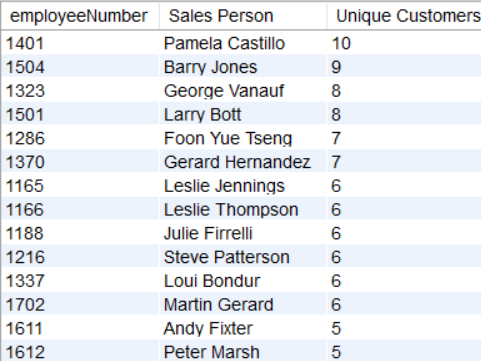
**Day 7**

1. Show employee number, Sales Person (combination of first and last names of employees), unique customers for each employee number and sort the data by highest to lowest unique customers.

Tables: Employees, Customers

**Expected output:**



1. Show total quantities, total quantities in stock, left over quantities for each product and each customer. Sort the data by customer number.

Tables: Customers, Orders, Orderdetails, Products

**Expected output:**

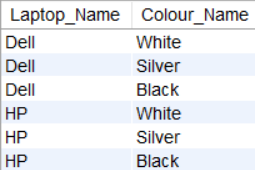


1. Create below tables and fields. (You can add the data as per your wish)

* Laptop: (Laptop\_Name)
* Colours: (Colour\_Name)

Perform cross join between the two tables and find number of rows.

**Expected output:**



1. Create table project with below fields.

* EmployeeID
* FullName
* Gender
* ManagerID

Add below data into it.

INSERT INTO Project VALUES(1, 'Pranaya', 'Male', 3);

INSERT INTO Project VALUES(2, 'Priyanka', 'Female', 1);

INSERT INTO Project VALUES(3, 'Preety', 'Female', NULL);

INSERT INTO Project VALUES(4, 'Anurag', 'Male', 1);

INSERT INTO Project VALUES(5, 'Sambit', 'Male', 1);

INSERT INTO Project VALUES(6, 'Rajesh', 'Male', 3);

INSERT INTO Project VALUES(7, 'Hina', 'Female', 3);

Find out the names of employees and their related managers.

**Expected output:**



**Day 8**

Create table facility. Add the below fields into it.

* Facility\_ID
* Name
* State
* Country

i) Alter the table by adding the primary key and auto increment to Facility\_ID column.

ii) Add a new column city after name with data type as varchar which should not accept any null values.

**Expected output:**



**Day 9**

Create table university with below fields.

* ID
* Name

Add the below data into it as it is.

INSERT INTO University

VALUES (1, " Pune University "),

(2, " Mumbai University "),

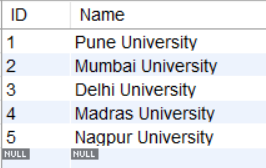
(3, " Delhi University "),

(4, "Madras University"),

(5, "Nagpur University");

Remove the spaces from everywhere and update the column like Pune University etc.

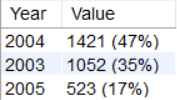
**Expected output:**



**Day 10**

Create the view products status. Show year wise total products sold. Also find the percentage of total value for each year. The output should look as shown in below figure.

**Expected output:**



**Day 11**

1. Create a stored procedure GetCustomerLevel which takes input as customer number and gives the output as either Platinum, Gold or Silver as per below criteria.

Table: Customers

* Platinum: creditLimit > 100000
* Gold: creditLimit is between 25000 to 100000
* Silver: creditLimit < 25000

1. Create a stored procedure Get\_country\_payments which takes in year and country as inputs and gives year wise, country wise total amount as an output. Format the total amount to nearest thousand unit (K)

Tables: Customers, Payments

**Expected output:**



**Day 12**

1. Calculate year wise, month name wise count of orders and year over year (YoY) percentage change. Format the YoY values in no decimals and show in % sign.

Table: Orders

**Expected output:**



1. Create the table emp\_udf with below fields.

* Emp\_ID
* Name
* DOB

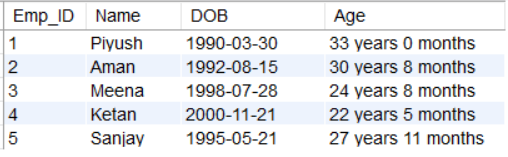
Add the data as shown in below query.

INSERT INTO Emp\_UDF(Name, DOB)

VALUES ("Piyush", "1990-03-30"), ("Aman", "1992-08-15"), ("Meena", "1998-07-28"), ("Ketan", "2000-11-21"), ("Sanjay", "1995-05-21");

Create a user defined function calculate\_age which returns the age in years and months (e.g. 30 years 5 months) by accepting DOB column as a parameter.

**Expected output:**



**Day 13**

1. Display the customer numbers and customer names from customers table who have not placed any orders using subquery

Table: Customers, Orders

**Expected output:**



1. Write a full outer join between customers and orders using union and get the customer number, customer name, count of orders for every customer.

Table: Customers, Orders

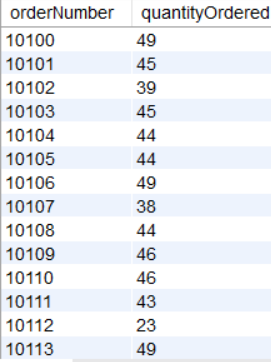
**Expected output:**



1. Show the second highest quantity ordered value for each order number.

Table: Orderdetails

**Expected output:**



1. For each order number count the number of products and then find the min and max of the values among count of orders.

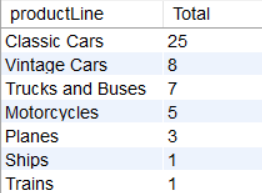
Table: Orderdetails

**Expected output:**



1. Find out how many product lines are there for which the buy price value is greater than the average of buy price value. Show the output as product line and its count.

**Expected output:**



**Day 14**

Create the table Emp\_EH. Below are its fields.

* EmpID (Primary Key)
* EmpName
* EmailAddress

Create a procedure to accept the values for the columns in Emp\_EH. Handle the error using exception handling concept. Show the message as “Error occurred” in case of anything wrong.

**Day 15**

Create the table Emp\_BIT. Add below fields in it.

* Name
* Occupation
* Working\_date
* Working\_hours

Insert the data as shown in below query.

INSERT INTO Emp\_BIT VALUES

('Robin', 'Scientist', '2020-10-04', 12),

('Warner', 'Engineer', '2020-10-04', 10),

('Peter', 'Actor', '2020-10-04', 13),

('Marco', 'Doctor', '2020-10-04', 14),

('Brayden', 'Teacher', '2020-10-04', 12),

('Antonio', 'Business', '2020-10-04', 11);

Create before insert trigger to make sure any new value of Working\_hours, if it is negative, then it should be inserted as positive.