**Assignment 1**

June 1, 2023

Group 9

John Gu - 200327450

Dain Shin - 200535561

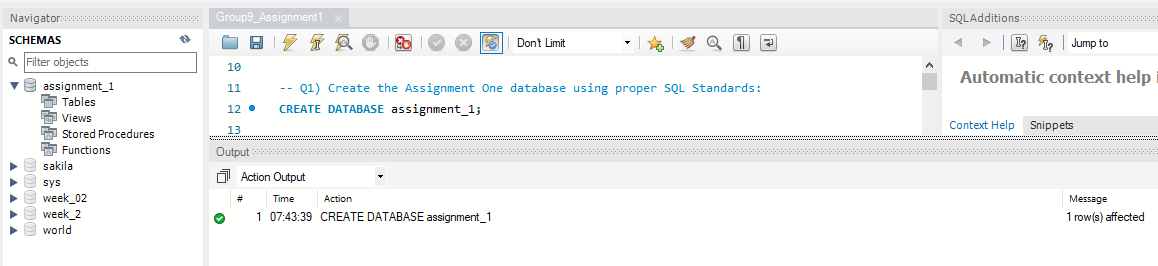
Matthew Antonis – 200373088

Emily Rose – 200553504

Karsten Leung - 200547539

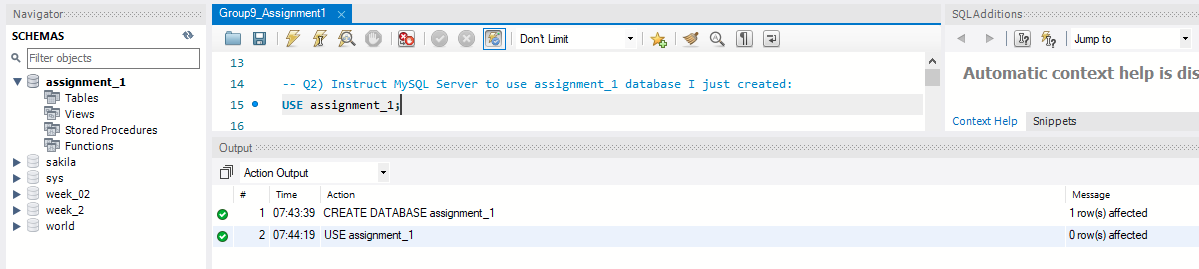
1. Create a database for this assignment.

|  |
| --- |
| -- Q1) Create the Assignment One database using proper SQL Standards: CREATE DATABASE assignment\_1; |



1. Instruct MySQL Server to use the database you just created.

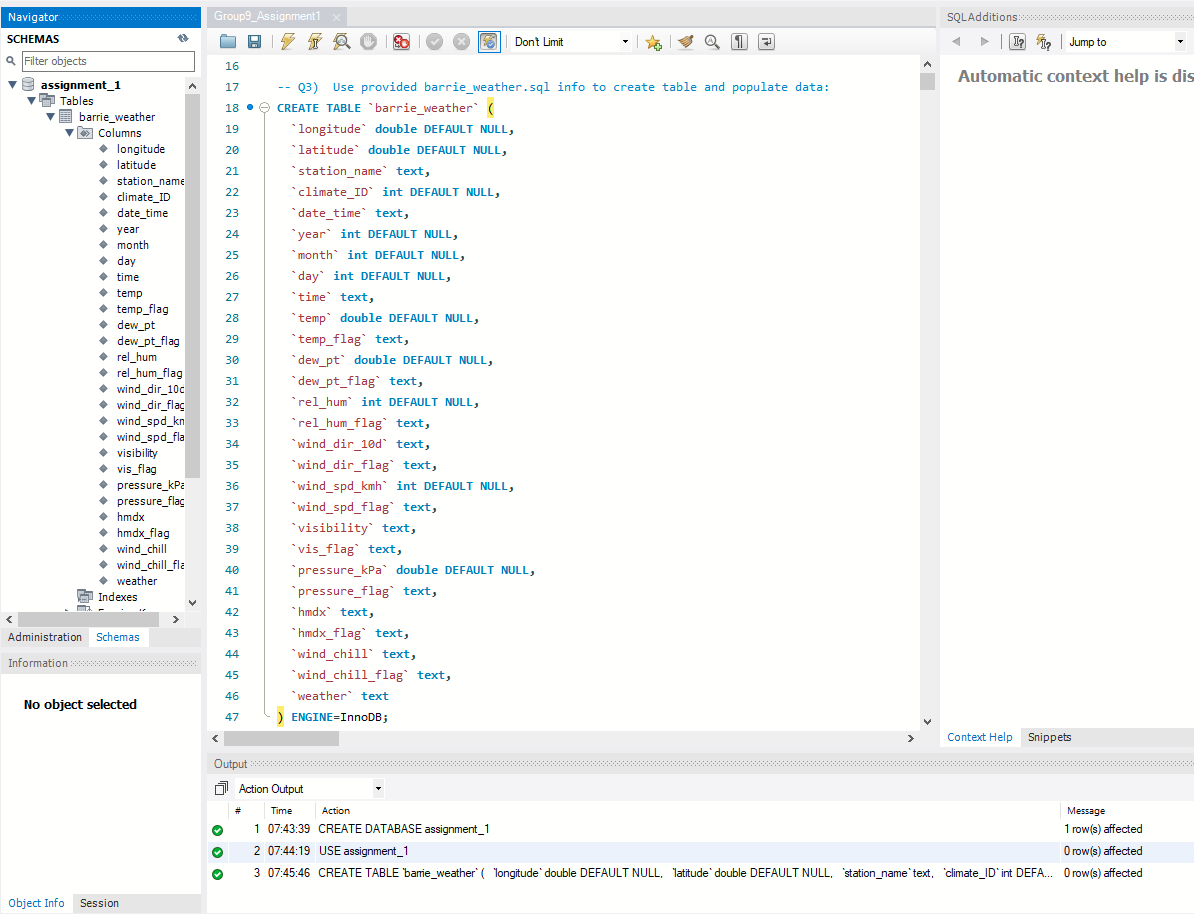
|  |
| --- |
| -- Q2) Instruct MySQL Server to use assignment\_1 database I just created: USE assignment\_1; |

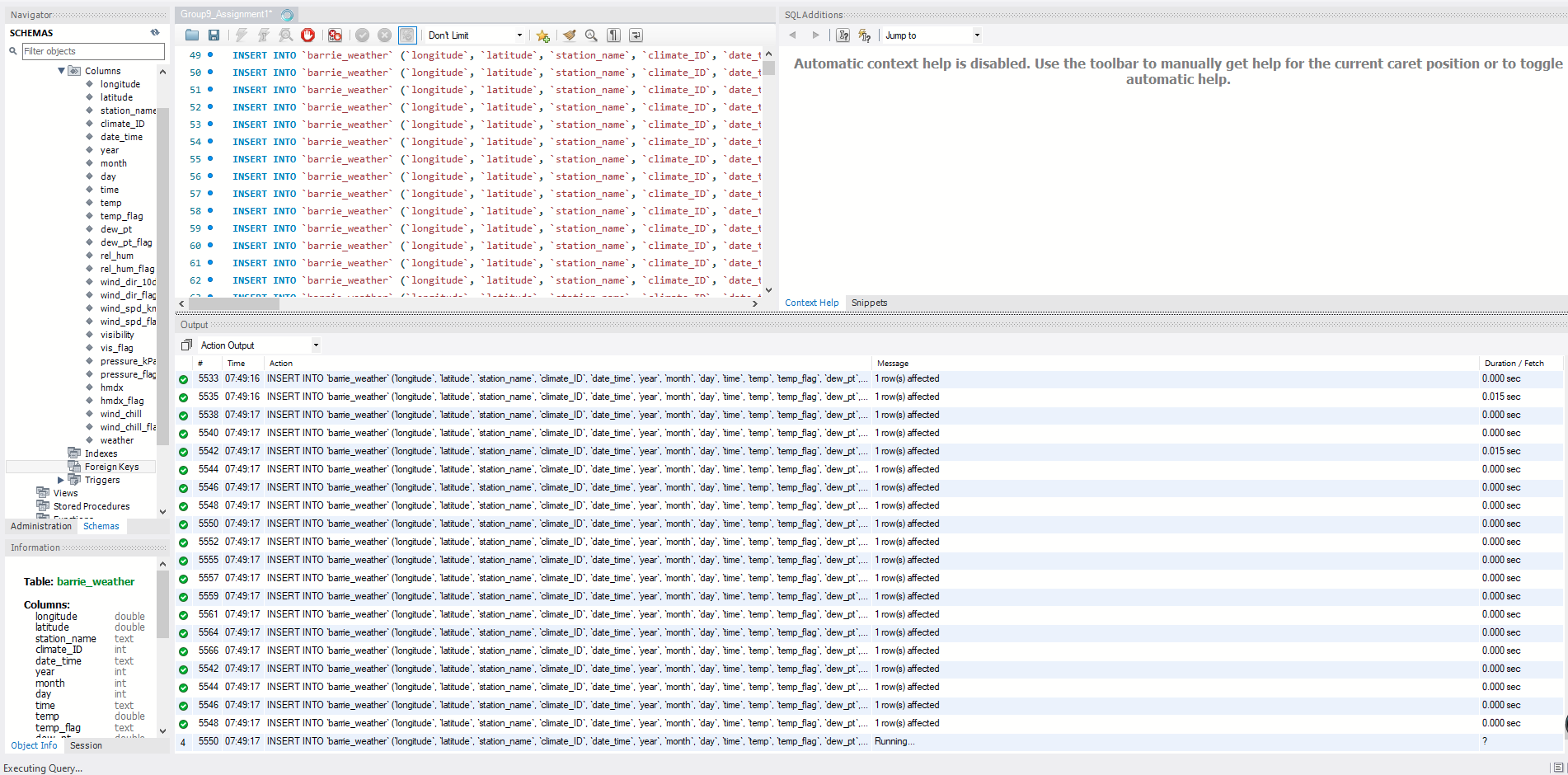


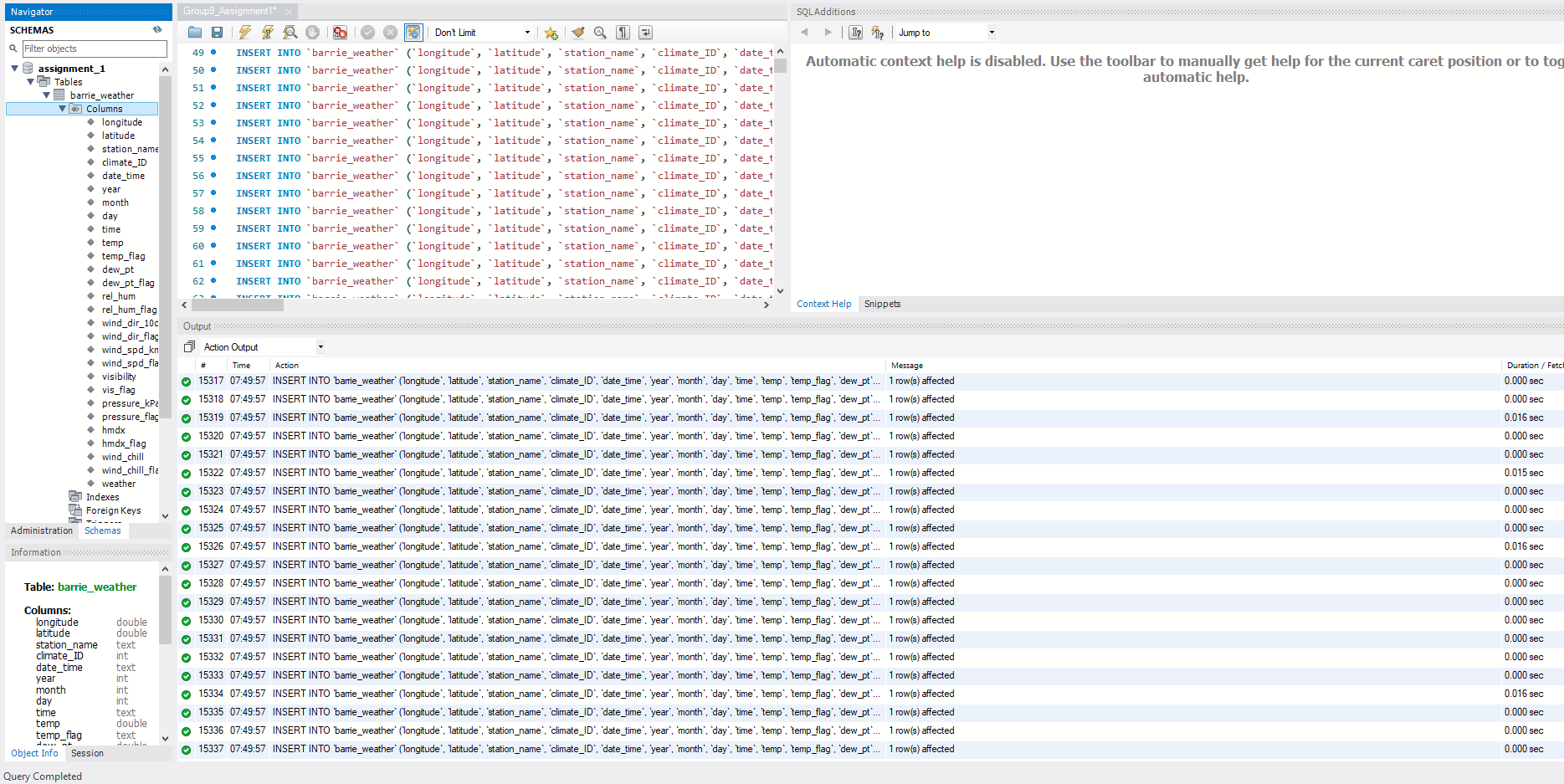
1. In a separate SQL tab, open the barrie\_weather.sql file provided. Under question 3, copy its contents into your assignment1 script, then run the commands to create the table (also called barrie\_weather, and will be used to answer following questions) and then populate it with data using the provided insert statements.

|  |
| --- |
| -- Q3) : Use provided barrie\_weather.sql info to create table and populate data: CREATE TABLE `barrie\_weather` (  `longitude` double DEFAULT NULL,  `latitude` double DEFAULT NULL,  `station\_name` text,  `climate\_ID` int DEFAULT NULL,  `date\_time` text,  `year` int DEFAULT NULL,  `month` int DEFAULT NULL,  `day` int DEFAULT NULL,  `time` text,  `temp` double DEFAULT NULL,  `temp\_flag` text,  `dew\_pt` double DEFAULT NULL,  `dew\_pt\_flag` text,  `rel\_hum` int DEFAULT NULL,  `rel\_hum\_flag` text,  `wind\_dir\_10d` text,  `wind\_dir\_flag` text,  `wind\_spd\_kmh` int DEFAULT NULL,  `wind\_spd\_flag` text,  `visibility` text,  `vis\_flag` text,  `pressure\_kPa` double DEFAULT NULL,  `pressure\_flag` text,  `hmdx` text,  `hmdx\_flag` text,  `wind\_chill` text,  `wind\_chill\_flag` text,  `weather` text ) ENGINE=InnoDB;  INSERT INTO `barrie\_weather` (`longitude`, `latitude`, `station\_name`, `climate\_ID`, `date\_time`, `year`, `month`, `day`, `time`, `temp`, `temp\_flag`, `dew\_pt`, `dew\_pt\_flag`, `rel\_hum`, `rel\_hum\_flag`, `wind\_dir\_10d`, `wind\_dir\_flag`, `wind\_spd\_kmh`, `wind\_spd\_flag`, `visibility`, `vis\_flag`, `pressure\_kPa`, `pressure\_flag`, `hmdx`, `hmdx\_flag`, `wind\_chill`, `wind\_chill\_flag`, `weather`) VALUES (-79.55,44.48,'BARRIE-ORO',6117700,'9/1/2019 0:00',2019,9,1,'0:00',11.3,'',9.1,'',86,'','36','',4,'','','',99.22,'',NULL,'','','','NA'); |

Etc…

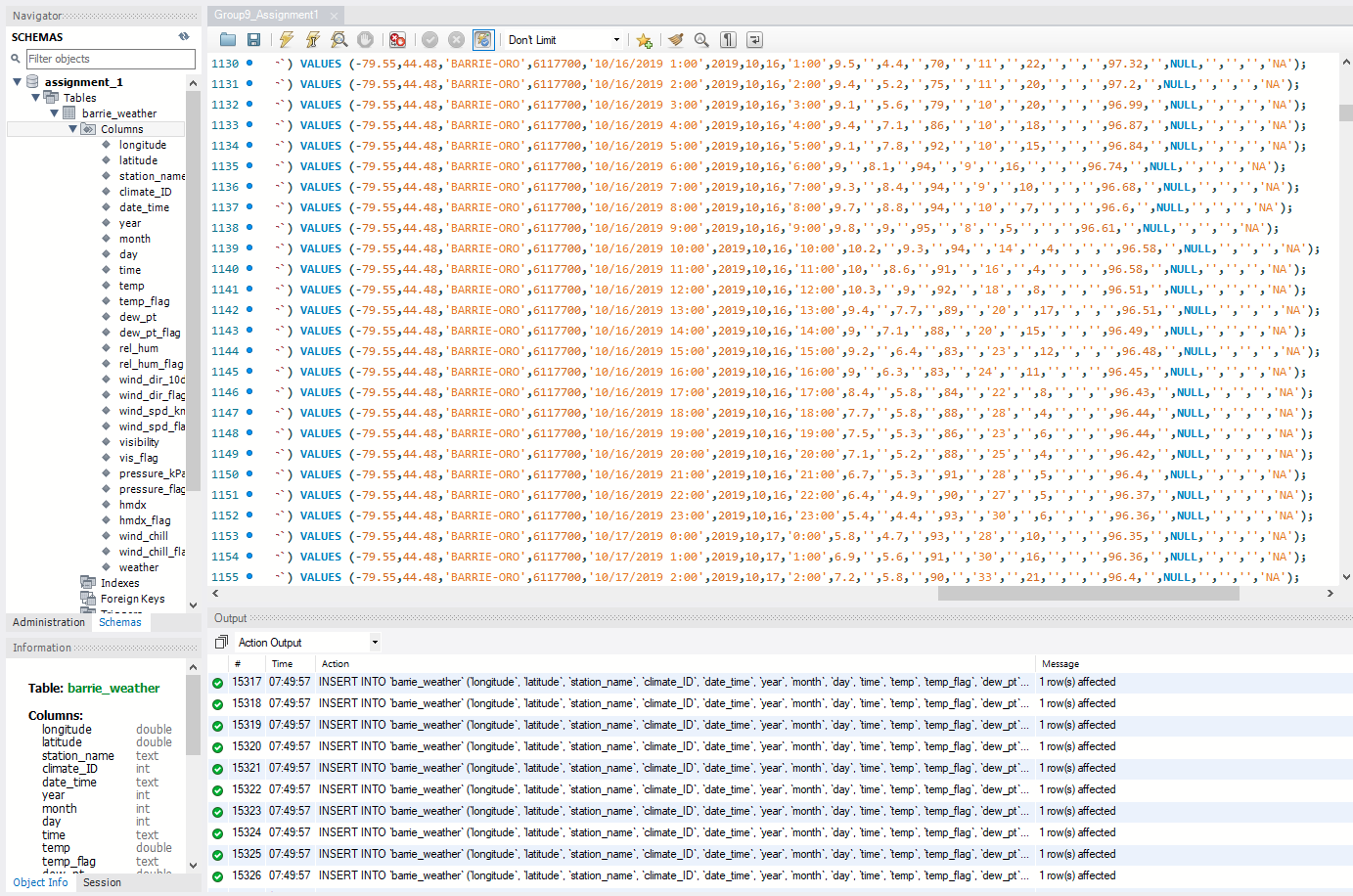






1. **Explain how the provided CREATE TABLE statement could be improved.  
   Please focus on the data types and parameters for this discussion.**

If we look at the insert statements we can see that the numeric and text values are rather low compared to the total byte size of int, double, and text.



The data types used in the create table statement can be converted into smaller data types in order to reduce memory use and improve performance. Since the create table statement only uses int, double, or text data types, we could instead replace them with tinyInt/smallInt, float, and varchar data types respectively.

For example, the text data type uses 65,535 bytes, but none of the text data columns ever use that many bytes. If we use varchar, then we can set the limit from anywhere between 0 to 65,535 bytes and improve the overall performance of our database.

Additionally, depending on how we want to use our data, we could change the data type values of our date and time related columns to the date, datetime, or timestamp data values in order to preserve more space. While this could make querying more difficult depending on how we intend to use the data, it could also preserve additional space by further lowering the used memory space and reducing the amount of columns in our table.

1. **The provided INSERT statements in barrie\_weather were generated automatically and load data very slowly because of the way that the script’s commands have been structured. What is it about the structure of this script that causes the script to run slowly? In the report only, explain how you could restructure the commands if you were tasked with inserting the same data into barrie\_weather.**

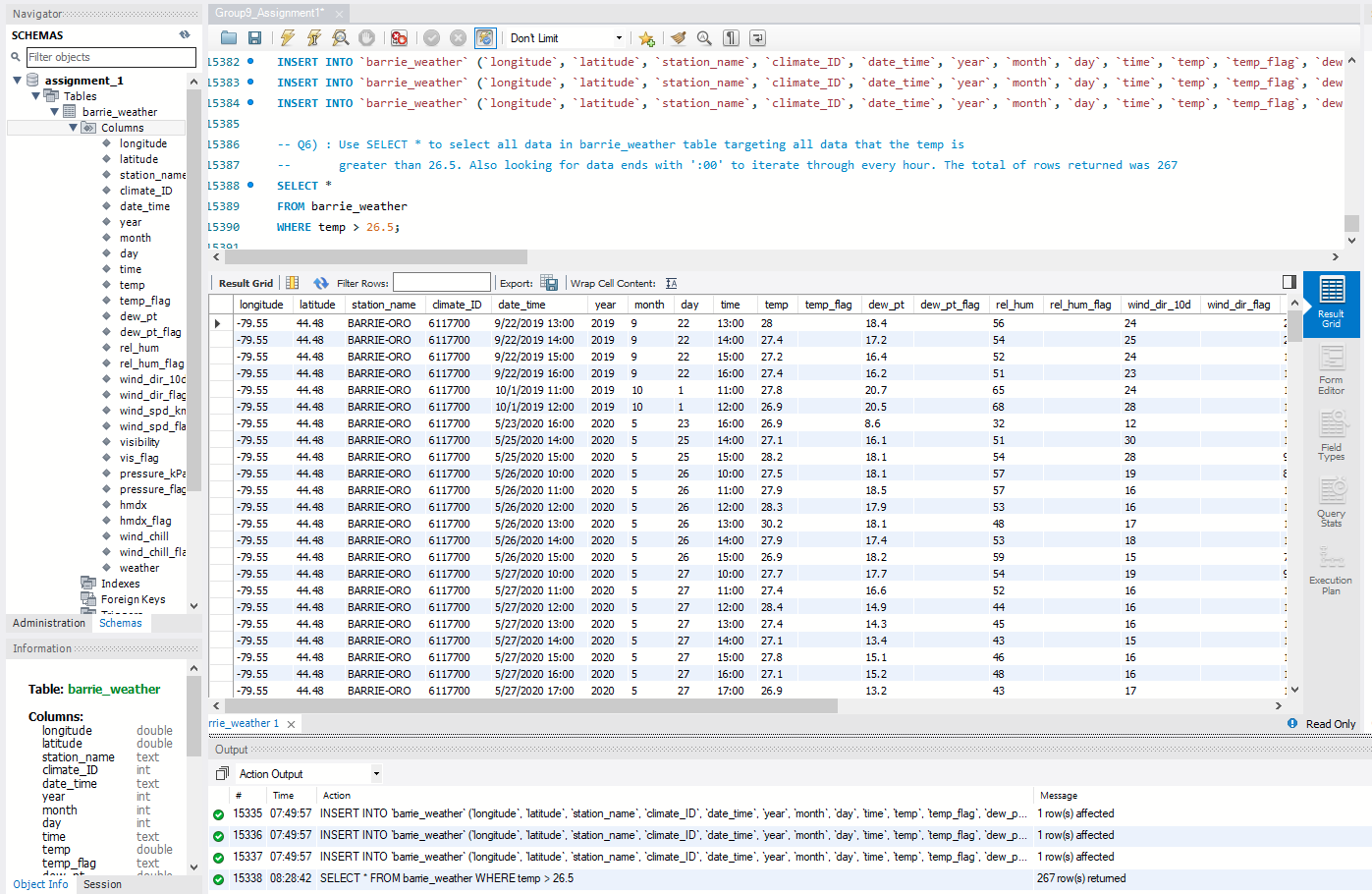
The cause of our insert statements running slowly is due to the repetitiveness of the way it is structured; it runs thousands of separate “insert into” statements as opposed to condensing them into one or fewer statements by using an extended insert statement.

By using the “INSERT INTO table() VALUES ()” syntax, we can insert multiple rows of data under the same statement, which is less reading and execution for the program, in turn making our database run more smoothly.

Alternatively, we could keep our data on a separate .csv file and read the data from there if that approach is appropriate for the way we wish to use our data.

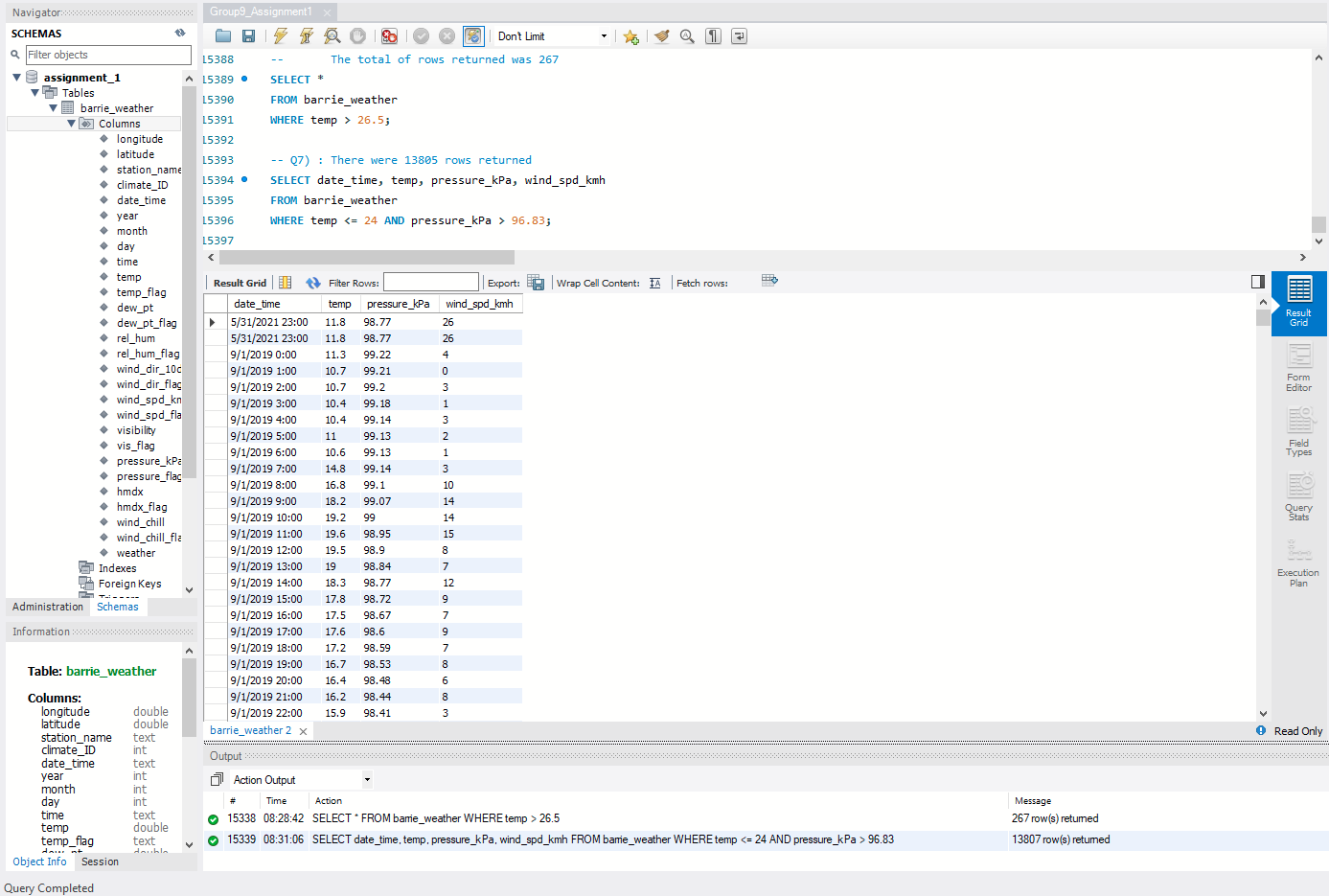
1. Select all data for every hour of data where the temperature (temp) was greater than 26.5. Provide an additional comment in your SQL script that states the number of rows returned.

|  |
| --- |
| -- Q6) : Use SELECT \* to select all data in barrie\_weather table targeting all data that the temp is greater than 26.5. Also looking for data ends with ':00' to iterate through every hour.  -- The total of rows returned was 267 SELECT \* FROM barrie\_weather WHERE temp > 26.5; |



1. Select the date\_time, temp, pressure\_kPA and wind\_spd\_kmh for every hour of data where the temperature was 24 or less and the air pressure (pressure\_kPa) was greater than 96.83. Provide an additional comment in your SQL script that states the number of rows returned.

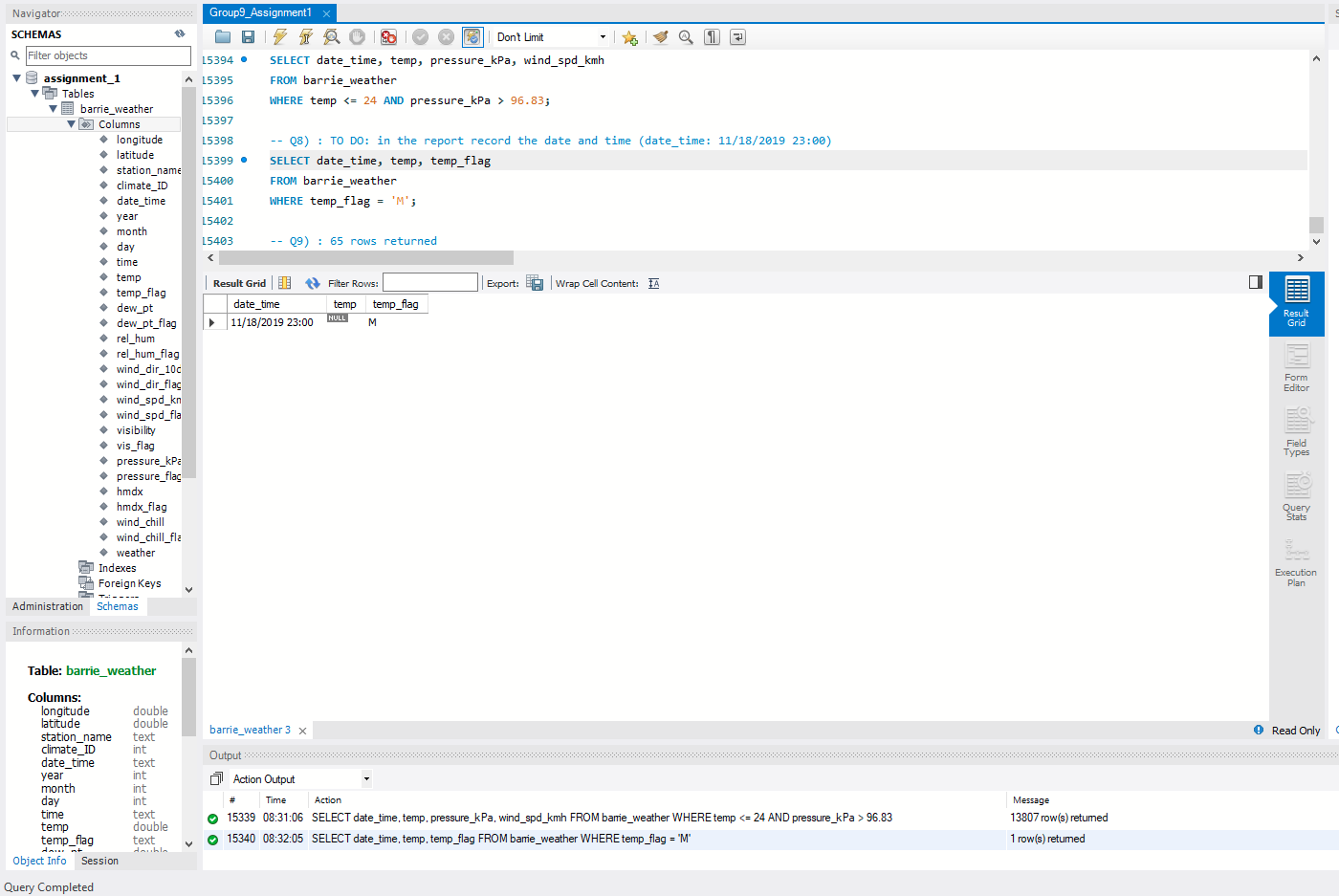
|  |
| --- |
| -- Q7) : Find the date\_time, temp, pressure\_kPA and wind\_spd\_kmh for every hour where the temperature was less than or equal to 24 degrees  -- There were 13805 rows returned SELECT date\_time, temp, pressure\_kPa, wind\_spd\_kmh FROM barrie\_weather WHERE temp <= 24 AND pressure\_kPa > 96.83; |



1. Select the date and time (date\_time), temperature (temp) and temperature flag (temp\_flag) columns for every hour where temperature data has been marked as missing using an ‘M’ value in the temp\_flag column. In the report, record the date and time.

Hint: remember that single quotes are required for text values.

|  |
| --- |
| -- Q8) : Find the date\_time, temp, and temp\_flag columns for every hour where temp\_flag is 'M' SELECT date\_time, temp, temp\_flag FROM barrie\_weather WHERE temp\_flag = 'M'; |

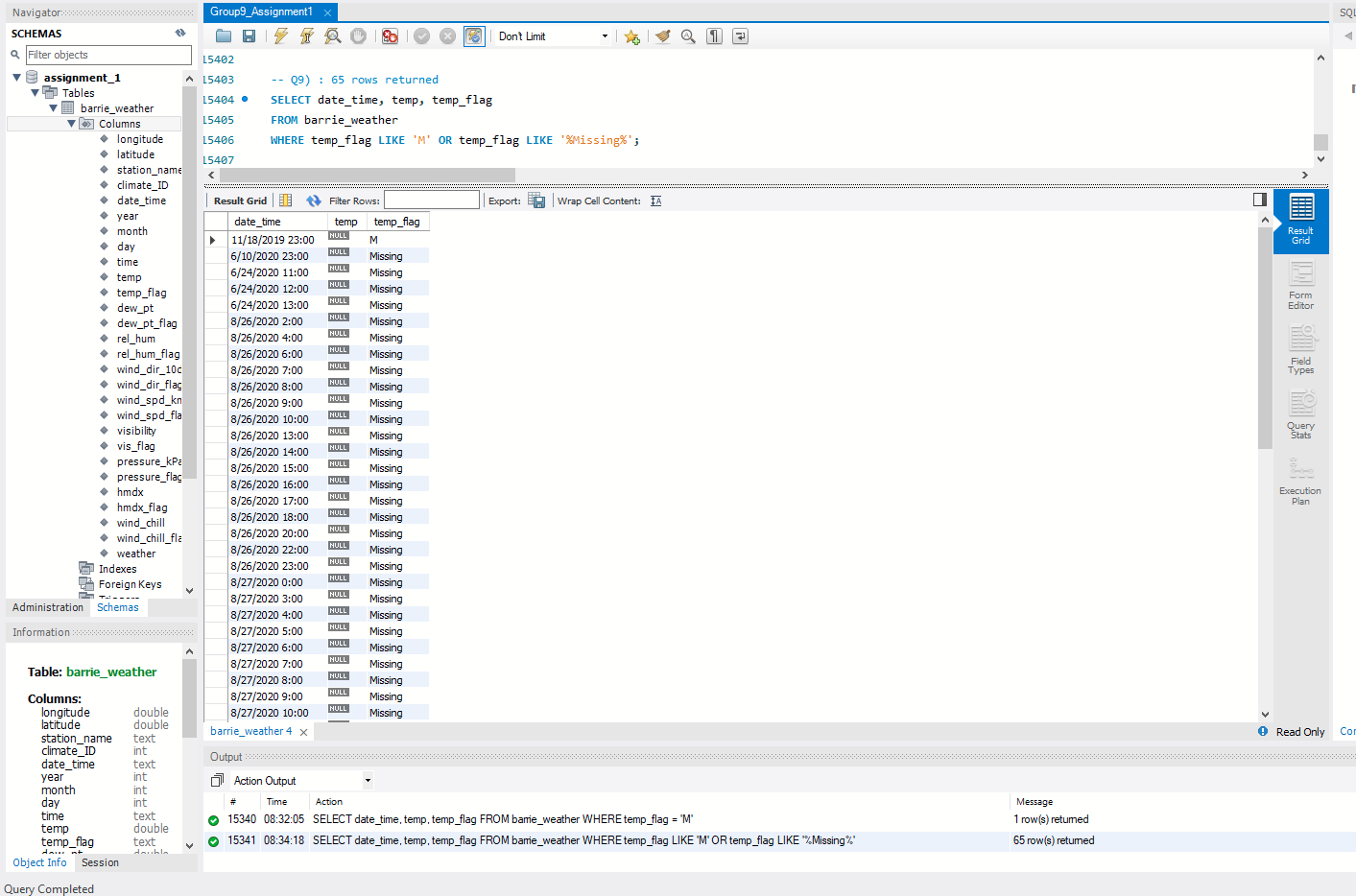


\* The date\_time returned is: date\_time: **11/18/2019 23:00**

1. Return the same results as in Question 8, except construct the query to return both ‘M’ and ‘Missing’ values.

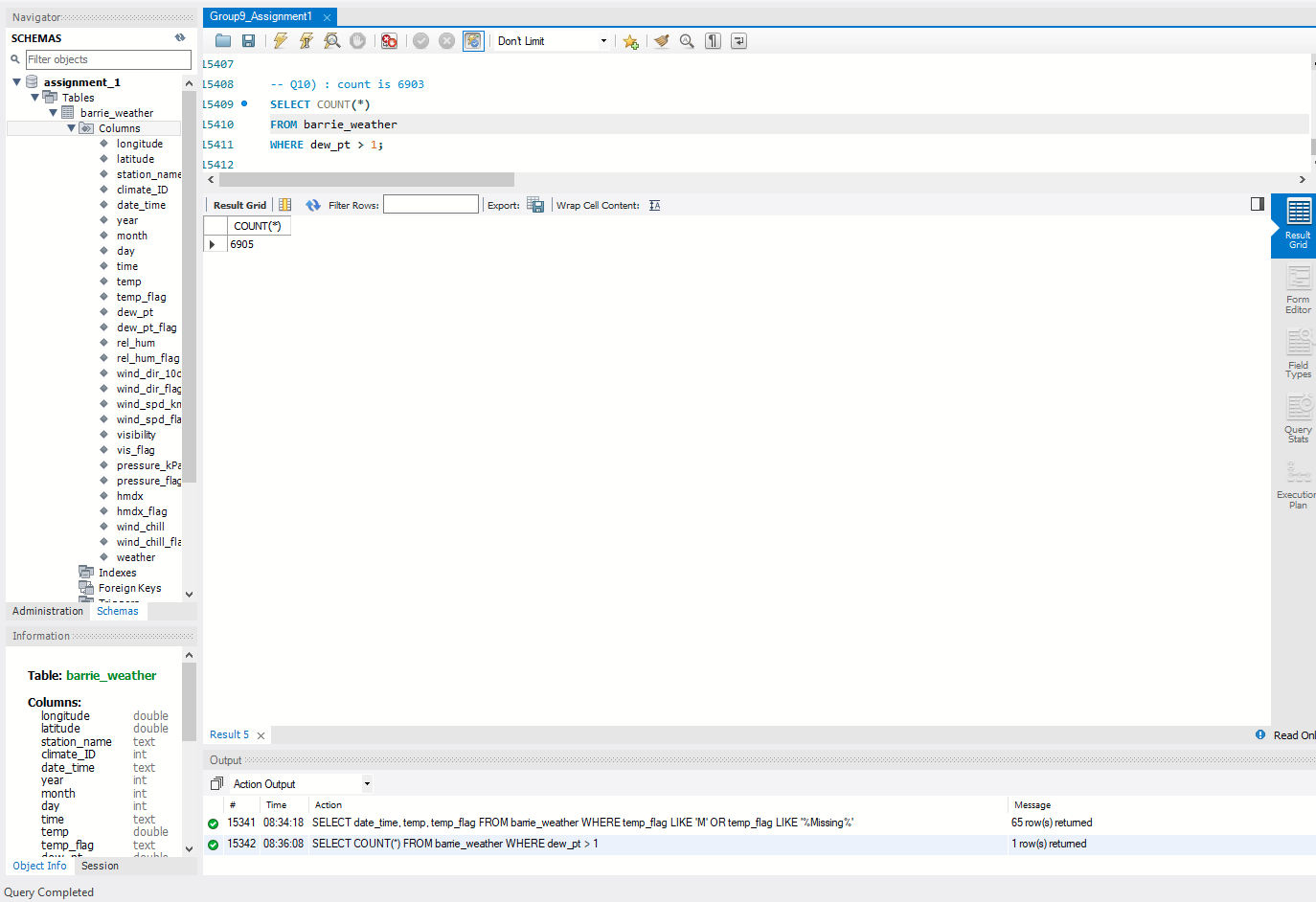
Note: There are a number of ways of doing this, please use LIKE.

|  |
| --- |
| -- Q9) : Find the same data as in Q8, except where temp\_flag is 'M' and 'Missing'  -- 65 rows returned SELECT date\_time, temp, temp\_flag FROM barrie\_weather WHERE temp\_flag LIKE 'M' OR temp\_flag LIKE '%Missing%'; |



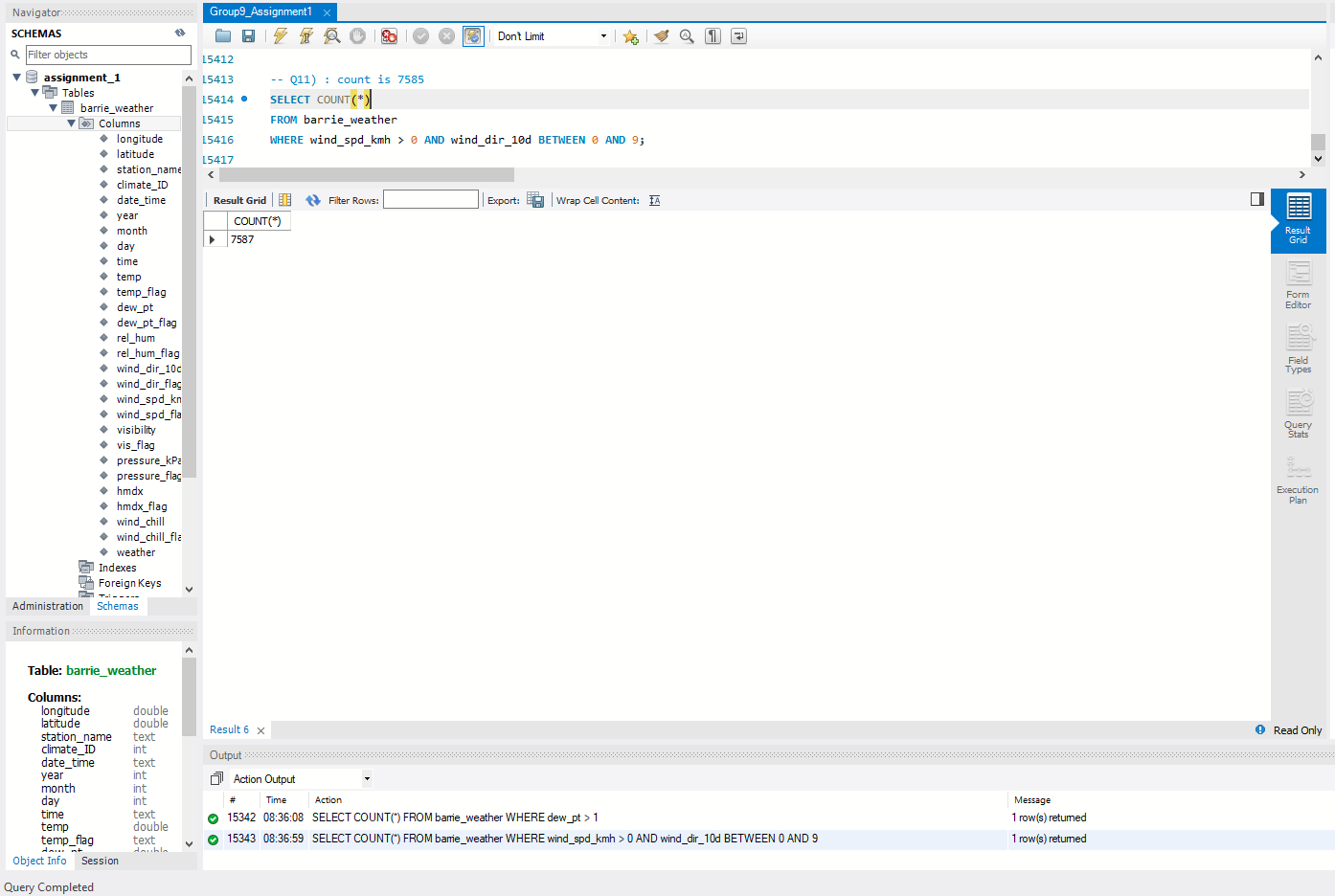
1. Count (SELECT COUNT) the number of hours contained in this dataset, where the dew point temperature (dew\_pt) is greater than one.

|  |
| --- |
| -- Q10) : Find the number of hours where dew\_pt is greater than 1  -- count is 6903 SELECT COUNT(\*) FROM barrie\_weather WHERE dew\_pt > 1; |



1. Count the number of hours that the wind was blowing (wind\_spd\_kmh is greater than 0) and from a direction ranging between north and east (wind\_dir\_10d is between 0 and 9).

|  |
| --- |
| -- Q11) : Find the number of hours where wind\_spd\_kmh is greater than 0, and the wind\_dir\_10d is between north (0) and east (9)  -- count is 7585 SELECT COUNT(\*) FROM barrie\_weather WHERE wind\_spd\_kmh > 0 AND wind\_dir\_10d BETWEEN 0 AND 9; |

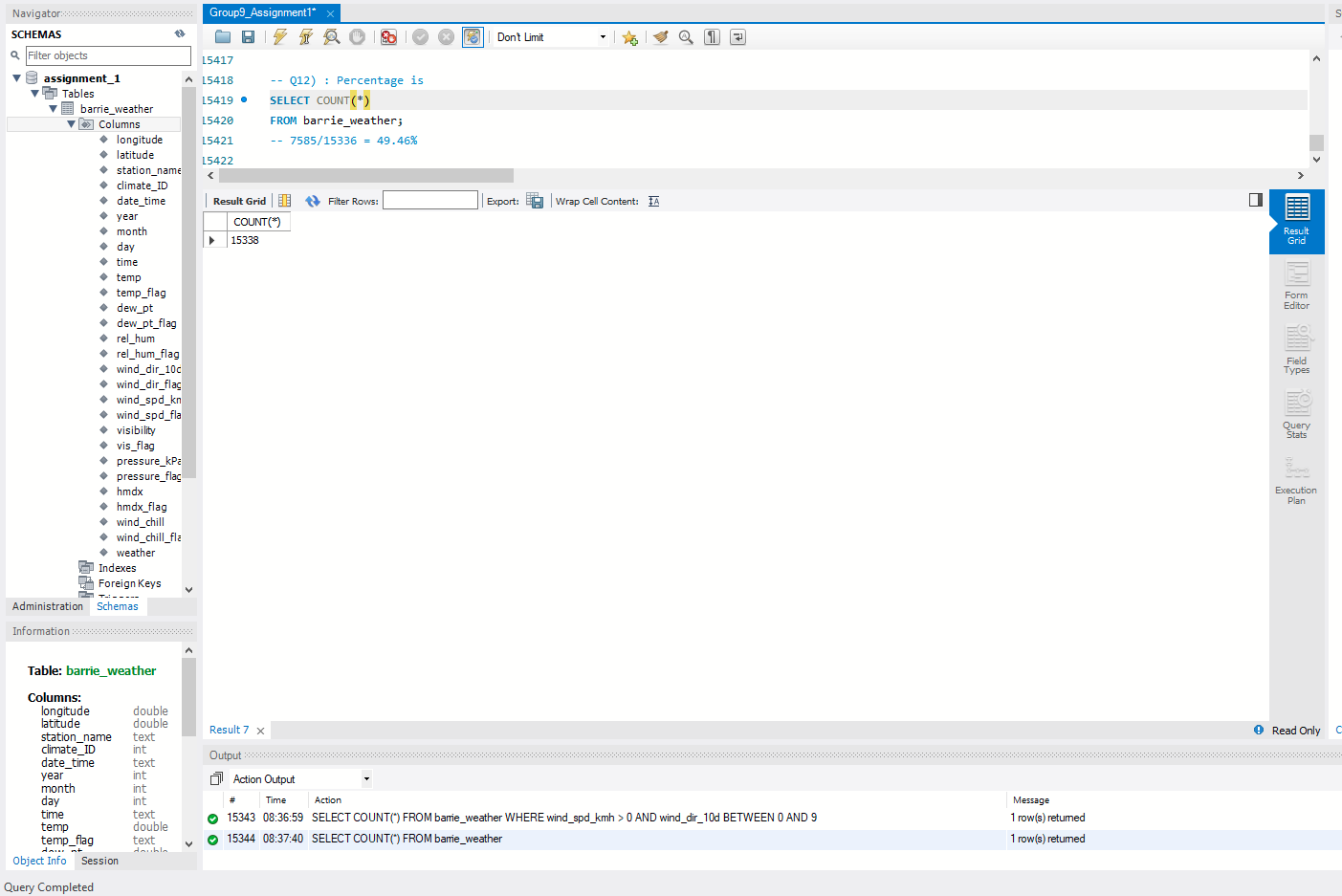


1. **In Barrie, what percent of the time are the conditions listed in Q11 true?**

Given that the count we obtained in Q11 is 7585, we can run another query for the total count in barrie\_weather in any condition, which returns 15336. We then calculate (7585/15336) \* 100 and get 49.458%.

Thus, the conditions in Q11 are true approximately 49.5% of the time.

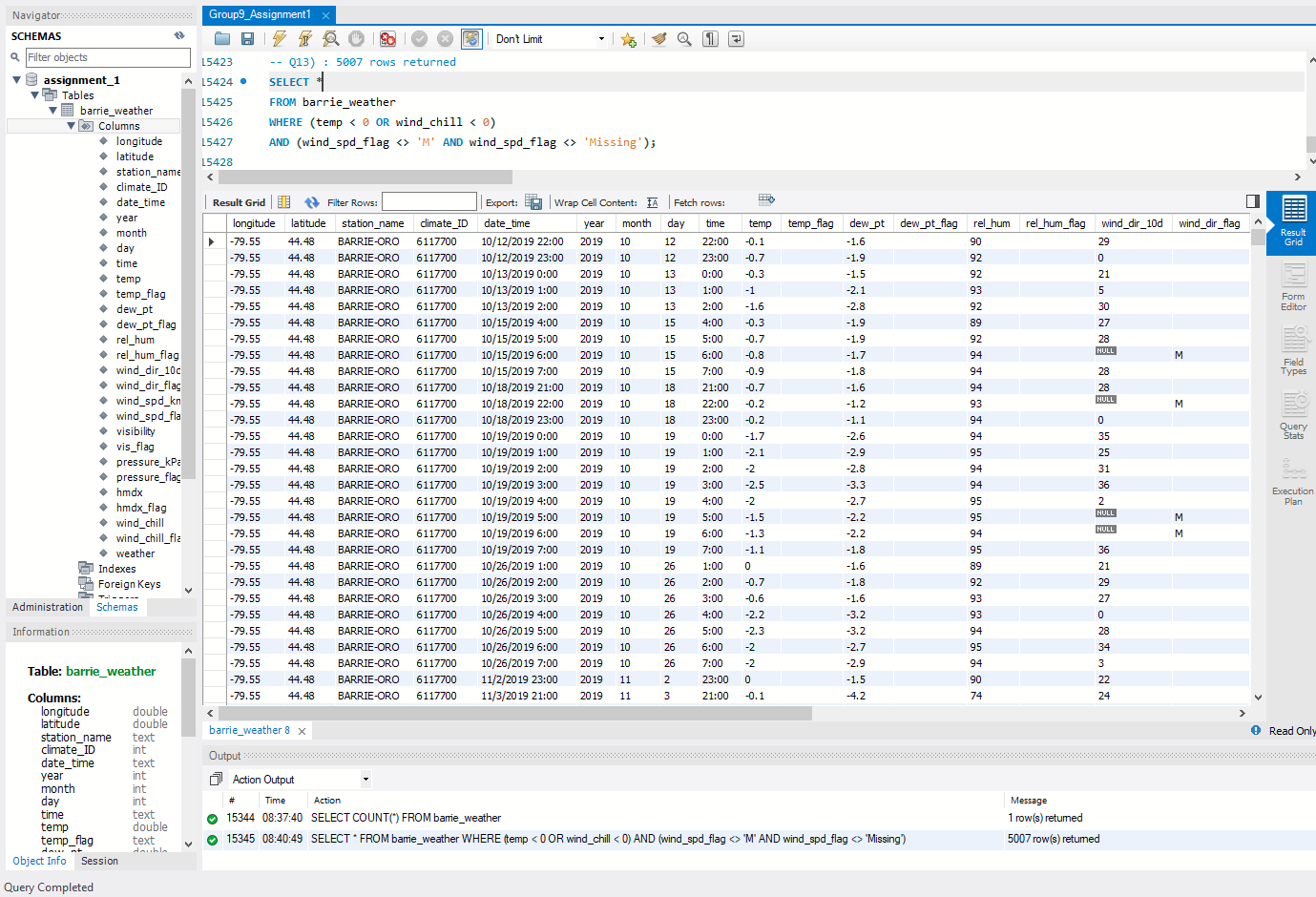
|  |
| --- |
| -- Q12) : Find how often are the conditions in Q11 true SELECT COUNT(\*) FROM barrie\_weather; -- 7585/15336 = 49.46% |



1. Select all data for every hour of data where either the temperature (temp) is below freezing, or wind chill (wind\_chill) was below zero. Additionally, the wind speed flag (wind\_spd\_flag) must indicate that there were no problems with the wind speed instrument (i.e. no M or Missing values present)

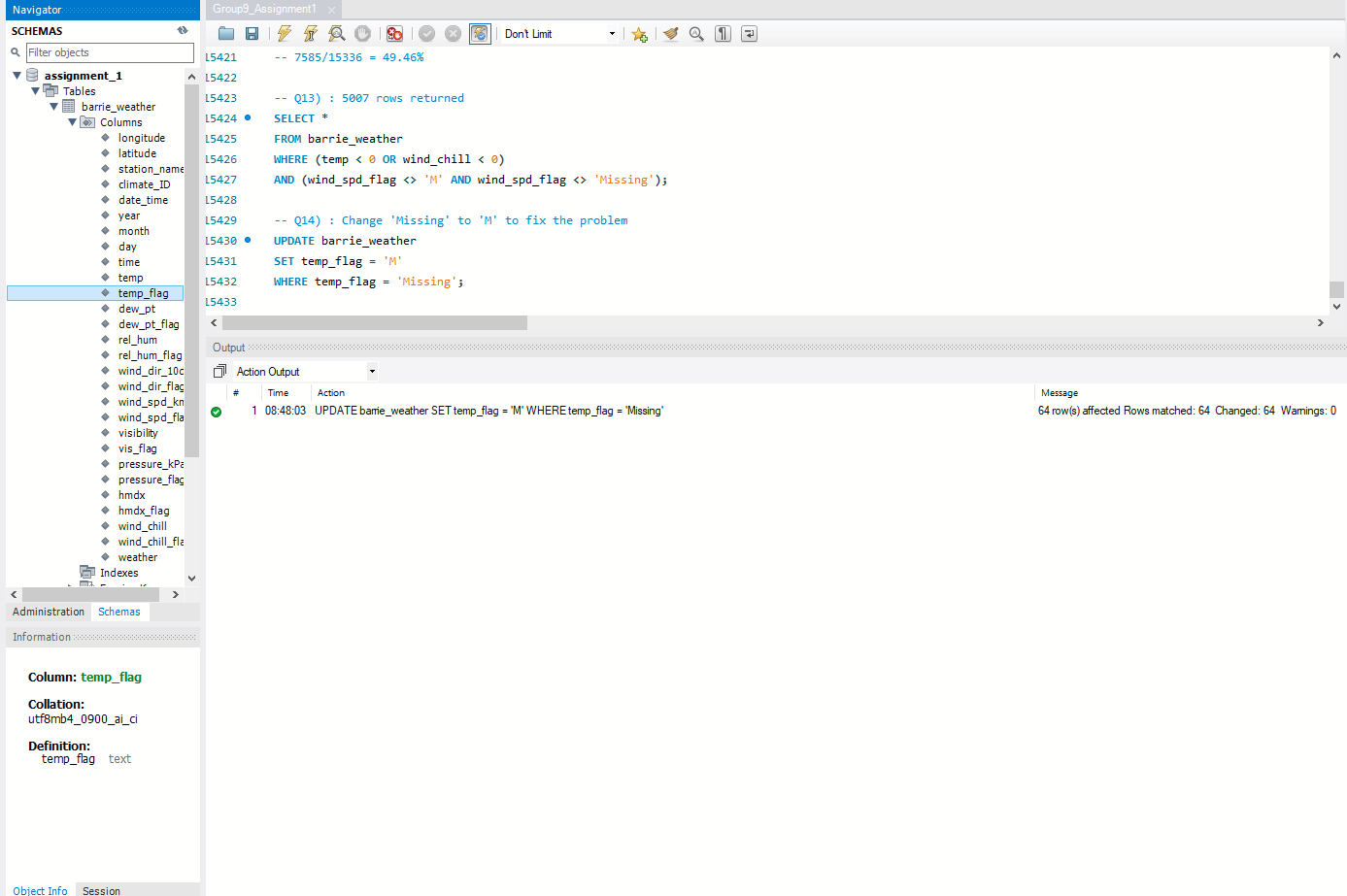
BTW: an anemometer is an instrument that measures wind speed

|  |
| --- |
| -- Q13) : Find every hour where temp is below 0 or wind\_chill is below 0, and wind\_spd\_flag must not  -- equal 'M' or 'Missing'  -- 5007 rows returned SELECT \* FROM barrie\_weather WHERE (temp < 0 OR wind\_chill < 0) AND (wind\_spd\_flag <> 'M' AND wind\_spd\_flag <> 'Missing'); |



1. Fix the M vs Missing problem in the temperature flag field by using UPDATE to change any instances of Missing values to just plain M

|  |
| --- |
| -- Q14) : Change temp\_flag 'Missing' to 'M'  UPDATE barrie\_weather SET temp\_flag = 'M' WHERE temp\_flag = 'Missing'; |



1. Use an update statement to change all ' ' values in the temp\_flag column to NULL (no quotes, just the letters N-U-L-L) this will change.

Hint: The values should end up looking like those in the humidex (hmdx) column.

FYI: The best way to store null data is using the NULL value. Look to the humidex (hmdx) column for an example of this being implemented properly.

|  |
| --- |
| -- Q15) : Change temp\_flag values with '' to NULL UPDATE barrie\_weather SET temp\_flag = NULL WHERE temp\_flag = ''; |

