot shows the MySQL Workbench interface. The query editor window contains the following SQL code:

```
42
43 #Task_6
44 #Create a SQL Statement to find only those cities from city table whose population is larger than 2000000
45 • SELECT *
  From City
46 WHERE Population > 2000000;
47
```

The result grid displays a table with the following data:

ID	Name	CountryCode	District	Population
35	Alger	DZA	Alger	2168000
56	Luanda	AGO	Luanda	2022000
69	Buenos Aires	ARG	Distrito Federal	2982146
130	Sydney	AUS	New South Wales	3276207
131	Melbourne	AUS	Victoria	2865329
150	Dhaka	BGD	Dhaka	3612850
206	São Paulo	BRA	São Paulo	9968485
207	Rio de Janeiro	BRA	Rio de Janeiro	5598953
208	Salvador	BRA	Bahia	2302832
209	Belo Horizonte	BRA	Minas Gerais	2139125
210	Fortaleza	BRA	Ceará	2097757
456	London	GBR	England	7285000
554	Santiago de...	CHL	Santiago	4703954
593	Guayaquil	ECU	Guayas	2070040
608	Cairo	EGY	Kairo	6789479
609	Alexandria	EGY	Aleksandria	3328196
610	Giza	EGY	Giza	2221868
653	Madrid	ESP	Madrid	2879052
712	Cape Town	ZAF	Western Cape	2352121
756	Addis Abeba	ETH	Addis Abeba	2495000
765	Quezon	PHL	National Capital...	2173831
939	Jakarta	IDN	Jakarta Raya	9604900
940	Surabaya	IDN	East Java	2663820
941	Bandung	IDN	West Java	2429000
1024	Mumbai (Bo...)	IND	Maharashtra	10500000
1025	Delhi	IND	Delhi	7206704
1026	Calcutta [Kol...]	IND	West Bengal	4399819
1027	Chennai (Ma...)	IND	Tamil Nadu	3841396
1028	Hyderabad	IND	Andhra Pradesh	2964638
1029	Ahmedabad	IND	Gujarat	2876710



Task 7

- - - - X

Create an SQL statement to find all city names from city table whose name begins with Be prefix.

For Task 7 I have used the ‘SELECT’ statement to select the ‘Name’ column and the ‘FROM’ statement to specify this is to be retrieved from the ‘City’ table.

I used the ‘WHERE’ statement to apply a condition using the ‘LIKE ‘Be%’’ function to instruct the query to return only cities whose names start with ‘Be’.

The screenshot shows the MySQL Workbench interface. The SQL editor tab contains the following code:

```
55  #Task_7
56  #Create a SQL statement to find all city names from city table whose name begins with "Be" prefix
57  •  SELECT Name
58  FROM City
59  WHERE Name LIKE 'Be%';
```

The results grid below shows a list of city names starting with 'Be':

Name
Béjaïa
Béchar
Benguela
Berazategui
Belize City
Belmopan
Belo Horizonte
Belém
Belford Roxo
Betim
Bento Gonçal...
Belfast
Benoni
Bekasi
Bengkulu
Belgaum
Bellary
Berhampore...
Beawar
Bettiah
Beerseba
Bene Beraq
Bergamo
Beira

At the bottom of the results grid, it says 'City 26' and 'Read Only'.



Task 8

- - - - X

Create an SQL statement to find only those cities from city table whose population is between 500000-1000000.

For this task, I used the asterisk (*) symbol with the 'SELECT' statement to select all the columns from the 'City' table.

I then used the 'WHERE Population BETWEEN 50000 AND 1000000' clause to apply a condition so the results are filtered to only include cities where the data in the population column is inclusive of and falls between 50,000 and 1,000,000.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Administration' tab, 'SCHEMAS' section with 'new_schema' and 'world_database' selected, and the 'Tables' section. The main area is titled 'MySQL Assignment Part 1 (TT)' and contains the following SQL code:

```
61 #Task_8
62 #Create a SQL statement to find only those cities from city table whose population is between 50000- 1000000.
63 • SELECT *
64   From City
65 WHERE Population BETWEEN 50000 AND 1000000;
66
```

The 'Result Grid' below shows the query results:

ID	Name	CountryCode	District	Popul...
2	Qandahar	AFG	Qandahar	237500
3	Herat	AFG	Herat	186800
4	Mazar-e-Sharif	AFG	Balkh	127800
5	Amsterdam	NLD	Noord-Holland	731200
6	Rotterdam	NLD	Zuid-Holland	502321
7	Haag	NLD	Zuid-Holland	440900
8	Utrecht	NLD	Utrecht	234323
9	Eindhoven	NLD	Noord-Brabant	201843
10	Tilburg	NLD	Noord-Brabant	193238
11	Groningen	NLD	Groningen	172701
12	Breda	NLD	Noord-Brabant	163988
13	Apeldoorn	NLD	Gelderland	153491
14	Nijmegen	NLD	Gelderland	152483
15	Enschede	NLD	Overijssel	149544
16	Haarlem	NLD	Noord-Holland	148772
17	Almere	NLD	Flevoland	142465
18	Arnhem	NLD	Gelderland	138020
19	Zaanstad	NLD	Noord-Holland	135621
20	's-Hertogenb...	NLD	Noord-Brabant	129170
21	Amersfoort	NLD	Utrecht	126270
22	Maastricht	NLD	Limburg	122087
23	Dordrecht	NLD	Zuid-Holland	119811
24	Leiden	NLD	Zuid-Holland	117196
25	Haarlemmer	NL D	Noord-Holland	117799



Task 9

- - - - X

Create an SQL statement to display all cities from the city table sorted by Name in ascending order.

For this task, I used the asterisk (*) symbol with the ‘SELECT’ statement to select all the columns from the ‘City’ table.

I then used the ‘ORDER BY’ statement to organise results based on the ‘Name’ column. The ‘ASC’ statement enables the column to be sorted in ascending order so information about all cities in the ‘City’ table is arranged alphabetically by the city name when the query is run.

The screenshot shows the MySQL Workbench interface. The SQL editor window contains the following code:

```
67  #Task_9 Create a SQL statement to display all cities from the city table sorted by Name in ascending order.
68 •  SELECT *
69   FROM City
70   ORDER BY Name ASC;
71
```

The Result Grid displays the following data:

ID	Name	CountryCode	District	Population
698	[San Cristóbal de] la Laguna	ESP	Canary Islands	127945
20	's-Hertogenbosch	NLD	Noord-Brabant	129170
670	A Coruña (La Coruña)	ESP	Galicia	243402
3097	Aachen	DEU	Nordrhein-Westfalen	243825
3318	Aalborg	DNK	Nordjylland	161161
2760	Aba	NGA	Imo & Abia	298900
1404	Abadan	IRN	Khuzestan	206073
395	Abaetetuba	BRA	Pará	111258
3683	Abakan	RUS	Hakassia	169200
1849	Abbotsford	CAN	British Columbia	105403
2747	Abeokuta	NGA	Ogun	427400
478	Aberdeen	GBR	Scotland	213070
3191	Abha	SAU	Asir	112300
2812	Abidjan	CIV	Abidjan	2500000
1703	Abiko	JPN	Chiba	126670
3989	Abilene	USA	Texas	115930
1309	Abohar	IND	Punjab	107163
2866	Abottabad	PAK	Northwest Border P...	106000
65	Abu Dhabi	ARE	Abu Dhabi	398695
2754	Abuja	NGA	Federal Capital Dist	350100
2655	Acámbaro	MEX	Guanajuato	110487
2528	Acapulco de Juárez	MEX	Guerrero	721011
3561	Acarigua	VEN	Portuguesa	158954
910	Accra	GHA	Greater Accra	1070000
1344	Achalpur	IND	Maharashtra	96216
2069	Archenh	CHN	Heilongjiang	197595



Task 10

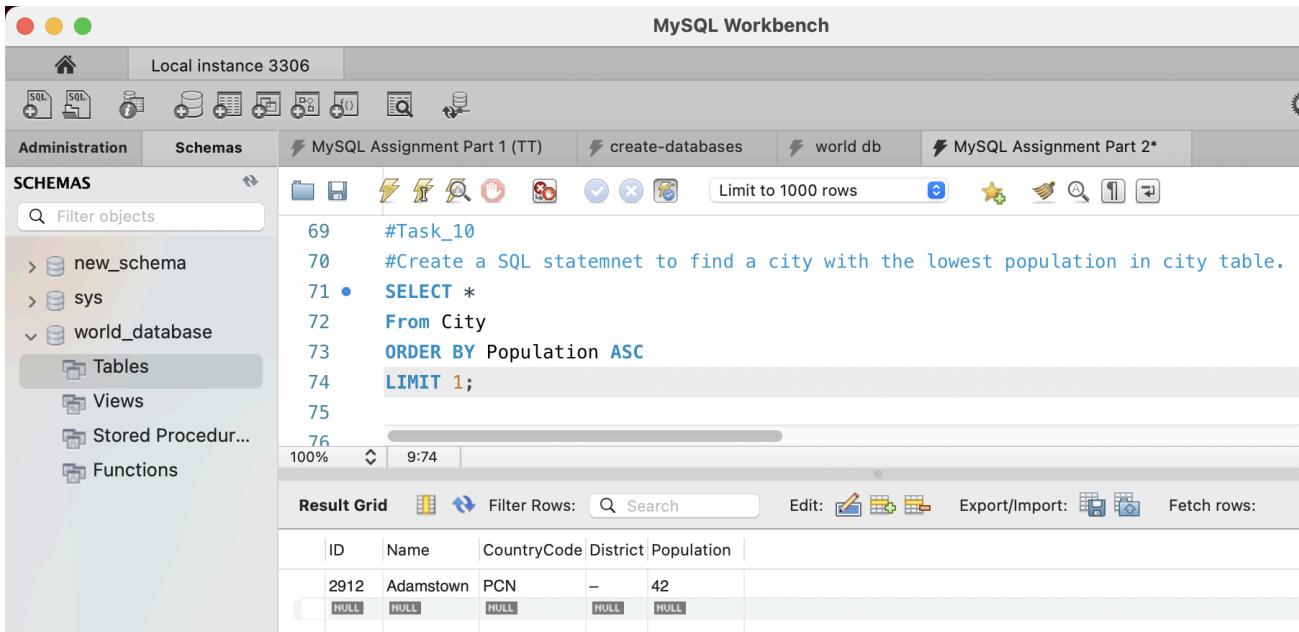
- - - - X

Create an SQL statement to find a city with the lowest population in the city table.

For task 10, I used the asterisk (*) symbol with the 'SELECT' statement to select all the columns from the 'City' table.

I used the 'ORDER BY' statement to organise results based on the 'Population' column. The 'ASC' statement enables the column to be sorted in ascending order so cities with the lowest population will be listed first in the results.

I then used the 'LIMIT 1' function for the query to select only the first row, which now represents the city with the lowest population.



The screenshot shows the MySQL Workbench interface. The SQL tab contains the following code:

```
69  #Task_10
70  #Create a SQL statemnet to find a city with the lowest population in city table.
71 • SELECT *
72   From City
73   ORDER BY Population ASC
74   LIMIT 1;
75
76
```

The result grid below shows one row of data:

ID	Name	CountryCode	District	Population
2912	Adamstown	PCN	-	42



Task 11

- - - - X

Create an SQL statement to find a country with the largest population in the country table.

For task 11, I used the asterisk (*) symbol with the ‘SELECT’ statement to select all the columns from the ‘Country’ table.

I used the ‘ORDER BY’ statement to organise results based on the ‘Population’ column. The ‘DESC’ statement enables the column to be sorted in descending order so the country with the highest population will be listed first in the results.

I then used the ‘LIMIT 1’ function for the query to show only the first row, which now represents the country with the largest population.

The screenshot shows the MySQL Workbench interface. The top bar displays 'MySQL Workbench' and 'Local instance 3306'. The left sidebar shows the 'Administration' tab selected, with 'SCHEMAS' expanded to show 'new_schema', 'sys', and 'world_database'. Under 'world_database', 'Tables' is selected, showing 'Country' as the current table. The main pane contains the following SQL code:

```
80  #Task_11
81  #Create a SQL statement to find a country with the largest population in the country table.
82  • SELECT *
83  FROM Country
84  ORDER BY Population DESC
85  LIMIT 1;
86
```

The 'Result Grid' tab is active, displaying the results of the query. The table has the following columns: Code, Name, Continent, Region, SurfaceArea, IndepYear, Population, LifeExpectancy, GNP, GNPOld, Loca. The single row returned is:

Code	Name	Continent	Region	SurfaceArea	IndepYear	Population	LifeExpectancy	GNP	GNPOld	Loca
CHN	China	Asia	Eastern Asia	9572900.00	-1523	1277558000	71.4	982268.00	917719.00	Zhon



Bonus Tasks

- - - - X

Task 1

Create SQL statement to find the capital of Spain (ESP)

Answer: Madrid

Within this query, the first line selects the 'Name' column from the 'City' table, and using the AS statement I renamed it "Capital" to improve understanding and readability. This means that the selected column will be displayed as "Capital" when the query is run.

The JOIN statement uses the 'ID' column from the 'City' table and the 'Capital' column from the 'Country' table to join the two tables together. This ensures that only the capital city of each country is retrieved.

The WHERE statement filters the results to only show the country with the country code 'ESP', which represents Spain.

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
95 # ***** Bonus tasks *****
96 # Task 1 - Create SQL statement to find the capital of Spain (ESP)
97 • SELECT City.Name AS Capital
98   FROM City
99   JOIN Country ON City.ID = Country.Capital
100  WHERE Country.Code = 'ESP';
101
```

The result grid shows one row with the capital 'Madrid'.



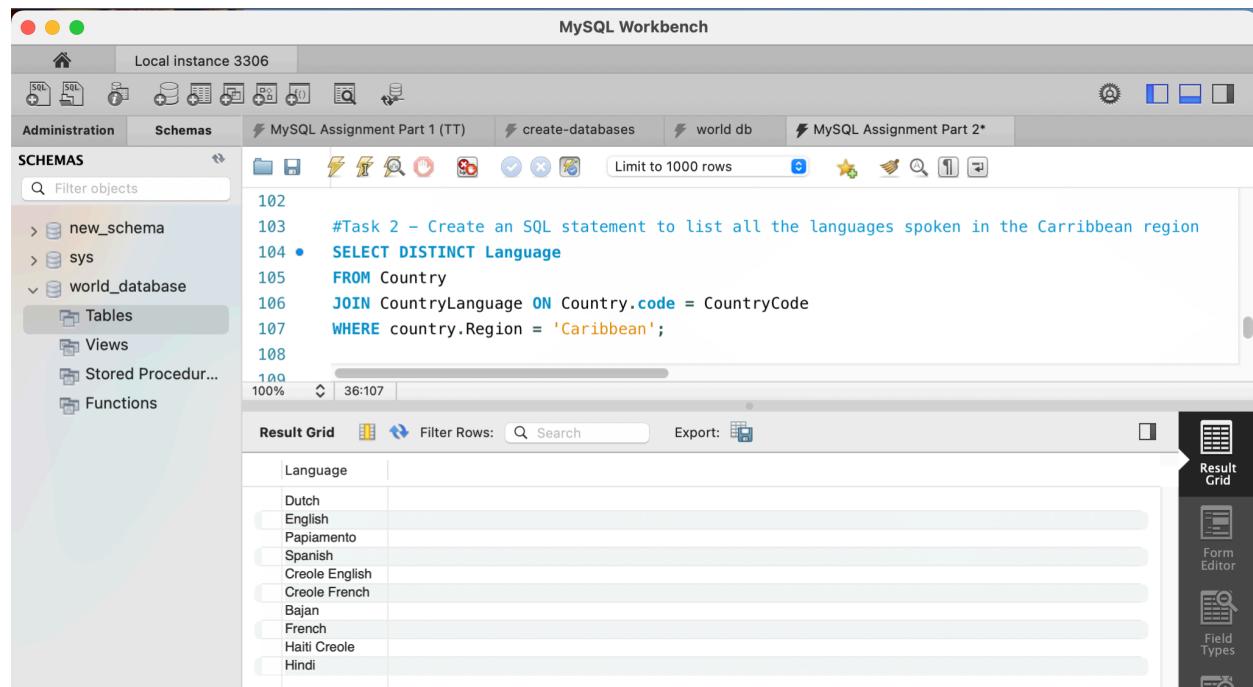
Task 2

Create an SQL statement to list all the languages spoken in the Caribbean region

For this task, the ‘Language’ column is selected from the ‘Country’ table. The DISTINCT statement was added to make sure each language is only included once in the result set.

The JOIN statement uses the ‘code’ column from the ‘Country’ table and the ‘CountryCode’ column from the ‘CountryLanguage’ table to join the two tables together. This ensures that language data is retrieved for each country.

The WHERE statement filters the results to only show countries located in the Caribbean region by applying the condition country.Region = ‘Caribbean’.



The screenshot shows the MySQL Workbench interface. The SQL tab contains the following code:

```
102
103  #Task 2 - Create an SQL statement to list all the languages spoken in the Caribbean region
104 •  SELECT DISTINCT Language
105   FROM Country
106   JOIN CountryLanguage ON Country.code = CountryCode
107   WHERE country.Region = 'Caribbean';
108
109
```

The Result Grid shows the following data:

Language
Dutch
English
Papiamento
Spanish
Creole English
Creole French
Bajan
French
Haiti Creole
Hindi



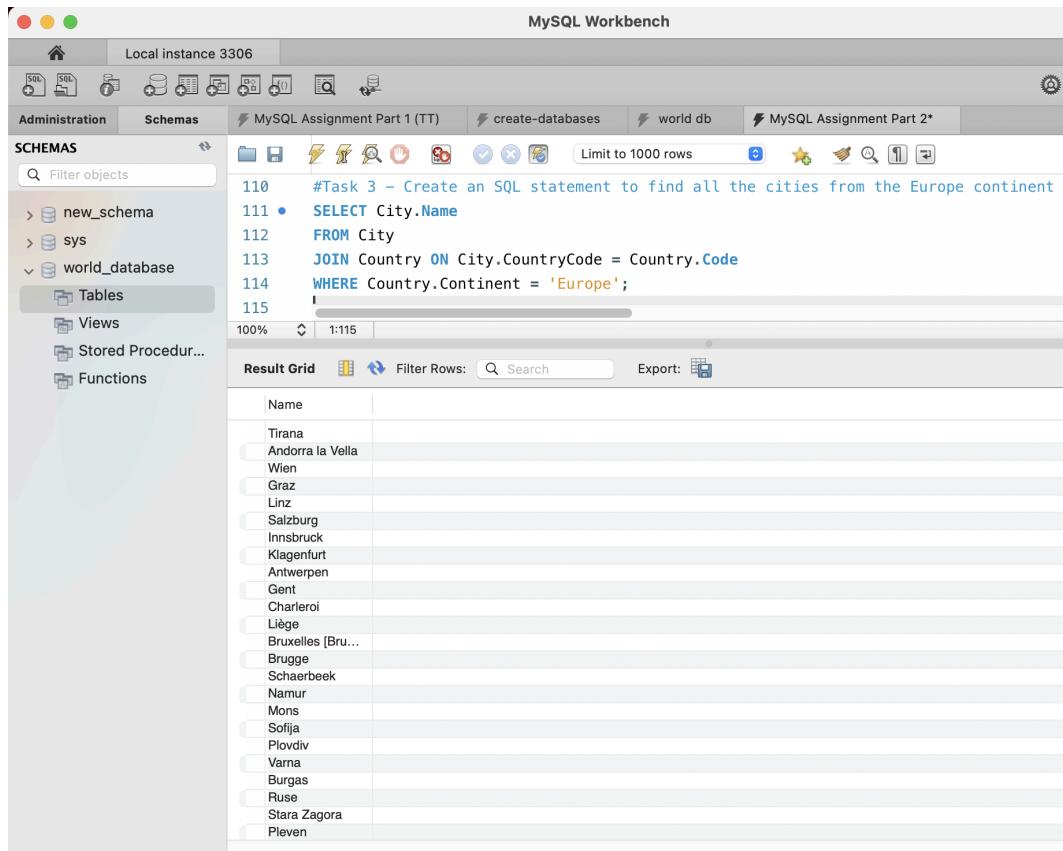
Task 3

Create an SQL statement to find all the cities from the Europe continent

For this bonus task, I have used the ‘SELECT’ statement to select the ‘City.Name’ column and the FROM statement to specify this is to be retrieved from the ‘City’ table.

The JOIN statement uses the ‘CountryCode’ column from the ‘City’ table and the ‘Code’ column from the ‘Country’ table to join the two tables together. This ensures that city data is retrieved for each country.

The WHERE statement here filters the results to only include countries located in the European continent by specifying the condition Country.Continent = ‘Europe’.



The screenshot shows the MySQL Workbench interface. The SQL editor pane contains the following SQL code:

```
110  #Task 3 – Create an SQL statement to find all the cities from the Europe continent
111 •  SELECT City.Name
112   FROM City
113   JOIN Country ON City.CountryCode = Country.Code
114   WHERE Country.Continent = 'Europe';
115  
```

The Result Grid pane displays the names of European cities:

Name
Tirana
Andorra la Vella
Wien
Graz
Linz
Salzburg
Innsbruck
Klagenfurt
Antwerpen
Gent
Charleroi
Liège
Bruxelles [Bru...
Brugge
Schaerbeek
Namur
Mons
Sofija
Plovdiv
Varna
Burgas
Ruse
Stara Zagora
Pleven



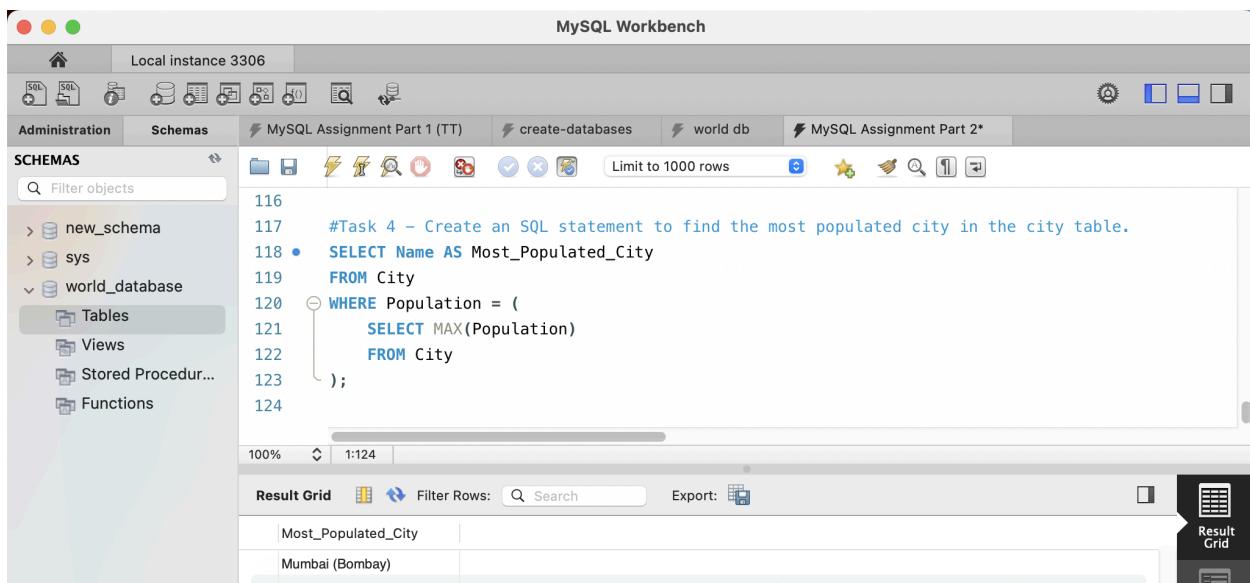
Task 4

Create an SQL statement to find the most populated city in the city table.

Answer: Mumbai (Bombay)

Within this query, the first line selects the 'Name' column from the 'City' table and using the AS statement I renamed it "Most_Populated_City" to improve understanding and readability. This means that the 'Name' column will be displayed as "Most_Populated_City" when the query is run.

The WHERE statement here filters the results to only include rows where the 'Population' column matches the maximum population value in the 'City' table. The subquery (SELECT MAX(Population) FROM City) finds the maximum population value in the 'City' table, and the outer query selects the city with that population.



The screenshot shows the MySQL Workbench interface. The SQL tab contains the following code:

```
116
117  #Task 4 – Create an SQL statement to find the most populated city in the city table.
118 •  SELECT Name AS Most_Populated_City
119   FROM City
120   WHERE Population = (
121       SELECT MAX(Population)
122       FROM City
123   );
124
```

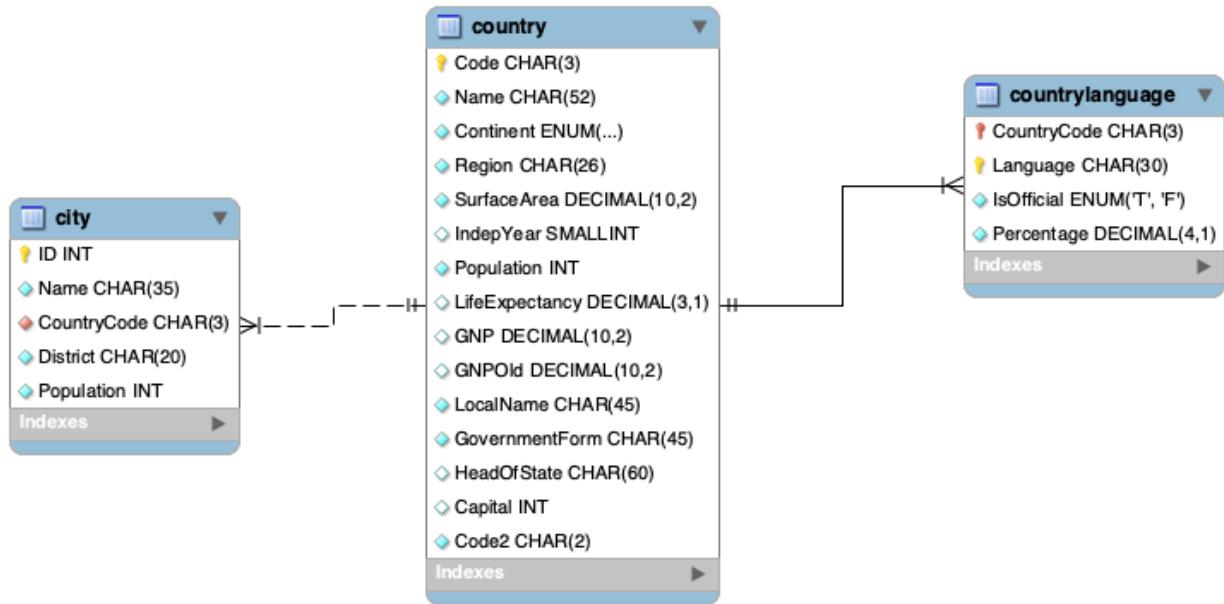
The Results tab shows the output of the query:

Most_Populated_City
Mumbai (Bombay)



EER Diagram

- - - - X



- Identify the primary key in country table: **Code CHAR(3)**
- Identify the primary key in city table: **ID INT**
- Identify the primary key in countrylanguage table: **Language CHAR(30)**
- Identify the foreign key in city table: **CountryCode CHAR(3)**
- Identify the foreign key in countrylanguage table: **CountryCode CHAR(3)**