**Project Name: -Blue Waters**

**Name: - Parveez Ali**

**Database systems-CPSC-50900-005**

**ID: L30062935**

**Assignment: Week 5 Project Part 4b**

**GitHub link: -**

**Current Date: -12/01/2022**

**Part 4b**

**Write the SQL commands for twelve queries. Two queries should be insert statements, two should update statements, one should be a delete statement, one should be a simple select statement that selects a subset of the rows and columns from one table, two queries should be a select statements that select data from a joining of two tables, three queries should use summary functions to generate statistics about the data, and one query should be a multi-table query. Show the queries and screenshots of the results in your Word document report and save your queries in a commented SQL script to GitHub.**

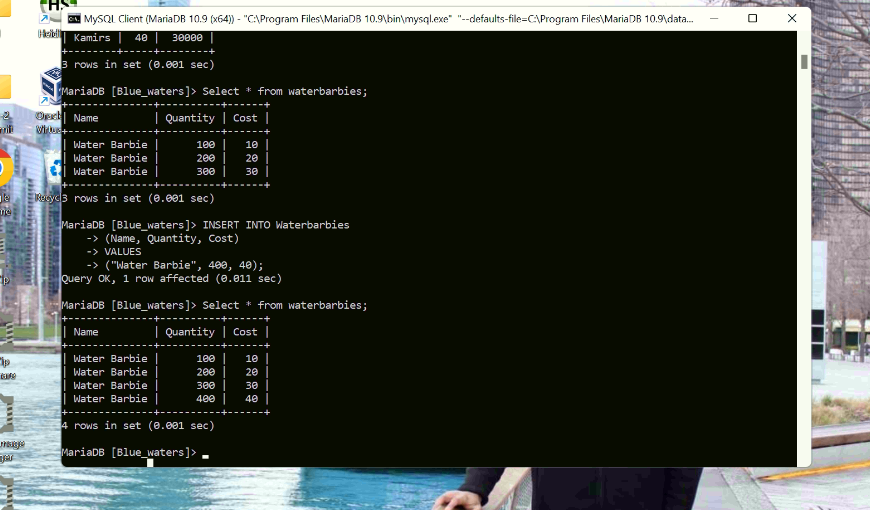
**1. Inserting record**

INSERT INTO Waterbarbies

(Name, Quantity, Cost)

VALUES

("Water Barbie", 400, 40.00);



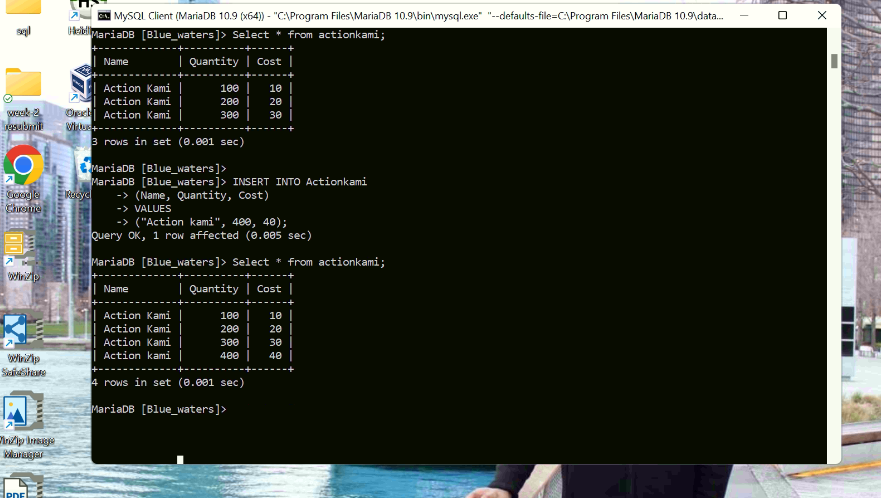
**2. Inserting record**

INSERT INTO Actionkami

(Name, Quantity, Cost)

VALUES

(“Action kami”, 400, 40.00);

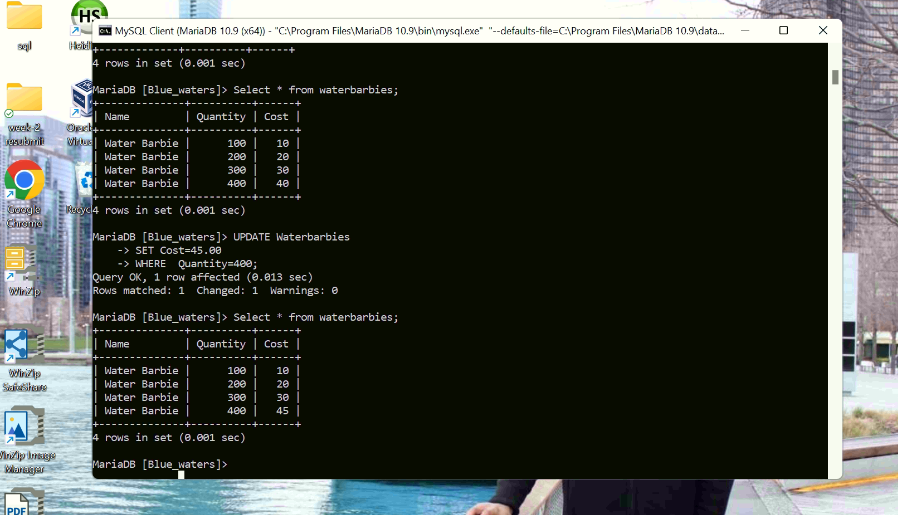


**3. updating Waterbarbies**

UPDATE Waterbarbies

SET Cost=45.00

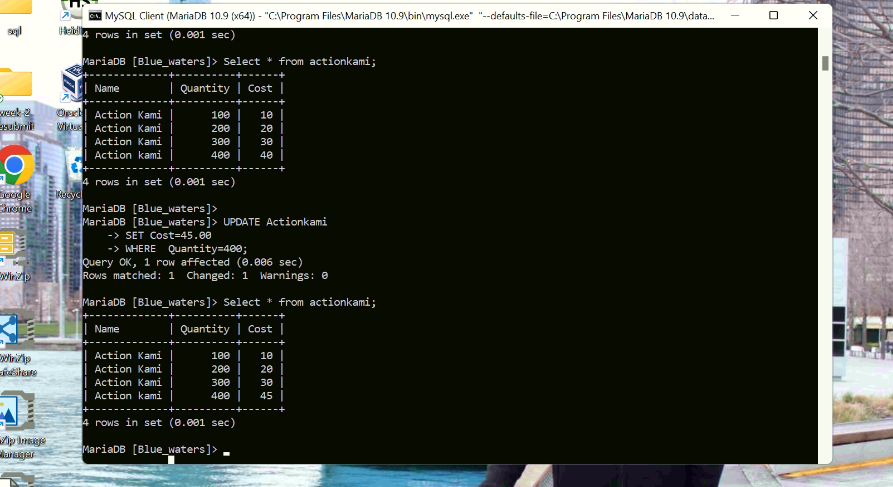
WHERE Quantity=400;



**4. updating Actionkami**

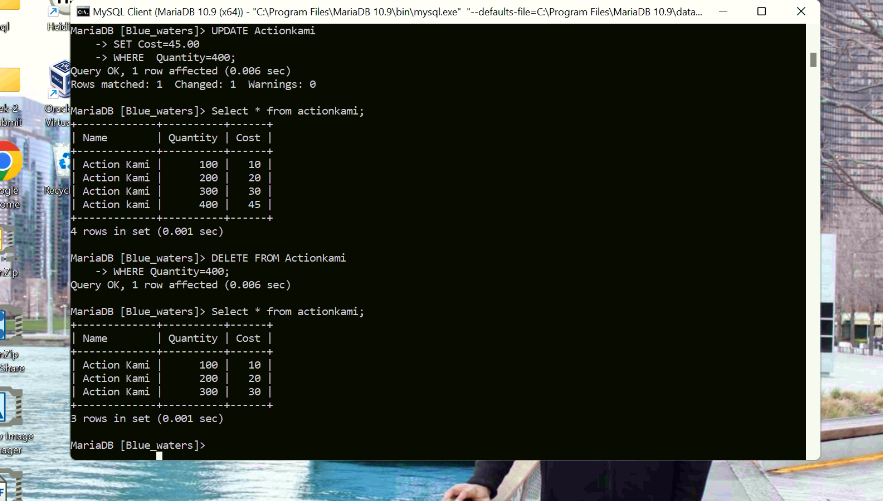
UPDATE Actionkami

SET Cost=45.00 WHERE Quantity=400;



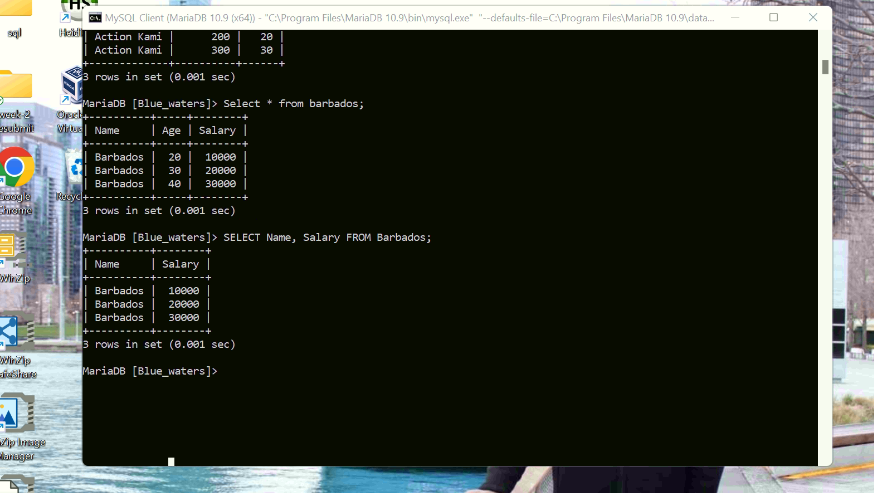
**5. Deleting record**

DELETE FROM Actionkami WHERE Quantity=400;



**6. Select statement**

SELECT Name, Salary FROM Barbados;

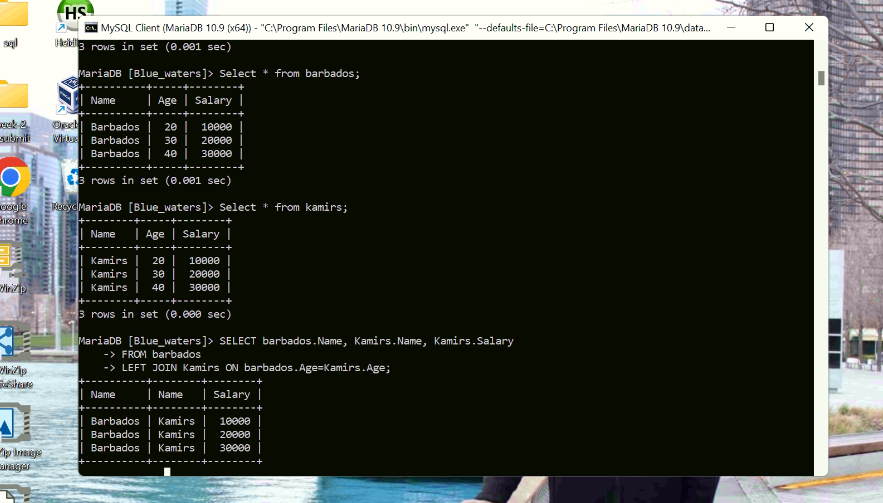


**7. LEFT JOIN**

SELECT barbados.Name, Kamirs.Name, Kamirs.Salary

FROM barbados

LEFT JOIN Kamirs ON barbados.Age=Kamirs.Age;

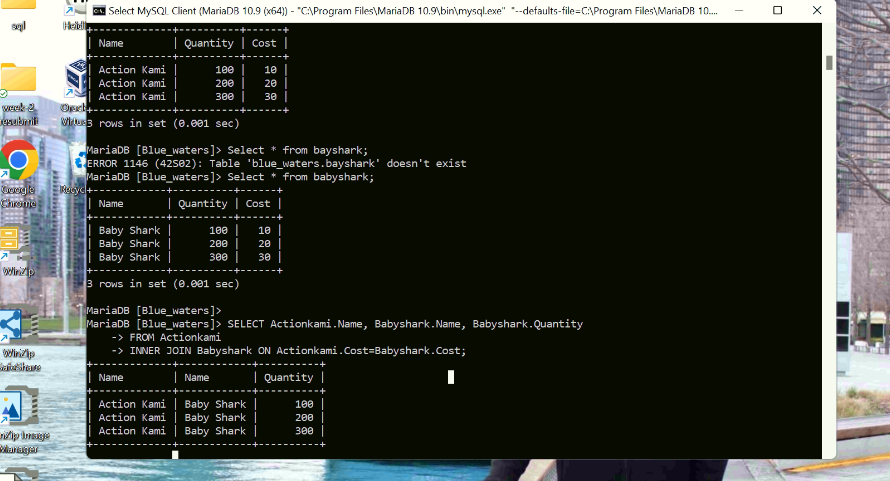


**8. INNER JOIN**

SELECT Actionkami.Name, Babyshark.Name, Babyshark.Quantity

FROM Actionkami

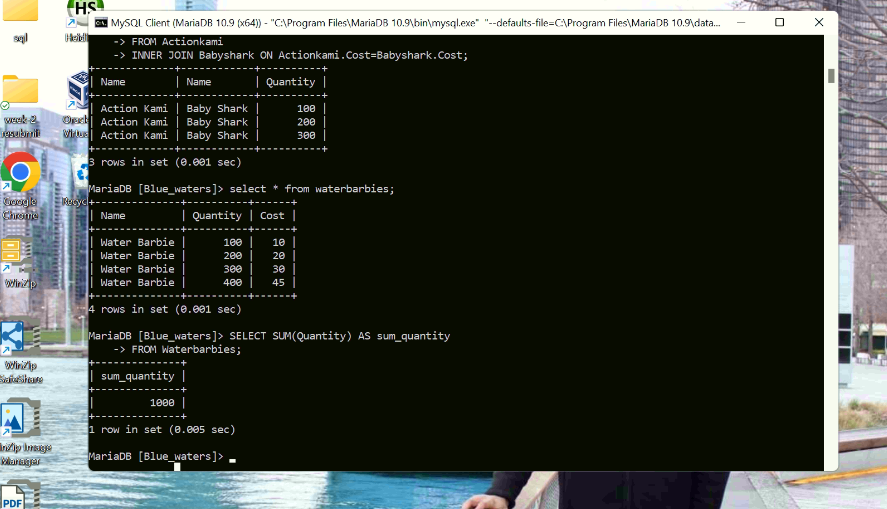
INNER JOIN Babyshark ON Actionkami.Cost=Babyshark.Cost;



**9. Summary function 1:**

SELECT SUM(Quantity) AS sum\_quantity

FROM Waterbarbies;



**10. Summary function 2:**

SELECT COUNT(\*) AS\_TOTAL

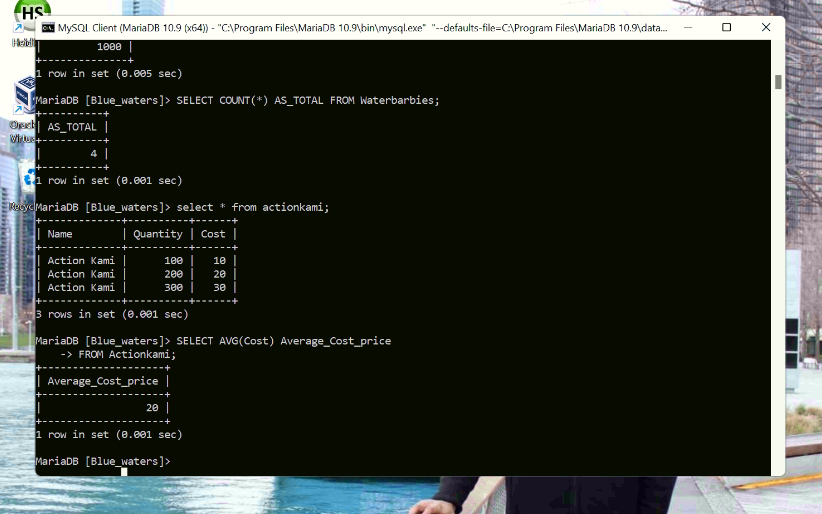
FROM Waterbarbies;



**11. Summary function 3:**

SELECT AVG(Cost) Average\_Cost\_price

FROM Actionkami;



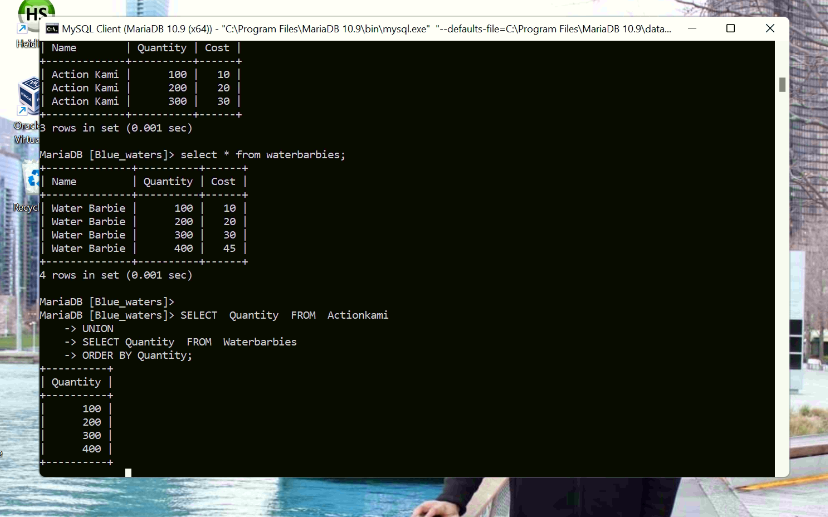
**12. Multi-table Query:**

SELECT Quantity FROM Actionkami

UNION

SELECT Quantity FROM Waterbarbies

ORDER BY Quantity;

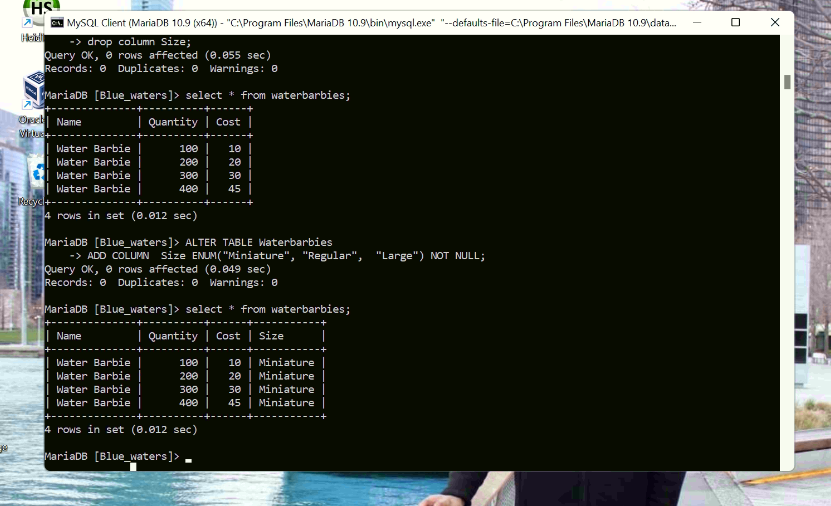


**1. Add a new column of type ‘ENUM’. Document in your report what the purpose of this column is, and what the choices represent. Add a constraint that prevents null values. Take screen captures of your table before and after the column is created**

**ADDING NEW COLUMN WITH ENUM DATATYPE:**

ALTER TABLE Waterbarbies

ADD COLUMN Size ENUM(“Miniature”, “Regular”, “Large”) NOT NULL;

****

**2. Insert a new record of data into the same table that you added the new column. This requires two steps:**

**a) show that the system throws an error if you try to leave the required ‘enum’ value blank.**

**b) modify your insert to include a valid entry so that the data works properly.**

**a)** INSERT INTO Waterbarbies

(Name, Quantity, Cost, Size)

VALUES

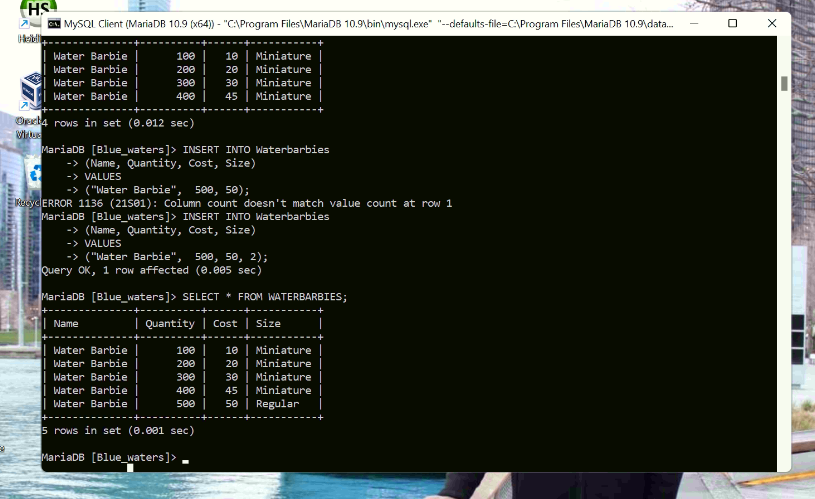
(“Water Barbie”, 500, 50);

**b)** INSERT INTO Waterbarbies

(Name, Quantity, Cost, Size)

VALUES

(“Water Barbie”, 500, 50, 2);

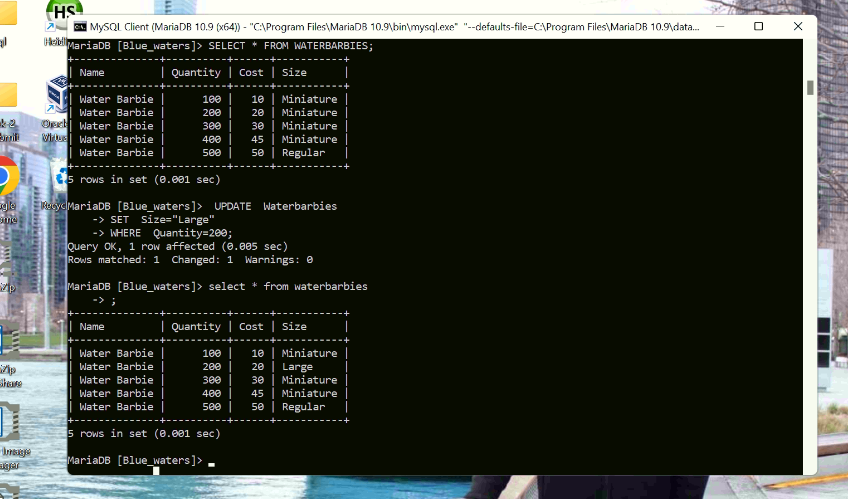


**3.**  **In your report, explain what changes you can make to each column of the record you just added in #2. Then, modify that data to show those changes. Include a screen capture that shows the ‘after’ result. The ‘before’ state is already documented from step 2.**

UPDATE Waterbarbies

SET Size=”Large”

WHERE Quantity=200;

****

**4.** **Delete the new record. Use the ‘where’ clause to specifically select just the new record. Show a screen capture of the command and the ‘after’ result.**

DELETE FROM Waterbarbies

WHERE Quantity=500;

****

**5. Demonstrate the use of the ‘AUTO\_INCREMENT’ constraint by adding a record to a table with this feature. If you do not have this feature already enabled, then alter a table to include a new column. Show a screen capture of the table before you add the new record, and another screen capture after you add the new record to demonstrate the autoincrement.**

AUTO\_ INCREMENT generates a unique number when a new record is inserted into a table. This is frequently the primary key field that we want to be automatically produced each time a new record is inserted. To enable this feature, I changed the employee field using the alter command.

ALTER TABLE Actionkami

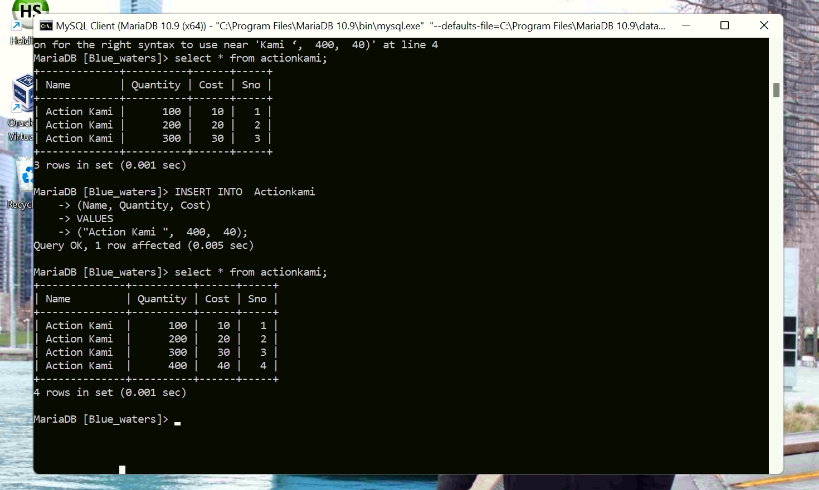
ADD COLUMN Sno INT AUTO\_INCREMENT NOT NULL PRIMARY KEY;

INSERT INTO Actionkami

(Name, Quantity, Cost)

VALUES

(“Action Kami “, 400, 40);



**6. Repeat step 5, except demonstrate the use of the ‘DEFAULT’ constraint by adding a record to a table with this feature. You may need to add a column with this feature if you don’t already have it. Prove that adding a new record of data without this value during entry will still get the default value after your ‘insert’ command executes.**

ALTER TABLE Actionkami

ALTER Name SET DEFAULT “Action Kami “;

INSERT INTO Actionkami

(Quantity, Cost)

VALUES

(500, 50);

