

# Database Assignment 2

## 1. [Japanese Cities' Names](#)

SQL Script Solution:

```
SELECT name FROM city WHERE countrycode = 'JPN';
```

Output Screenshot:

The screenshot shows a web-based MySQL editor interface. At the top, there's a dropdown menu set to 'MySQL' with expand and settings icons. Below this is a code editor with two lines of SQL: '1 SELECT name FROM city WHERE countrycode = 'JPN';' and '2'. A status bar at the bottom right of the editor indicates 'Line: 2 Col: 1'. Below the editor, there are three buttons: 'Upload Code as File', 'Run Code', and 'Submit Code'. A large green banner with the text 'Congratulations' and 'You solved this challenge. Would you like to challenge your friends?' with social media icons is present. At the bottom, there's a 'Test case 0' section with a green checkmark, a 'Compiler Message' box showing 'Success', and an 'Input (stdin)' box with the number '1'. A 'Download' button is also visible.

```
MySQL
```

```
1 SELECT name FROM city WHERE countrycode = 'JPN';
2
```

Line: 2 Col: 1

Upload Code as File Run Code Submit Code

### Congratulations

You solved this challenge. Would you like to challenge your friends? [f](#) [t](#) [in](#) [Next Challenge](#)

Test case 0

Compiler Message

Success

Input (stdin)

1

Download

## 2. [Weather Observation Station 3](#)

SQL Script Solution:

```
SELECT DISTINCT CITY
FROM STATION
WHERE MOD(ID, 2) = 0;
```

Output Screenshot:

HackerRank | Prepare > SQL > Basic Select | Weather Observation Station 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.

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.

```
1 SELECT DISTINCT CITY
2 FROM STATION
3 WHERE MOD(ID, 2) = 0;
```

Line: 3 Col: 22

Upload Code as File

Run Code Submit Code

**Congratulations**

You solved this challenge. Would you like to challenge your friends? [f](#) [t](#) [in](#)

Next Challenge

Test case 0

Compiler Message

Success

Input (stdin)

Download

INPUT

## 3. [Weather Observation Station 5](#)

SQL Script Solution:

```
// for shortest city
```

```
SELECT city, LENGTH(city) AS city_length
FROM station
ORDER BY LENGTH(city) ASC, city ASC
LIMIT 1;
```

```
// for longest city
```

```
SELECT city, LENGTH(city) AS city_length  
FROM station  
ORDER BY LENGTH(city) DESC, city ASC  
LIMIT 1;
```

Output Screenshot:

The screenshot shows a code editor with two SQL queries. The first query is commented out, and the second query is active. Below the code editor, there is a green banner with a 'Congratulations' message and a 'Next Challenge' button. At the bottom, there is a 'Test case 0' section showing the compiler message 'Success', the input 'INPUT', and the expected output 'Amo 3' and 'Marine On Saint Croix 21'.

```
3  
4 SELECT city, LENGTH(city) AS city_length  
5 FROM station  
6 ORDER BY LENGTH(city) ASC, city ASC  
7 LIMIT 1;  
8  
9 SELECT city, LENGTH(city) AS city_length  
10 FROM station  
11 ORDER BY LENGTH(city) DESC, city ASC  
12 LIMIT 1;
```

Line: 3 Col: 1

Upload Code as File

Run Code Submit Code

### Congratulations

You solved this challenge. Would you like to challenge your friends? [f](#) [t](#) [in](#)

Next Challenge

Test case 0

Compiler Message

Success

Input (stdin) Download

```
1 INPUT
```

Expected Output Download

```
1 Amo 3  
2 Marine On Saint Croix 21
```

## 4. [The Blunder](#)

SQL Script Solution:

```
SELECT CEILING(AVG(salary)-AVG(REPLACE(salary, 0,'')))
FROM employees;
```

Output Screenshot:

**Problem**

Samantha was tasked with calculating the average monthly salaries for all employees in the **EMPLOYEES** table, but did not realize her keyboard's 0 key was broken until after completing the calculation. She wants your help finding the difference between her miscalculation (using salaries with any zeros removed), and the actual average salary.

Write a query calculating the amount of error (i.e.: *actual* – *miscalculated* average monthly salaries), and round it up to the next integer.

**Input Format**

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

Column	Type
ID	Integer
Name	String
Salary	Integer

**Note:** Salary is per month.

**Constraints**

1000 < Salary < 10<sup>5</sup>.

**Sample Input**

ID	Name	Salary
1	Kristeen	1420
2	Ashley	2006

**SQL Script Solution:**

```
SELECT CEILING(AVG(salary)-AVG(REPLACE(salary, 0,'')))
FROM employees;
```

**Output Screenshot:**

**Congratulations**

You solved this challenge. Would you like to challenge your friends? [Facebook](#) [Twitter](#) [LinkedIn](#) [Next Challenge](#)

**Test case 0**

Compiler Message: **Success**

Input (stdin): **INPUT**

Expected Output: **2253**

## 5. [Weather Observation Station 18](#)

SQL Script Solution:

```
WITH diff AS (
SELECT
    MIN(LAT_N) AS a_latn_min,
    MAX(LAT_N) AS c_latn_max,
    MIN(LONG_W) AS b_longw_min,
    MAX(LONG_W) AS d_longw_max
```

```
FROM station
)

SELECT
  ROUND(
    ABS(c_latn_max - a_latn_min)
    +
    ABS(d_longw_max - b_longw_min),
    4)
  AS absolute_value_rounded
FROM diff;
```

Output Screenshot:

The screenshot displays a MySQL code editor interface. The top bar shows 'MySQL' and some icons. The main area contains a SQL query with line numbers 1 through 13. The query defines a CTE named 'diff' and then selects a rounded absolute value. Below the code editor, there are buttons for 'Upload Code as File', 'Run Code', and 'Submit Code'. A notification banner states 'You have earned 25.00 points!' and shows progress towards a badge. A green 'Congratulations' banner follows, with a 'Next Challenge' button. At the bottom, a 'Test case 0' section shows a 'Success' compiler message.

```
1 WITH diff AS (
2   SELECT
3     MIN(LAT_N) AS a_latn_min,
4     MAX(LAT_N) AS c_latn_max,
5     MIN(LONG_W) AS b_longw_min,
6     MAX(LONG_W) AS d_longw_max
7   FROM station
8 )
9
10 SELECT
11   ROUND(ABS(c_latn_max - a_latn_min) + ABS(d_longw_max - b_longw_min), 4)
12   AS absolute_value_rounded
13 FROM diff;
```

Line: 1 Col: 1

Upload Code as File Run Code Submit Code

You have earned 25.00 points!  
You are now 85 points away from the 2nd star for your sql badge. 11% 90/175

**Congratulations**  
You solved this challenge. Would you like to challenge your friends? [Next Challenge](#)

**Test case 0** Compiler Message  
Success

## 6. [Average Population of Each Continent](#)

SQL Script Solution:

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

Output Screenshot:

The screenshot shows the HackerRank interface for the 'Average Population of Each Continent' challenge. On the left, the 'Problem' tab is active, displaying a note that CITY.CountryCode and COUNTRY.Code are matching key columns. Below this, the 'Input Format' section describes the CITY and COUNTRY tables. The CITY table has columns: ID (NUMBER), NAME (VARCHAR2(17)), COUNTRYCODE (VARCHAR2(3)), DISTRICT (VARCHAR2(20)), and POPULATION (NUMBER). The COUNTRY table has columns: CODE (VARCHAR2(3)), NAME (VARCHAR2(44)), CONTINENT (VARCHAR2(13)), REGION (VARCHAR2(25)), SURFACEAREA (NUMBER), INDEPYEAR (VARCHAR2(5)), POPULATION (NUMBER), and LIFEEXPECTANCY (VARCHAR2(4)).

On the right, the SQL editor shows the following solution:

```
1 SELECT COUNTRY.Continent, FLOOR(AVG(CITY.Population)) AS avg_city_population
2 FROM CITY
3 JOIN COUNTRY ON CITY.CountryCode = COUNTRY.Code
4 GROUP BY COUNTRY.Continent;
5
```

Below the editor, there are buttons for 'Run Code' and 'Submit Code'. A green banner indicates 'You have earned 10.00 points!' and 'You are now 45 points away from the 2nd star for your sql badge.' A 'Congratulations' message states 'You solved this challenge. Would you like to challenge your friends?' with social media icons and a 'Next Challenge' button. At the bottom, a 'Test case 0' section shows a 'Success' compiler message.

## 7. [The PADS](#)

SQL Script Solution:

```
SELECT
  CONCAT(name, '(', LEFT(occupation, 1), ')')
FROM occupations
ORDER BY name ASC;
```

```
SELECT
    CONCAT('There are a total of ', COUNT(occupation), ' ',
    LOWER(occupation), 's.')
FROM occupations
GROUP BY occupation
ORDER BY COUNT(occupation) ASC, occupation ASC;
```


Output Screenshot:




The screenshot displays a SQL challenge interface. At the top, a code editor shows two SQL queries. The first query is for a leaderboard, and the second query is the solution provided by the user. Below the code editor, there are buttons for 'Upload Code as File', 'Run Code', and 'Submit Code'. A notification banner indicates that the user has earned 30.00 points and is 55 points away from the 2nd star for their SQL badge. The progress is shown as 42% (120/175). A green 'Congratulations' banner follows, with a 'Next Challenge' button. At the bottom, the 'Test case 0' results are shown as 'Success'.


```
1 SELECT
2     CONCAT(name, '(', LEFT(occupation, 1), ')')
3 FROM occupations
4 ORDER BY name ASC;
5
6 SELECT CONCAT('There are a total of ', COUNT(occupation), ' ', LOWER(occupation), 's.')
7 FROM occupations
8 GROUP BY occupation
9 ORDER BY COUNT(occupation) ASC, occupation ASC;
```

Line: 6 Col: 89

Upload Code as File Run Code Submit Code

 You have earned 30.00 points!  
You are now 55 points away from the 2nd star for your sql badge. 42% 120/175

**Congratulations**  
You solved this challenge. Would you like to challenge your friends?    [Next Challenge](#)

 **Test case 0**

Compiler Message  
Success

Input (stdin) Download  
INPUT

## 8. Type of Triangle

SQL Script Solution:

```
SELECT
CASE
    WHEN A + B <= C OR A + C <= B OR B + C <= A THEN 'Not A
Triangle'
    WHEN A = B AND B = C THEN 'Equilateral'
    WHEN A = B OR A = C OR B = C THEN 'Isosceles'
    ELSE 'Scalene'
END AS triangleType
FROM
    triangles;
```

Output Screenshot:

The screenshot displays a SQL challenge interface. At the top, a code editor shows the SQL script for determining the type of a triangle based on side lengths A, B, and C. The script uses a CASE statement to categorize the triangle as 'Not A Triangle', 'Equilateral', 'Isosceles', or 'Scalene'. Below the code editor, there are buttons for 'Run Code' and 'Submit Code'. A notification banner indicates that the user has earned 20.00 points and is 25 points away from the 2nd star for their SQL badge. A large green 'Congratulations' banner follows, with a 'Next Challenge' button. At the bottom, the 'Test case 0' results are shown, indicating a 'Success' status with a 'Compiler Message' and 'Input (stdin)' section.

```
1 SELECT
2     CASE
3         WHEN A + B <= C OR A + C <= B OR B + C <= A THEN 'Not A Triangle'
4         WHEN A = B AND B = C THEN 'Equilateral'
5         WHEN A = B OR A = C OR B = C THEN 'Isosceles'
6         ELSE 'Scalene'
7     END AS triangleType
8 FROM
9     triangles;
```

Line: 1 Col: 8

Upload Code as File

Run Code Submit Code

You have earned 20.00 points!  
You are now 25 points away from the 2nd star for your sql badge. 74% 150/175

**Congratulations**  
You solved this challenge. Would you like to challenge your friends? [Next Challenge](#)

Test case 0

Compiler Message

Success

Input (stdin) Download



## 9. [Weather Observation Station 13](#)

SQL Script Solution:

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

Output Screenshot:

The screenshot displays a web-based SQL challenge interface. At the top, a tab labeled 'MySQL' is visible. The code editor contains the following SQL script:

```
1 SELECT ROUND(SUM(LAT_N), 4) AS sum
2 FROM station
3 WHERE LAT_N > 38.7880 AND LAT_N < 137.2345;
4
```

Below the code editor, there are buttons for 'Run Code' and 'Submit Code'. A notification banner indicates that the user has earned 10.00 points, bringing their total to 160/175, and is 15 points away from the 2nd star badge. A green 'Congratulations' banner follows, with a 'Next Challenge' button. At the bottom, the 'Test case 0' section shows a 'Success' message from the compiler.

MySQL

```
1 SELECT ROUND(SUM(LAT_N), 4) AS sum
2 FROM station
3 WHERE LAT_N > 38.7880 AND LAT_N < 137.2345;
4
```

Line: 1 Col: 35

Upload Code as File

Run Code Submit Code

You have earned 10.00 points!  
You are now 15 points away from the 2nd star for your sql badge.

84% 160/175

**Congratulations**

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

Next Challenge

Test case 0

Compiler Message

Success

## 10. [The Report](#)

SQL Script Solution:

```
SELECT
  CASE
    WHEN grade >= 8 THEN name
    ELSE 'NULL'
  END AS name, grade, marks
FROM Students s
JOIN Grades g ON s.marks BETWEEN g.min_mark AND
g.max_mark
ORDER BY grade desc, name asc, marks asc;
```

Output Screenshot:

The screenshot displays a web-based SQL challenge interface. At the top, a dropdown menu shows 'MySQL'. Below it, a code editor contains the following SQL script:

```
1 SELECT
2   CASE
3     WHEN grade >= 8 THEN name
4     ELSE 'NULL'
5   END AS name, grade, marks
6 FROM Students s
7 JOIN Grades g ON s.marks BETWEEN g.min_mark AND g.max_mark
8 ORDER BY grade desc, name asc, marks asc
9
```

Below the code editor, there are two buttons: 'Run Code' and 'Submit Code'. A notification banner states: 'You have earned 20.00 points! You are now 120 points away from the 3rd star for your sql badge.' To the right of this banner, it shows '4%' and '180/300'. A large green banner with the text 'Congratulations' and 'You solved this challenge. Would you like to challenge your friends?' is followed by social media icons for Facebook, Twitter, and LinkedIn, and a 'Next Challenge' button. At the bottom, a 'Test case 0' section shows a 'Success' message in the 'Compiler Message' area.