**Clothing Database Management System**

Shaik Masthan Vali

<https://github.com/Masthan9712/True-Style-Clothing.git>

May 1, 2022

I have created a business for selling textiles known as “True Style Clothing”.

For this business I will be creating a new database which stores all the information regarding Employees (Employee ID, Names, Job role & Salary), Customers (Customer Name, Date of Birth, Phone, Email & Address), Transactions (Transaction amount & transaction method), Costume (Costume ID, Brand, Quantity, Price & Size) and Categories (Men, Women, Kids). This data is very essential for running the store to keep track of all the orders, employees and their salary pay, offering discounts to the regular customers and maintain the stock up to date. If the stock is less, more collection needs to be ordered. Everyday transactions gives the overview of the most salable and liked costumes which helps to order those costumes more in quantity.

I have to enter the employee data at the time of their joining and it contains all details about their role and pay. Customer data is generated when they purchase any costume from the store.

As I am running this store, I should be able to view all the data to get a detailed overview of how the business is running and what needs to be taken care of. Store managers needs access to all the employee data and the costumes data. In case if any particular costume is running low in that store they can check whether it is available in the near store. Sales team needs to have access to the transactions data to check how well the costumes are getting sold. If the business is running good, sales team can make analysis to put some offers on the most salable costumes and even encourage customers to buy it.

**Week 2 - Project part 2**

For my True Style Clothing business application I will be creating 6 entities and described them below:

1. Customer Info - This entity describes the basic information of a customer like Customer First Name, Customer Last Name, Customer Email, Customer DOB, Customer Orders.
2. Employee Info - This entity describes the employee information like Employee ID, Employee First Name, Employee Last Name, Employee Role, Employee Salary, Employee Sales.
3. Costume Info - This entity describes the Costume ID, Brand, Shirts Price, Trousers Price.
4. Category info - This describes the available sizes for all categories. S - small, M - Medium and L - Large, Gender - Male and Female and Age.
5. Vendor Info - This entity describes the Vendor name, Delivery Date, Shirts Quantity, Trousers Quantity.
6. Sales Info - This entity describes the sales details like Date, Number of costumes sold, Total sale value.

Below is the chart representation for my entities:

|  |  |
| --- | --- |
| **Customer Info** | **Data type** |
| Customer First Name | varChar[30] |
| Customer Last Name | varChar[30] |
| Customer Email | varChar[30] |
| Customer DOB | date |
| Customer Orders | Integer |

|  |  |
| --- | --- |
| **Employee Info** | **Data type** |
| Employee First Name | varChar[30] |
| Employee Last Name | varChar[30] |
| Role | varChar[30] |
| Salary | Float |
| Sales Completed | Integer |

|  |  |
| --- | --- |
| **Costume Info** | **Data type** |
| Costume ID | Integer |
| Brand | varChar[20] |
| Shirts price | Float |
| Trousers price | Float |

|  |  |
| --- | --- |
| **Categories Info** | **Data type** |
| Gender | char |
| Size | char |
| Age | int |

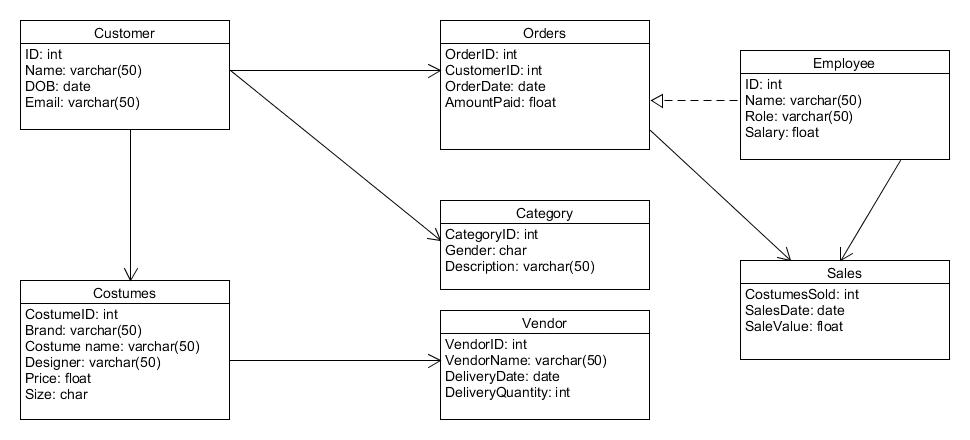
|  |  |
| --- | --- |
| **Vendor Info** | **Data type** |
| Vendor Name | varChar[20] |
| Delivery date | date |
| Shirts Quantity | Integer |
| Trousers Quantity | Integer |

|  |  |
| --- | --- |
| **Sales Info** | **Data type** |
| Date | date |
| Costumes Sold | Integer |
| Sale value | Float |

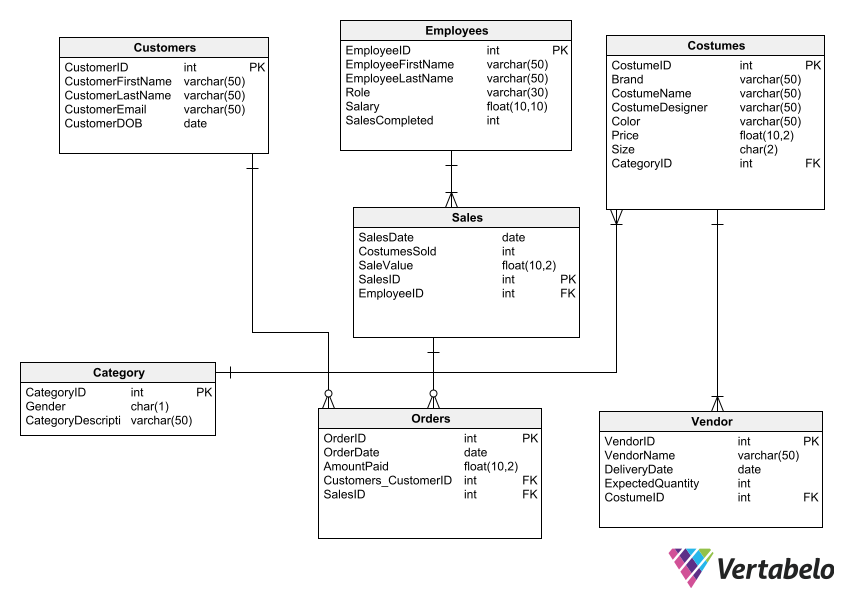
**Week 3 - Project Part 3**

**UML diagram - UMLet**

I have used the UMLet tool for drawing the UML diagram for my business application TrueStyleClothing.



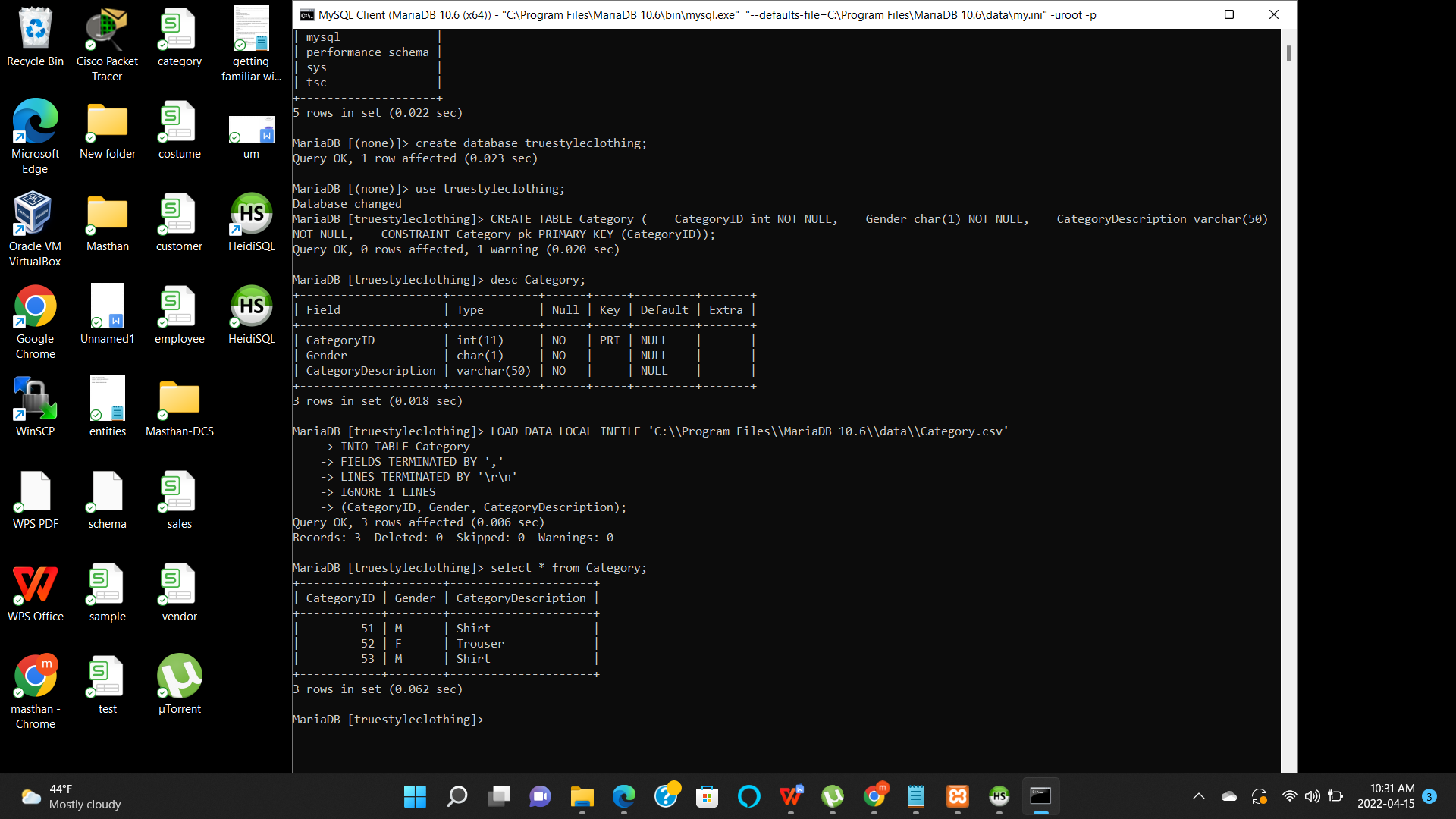
**ER Diagram - Crow’s Foot Model:**

******

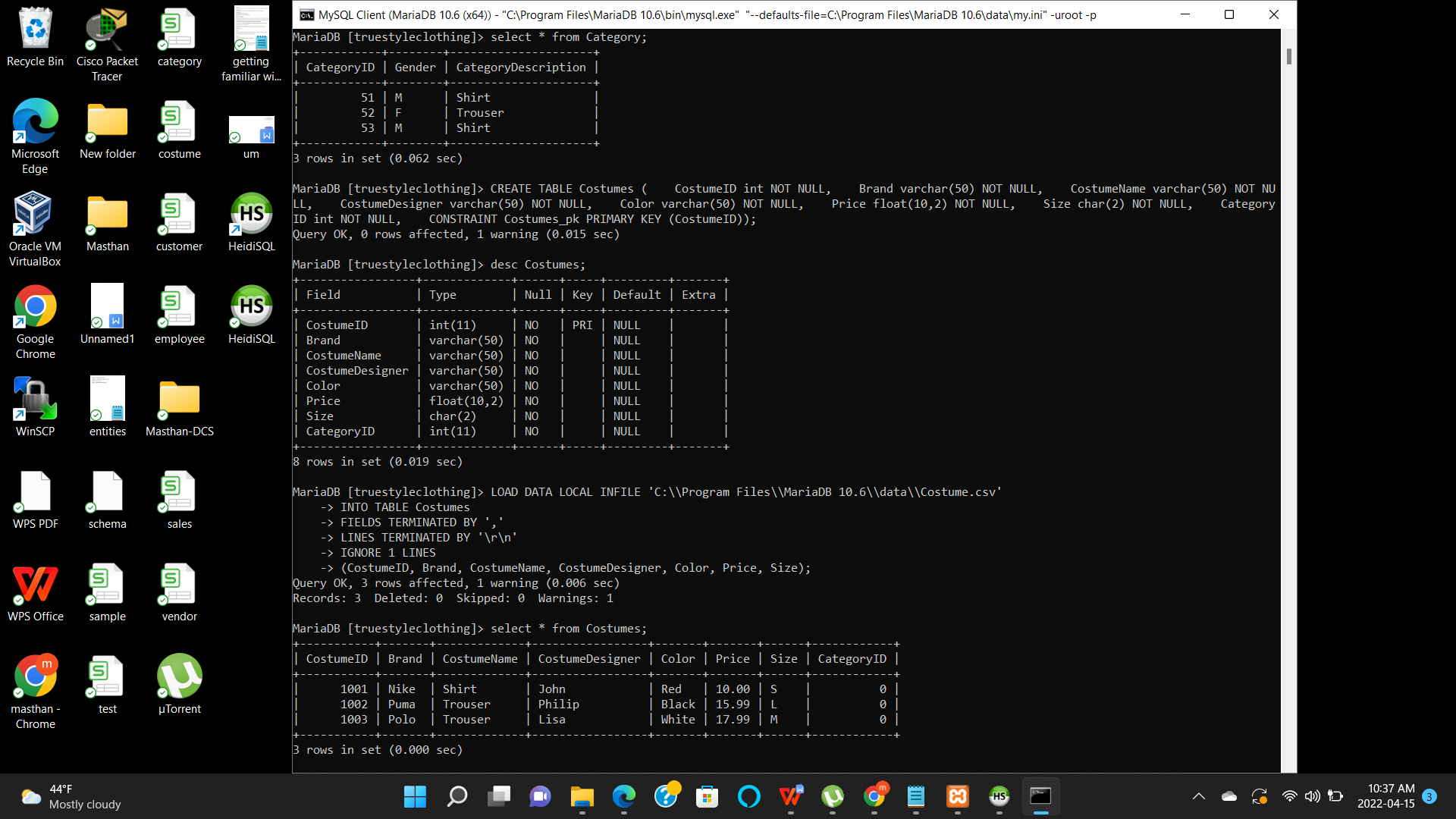
**Week 4 - Project Part 4**

In this week’s project, I am going to use the SQL query generated by Vertabelo and load the data in to tables using the csv files.

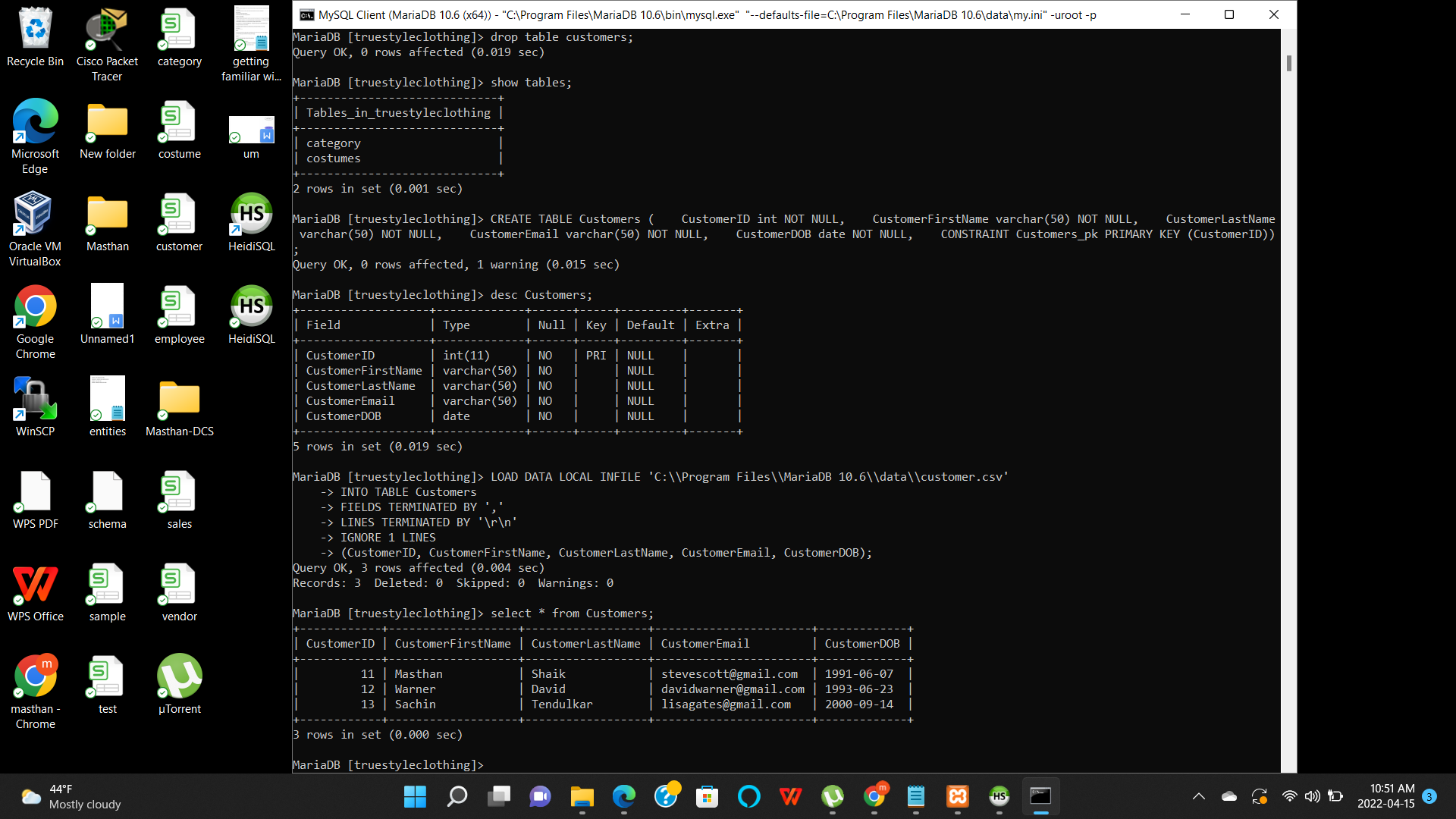
Category information is loaded into Category table using category.csv file. Fields terminated by uses the parameter “,” to represent each element, Lines terminated by uses the parameter “\r\n” to represent the end of the line, Ignore 1 lines is used to ignore the headings in the CSV file.



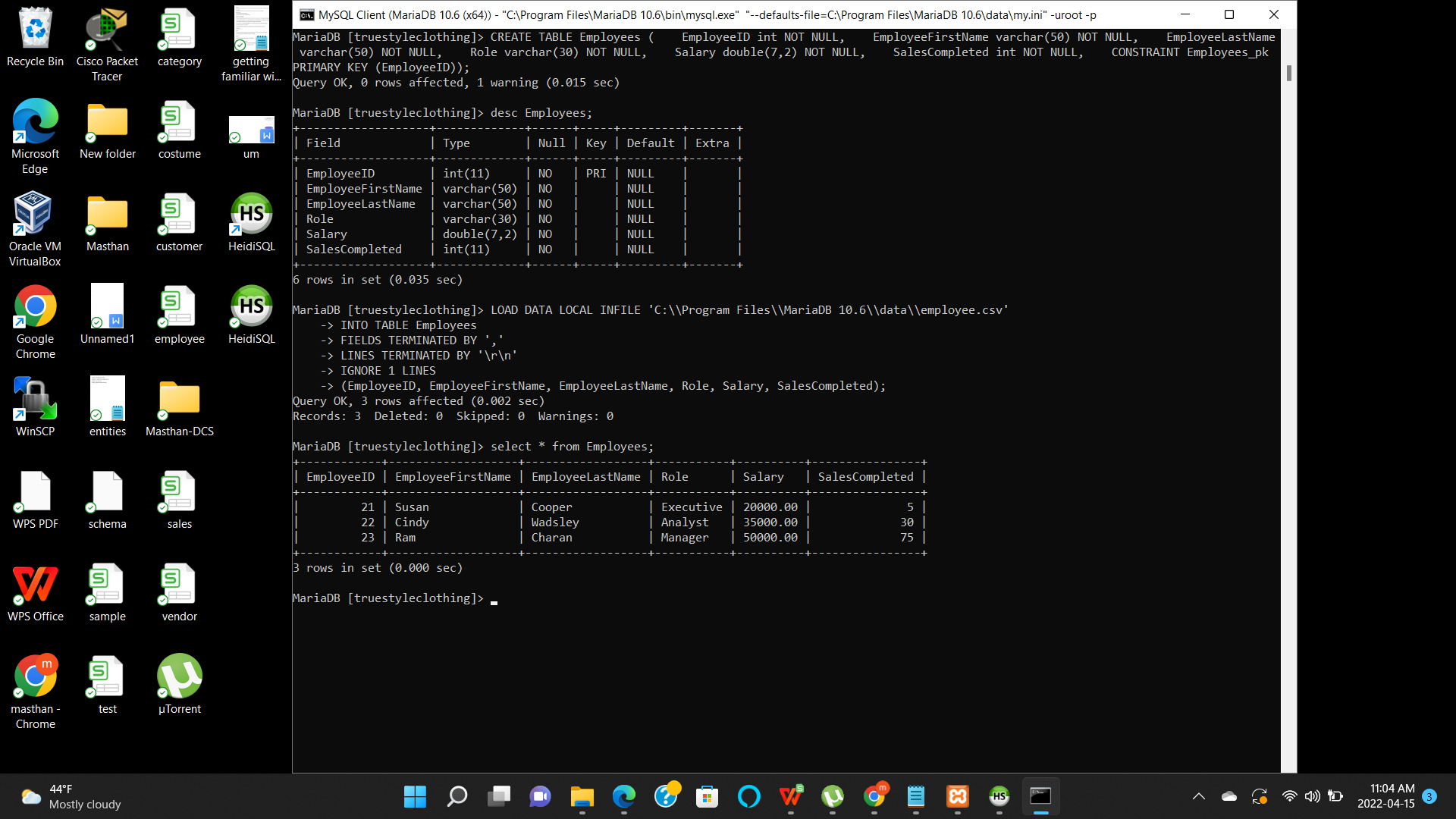
Costume information is loaded into Costumes table using costume.csv file. Fields terminated by uses the parameter “,” to represent each element, Lines terminated by using the parameter “\r\n” to represent the end of the line, Ignore 1 lines is used to ignore the headings in the CSV file.



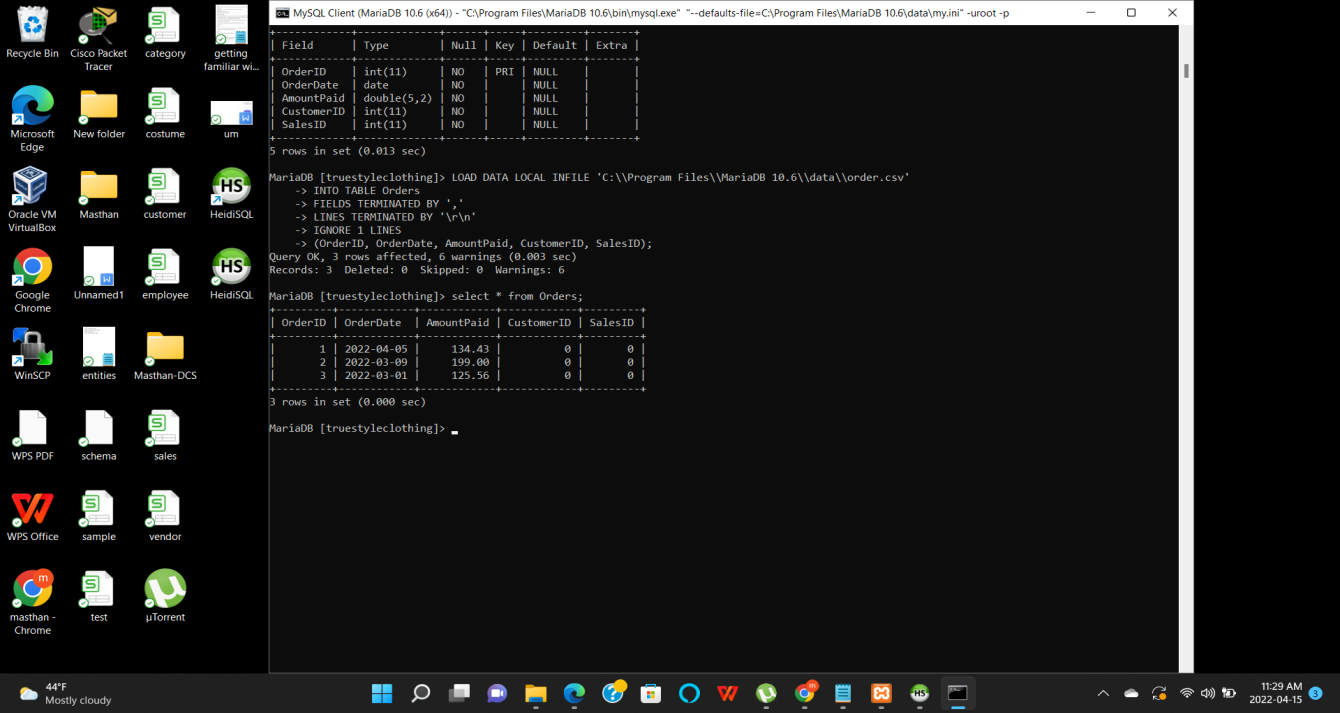
Customer information is loaded into Customers table using customer.csv file. Fields terminated by uses the parameter “,” to represent each element, Lines terminated by uses the parameter “\r\n” to represent the end of the line, Ignore 1 lines is used to ignore the headings in the CSV file.



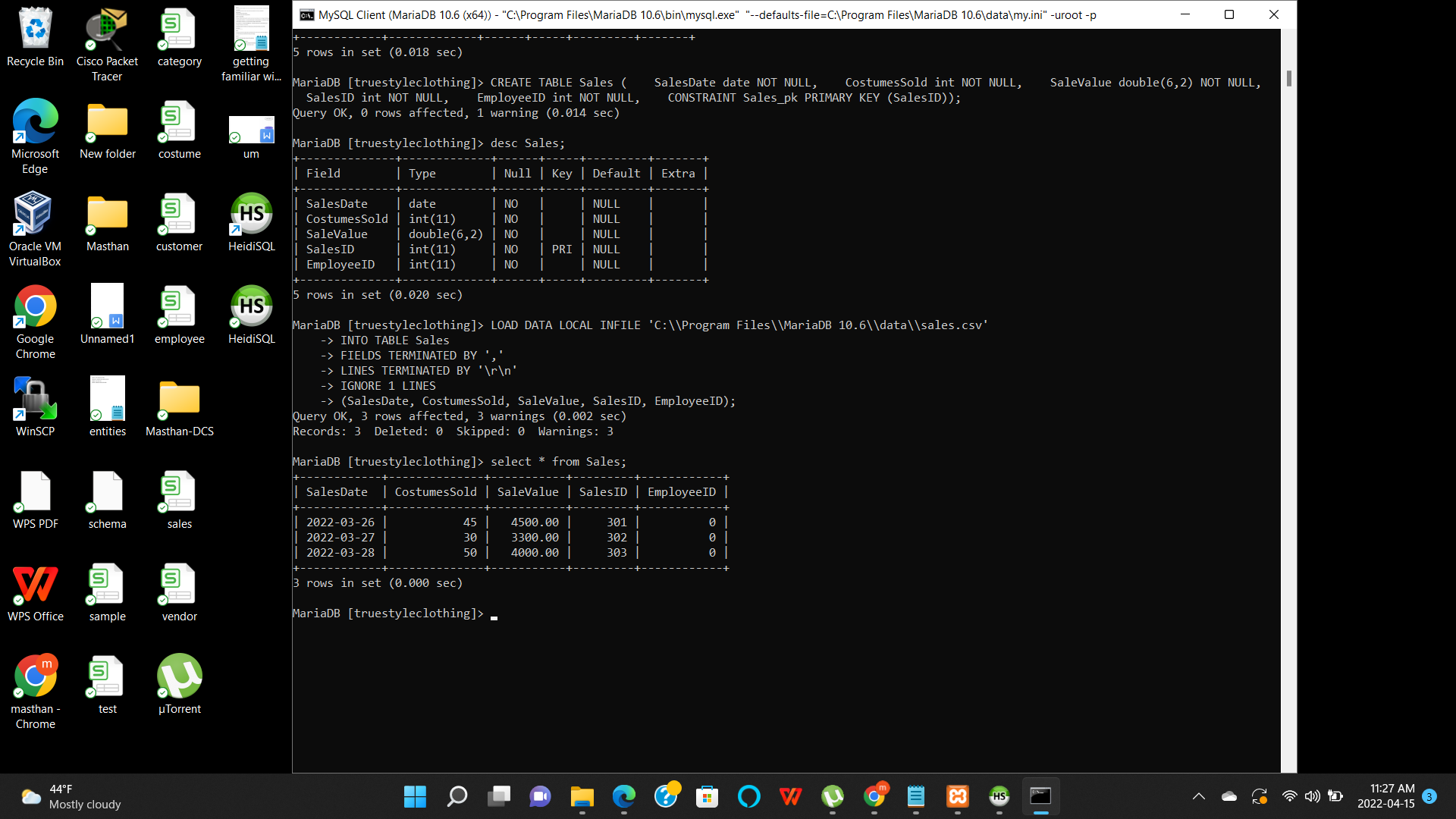
Employee information is loaded into Employees table using employee.csv file. Fields terminated by uses the parameter “,” to represent each element, Lines terminated by using the parameter “\r\n” to represent the end of the line, Ignore 1 lines is used to ignore the headings in the CSV file.



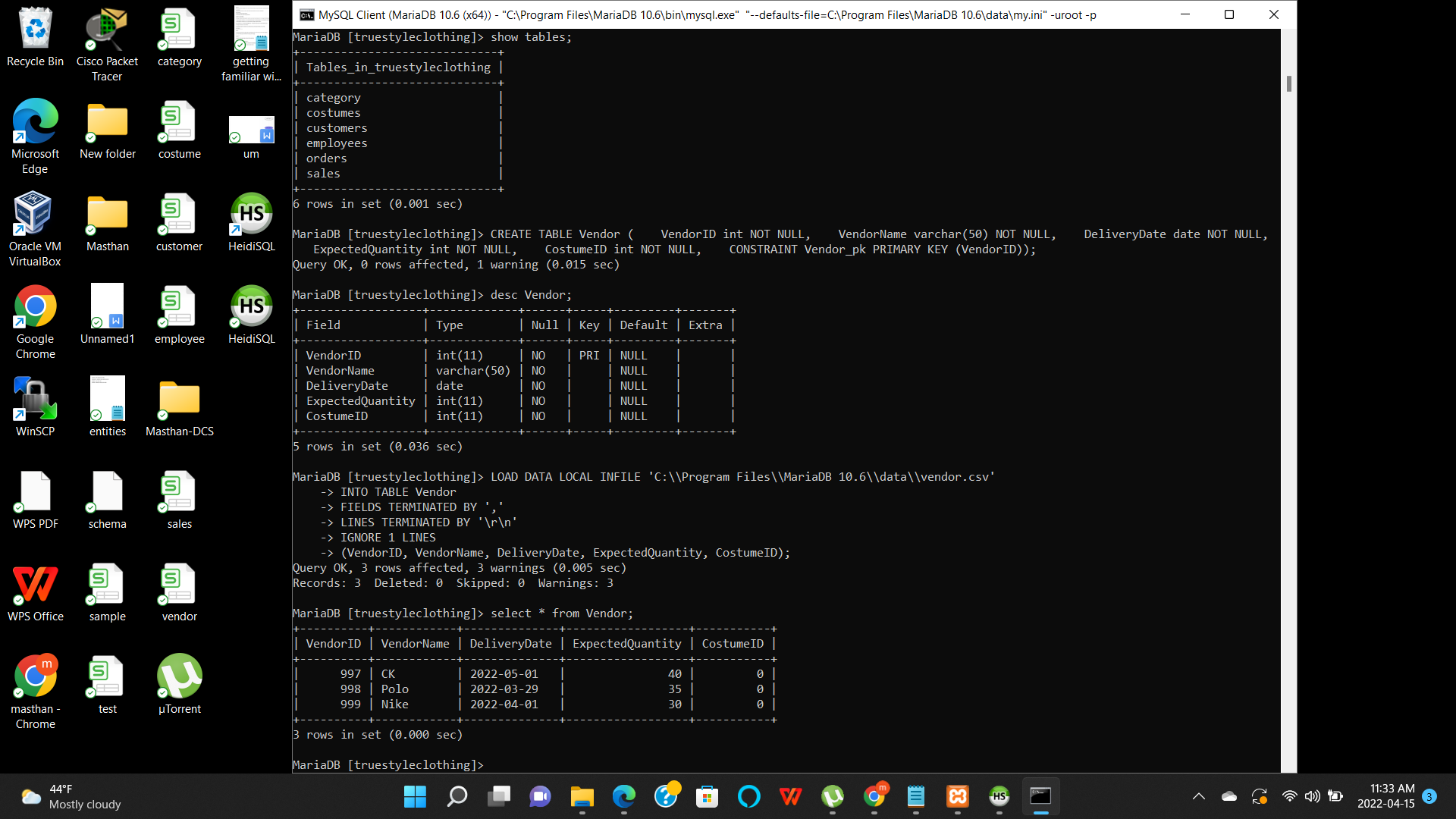
Orders information is loaded into Orders table using order.csv file. Fields terminated by uses the parameter “,” to represent each element, Lines terminated by using the parameter “\r\n” to represent the end of the line, Ignore 1 lines is used to ignore the headings in the CSV file.



Sales information is loaded into Sales table using sale.csv file. Fields terminated by uses the parameter “,” to represent each element, Lines terminated by using the parameter “\r\n” to represent the end of the line, Ignore 1 lines is used to ignore the headings in the CSV file.



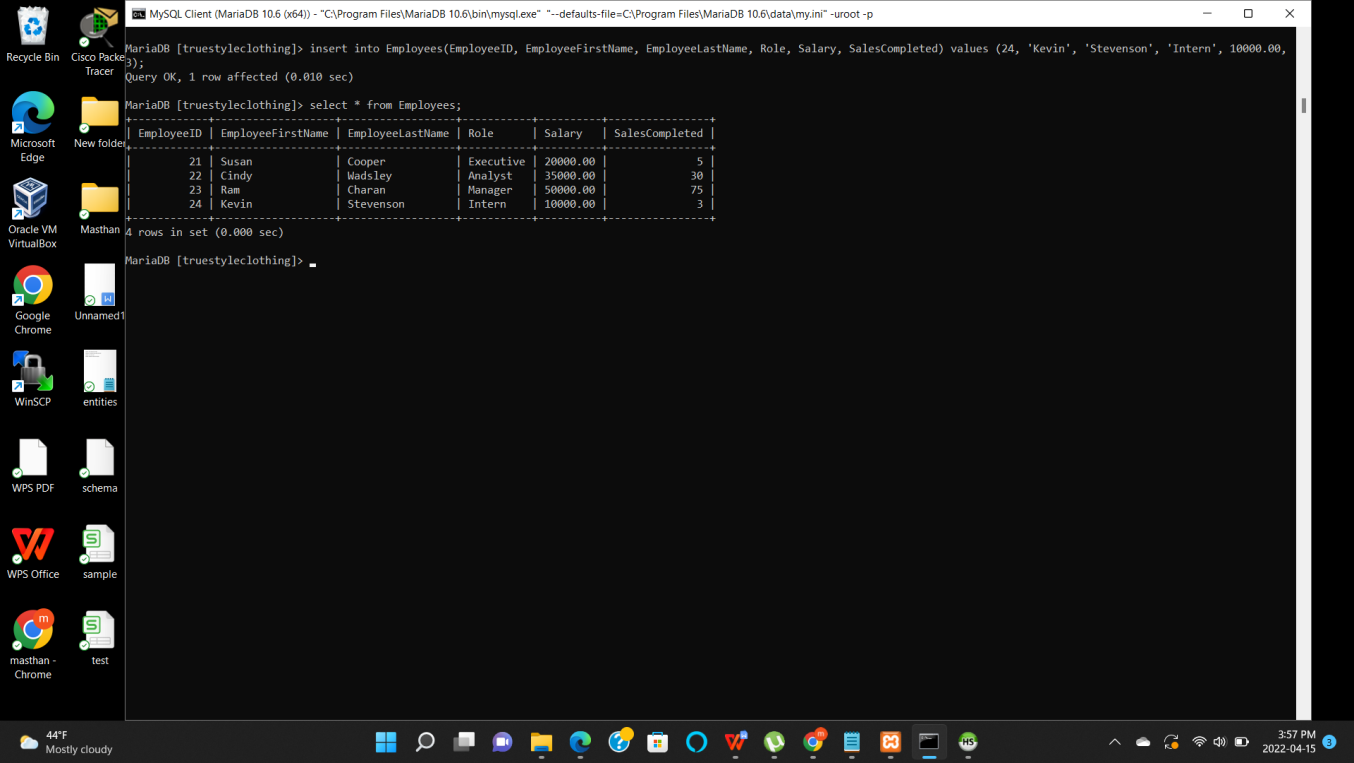
Vendor information is loaded into Vendor table using vendor.csv file. Fields terminated by uses the parameter “,” to represent each element, Lines terminated by using the parameter “\r\n” to represent the end of the line, Ignore 1 lines is used to ignore the headings in the CSV file.



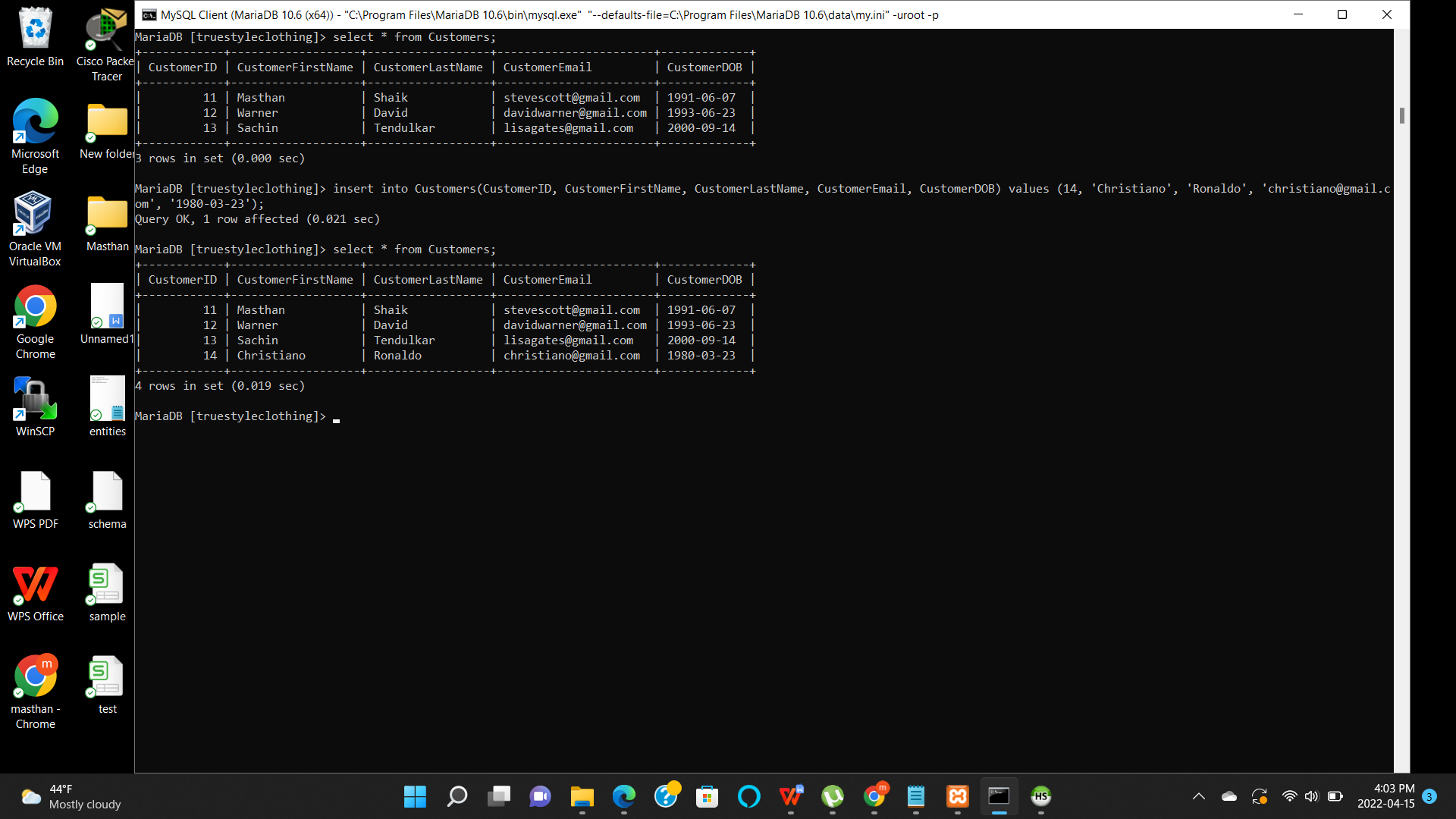
**DML Commands:**

**Insert commands:**

In the first insert query, a new record has been added to the Employees table.

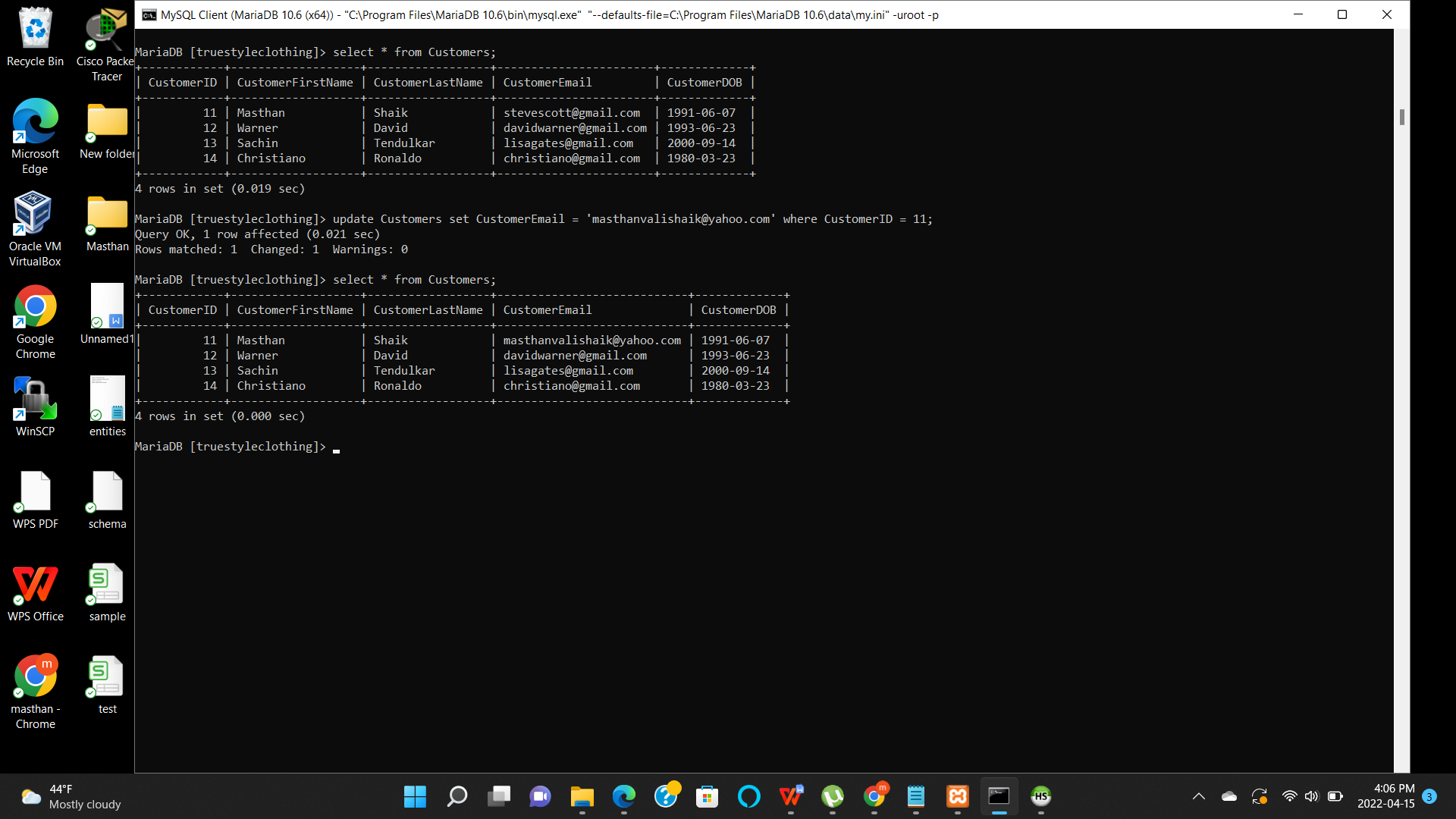


In the second insert query, a new record has been inserted into Customers table.

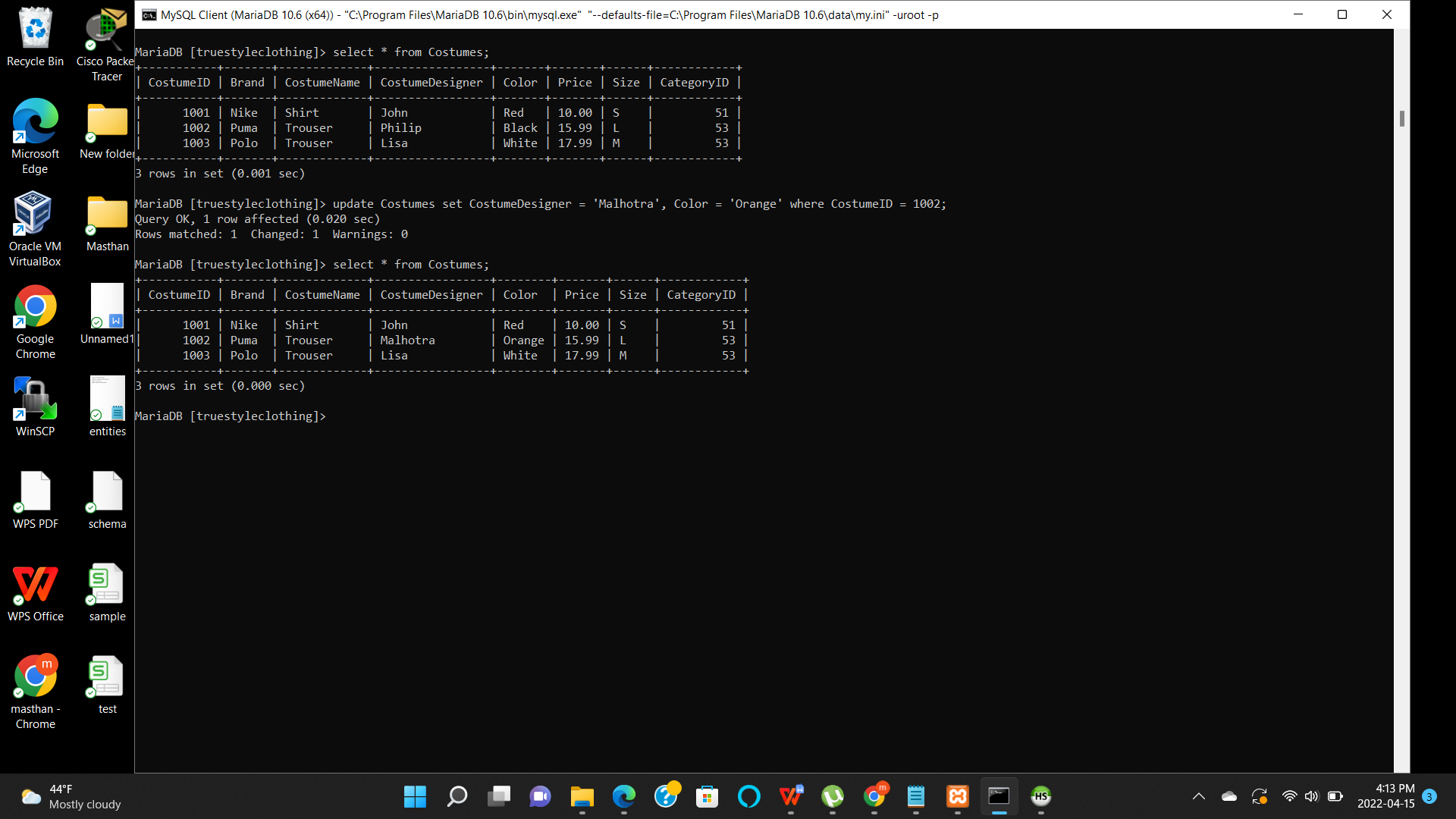


**Update commands:**

In the first update command, email address for Customer ID has changed from [stevescott@gmail.com](mailto:stevescott@gmail.com) to masthanvalishaik@yahoo.com.

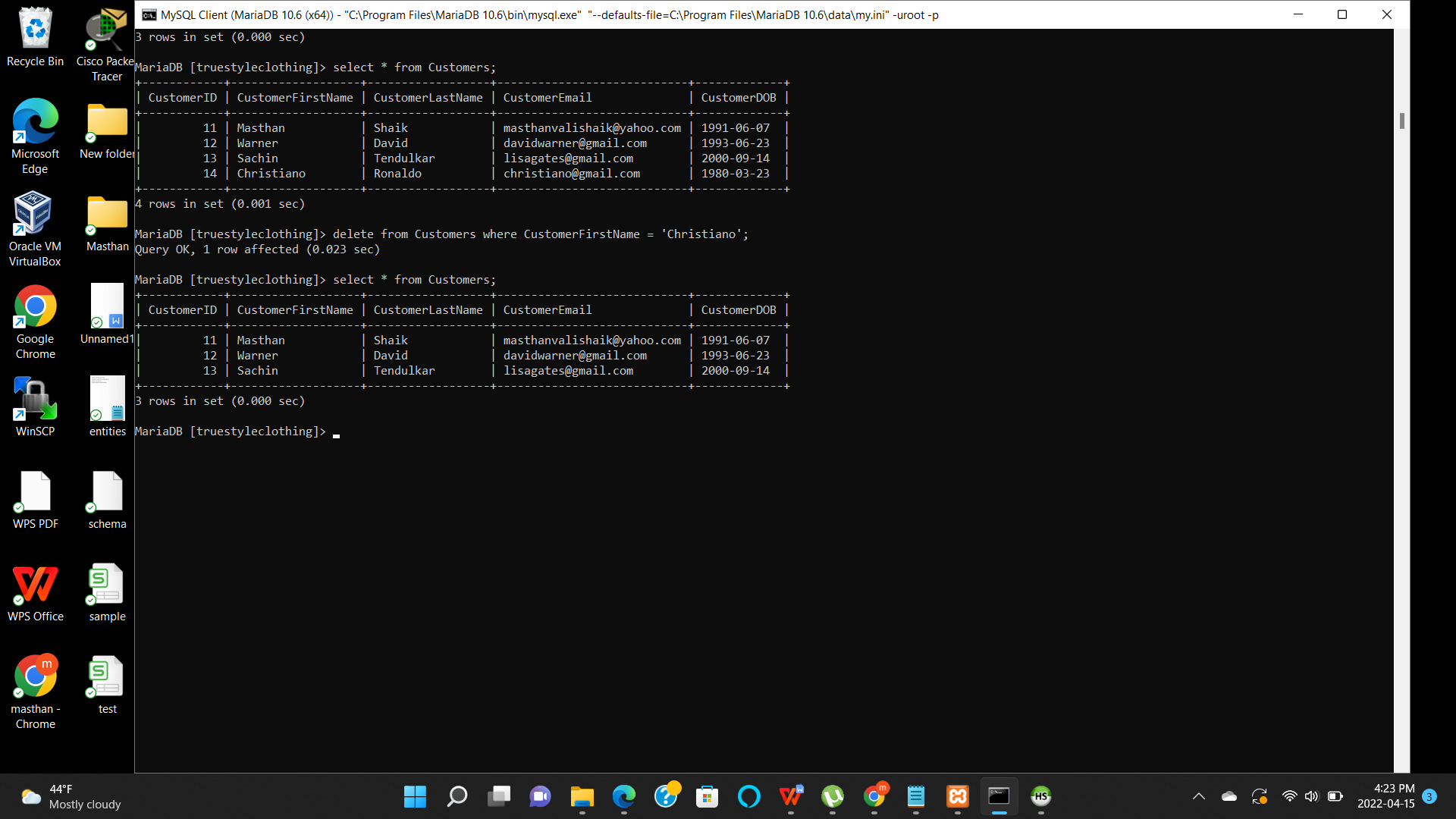


In the second update command, CostumeDesigner is changed to Malhotra and color is change to Orange from Black.



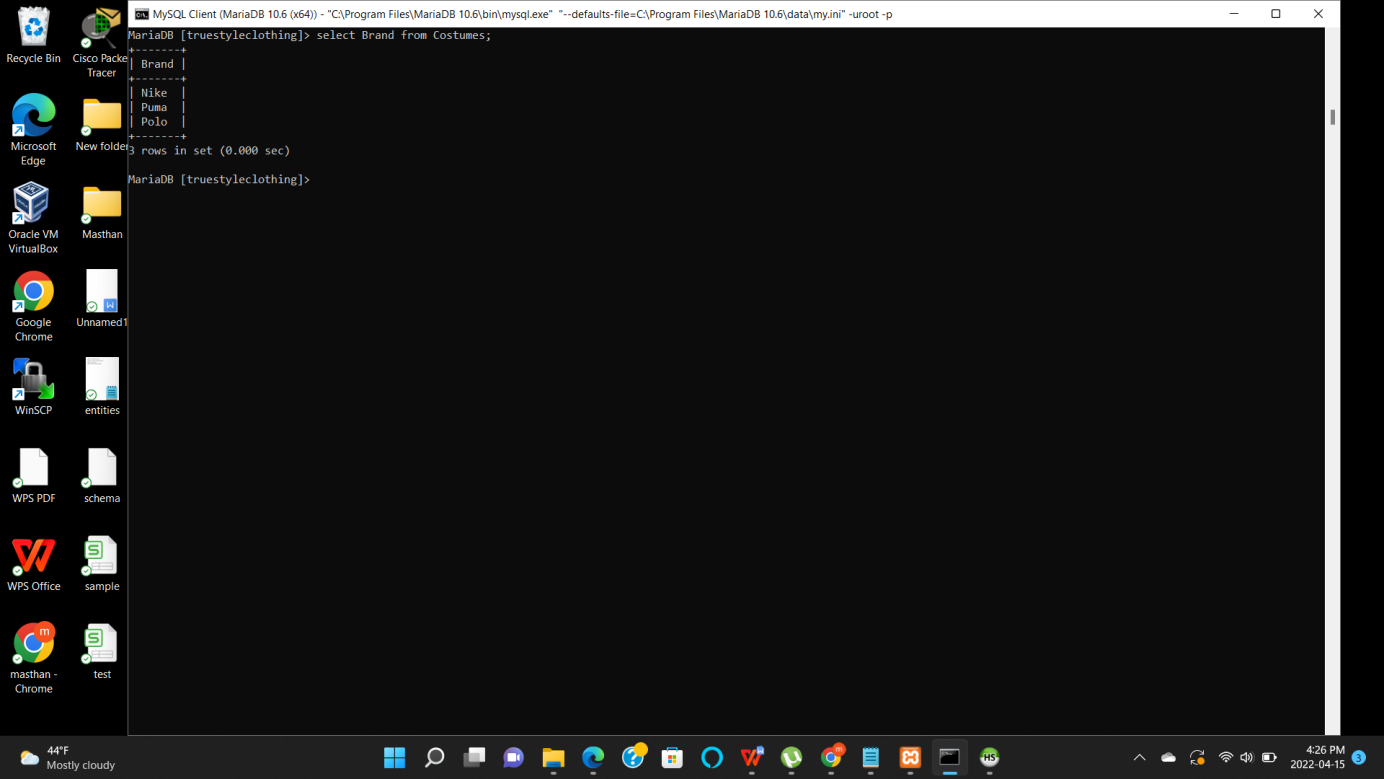
**Delete Command:**

Using delete command, a record has been removed from the Customers table with CustomerID = 14.



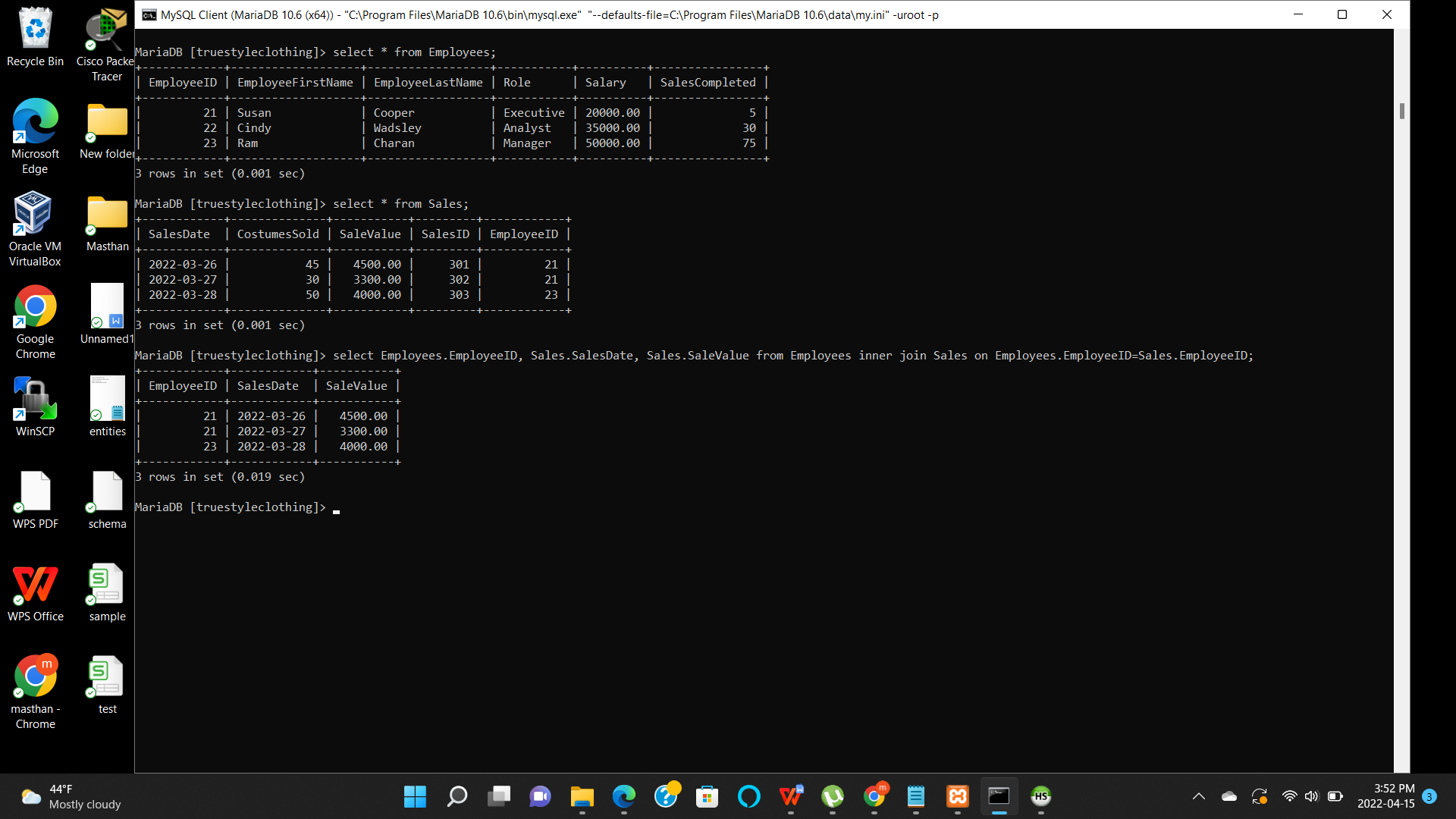
**Select Command:**

Using the simple select command, only the Brand column is selected from Costumes table.

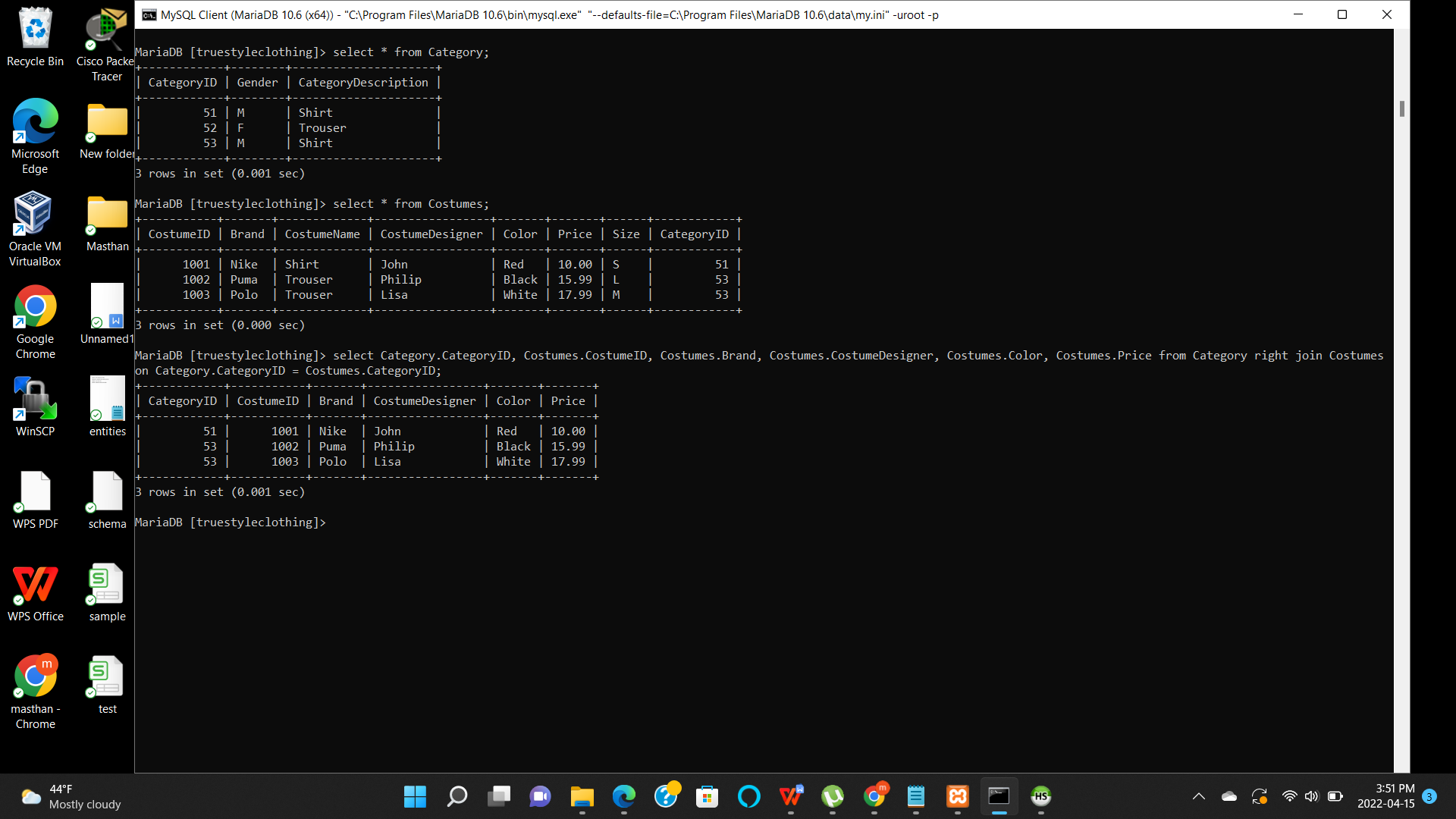


**Join Commands:**

Using the inner join, two tables Employees and Sales are joined which shows the selected columns from both tables.

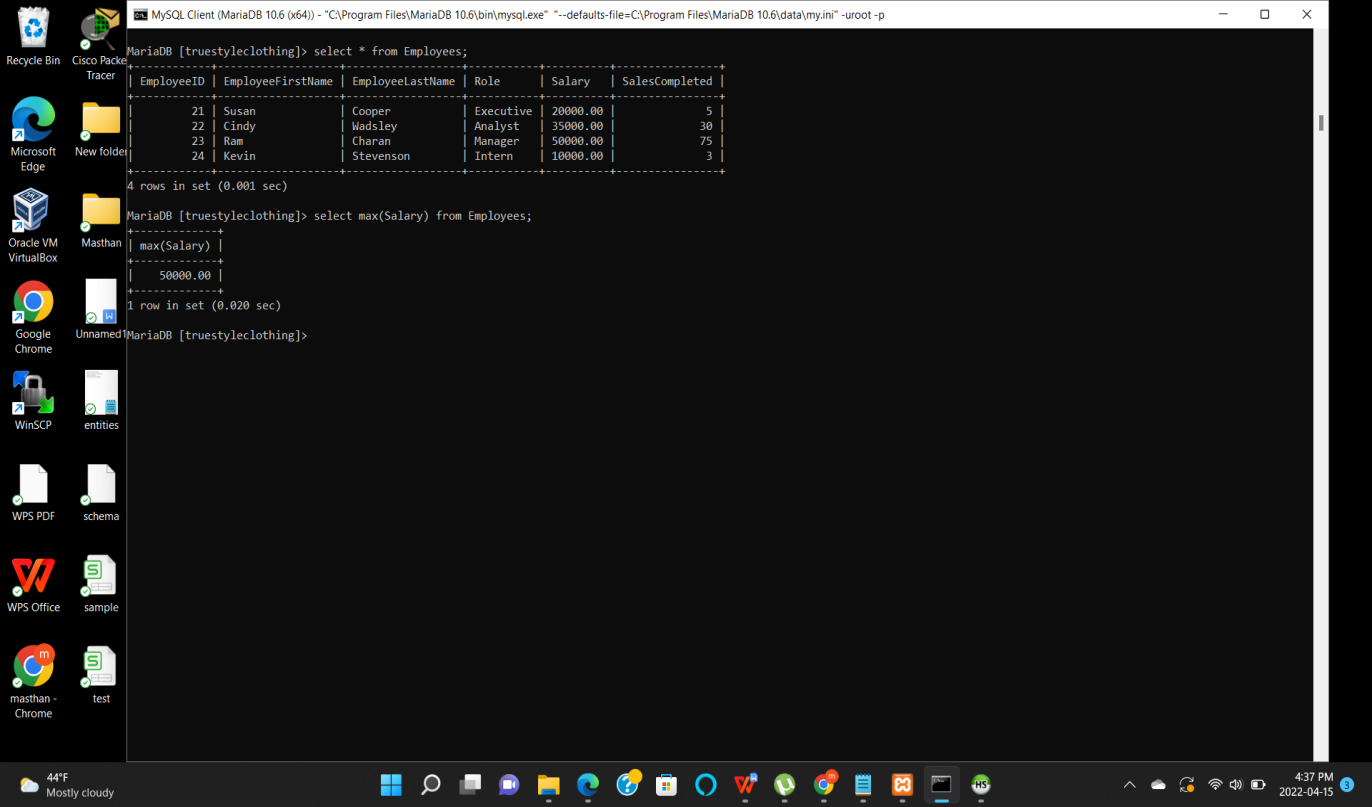


Using the right join, two tables Category and Costumes are joined. Right join shows all the records from the right table (Costumes)and the common records from the left table (Category).

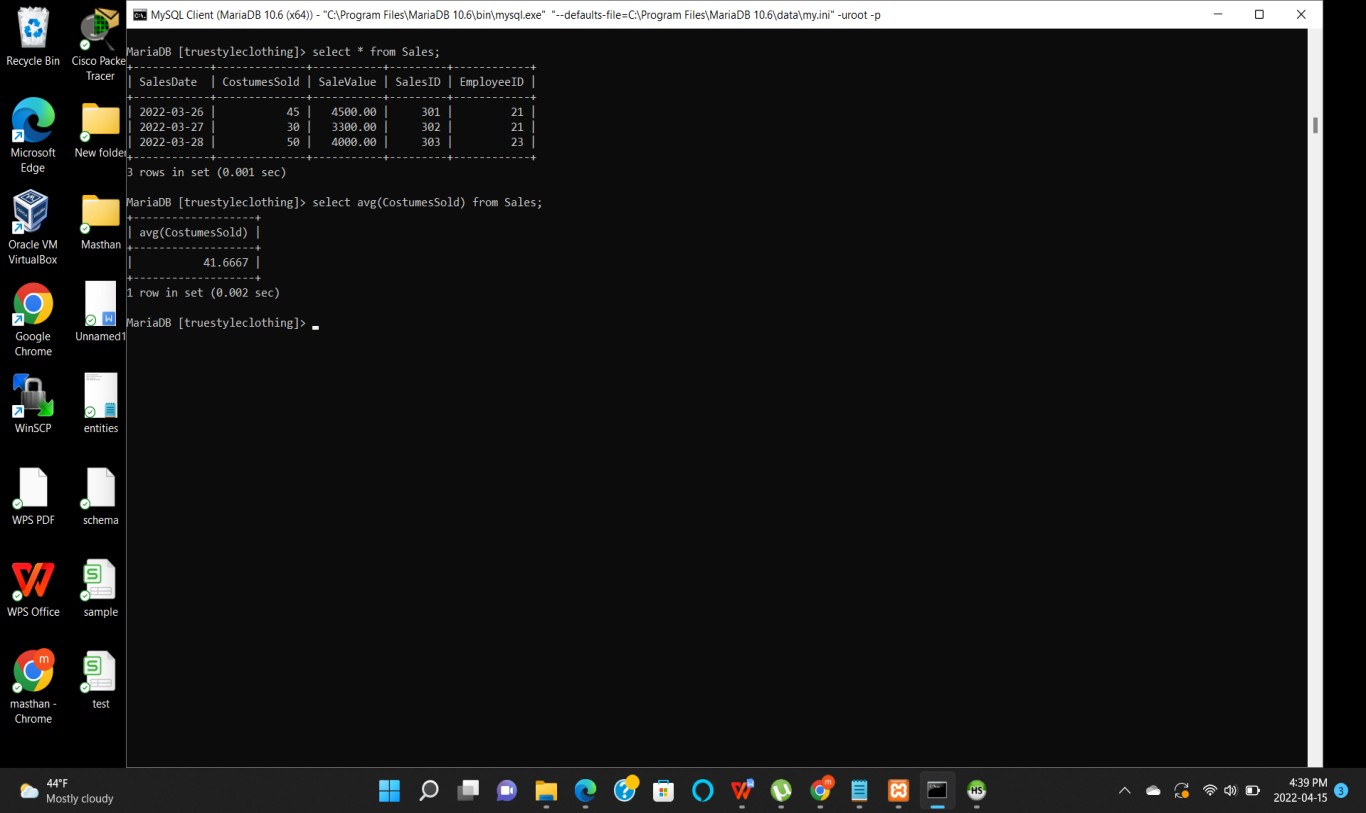


**Summary Commands:**

**Max** - Using the max command, the maximum value in a table can be identified.



**Avg** - Using the avg command, the average of the values can be taken from any table.

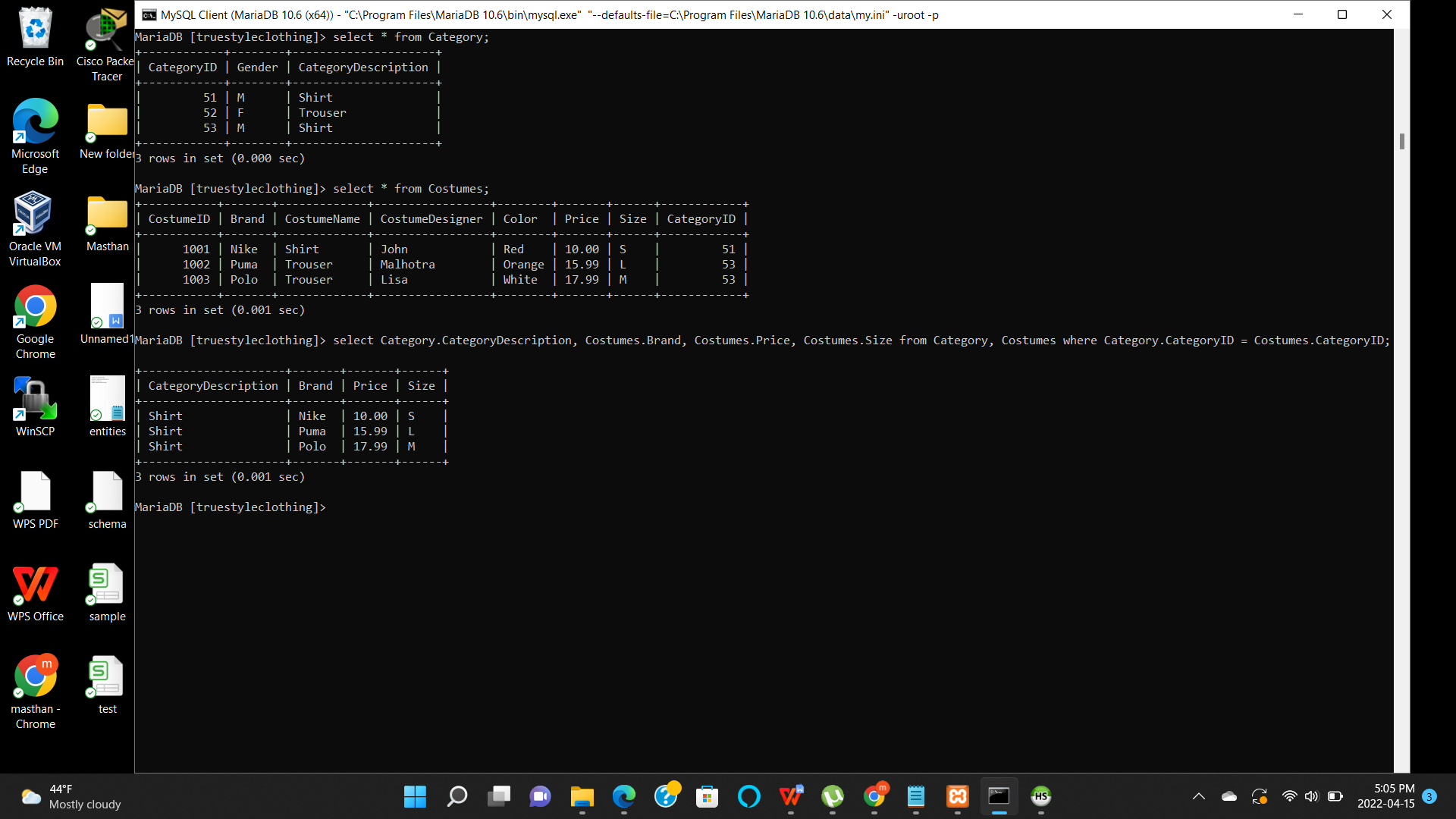


**Sum** - By using sum command, total value of any particular column can be generated.



**Multi table query:**

Category and Costumes tables were used in this example to generate multi table query.



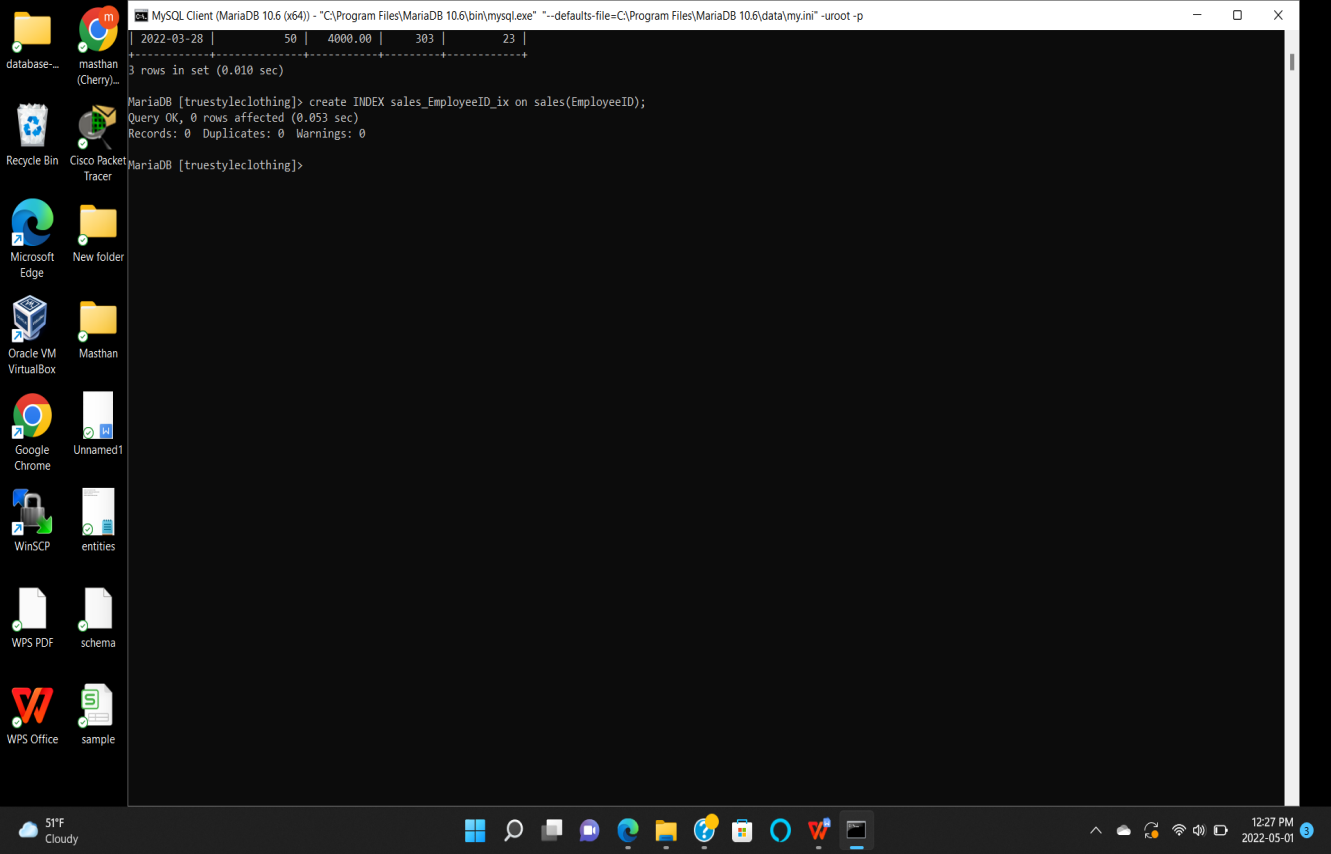
**Week 5 - Project part 5**

In this part of the project, I am working on indexes and views.

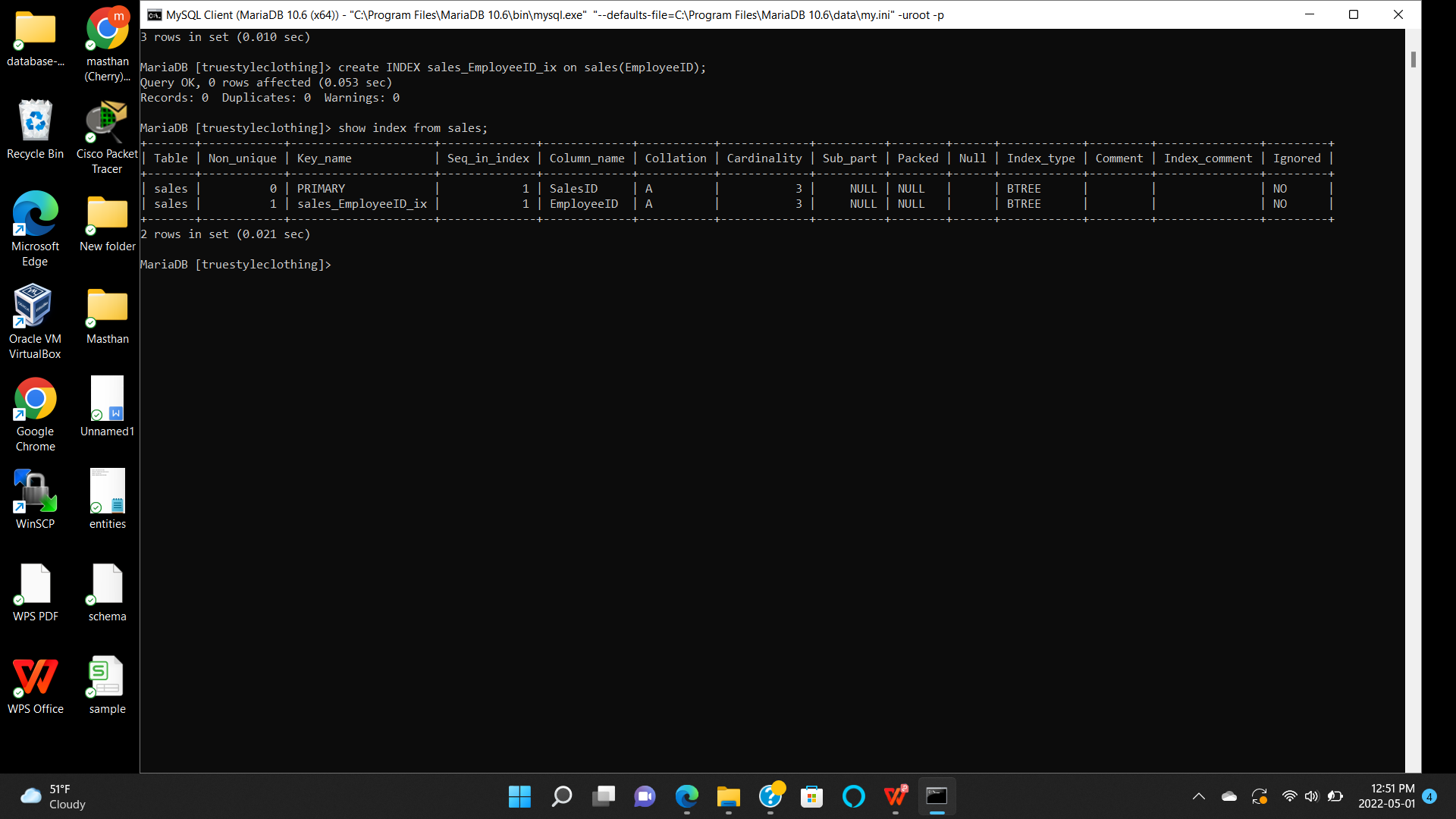
**Indexes**:

Indexes are used for fast searching in the database. I am going to select my Sales and Customers tables for adding the indexes. Sales table has a column **EmployeeID**. Up on adding index to it, it makes easier to search for the Employee who has done more sales till current date. Also, I am adding index to my customers table to the columns **FisrtName** and **LastName**. Similar kind of first names are very common for the customers to have. If an index is added for the FirstName and LastName it helps in fast searching of the customers with similar names.

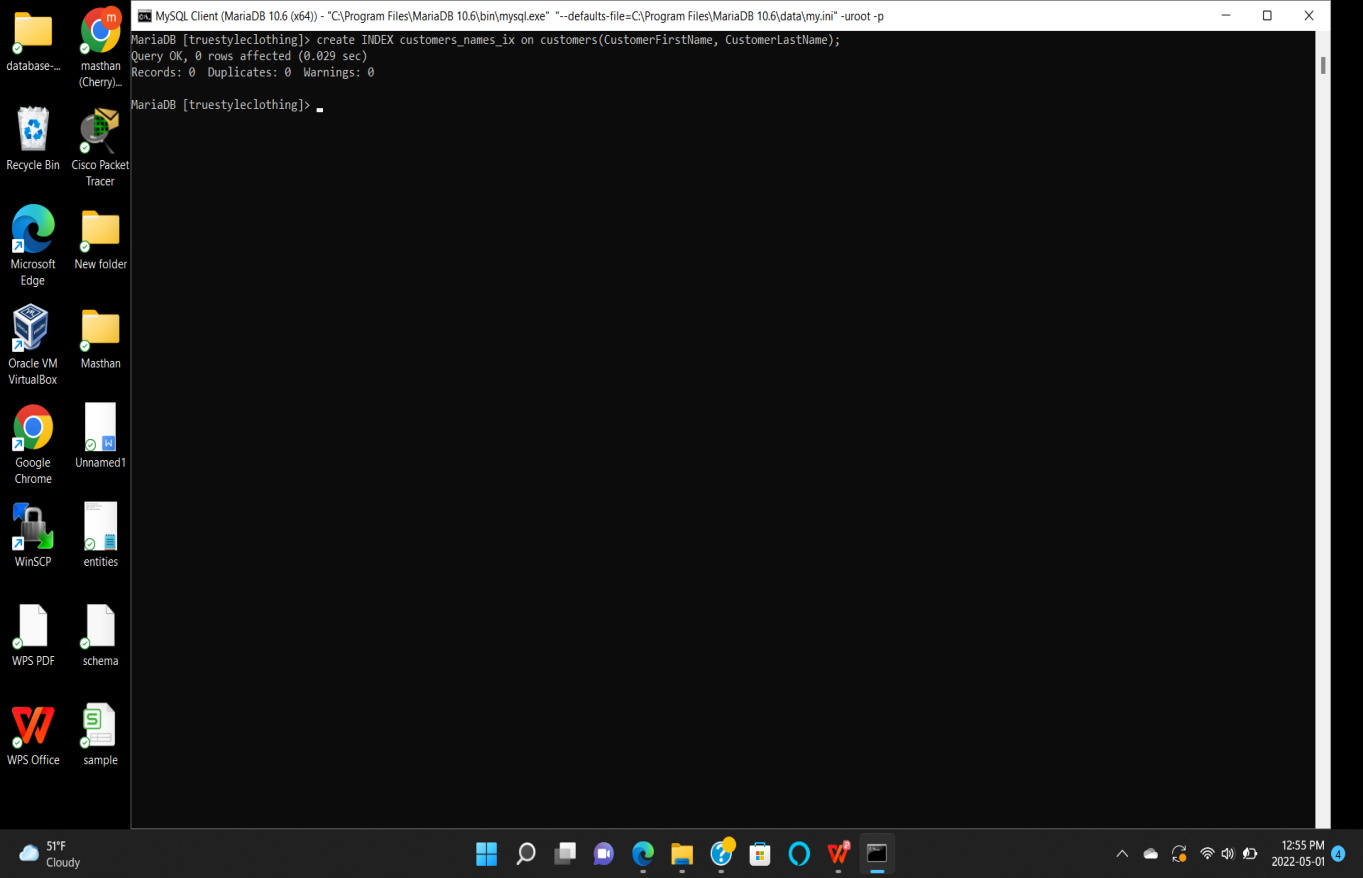
Here is the command I used for creating an index to the Sales table.



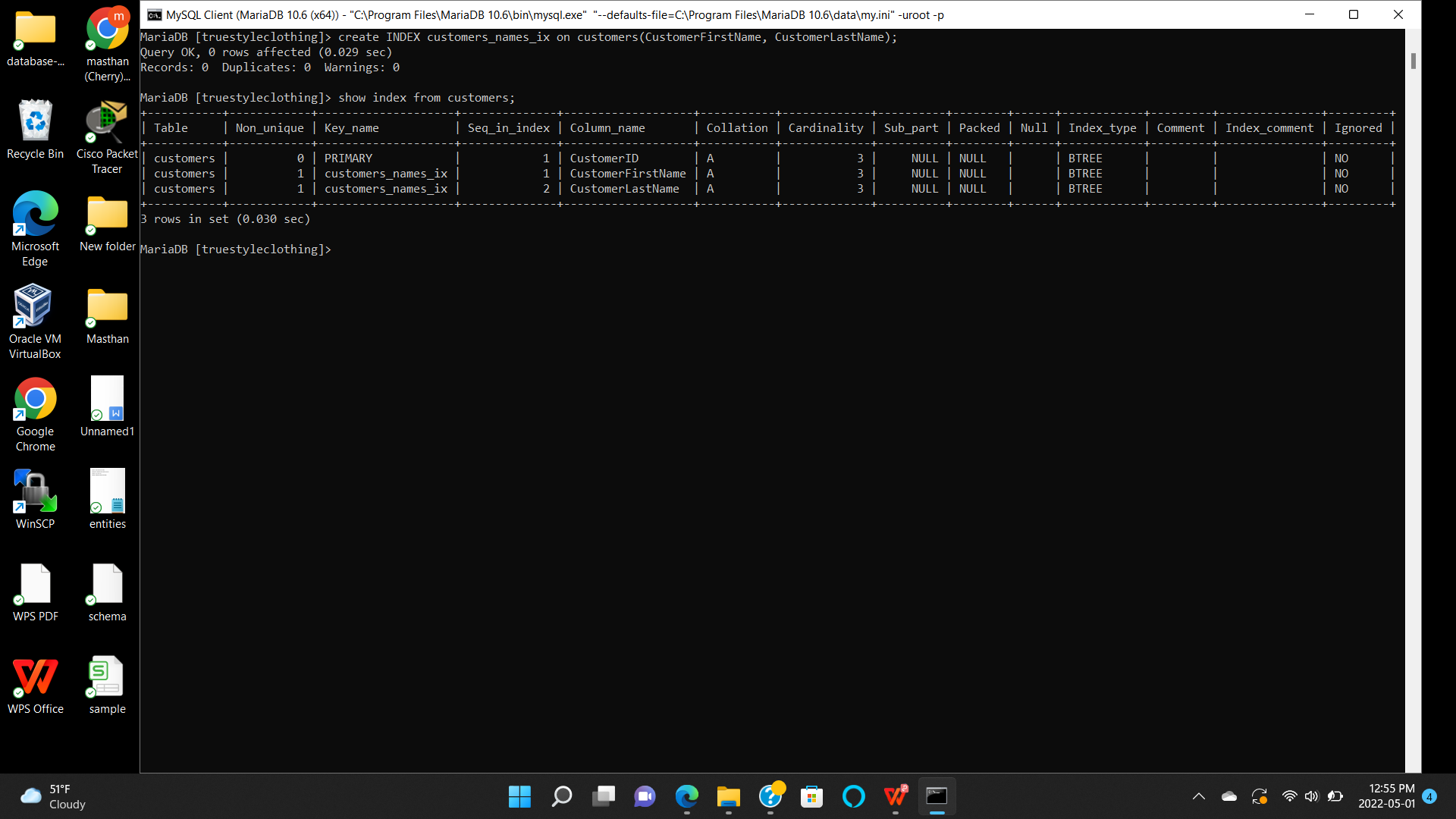
Command used to show the index of Sales table.



Command used to create index to the Customers table.



Command to show the index of the Customers table.



By using Clustered index, the records are arranged in sorted manner and it gets easier to get the value of any specific record by using index. Without the index, the data is searched in the entire database which takes more time to get the desired record when compared to searching with index value.

**View**:

View is like a similar kind of table which contains the rows and columns based on the main table. The data displayed is a subset of the main table.

I am going to create two views on Employees and Sales table. Main advantage of the view is like it also works as a table by giving the desired output that we give when creating a view.

**SQL for creating views for both tables:**

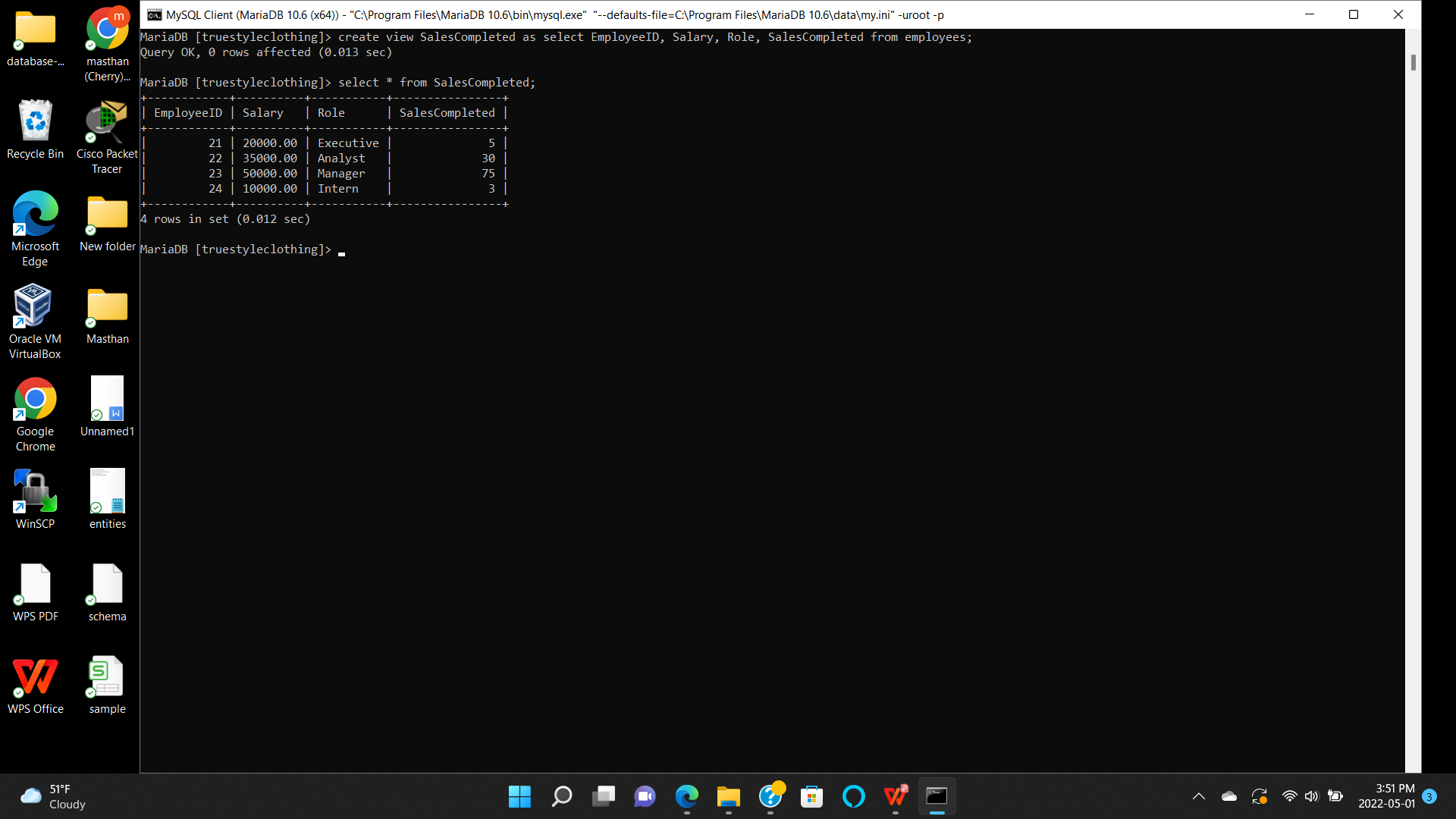
create view SalesCompleted as select EmployeeID, Salary, Role, SalesCompleted from employees;

Select \* from SalesCompleted;

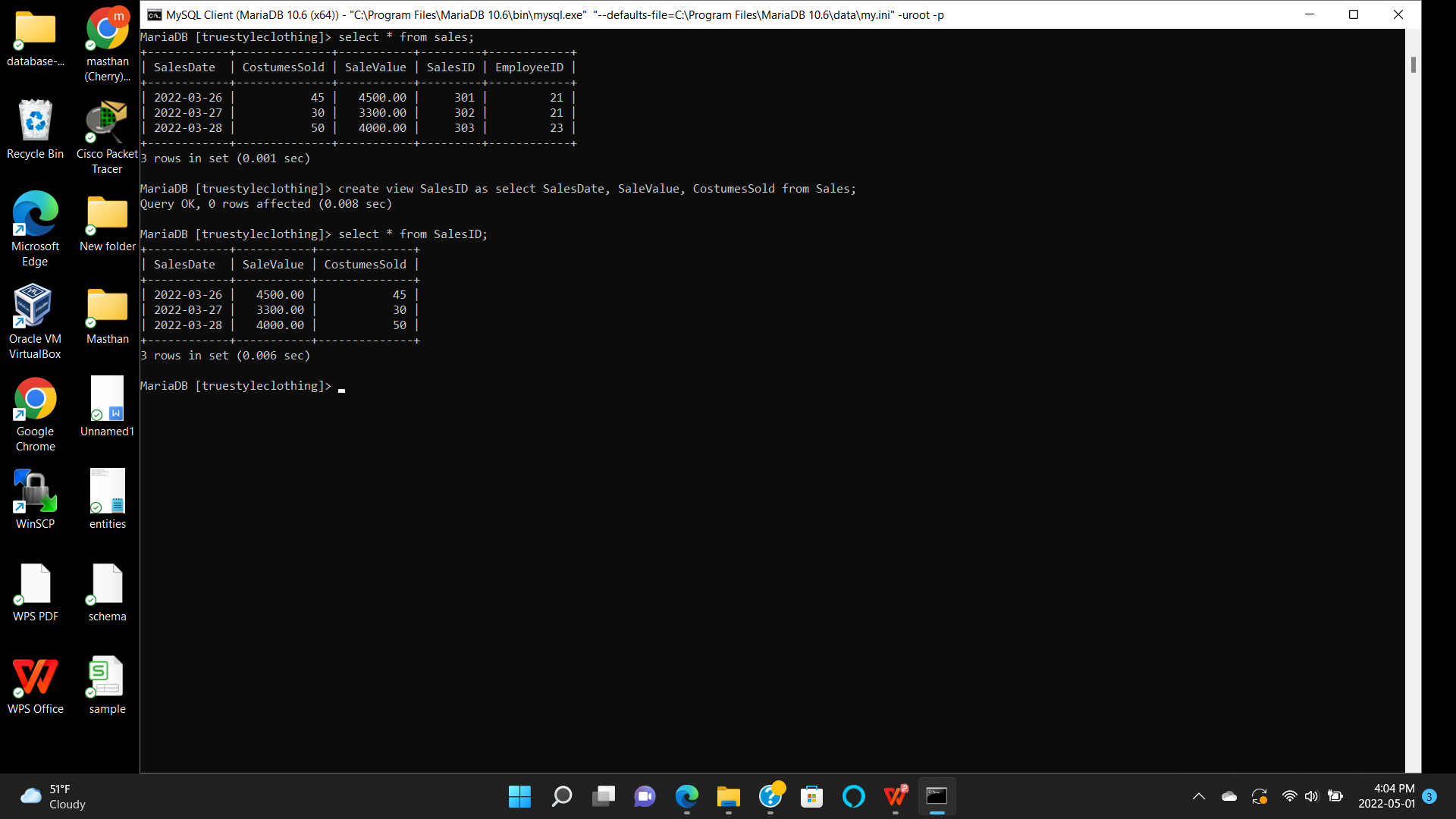
create view SalesID as select SalesDate, SaleValue, CostumesSold from Sales;

Select \* from SalesID;

Command to create and display view for the employees table.



Command to create and display view for the Sales table.



Sales view is very important to my database. It is very easy to get the sales data like number of sales completed on a particular date,total sale value and the number of costumes sold.

Employee view is also crucial as it gets employees data like EmployeeID, Salary, role and sales completed. Employees are directly involved with the sales. All of this information is stored in the database. View helps in limiting the desired data in the form of columns.

These two view helps the administrator, manager and the sales team to get quick access to the data they wanted to see. When ever they want to check the details they can just use these views to look for the data and the recent updated data to the main tables. For huge data sets these views are really helpful to limit the columns.