ANALYSIS PROJECT

On Microsoft online store

20th October 2020

Group Members

Aishwarya Chaubal
Aniketh Satyanarayana
Bhupinder Jagwani
Saurabh Somani
Veni Are

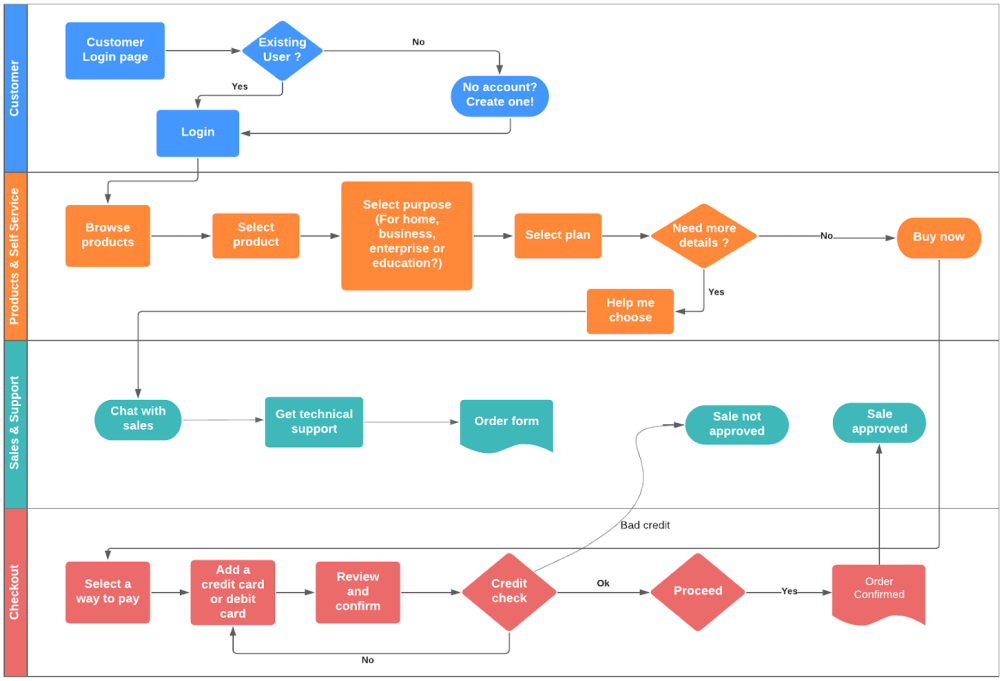
Prof. Ravikumar Nandagopalan

TABLE OF CONTENTS

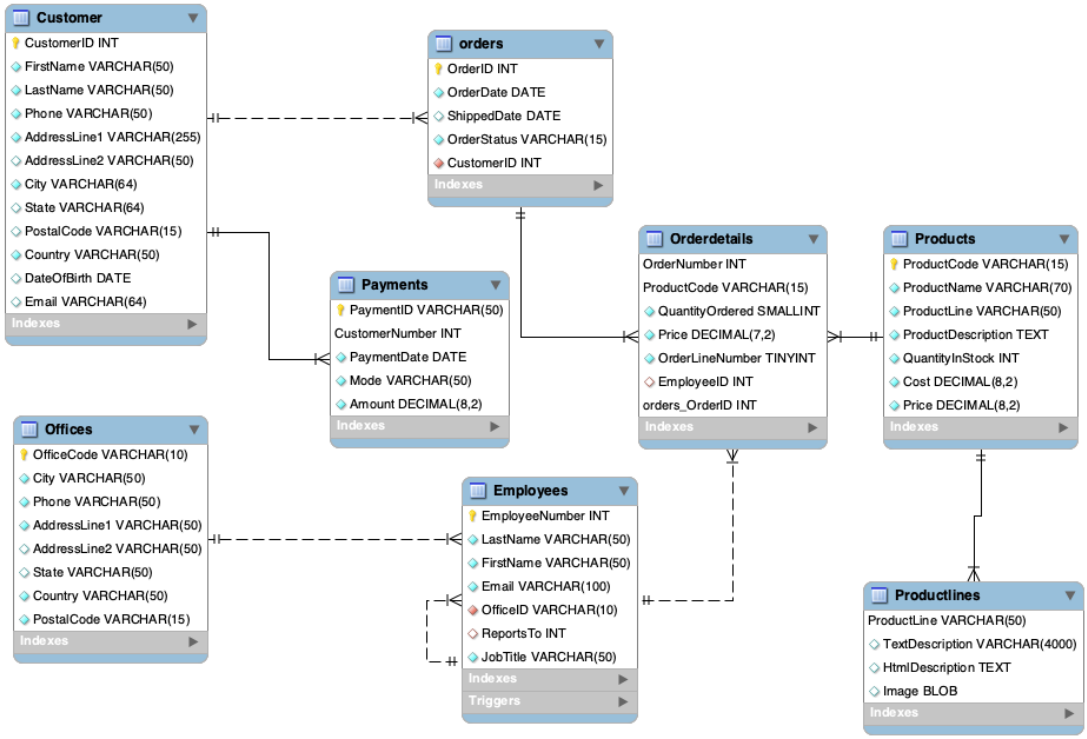
BUSINESS SCENARIO

Joey, who had recently launched a business ,wanted to purchase software for his office. He decided to check the Microsoft website for the same. After successfully logging in using his user account, he started checking the specifications and the details of the products. As he wanted more understanding regarding each product and services, he decided to reach out to the tech support via the ‘chat with sales’ option. The representative guided him with selecting the right plan for his organization of 50 employees. He selected Microsoft 365 Business ‘Standard' as the best plan as it catered for his business requirements by providing tools which enabled remote working and collaboration tools including Microsoft Teams, Secure cloud storage like Exchange, OneDrive, SharePoint; Business email using Outlook and premium Office applications across devices. He finally checked out and purchased the plan. Satisfied with the services provided and the ease of the entire process, he logged off.

SWIMLANE DIAGRAM



ENTITY-RELATIONSHIP(ER) DIAGRAM



DDL COMMANDS (CREATE/ALTER TABLES)

1. **Customer**

```
CREATE TABLE Customer (
    CustomerID INT NOT NULL AUTO_INCREMENT,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    Phone VARCHAR(50) NOT NULL,
    AddressLine1 VARCHAR(255) NOT NULL,
    AddressLine2 VARCHAR(50) DEFAULT NULL,
    City VARCHAR(64) NOT NULL,
    State VARCHAR(64) DEFAULT NULL,
    PostalCode VARCHAR(15) DEFAULT NULL,
    Country VARCHAR(50) NOT NULL,
    DateOfBirth DATE,
    PRIMARY KEY (CustomerID)
)AUTO_INCREMENT=100;
```

```sql
ALTER TABLE Customer ADD Email VARCHAR(64);

CREATE INDEX Idx_CustName
ON Customer (LastName, FirstName);
```

2. **Offices**

```sql
CREATE TABLE Offices (
   OfficeCode VARCHAR(10)  NOT NULL,
   City VARCHAR(50)  NOT NULL,
   Phone VARCHAR(50)  NOT NULL,
   AddressLine1 VARCHAR(50)  NOT NULL,
   AddressLine2 VARCHAR(50)  DEFAULT NULL,
   State VARCHAR(50)  DEFAULT NULL,
   Country VARCHAR(50)  NOT NULL,
   PostalCode VARCHAR(15)  NOT NULL,
   PRIMARY KEY  (OfficeCode),
   INDEX (Phone),
   INDEX (City)
);
```

3. **Employees**

```sql
CREATE TABLE Employees (
 EmployeeNumber INT NOT NULL AUTO_INCREMENT,
 LastName VARCHAR(50) NOT NULL,
 FirstName VARCHAR(50) NOT NULL,
 Email VARCHAR(100) NOT NULL,
 OfficeID VARCHAR(10) NOT NULL,
 ReportsTo INT DEFAULT NULL,
 JobTitle VARCHAR(50)   NOT NULL,
 PRIMARY KEY (EmployeeNumber),
 INDEX (lastName),
 INDEX (firstName),
 FOREIGN KEY (reportsTo) REFERENCES Employees (EmployeeNumber),
 FOREIGN KEY (officeID) REFERENCES Offices (OfficeCode)
);
```

4. **Products**

```
CREATE TABLE Products(
   ProductCode VARCHAR(15) NOT NULL,
   ProductName VARCHAR(70) NOT NULL,
   ProductLine VARCHAR(50) NOT NULL,
   ProductDescription TEXT NOT NULL,
   QuantityInStock INT NOT NULL,
   Cost DECIMAL(8,2) NOT NULL,
   Price  DECIMAL(8,2) NOT NULL,
   PRIMARY KEY (ProductCode),
   INDEX (ProductLine)
);
```

5. **ProductLines**

```
CREATE TABLE Productlines (
   ProductLine VARCHAR(50)   NOT NULL,
   TextDescription VARCHAR(4000) DEFAULT NULL,
   HtmlDescription TEXT DEFAULT NULL,
   Image BLOB DEFAULT NULL,
   PRIMARY KEY (ProductLine),
   FOREIGN KEY (ProductLine) REFERENCES Products(ProductLine)
);
```

6. **Orders**

```
CREATE TABLE Orders (
    OrderID INT NOT NULL AUTO_INCREMENT,
    OrderDate DATE NOT NULL,
    ShippedDate DATE DEFAULT NULL,
    OrderStatus VARCHAR(15) NOT NULL,
    CustomerID INT NOT NULL,
    PRIMARY KEY (OrderID),
    INDEX (CustomerID),
    INDEX (OrderDate),
    FOREIGN KEY (CustomerID) REFERENCES Customer (CustomerID)
);
```

7. **OrderDetails**

```
CREATE TABLE Orderdetails (
   OrderNumber INT NOT NULL,
   ProductCode VARCHAR(15) NOT NULL,
   QuantityOrdered SMALLINT NOT NULL,
   Price DECIMAL(7,2) NOT NULL,
   OrderLineNumber  TINYINT NOT NULL,
   EmployeeID INT DEFAULT NULL,
   PRIMARY KEY  (OrderNumber,ProductCode),
   FOREIGN KEY (OrderNumber) REFERENCES Orders (OrderID),
   FOREIGN KEY (ProductCode) REFERENCES Products (ProductCode),
   FOREIGN KEY (EmployeeID) REFERENCES Employees(EmployeeNumber)
);
```

8. **Payments**

```
CREATE TABLE Payments (
   PaymentID VARCHAR(50) NOT NULL,
   CustomerNumber INT NOT NULL,
   PaymentDate DATE NOT NULL,
   Mode VARCHAR(50) NOT NULL,
   Amount DECIMAL(8,2) NOT NULL,
   PRIMARY KEY  (CustomerNumber,PaymentID),
   FOREIGN KEY (CustomerNumber) REFERENCES Customer (CustomerID)
);
```

DML COMMANDS (INSERT/UPDATE)

INSERT command to enter values into payments table:

```
INSERT INTO PAYMENTS
(CustomerNumber,PaymentID,PaymentDate,Mode,Amount)
VALUES
(100,'ER54537','2004-09-28','Card','31310.09');
```

UPDATE command to set sales representative for a particular order:

```
UPDATE `DBMSProject`.`Orderdetails`
SET
    `EmployeeID` = '1165'
WHERE
    (`OrderNumber` = '10125')
        AND (`ProductCode` = 'S10_2942');
```

QUERYING THE TABLES

Without Joins

Query#1 – List all the products that are less than 2000 in stock

```
SELECT
    *
FROM
    Products
WHERE
    QuantityInStock < 2000;
```

100% 6:2

Result Grid Filter Rows: Search Edit: Export/Import:

ProductCode	ProductName	ProductLine	ProductDescripti...	QuantityInSto...	Cost	Price
S18_3029	Surface Book 3	PCs & Devices	Surface Book 3	1000	1379.99	1399.99
S18_3782	Surface Pro X	PCs & Devices	Surface Pro X	1800	1339.99	1359.99
S700_3167	Surface Pro 7	PCs & Devices	Surface Pro 7	1299	729.99	749.99
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Query#2 – List all the offices located in United States

```
SELECT
    *
FROM
    Offices
WHERE
    Country = 'USA';
```

100% 5:3

Result Grid Filter Rows: Search Edit: Export/Import:

OfficeCode	City	Phone	AddressLine1	AddressLine2	State	Country	PostalCo
1	San Francisco	+1 650 219 4782	100 Market Street	Suite 300	CA	USA	94080
2	Boston	+1 215 837 0825	1550 Court Place	Suite 102	MA	USA	2107
3	NYC	+1 212 555 3000	523 East 53rd Street	apt. 5A	NY	USA	10022
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Query#3 – List out the payments done by cash

```
SELECT
    *
FROM
    Payments
WHERE
    Mode = 'Cash';
```

100% 19:6

Result Grid Filter Rows: Search Edit:

PaymentID	CustomerNumb...	PaymentDate	Mode	Amount
AH612904	101	2003-09-28	Cash	6897.09
HQ336336	103	2004-10-19	Cash	6066.78
EP227123	105	2004-02-10	Cash	5759.42
GB361972	105	2003-12-07	Cash	27988.47
HN114306	105	2003-07-18	Cash	23419.47
IO164641	105	2004-08-30	Cash	37527.58
AD304085	106	2003-10-24	Cash	36798.88
DR578578	106	2004-10-28	Cash	47411.33

Query#4 – List all the employees who work at 'Sydney' Office

```
SELECT
    *
FROM
    Employees
WHERE
    OfficeID = (SELECT
            OfficeCode
        FROM
            Offices
        WHERE
            City = 'Sydney');
```

100% 30:11

Result Grid Filter Rows: Search Edit: Export/Import:

EmployeeNumber	LastName	FirstName	Email	OfficeID	ReportsTo	JobTitle
1088	Patterson	William	wpatterson@microsoft.com	6	1056	Sales Manager
1611	Fixter	Andy	afixter@microsoft.com	6	1102	Sales Rep
1612	Marsh	Peter	pmarsh@microsoft.com	6	1102	Sales Rep
1619	King	Tom	tking@microsoft.com	6	1102	Sales Rep
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Query#5 – List employee details who works as human resources representative

```sql
SELECT
    *
FROM
    Employees
WHERE
    JobTitle = 'VP HR';
```

100% 24:6

Result Grid Filter Rows: Search Edit: Export/Impo

EmployeeNumber	LastName	FirstName	Email	OfficeID	ReportsTo	JobTitle
1086	Anton	Yves	antonyves@microsoft.com	1	1002	VP HR
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Query#6 – Sales done by Employee “Andy Fixter” as Sales representative

```sql
SELECT
    EmployeeID SalesRep,
    OrderNumber AS 'List Of Orders',
    SUM(QuantityOrdered * Price) SalesAmount
FROM
    OrderDetails
WHERE
    EmployeeID = (SELECT
            EmployeeNumber
        FROM
            Employees
        WHERE
            FirstName = 'Andy'
                AND LastName = 'Fixter')
GROUP BY OrderNumber , EmployeeID;
```

100% 35:15

Result Grid Filter Rows: Search Export:

SalesRep	List Of Orders	SalesAmount
1611	10159	3749.24
1611	10160	2811.00
1611	10161	31882.16
1611	10162	4299.70

Query#7 – List out the products which have no sales

```
SELECT
    P.ProductName
FROM
    Products P
WHERE
    P.ProductCode NOT IN (SELECT DISTINCT
            ProductCode
        FROM
            OrderDetails);
```

100% 27:9

Result Grid Filter Rows: Search Export:

ProductName
Surface Laptop 3

Query#8 – List out the minimum and maximum price of products for different product lines

```
SELECT
    ProductLine, MIN(Price), MAX(Price)
FROM
    Products
GROUP BY ProductLine;
```

100% 22:5

Result Grid Filter Rows: Search Export:

ProductLine	Min(Price)	Max(Price)
Business	39.99	3999.99
Developer & IT	58.99	600.00
Entertainment	148.99	2999.00
Other	78.99	3399.00
PCs & Devices	749.99	1399.99
Software	40.99	530.00

Query#9 – Record total number of sales per month

```sql
SELECT
    DATE_FORMAT(OrderDate, '%b') AS Month,
    COUNT(orderid) AS "Total Orders"
FROM
    Orders
GROUP BY DATE_FORMAT(OrderDate, '%b');
```

100% 37:3

Result Grid Filter Rows: Search Export:

Month	Total Orders
Oct	9
May	6
Sep	8
Apr	7
Jan	5
Jun	7
Mar	6
Jul	7
Aug	5
Feb	3

Query#10 – List out the orders of the customers who locate in New York

```sql
SELECT
    OrderID, OrderDate, CustomerID
FROM
    Orders
WHERE
    CustomerID IN (SELECT
            CustomerID
        FROM
            Customer
        WHERE
            City = 'NYC')
ORDER BY OrderDate DESC;
```

100% 24:11

Result Grid Filter Rows: Search

OrderID	OrderDate	CustomerID
10161	2020-10-17	127
10145	2020-08-25	205
10130	2020-06-16	198
NULL	NULL	NULL

With Joins

Query#1 – List out total sales done by each country

```
SELECT
    C.Country, SUM(P.Amount) "Total Sales"
FROM
    Customer C,
    Payments P
WHERE
    C.CustomerID = P.Customernumber
GROUP BY Country;
```

100% 18:8

Result Grid Filter Rows: Search Export:

Country	Total Sales
France	141335.59
USA	2910625.51
Poland	494958.59
Spain	202774.19
Sweden	91010.54
Denmark	109767.13
Singapore	471892.20
Norway	768872.79
Portugal	199771.25
Finland	75946.45
UK	74644.51
Germany	610658.25
Australia	22.77
South Africa	72497.64
Switzerland	148410.09
Belgium	49898.27
Italy	45480.79
New Zealand	48667.67

Query#2 – List all the employees working in different cities

```sql
SELECT
    COUNT(EmployeeNumber) AS 'Employees', O.City
FROM
    Employees E
        INNER JOIN
    Offices O ON O.OfficeCode = E.OfficeID
GROUP BY O.City;
```

100% 17:7

Result Grid Filter Rows: Search Export:

Employees	City
2	Boston
2	London
2	NYC
4	Paris
7	San Francisco
4	Sydney
2	Tokyo

Query#3 – List the products whose order quantity was below 25 in desc order

```sql
SELECT
    P.ProductName, P.ProductLine, O.QuantityOrdered
FROM
    OrderDetails O
        JOIN
    Products P ON O.ProductCode = P.ProductCode
WHERE
    QuantityOrdered < 25
ORDER BY O.QuantityOrdered DESC;
```

100% 1:1

Result Grid Filter Rows: Search Export:

ProductName	ProductLine	QuantityOrder...
Office for Mobile	Business	24
Gift Card Extreme	Other	24
Security for Home	Other	24
Microsoft Teams	Software	24
Security for Home	Other	24
PowerPoint	Software	24
Microsoft Teams	Software	24
Access	Software	24
OneDrive for Business	Software	24
Microsoft Surface	PCs & Devices	24
Microsoft 365 for home	Business	24
Xbox One S	Entertainment	24
OneDrive for Business	Software	23
Security for Business	Other	23

Query#4 – List out all the products sold in quarter 1

```sql
SELECT
    SUM(OD.QuantityOrdered) AS Quantity_Sold, P.ProductName
FROM
    Orders O
        INNER JOIN
    Orderdetails OD ON O.OrderID = OD.OrderNumber
        INNER JOIN
    Products P ON OD.ProductCode = P.ProductCode
WHERE
    O.OrderDate BETWEEN '2020-01-01' AND '2020-03-31'
GROUP BY P.ProductName
ORDER BY Quantity_Sold DESC;
```

100% 23:11

Result Grid Filter Rows: Search Export:

Quantity_Sold	ProductName
171	Visual Studio
142	Microsoft 365 for business
140	WINDOWS 10 HOME
127	Windows Dev Center
114	Office for Students & teachers
112	Bing
110	Microsoft 365 for home
104	Office 365 Professional
102	MSN
91	Hardware
88	Office Dev Center
88	Surface Pro 7
88	Skype for Business
88	Microsoft 365
86	Gift Card for Students

Query#5 – List out top orders (Max purchase price) in each country

```sql
SELECT
    MAX(Purchase_Price), A.Country
FROM
    (SELECT
        C.FirstName,
            C.LastName,
            SUM(od.Price) AS 'Purchase_Price',
            C.Country
    FROM
        Customer C
    JOIN Orders O ON O.CustomerID = C.CustomerID
    JOIN Orderdetails OD ON OD.OrderNumber = O.OrderID
    GROUP BY OD.OrderNumber) AS a
GROUP BY A.Country;
```

100% 20:14

Result Grid Filter Rows: Search Export:

MAX(Purchase_Price)	Country
1601.39	France
432.34	Sweden
976.51	Denmark
1322.67	Singapore
1520.37	Portugal
1623.71	USA
352.00	Finland
460.16	UK
997.50	Canada
1307.47	Germany
1374.90	Spain
1093.98	Norway

VIEWS, TRIGGERS & STORED PROCEDURE

View#1 - This view lists the number of customers per country

```
CREATE OR REPLACE VIEW Customer_Group AS
SELECT COUNT(CustomerID), Country
FROM Customer
GROUP BY Country
ORDER BY COUNT(CustomerID) DESC;

SELECT * FROM Customer_Group;
```

100% 30:7

Result Grid Filter Rows: Search Export:

COUNT(CustomerID)	Country
36	USA
13	Germany
12	France
7	Spain
5	Australia
5	UK
4	Italy
4	New Zealand
3	Norway
3	Singapore
3	Finland
3	Canada
3	Switzerland
2	Sweden

Customer_Group 1

Action Output

	Time	Action	Response
1	18:03:47	SELECT * FROM Customer_Group	27 row(s) returned

View#2 - This view lists all the customer orders group by order number and the sum of each transaction

Using this view user can query top 5 sales as shown:

```
CREATE OR REPLACE VIEW CustomerOrders AS
    SELECT
        OrderNumber,
        CustomerID,
        SUM(quantityOrdered * price) TotalAmount
    FROM
        OrderDetails OD
            INNER JOIN
        Orders O ON OD.OrderNumber = O.OrderID
            INNER JOIN
        Customer USING (customerid)
    GROUP BY OrderNumber;

-- Top 5 sales
SELECT * FROM CustomerOrders
ORDER BY TotalAmount DESC limit 5;
```

100% 35:16

Result Grid Filter Rows: Search Export:

OrderNumber	CustomerID	TotalAmount
10127	151	58841.35
10126	135	57131.92
10142	124	56052.56
10135	124	55601.84
10105	145	53959.21

CustomerOrders 2

Action Output

	Time	Action	Response
1	18:10:56	CREATE OR REPLACE VIEW CustomerOrders AS SELECT OrderNumber, CustomerID, S...	0 row(s) affected
2	18:10:56	SELECT * FROM CustomerOrders ORDER BY TotalAmount DESC limit 5	5 row(s) returned

Trigger – Below trigger inserts employee details into Organization Alumni table before a record is deleted from employee table

```
CREATE TABLE MicrosoftAlumniNetwork (
    EmployeeNumber INT NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    FirstName VARCHAR(50) NOT NULL,
    Email VARCHAR(100) NOT NULL,
    OfficeID VARCHAR(10) NOT NULL,
    PRIMARY KEY (EmployeeNumber),
    FOREIGN KEY (OfficeID)
        REFERENCES Offices (OfficeCode)
);
```

```
DELIMITER $$
CREATE TRIGGER before_employees_delete
BEFORE DELETE
ON Employees FOR EACH ROW
BEGIN
    INSERT INTO MicrosoftAlumniNetwork(EmployeeNumber,LastName,FirstName,Email,OfficeID)
    VALUES(OLD.EmployeeNumber,OLD.LastName,OLD.FirstName,OLD.Email,OLD.OfficeID);
END$$
DELIMITER ;

DELETE FROM employees
WHERE
    employeenumber = '1337';

Select  * from MicrosoftAlumniNetwork;
```

100% 38:54

Result Grid Filter Rows: Search Edit: Export/Import:

EmployeeNumber	LastName	FirstName	Email	OfficeID
1337	Bondur	Loui	lbondur@microsoft.com	4
NULL	NULL	NULL	NULL	NULL

Stored Procedure – To get order count based on order status
Total orders with status “Cancelled”:

```
DELIMITER $$
CREATE PROCEDURE GetOrderCountByStatus(
    IN  Status VARCHAR(15),
    OUT Total INT
)
BEGIN
    SELECT COUNT(OrderID)
    INTO Total
    FROM Orders
    WHERE OrderStatus = Status;
END$$
DELIMITER ;

-- Procedure Call
CALL GetOrderCountByStatus('Cancelled',@Total);
SELECT @Total AS "Total Orders Cancelled";
```

100% 43:16

Result Grid Filter Rows: Search Export:

Total Orders Cancelled
3

Result 1 Result 2

Action Output

	Time	Action	Response
1	18:38:12	CALL GetOrderCountByStatus('Cancelled',@Total)	1 row(s) affected
2	18:38:12	SELECT @Total AS "Total Orders Cancelled"	1 row(s) returned

Total orders with status “pending payment”:

```
-- Procedure Call

CALL GetOrderCountByStatus('Pending Payment',@Total);
SELECT @Total AS "Total Orders with Pending Payment";
```

100% 54:4

Result Grid Filter Rows: Search Export:

Total Orders with Pending Payment
1

TABLEAU REPORT

CONCLUSION

1. One of the key challenges while collaborating with teammates on this project was the online perspective. Given the unique COVID-19 situation, the team adapted to a harmonious online discussion. Scheduled zoom calls, Google docs as a single source of communication & MySQL Workbench app were the 3 key resources for the success of this project execution.
2. Generating the data would be the highlight of this project. As we know, data is the heart of each and every project. With significant efforts in finalizing the business requirements, making the ER diagrams, generating/manipulating/cleaning the data paved the way for successful creation & execution of queries.
3. Since we modelled our project on “Microsoft” which is a very well-known brand across the globe, visualizing the queries became a bit easier.
4. The queries implemented in the project along with the swim lane and ER diagrams gives out a small POC (proof of concept) of the ecommerce aspect of Microsoft’s business model.
5. We would also like to extend our gratitude towards Prof. Nandagopalan for planning the lectures & assignments in a very specific manner. The project building phase was in sync with the contents covered in the class, helping massively to implement the concepts in the project.

FUTURE SCOPE

1. Would like to use a version control system like “Git” to keep track of project code and use of platforms like “Confluence” to maintain the documentation.
2. Would like to implement this project in a cloud based RDS (relational database service) to explore the cloud environment.