1. Query a list of CITY and STATE from the STATION table.

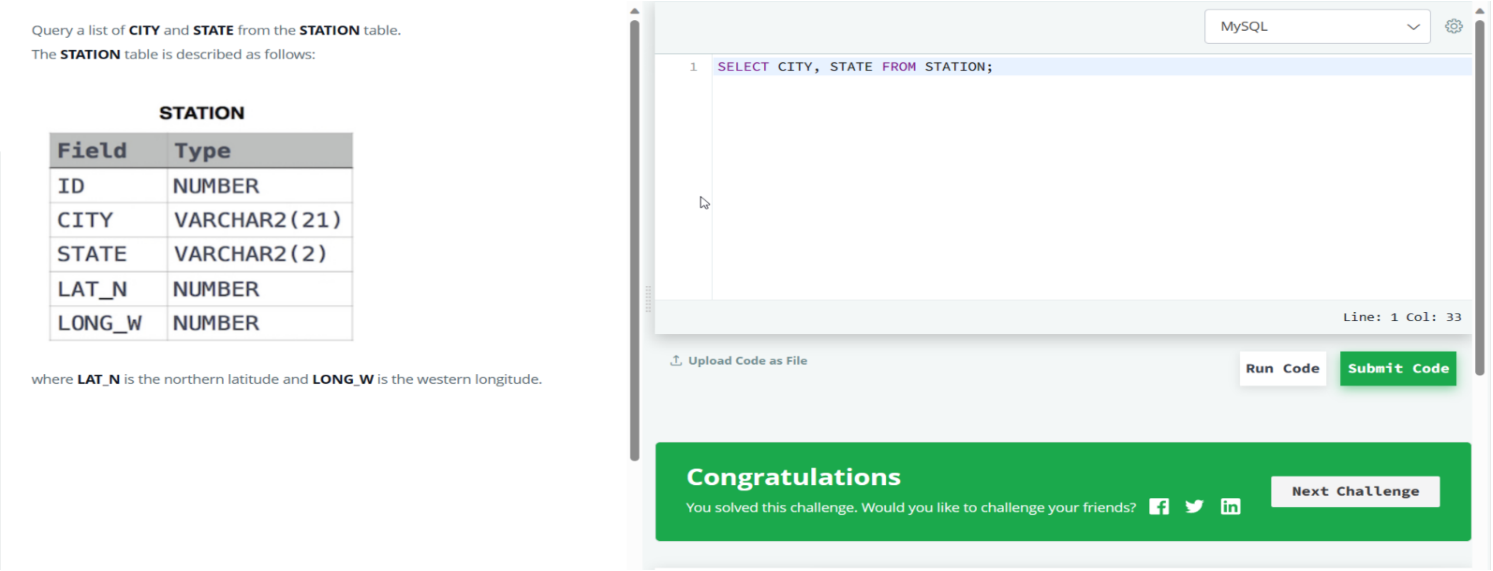
The STATION table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

SQL query: **SELECT CITY, STATE FROME STATION;**



Query a list of **CITY** and **STATE** from the **STATION** table.

The **STATION** table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where **LAT_N** is the northern latitude and **LONG_W** is the western longitude.

```
SELECT CITY, STATE FROM STATION;
```

Congratulations

You solved this challenge. Would you like to challenge your friends?

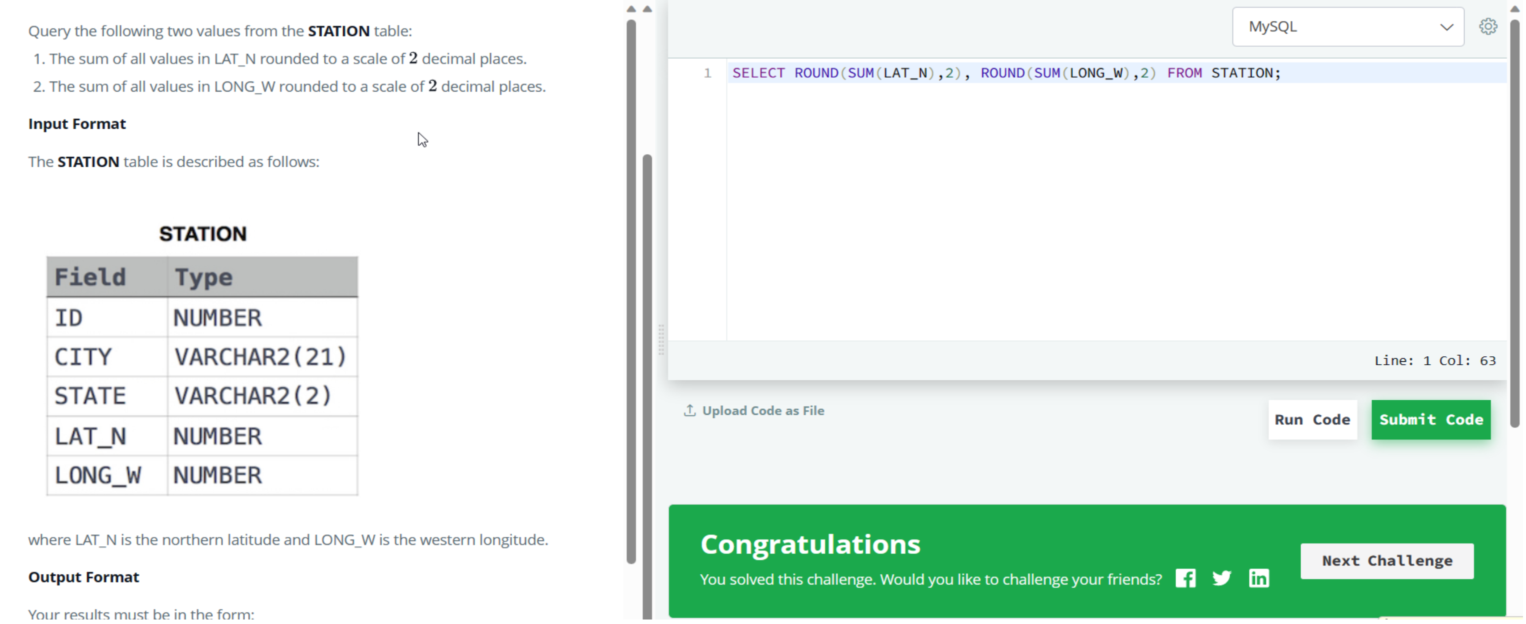
2. Query the following two values from the STATION table:

- The sum of all values in LAT_N rounded to a scale of 2 decimal places.

- The sum of all values in LONG_W rounded to a scale of 2 decimal places.

SQL query:

SELECT ROUND(SUM(LAT_N),2), ROUND(SUM(LONG_2),2) FROM STATION;



Query the following two values from the **STATION** table:

1. The sum of all values in LAT_N rounded to a scale of 2 decimal places.
2. The sum of all values in LONG_W rounded to a scale of 2 decimal places.

Input Format

The **STATION** table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

Output Format

Your results must be in the form:

```
SELECT ROUND(SUM(LAT_N),2), ROUND(SUM(LONG_W),2) FROM STATION;
```

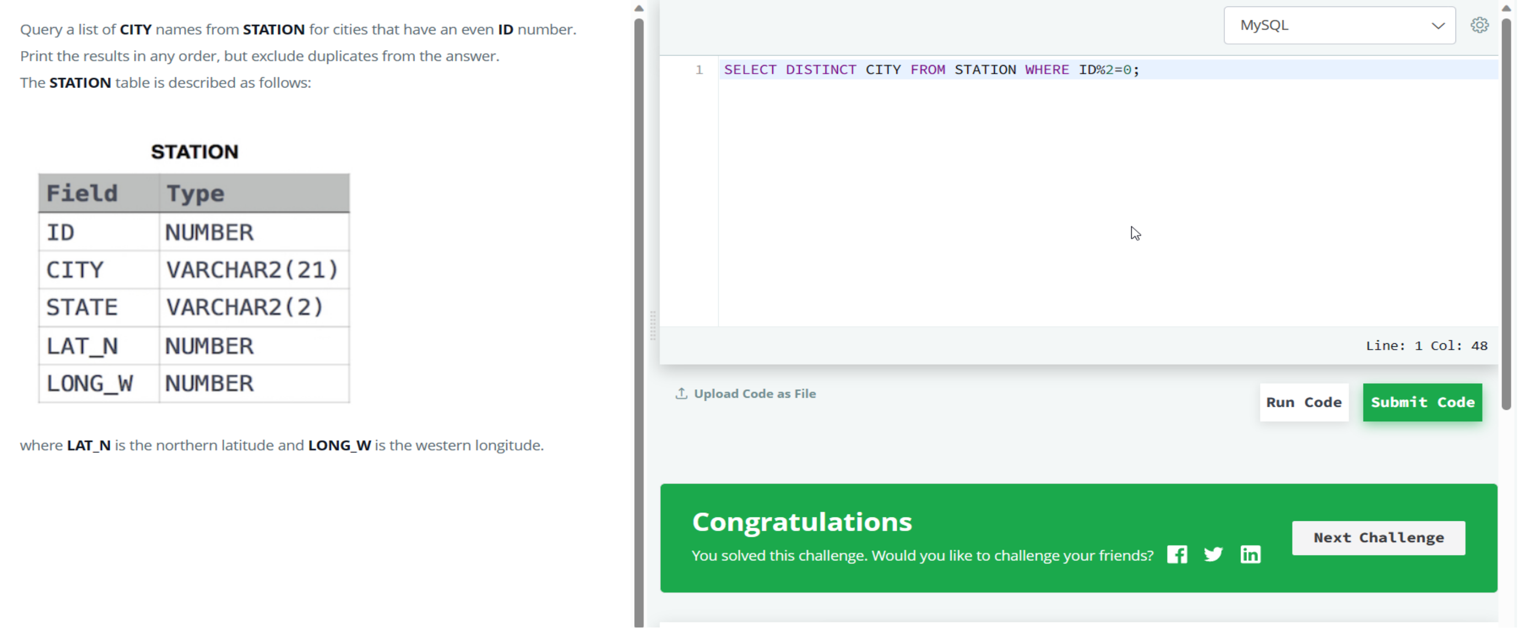
Congratulations

You solved this challenge. Would you like to challenge your friends?

3. Query a list of CITY names from STATION for cities that have an even ID number. Print the results in any order, but exclude duplicates from the answer.

SQL query:

SELECT DISTINCT CITY FROM STATION WHERE ID%2=0;



Query a list of **CITY** names from **STATION** for cities that have an even **ID** number. Print the results in any order, but exclude duplicates from the answer.
The **STATION** table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where **LAT_N** is the northern latitude and **LONG_W** is the western longitude.

```
SELECT DISTINCT CITY FROM STATION WHERE ID%2=0;
```

Congratulations

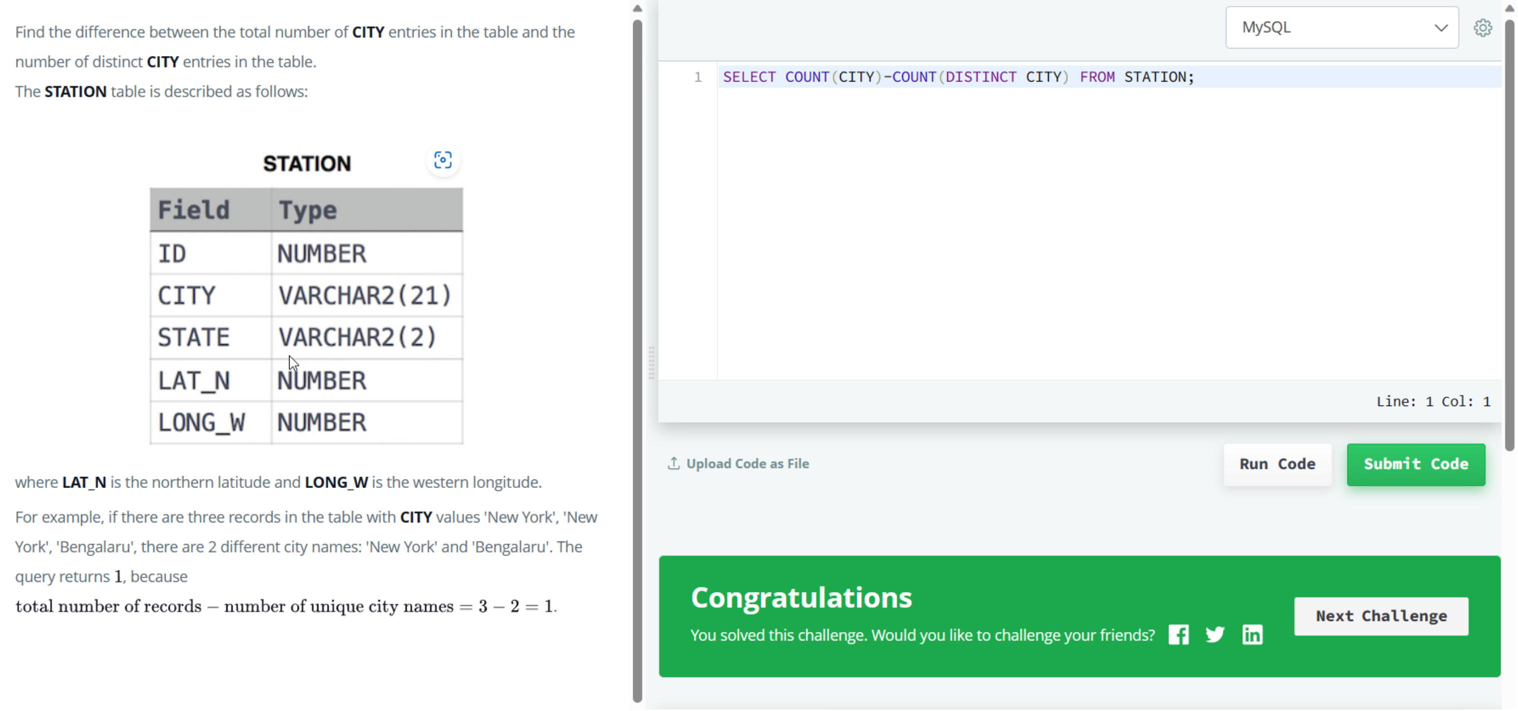
You solved this challenge. Would you like to challenge your friends?

4. Find the difference between the total number of CITY entries in the table and the number of distinct CITY entries in the table.

For example, if there are three records in the table with CITY values 'New York', 'New York', 'Bengalaru', there are 2 different city names: 'New York' and 'Bengalaru'. The query returns 1, because:

Total number of records – number of unique city names = 3 – 2 = 1.

SQL query: **SELECT COUNT(CITY) – COUNT(DISTINCT CITY) FROM STATION;**



Find the difference between the total number of **CITY** entries in the table and the number of distinct **CITY** entries in the table.
The **STATION** table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where **LAT_N** is the northern latitude and **LONG_W** is the western longitude.

For example, if there are three records in the table with **CITY** values 'New York', 'New York', 'Bengalaru', there are 2 different city names: 'New York' and 'Bengalaru'. The query returns 1, because

$\text{total number of records} - \text{number of unique city names} = 3 - 2 = 1.$

```
SELECT COUNT(CITY)-COUNT(DISTINCT CITY) FROM STATION;
```

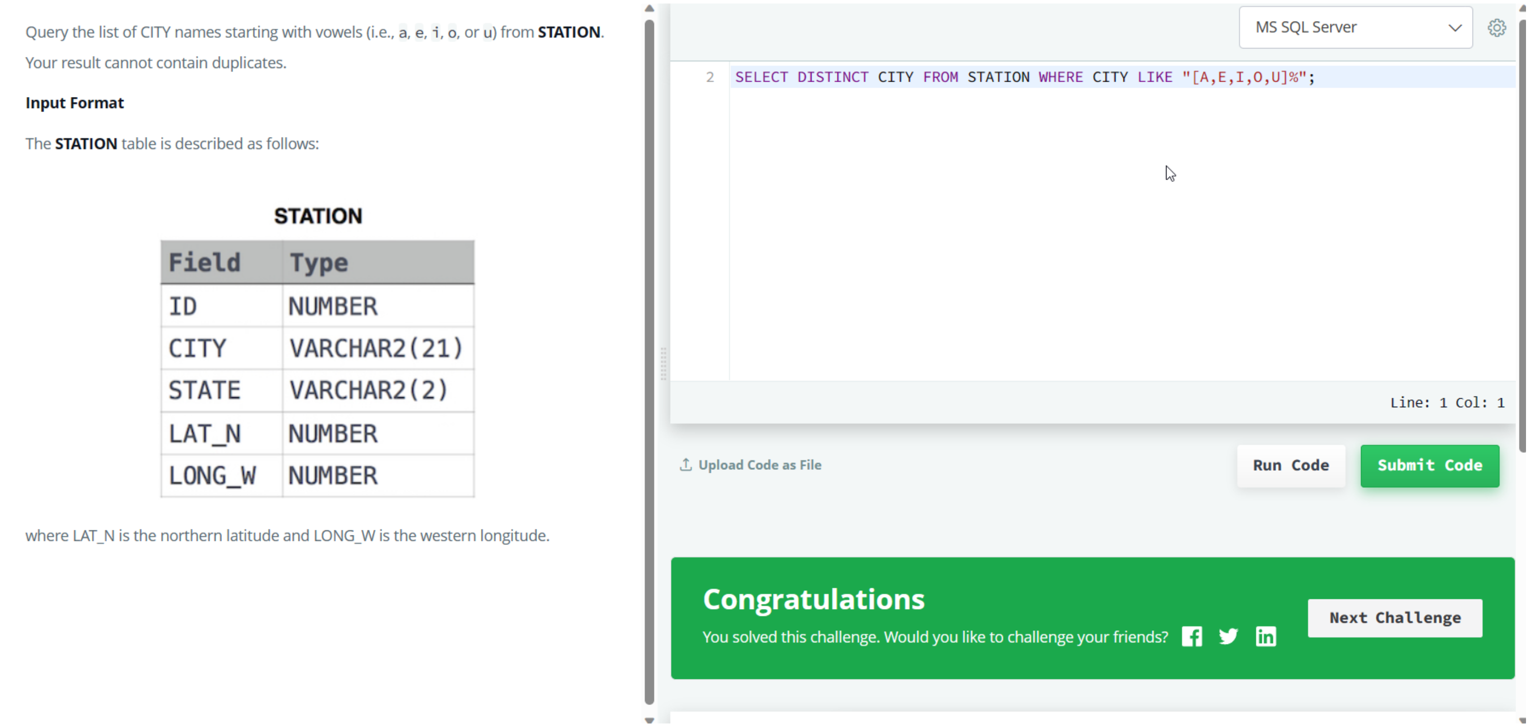
Congratulations

You solved this challenge. Would you like to challenge your friends?

5. Query the list of CITY names starting with vowels (i.e., a, e, i, o, or u) from STATION. Your result cannot contain duplicates.

SQL query:

SELECT DISTINCT CITY FROM STATION WHERE CITY LIKE “[A,E,I,O,U]%”;



Query the list of CITY names starting with vowels (i.e., a, e, i, o, or u) from **STATION**. Your result cannot contain duplicates.

Input Format

The **STATION** table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

MS SQL Server

```
SELECT DISTINCT CITY FROM STATION WHERE CITY LIKE "[A,E,I,O,U]%";
```

Line: 1 Col: 1

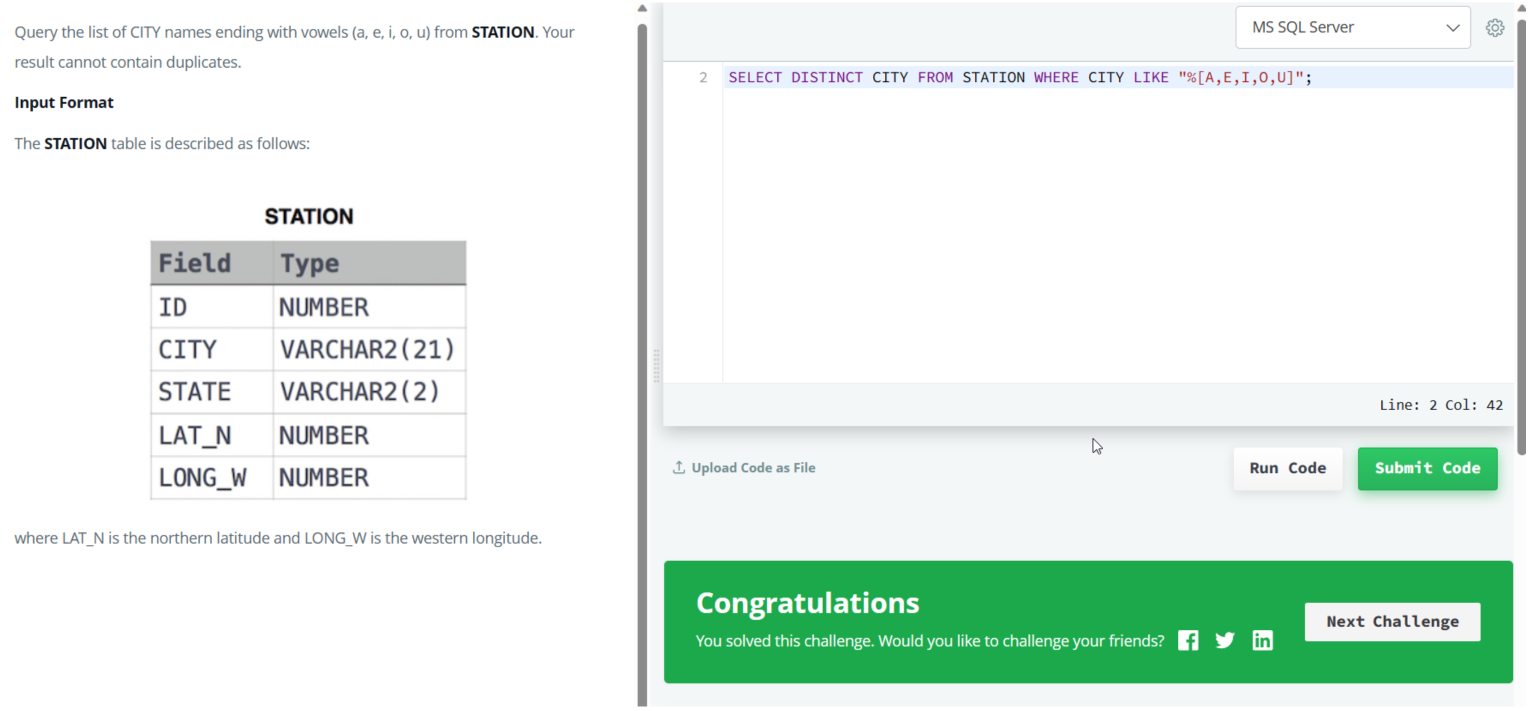
Congratulations

You solved this challenge. Would you like to challenge your friends?

6. Query the list of CITY names ending with vowels (a, e, i, o, u) from STATION. Your result cannot contain duplicates.

SQL query:

SELECT DISTINCT CITY FROM STATION WHERE CITY LIKE "%[A,E,I,O,U]";



Query the list of CITY names ending with vowels (a, e, i, o, u) from **STATION**. Your result cannot contain duplicates.

Input Format

The **STATION** table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

MS SQL Server

```
SELECT DISTINCT CITY FROM STATION WHERE CITY LIKE "%[A,E,I,O,U]";
```

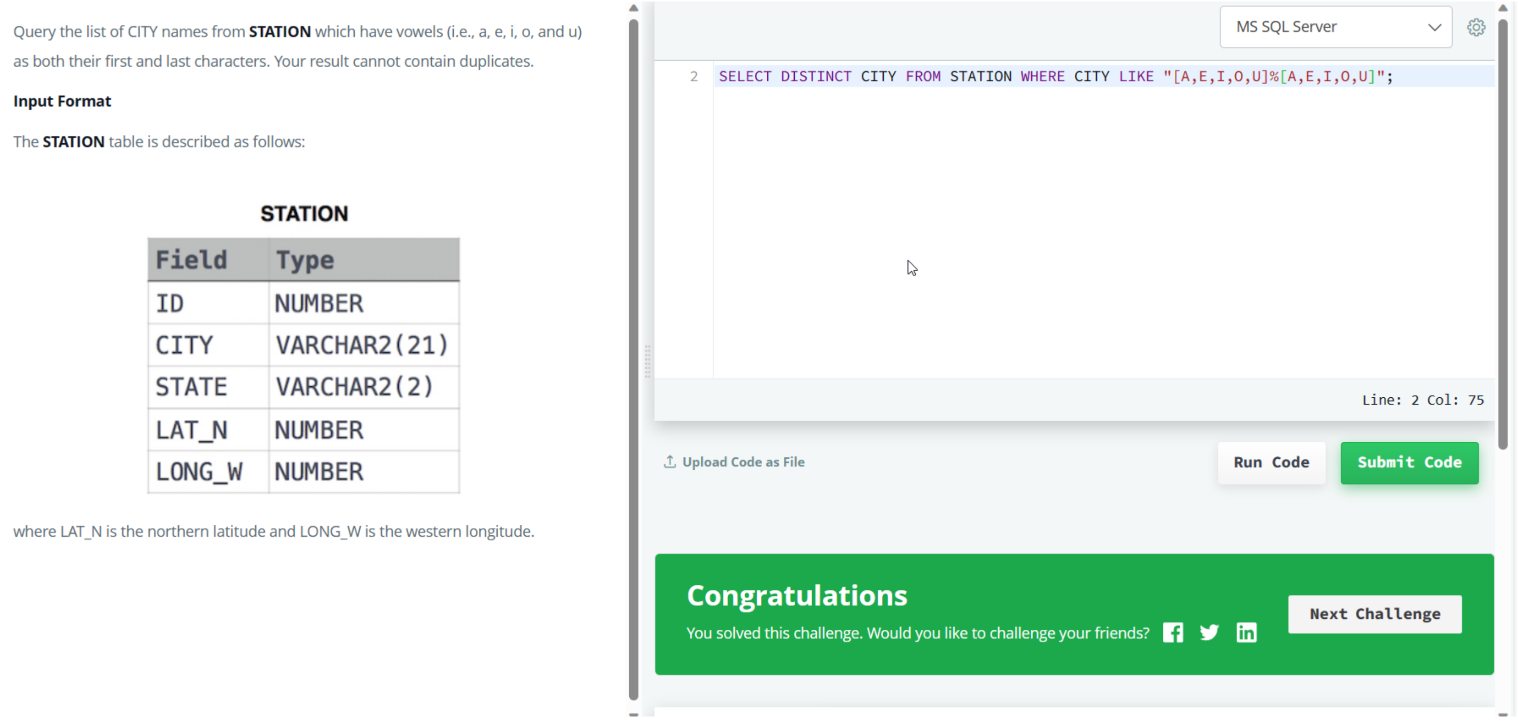
Line: 2 Col: 42

Congratulations

You solved this challenge. Would you like to challenge your friends?

7. Query the list of CITY names from STATION which have vowels (i.e., a, e, i, o, and u) as both their first and last characters. Your result cannot contain duplicates.

SQL query: **SELECT DISTINCT CITY FROM STATION WHERE CITY LIKE "[A,E,I,O,U]%[A,E,I,O,U]";**



Query the list of CITY names from **STATION** which have vowels (i.e., a, e, i, o, and u) as both their first and last characters. Your result cannot contain duplicates.

Input Format

The **STATION** table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

```
SELECT DISTINCT CITY FROM STATION WHERE CITY LIKE "[A,E,I,O,U]%[A,E,I,O,U]";
```

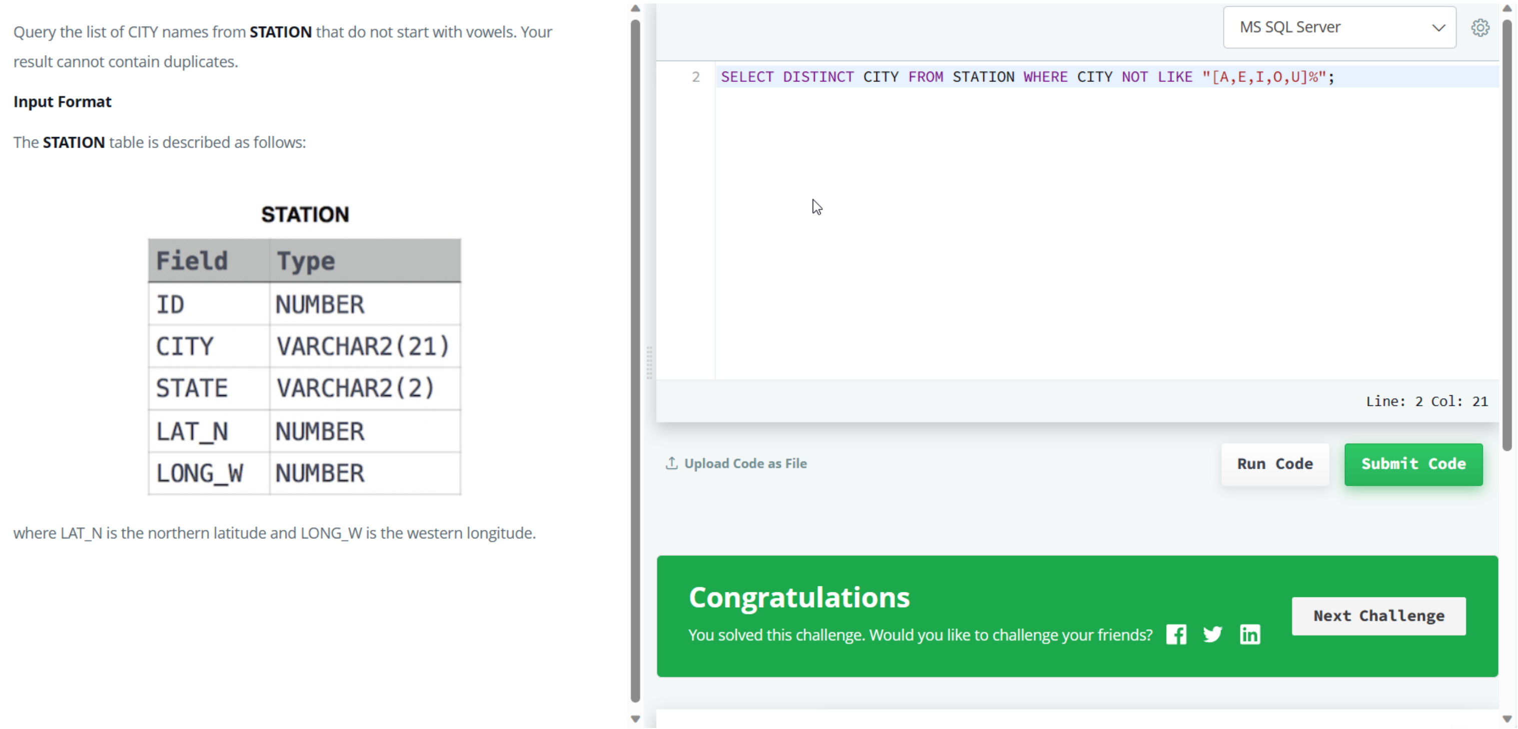
Congratulations

You solved this challenge. Would you like to challenge your friends?

8. Query the list of CITY names from STATION that do not start with vowels. Your result cannot contain duplicates.

SQL query:

SELECT DISTINCT CITY FROM STATION WHERE CITY NOT LIKE "[A,E,I,O,U]%";



Query the list of CITY names from **STATION** that do not start with vowels. Your result cannot contain duplicates.

Input Format

The **STATION** table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

```
SELECT DISTINCT CITY FROM STATION WHERE CITY NOT LIKE "[A,E,I,O,U]%";
```

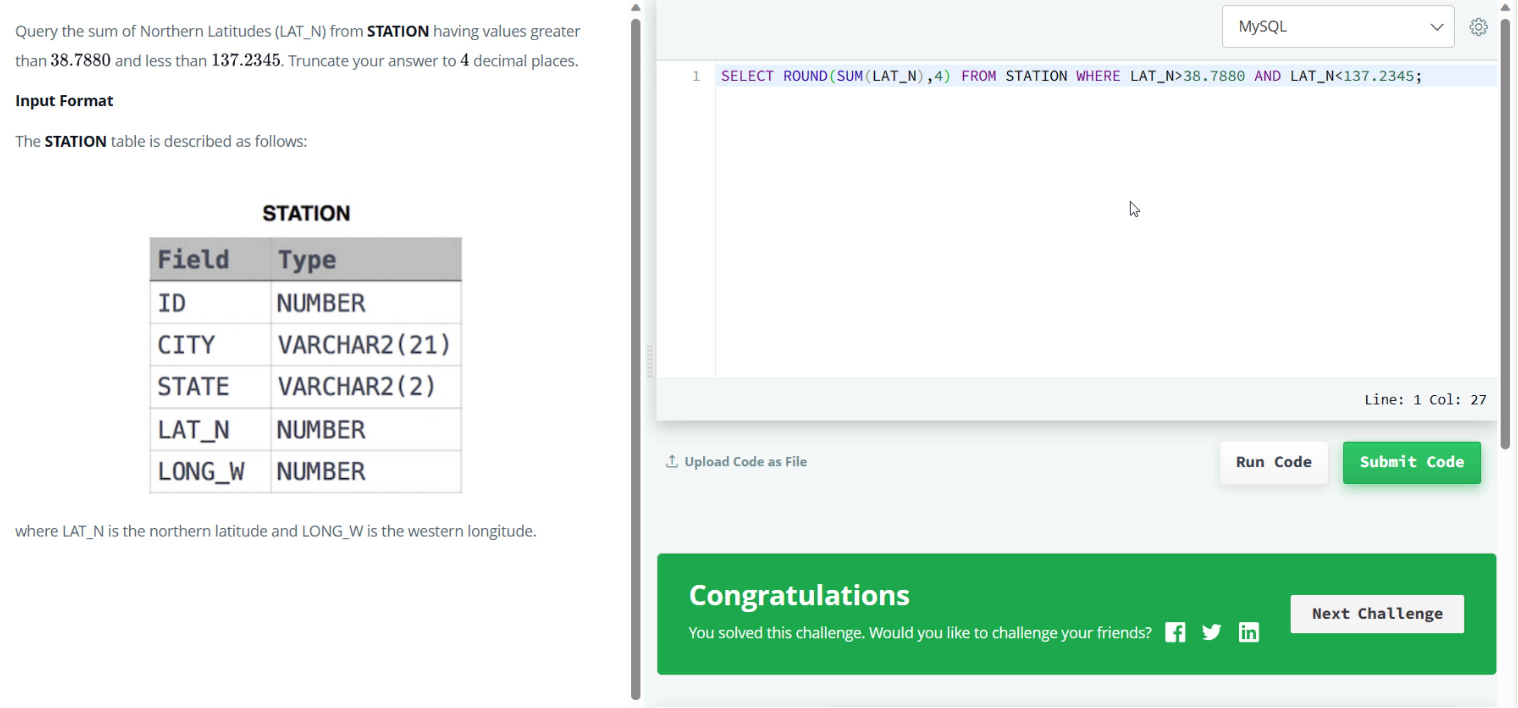
Congratulations

You solved this challenge. Would you like to challenge your friends?

9. Query the sum of Northern Latitudes (LAT_N) from STATION having values greater than 38.7880 and less than 137.2345. Truncate your answer to 4 decimal places.

SQL query:

SELECT ROUND(SUM(LAT_N),4) FROM STATION WHERE LAT_N>38.7880 AND LAT_N<137.2345;



Query the sum of Northern Latitudes (LAT_N) from **STATION** having values greater than 38.7880 and less than 137.2345. Truncate your answer to 4 decimal places.

Input Format

The **STATION** table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

MySQL

```
SELECT ROUND(SUM(LAT_N),4) FROM STATION WHERE LAT_N>38.7880 AND LAT_N<137.2345;
```

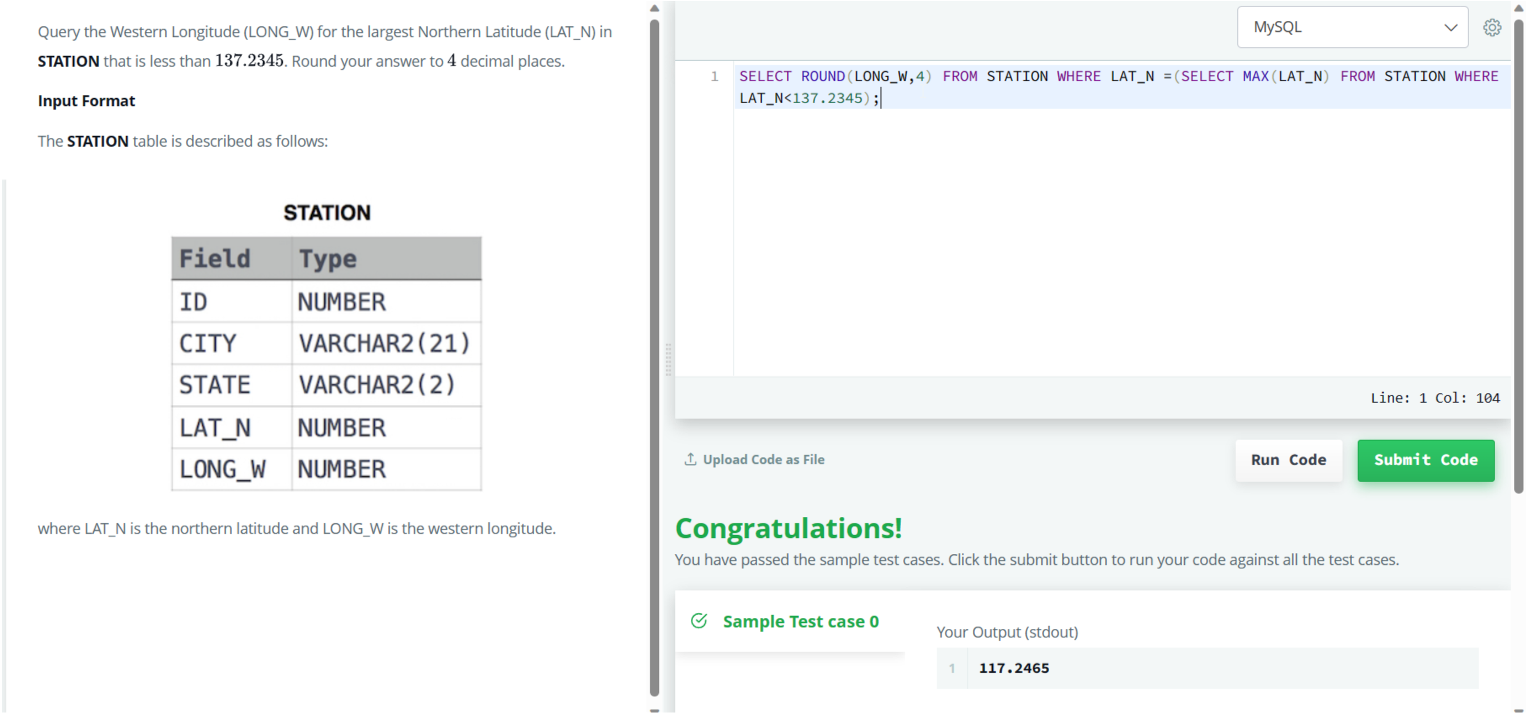
Line: 1 Col: 27

Congratulations

You solved this challenge. Would you like to challenge your friends?

10. Query the Western Longitude (LONG_W) for the largest Northern Latitude (LAT_N) in STATION that is less than 137.2345. Round your answer to 4 decimal places.

SQL query: **SELECT ROUND(LONG_W,4) FROM STATION WHERE LAT_N = (SELECT MAX(LAT_N) FROM STATION WHERE LAT_N<137.2345);**



Query the Western Longitude (LONG_W) for the largest Northern Latitude (LAT_N) in **STATION** that is less than 137.2345. Round your answer to 4 decimal places.

Input Format

The **STATION** table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

MySQL

```
SELECT ROUND(LONG_W,4) FROM STATION WHERE LAT_N =(SELECT MAX(LAT_N) FROM STATION WHERE
LAT_N<137.2345);
```

Line: 1 Col: 104

Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Your Output (stdout)

```
117.2465
```

11. Consider P_1(a,b) and P_2(c,d) to be two points on a 2D plane.

- a happens to equal the minimum value in Northern Latitude (LAT_N in STATION).

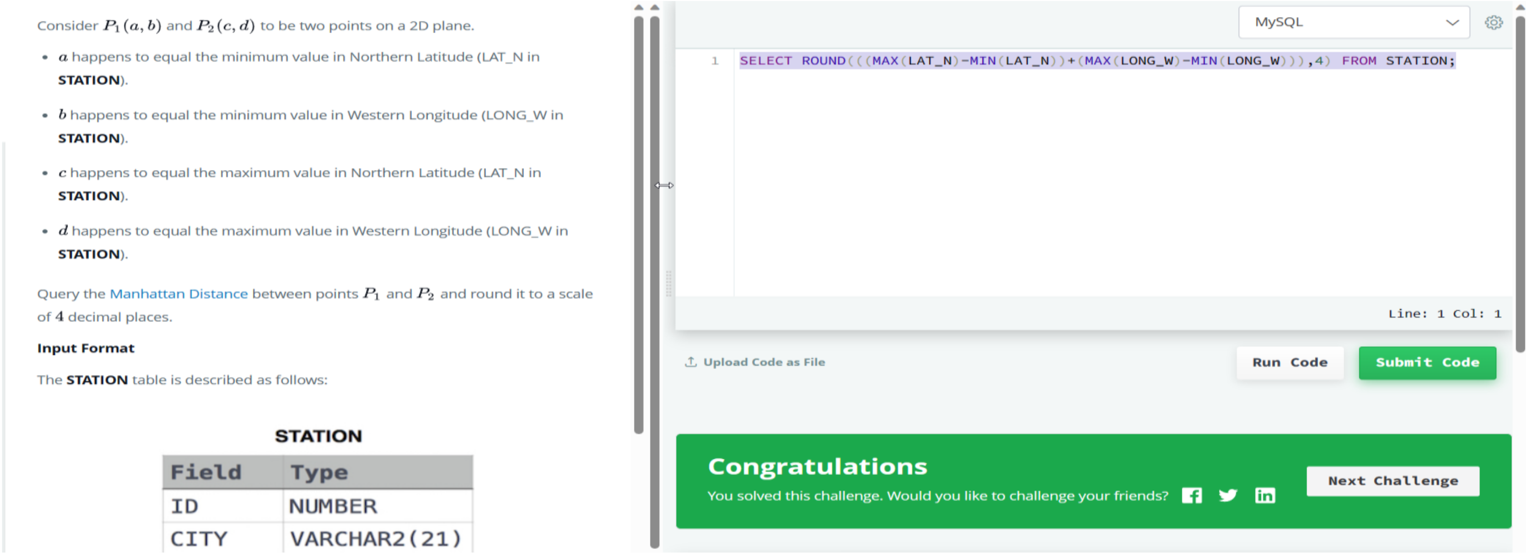
- b happens to equal the minimum value in Western Longitude (LONG_W in STATION).

- c happens to equal the maximum value in Northern Latitude (LAT_N in STATION).

- d happens to equal the maximum value in Western Longitude (LONG_W in STATION).

Query the Manhattan Distance between points P_1 and P_2 and round it to a scale of 4 decimal places.

SQL query: **SELECT ROUND(((MAX(LAT_N)-MIN(LAT_N))+(MAX(LONG_W)-MIN(LONG_W))),4) FROM STATION;**



Consider $P_1(a, b)$ and $P_2(c, d)$ to be two points on a 2D plane.

- a happens to equal the minimum value in Northern Latitude (LAT_N in **STATION**).
- b happens to equal the minimum value in Western Longitude (LONG_W in **STATION**).
- c happens to equal the maximum value in Northern Latitude (LAT_N in **STATION**).
- d happens to equal the maximum value in Western Longitude (LONG_W in **STATION**).

Query the Manhattan Distance between points P_1 and P_2 and round it to a scale of 4 decimal places.

Input Format

The **STATION** table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)

```
SELECT ROUND(((MAX(LAT_N)-MIN(LAT_N))+(MAX(LONG_W)-MIN(LONG_W))),4) FROM STATION;
```

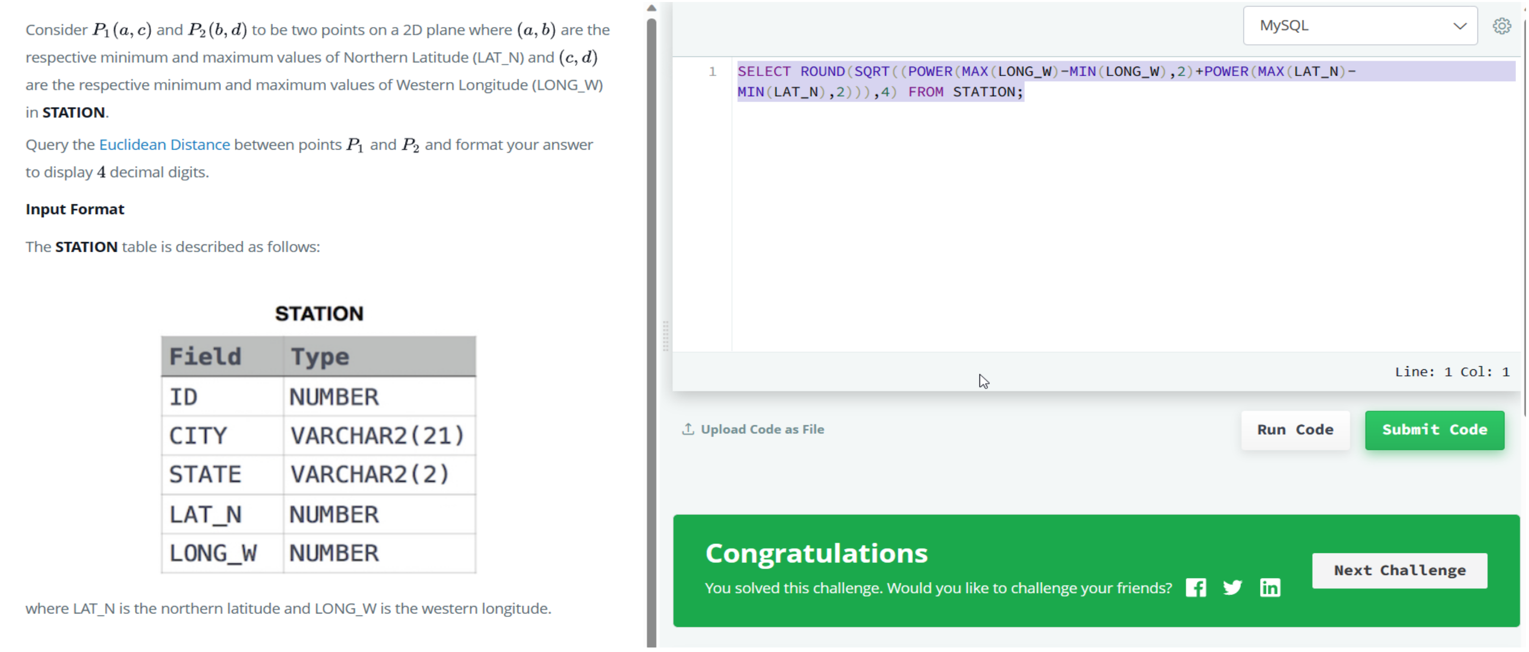
Congratulations

You solved this challenge. Would you like to challenge your friends?

12. Consider P_1(a,b) and P_2(c,d) to be two points on a 2D plane where (a,b) are the respective minimum and maximum values of Northern Latitude (LAT_N) and (c,d) are the respective minimum and maximum values of Western Longitude (LONG_W) in STATION.

Query the Euclidean Distance between points P_1 and P_2 and format your answer to display 4 decimal digits.

SQL query: **SELECT ROUND(SQRT((POWER(MAX(LONG_W)-MIN(LONG_W),2)+POWER(MAX(LAT_N)-MIN(LAT_N),2))),4) FROM STATION;**



Consider $P_1(a, c)$ and $P_2(b, d)$ to be two points on a 2D plane where (a, b) are the respective minimum and maximum values of Northern Latitude (LAT_N) and (c, d) are the respective minimum and maximum values of Western Longitude (LONG_W) in **STATION**.

Query the Euclidean Distance between points P_1 and P_2 and format your answer to display 4 decimal digits.

Input Format

The **STATION** table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

```
SELECT ROUND(SQRT((POWER(MAX(LONG_W)-MIN(LONG_W),2)+POWER(MAX(LAT_N)-
MIN(LAT_N),2))),4) FROM STATION;
```

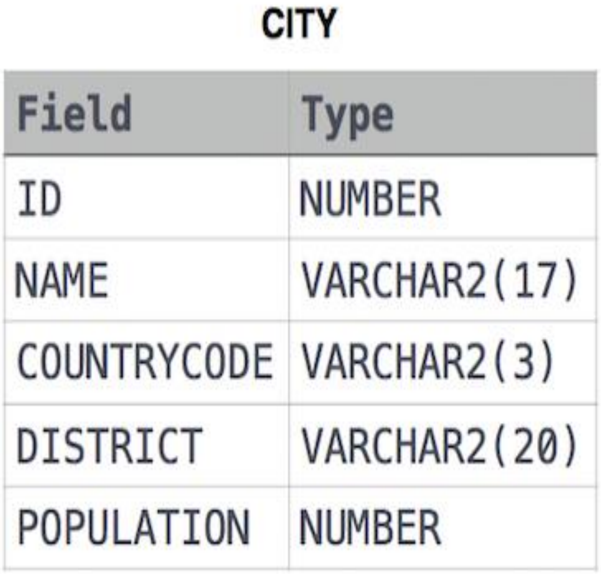
Congratulations

You solved this challenge. Would you like to challenge your friends?

13. Given the CITY and COUNTRY tables, query the sum of the populations of all cities where the CONTINENT is 'Asia'.

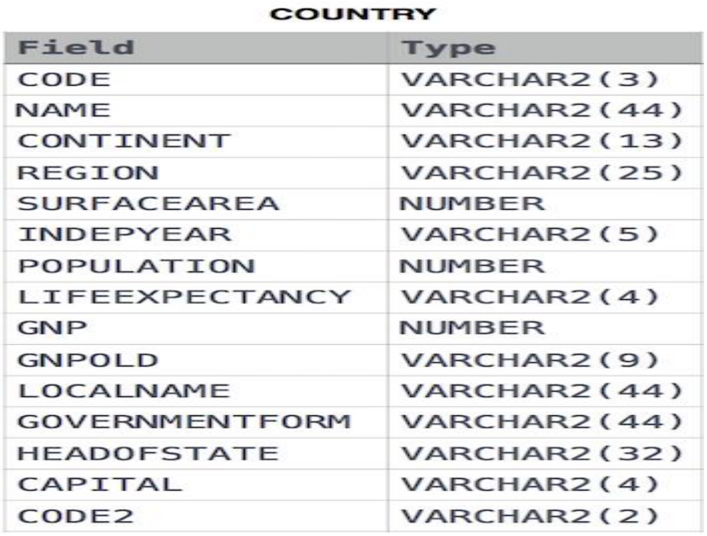
Note: CITY.CountryCode and COUNTRY.Code are matching key columns.

The CITY and COUNTRY tables are described as follows:



CITY

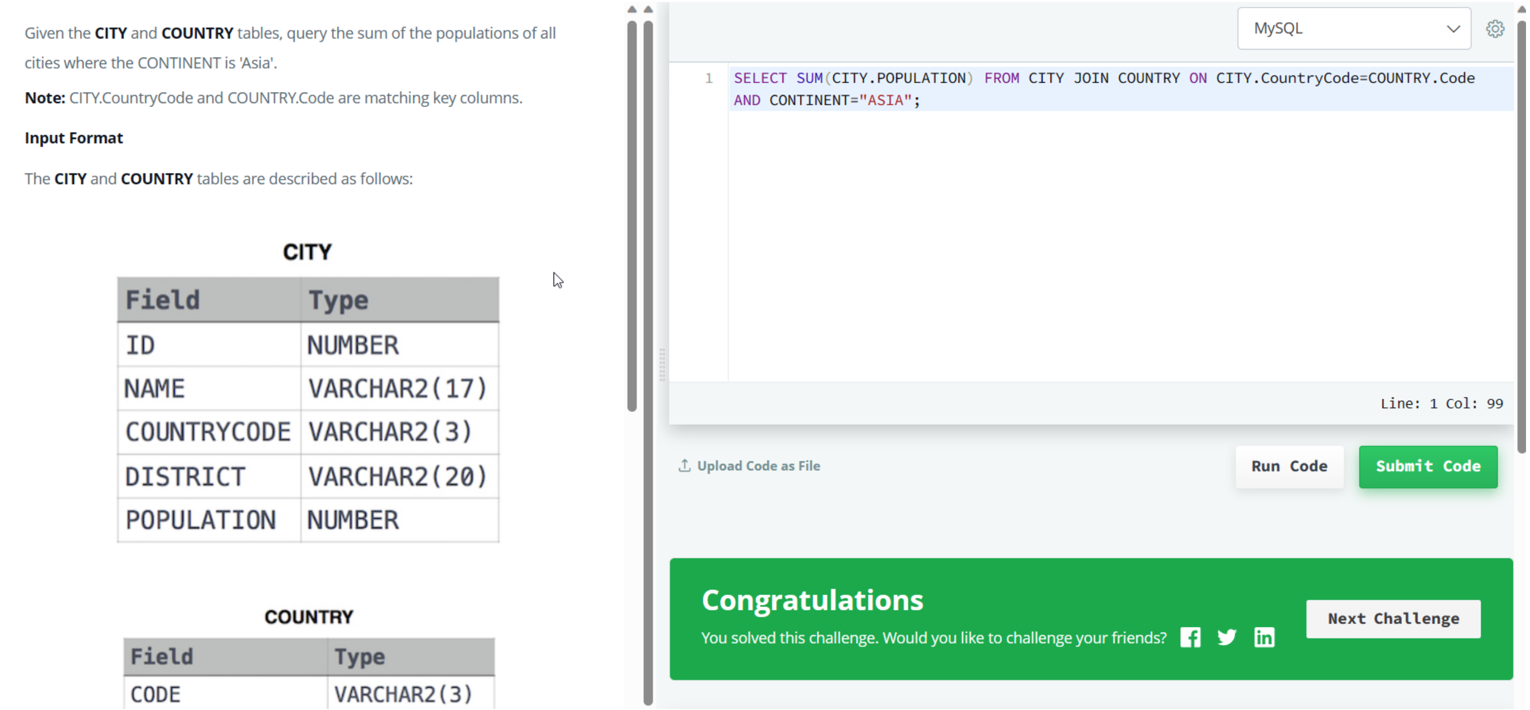
Field	Type
ID	NUMBER
NAME	VARCHAR2(17)
COUNTRYCODE	VARCHAR2(3)
DISTRICT	VARCHAR2(20)
POPULATION	NUMBER



COUNTRY

Field	Type
CODE	VARCHAR2(3)
NAME	VARCHAR2(44)
CONTINENT	VARCHAR2(13)
REGION	VARCHAR2(25)
SURFACEAREA	NUMBER
INDEPYEAR	VARCHAR2(5)
POPULATION	NUMBER
LIFEEXPECTANCY	VARCHAR2(4)
GNP	NUMBER
GNPOLD	VARCHAR2(9)
LOCALNAME	VARCHAR2(44)
GOVERNMENTFORM	VARCHAR2(44)
HEADOFSTATE	VARCHAR2(32)
CAPITAL	VARCHAR2(4)
CODE2	VARCHAR2(2)

SQL query: **SELECT SUM(CITY.POPULATION) FROM CITY JOIN COUNTRY ON CITY.CountryCode=COUNTRY.Code AND CONTINENT="ASIA";**



Given the **CITY** and **COUNTRY** tables, query the sum of the populations of all cities where the CONTINENT is 'Asia'.

Note: CITY.CountryCode and COUNTRY.Code are matching key columns.

Input Format

The **CITY** and **COUNTRY** tables are described as follows:

CITY

Field	Type
ID	NUMBER
NAME	VARCHAR2(17)
COUNTRYCODE	VARCHAR2(3)
DISTRICT	VARCHAR2(20)
POPULATION	NUMBER

COUNTRY

Field	Type
CODE	VARCHAR2(3)

```
SELECT SUM(CITY.POPULATION) FROM CITY JOIN COUNTRY ON CITY.CountryCode=COUNTRY.Code
AND CONTINENT="ASIA";
```

Line: 1 Col: 99

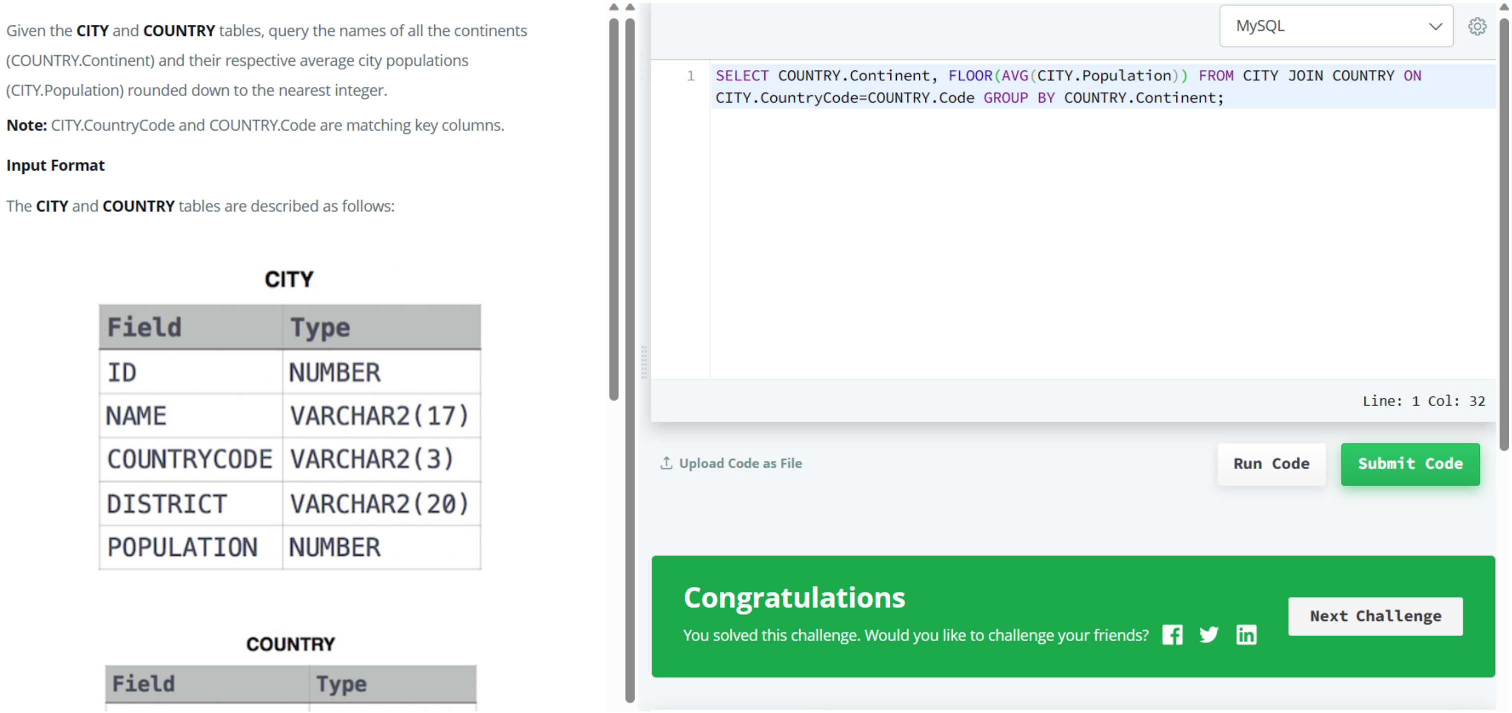
Congratulations

You solved this challenge. Would you like to challenge your friends?

14. Given the CITY and COUNTRY tables, query the names of all the continents (COUNTRY.Continent) and their respective average city populations (CITY.Population) rounded down to the nearest integer.

SQL query:

SELECT COUNTRY.Continent, FLOOR(AVG(CITY.Population)) FROM CITY JOIN COUNTRY ON CITY.CountryCode=COUNTRY.Code GROUP BY COUNTRY.Continent;



Given the **CITY** and **COUNTRY** tables, query the names of all the continents (COUNTRY.Continent) and their respective average city populations (CITY.Population) rounded down to the nearest integer.

Note: CITY.CountryCode and COUNTRY.Code are matching key columns.

Input Format

The **CITY** and **COUNTRY** tables are described as follows:

CITY

Field	Type
ID	NUMBER
NAME	VARCHAR2(17)
COUNTRYCODE	VARCHAR2(3)
DISTRICT	VARCHAR2(20)
POPULATION	NUMBER

COUNTRY

Field	Type

```
SELECT COUNTRY.Continent, FLOOR(AVG(CITY.Population)) FROM CITY JOIN COUNTRY ON
CITY.CountryCode=COUNTRY.Code GROUP BY COUNTRY.Continent;
```

Congratulations

You solved this challenge. Would you like to challenge your friends?