USE CASE STUDY REPORT

Group No: Group 14

Student Name: Batta Aditya and Soumitra Rajeev Bhagdikar

Executive Summary:

The principal goal of this case study is to discern the operational dynamics of manufacturing companies within the business domain. Employing a relational database model has been pivotal in our pursuit of uncovering valuable insights and understanding customer behaviours. This strategic choice in database architecture enables a systematic exploration of relationships within the data, facilitating a comprehensive analysis of the intricate patterns and trends that define the manufacturing sector. Through this approach, we aim to derive actionable intelligence that contributes to informed decision-making and strategic planning.

To enhance comprehension, we have implemented both an Entity-Relationship (EER) model and a Unified Modelling Language (UML) model. In conjunction with this, we have curated data from diverse sources and seamlessly integrated it into our MySQL database. This repository encompasses a substantial dataset, encompassing customer profiles, supplier information, and associated products. The architecture of our database is structured around manufacturing sectors and demand considerations, allowing for a systematic and efficient organization of information.

Moreover, we have executed purposeful queries aligned with our business model, providing actionable insights and fostering strategic decision-making. This meticulous approach not only ensures the integrity and coherence of our data but also positions our organization to derive maximum value from the information stored in our database.

The integration of our database with Python has been instrumental in the creation of descriptive visualizations, enabling a more nuanced exploration of the underlying data. This strategic coupling facilitates a comprehensive analysis aimed at extracting valuable insights. Furthermore, our choice to implement the database using MongoDB reflects a commitment to achieving a broader and deeper understanding of customer and market behaviour.

Introduction

Advancement in digitalization has paved a way for many opportunities for business and customers. In the age and era of social media, the world has become a diminished entity as everything is interconnected and inseparable. We aim to create an e-commerce application for a manufacturing company “Blue Manufacturers”. There are usually 5 product groups in a manufacturing company including Instrumentation, Fluid Connectors, Motion Systems, Aeronautics and Engineering Materials.
We will include all these product groups in our database and retrieve customer demand for these groups. Manufacturing process after receiving the order would be done via Trading or Warehouse where goods would be either traded from a third-party vendor or would be manufactured in the factory plant. We will then segregate the received order according to its manufacturing type and send the order to the concerned management. Customers are further divided into two groups. The reputed old customers and new customers. Both of these group’s pricing and negotiation strategy would be different. Delivery type of each order would be different but will be primarily done via different means each of whose price is

added in the final bill. Feedback tracking mechanism to ensure customer satisfaction and rating employee performance is necessary and would be included in the application.
Blue Manufactures has 5 product types exclusively for the needs of customers and demand of the market. Our target clients are mainly from Petroleum, Automobile and Airways industries. We basically sell our products at discounted amounts to old and regular customers. For new customers, a special offer is available where they can purchase at relatively lower prices. Also, we have a special offer for those customers who recommend and refer new customers to purchase our products.
When orders are made by customers, we note the order ID. Apart from this, supplier name, address, referral ID are recorded. Our products have particular Product id, product type and price. Our company will give discounts for bulk and heavy products.
We record our daily based customers and give more discounts as proportionate to their orders.

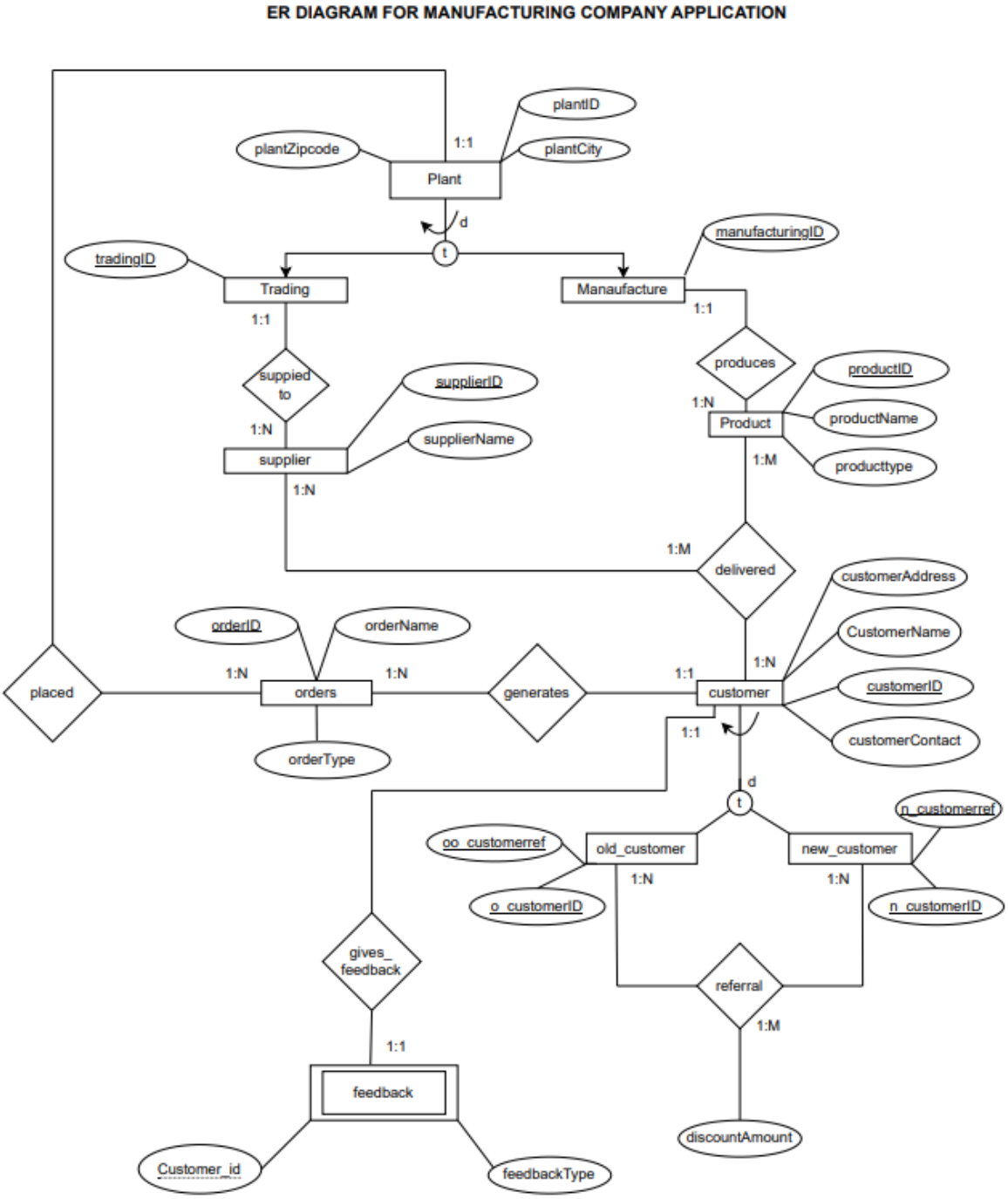
For example, we provide free transportation to all our customers who order bulk quantities. Using relational database model, it is efficient to predict the product type that are been ordered in bulk by the customers. It would be also beneficial to understand the location of plant and customers which would ease the transportation and supply chain of our product

Functioning of the Application

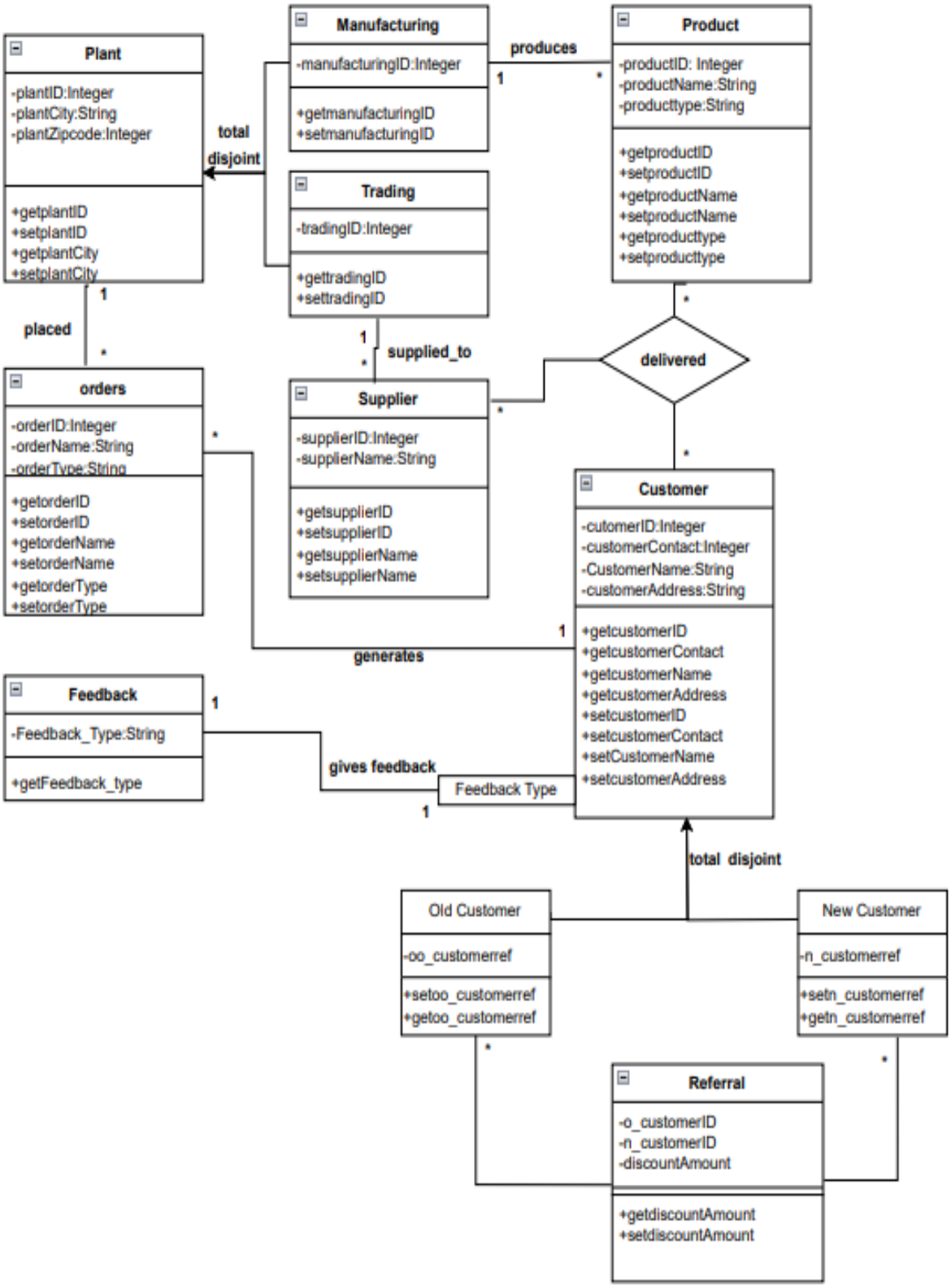
- A product can be provided by at least one supplier, a supplier can provide multiple but at least one product.
- A customer can give many orders and order can have only a single customer
- A product can be delivered to multiple customers and a customer can order multiple products
- An order can be placed to one plant and a plant can have several orders
- A trading unit can have many suppliers and a supplier must belong to one trading unit
- A customer can give his referral to many different customers and one customer can only use one referral
- A customer can give single feedback and feedback is given by a single customer
- A product can be manufactured in a single plant and a plant can manufacture multiple products

II Conceptual Modelling Diagram

a) ER Diagram



b) UML Diagram



III Mapping Conceptual Model to Relational Model

Primary Key – underlined **Foreign key – italicized**

Relational Model

- **Customer**(customerID, customerName, customerAddress, customerContact)
- **Plant** (plantID, plantCity, plantZipcode)
- **Trading**(tradingID,*t_plantID*)
  Foreign Key: t_plantID: refers to plantID in Plant, Not Null
- **Manufacturing**(manufacturingID,*m_plantID*)
  Foreign Key: m_plantID: refers to plantID in Plant, Not Null
- **Supplier**(supplierID,supplierName,*s_tradingID*)
  Foreign Key: s_tradingID: refers to tradingID in Trading Not Null
- **Product**(productID, productName,productType,*p_manufacturingID*)
  Foreign Key: p_manufacturingID: refers to manufacturingID in Manufacturing
- **Delivery**`(*d_productID, d_supplierID, d_customer*)
  Foreign Key: d_productID: refers to productID in Product, Not Null. d_supplierID: refers to supplierID in Supplier, Not Null. d_customerID: refers to customerID in Customer, Not Null
- **orders**(orderID, orderType, *o_customerID, o_plantID*, OrderName)
  Foreign Key: o_customerID: refers to customerID in Customer, Not Null. o_plantID: refers to plantID in Plant, Not Null
- **OldCustomer**(*o_customerID*,oo_customerref)
  Foreign Key: o_customerID: refers to customerID in Customer, Not Null
- **NewCustomer**(*n_customerID*, *n_customerref*)
  Foreign Key:n_customerID refers to customerID in Customer, Not Null. n_customerref refers to o_customerID in Old Customer
- **Feedback**(*f_customerID*, feedbackType)
  Foreign Key: f_customerID refers to customerID in Customer, Not Null
- **Referral**(*or_customerID, nr_customerID*, discountAmount)
  Foreign Key: or_customerID: refers to customerID in Customer, Not Null. nr_customerID: refers to customerID in Customer, Not Null

IV. IMPLEMENTATION OF RELATIONAL MODEL MySQL

The database is created in MySQL Workbench.

Query 1: (SIMPLE QUERY) Retrieve names of all the customers from NewYork City

```
SELECT customerName FROM customer
WHERE customerAddress LIKE %NY%;
```

customerName
Curtis Mullins
Alan Miller

Query 2: (SIMPLE AGGREGATE) Retrieve average, variance, maximum and minimum discountAmount from the referrals where new customerID is 8

```
SELECT AVG(discountAmount) AS avg_discount, variance(discountAmount) as
var_discount, min(discountAmount) as min_discount, max(discountAmount) as
max_discount FROM referral
WHERE nr_customerID = 8;
```

avg_discount	var_discount	min_discount	max_discount
92.0000	64	84	100

Query 3: (INNER JOIN)Retreive all the names of customers who get deliveries from same supplier

```
SELECT DISTINCT c.customerName
FROM customer c
INNER JOIN orders o ON c.customerID = o.o_customerID
INNER JOIN delivery d ON o.orderID = d.d_customerID
INNER JOIN supplier s ON d.d_supplierID = s.supplierID
INNER JOIN delivery d2 ON s.supplierID = d2.d_supplierID
INNER JOIN orders o2 ON d2.d_customerID = o2.o_customerID
WHERE o.o_customerID &lt; &gt; o2.o_customerID ;
```

customerName
Amanda Cruz
Sara Lee
Rebecca Gillespie
Joanna Lee
Michelle Campbell
Patrick Pierce
Jacob Edwards

Query 4:(LEFT JOIN) Retrieve the names of customers who have placed orders and their corresponding product names, along with any customers who have not placed orders yet, along with their referral discount information

```
SELECT c.customerName, p.productName, COALESCE(r.discountAmount, 0) AS
referralDiscount
FROM customer c
```

```
LEFT JOIN orders o ON c.customerID = o.o_customerID
LEFT JOIN product p ON o.o_plantID = p.productID
LEFT JOIN referral r ON c.customerID = r.nr_customerID;
```

customerName	productName	referralDiscount
Alicia Davila	NULL	47
Alicia Davila	NULL	43
Alicia Davila	NULL	9
Alicia Davila	NULL	98
Francisco Smith	HydraTrack System	21
Francisco Smith	HydraTrack System	89
Stephanie Turner	HydraPulse Joint	86

Query 5: (WINDOW FUNS)Retrieve count of all products and producttypes which are ordered by customers

```
SELECT productName, productType, orderCount,
RANK() OVER (ORDER BY orderCount DESC) AS ProductRank
FROM (
SELECT p.productName, p.productType, COUNT(*) AS orderCount
FROM orders o
JOIN product p ON o.o_plantID = p.productID
GROUP BY p.productName, p.productType) AS ProductOrders;
```

productName	productType	orderCount	ProductRank
NanoFlow Meter	Instrumentation	18	1
AeroRotor Blade	Aeronautics	16	2
FluidicCeramic Material	Engineering Materials	15	3
HydraPulse Joint	Fluid Connectors	15	3
NanoFlow Transmitter	Instrumentation	14	5
FlexiRotor Blade	Engineering Materials	14	5
FluidicLink Adapter	Fluid Connectors	13	7

Query 6: (NESTED QUERY)Retrieve customerID, customerName who can order producttype as Aeronautics

```
SELECT customerID, customerName
FROM customer
WHERE customerID IN (
SELECT DISTINCT o.o_customerID
FROM orders o
JOIN product p ON o.o_plantID = p.p_manufacturingID
WHERE p.productType =Aeronautics);
```

customerID	customerName
2	Francisco Smith
3	Stephanie Turner
5	Jason Wilson
7	Douglas Browning
8	Kimberly Weaver
9	Ebony Knight
10	Benjamin Jones

Query 7: (CORRELATED) Retrieve the customer number and name of customers who placed orders for a product with a product ID=10 and received the highest discount amount.

```
SELECT c.customerID AS customernumber, c.customerName
FROM customer c
JOIN orders o ON c.customerID = o.o_customerID
JOIN delivery d ON o.orderID = d.d_customerID
JOIN product p ON d.d_productID = p.productID
JOIN referral r ON o.orderID = r.nr_customerID
WHERE p.productID = 10
AND r.discountAmount = (
SELECT MAX(discountAmount)
FROM referral r_inner
WHERE r_inner.nr_customerID = o.orderID);
```

customernumber	customerName
28	Katrina Sanders
20	Alexandria Williams
86	Jonathan Callahan

Query 8: (ANY/ALL) Retrieve the name of Customer who ordered the most than others using ANY/ALL function

```
SELECT c.customerName
FROM customer c
WHERE c.customerID = ANY (
SELECT o.o_customerID
FROM orders o
GROUP BY o.o_customerID
HAVING COUNT(*) &gt;= ALL (
SELECT COUNT(*)
FROM orders
GROUP BY o_customerID
));
```

customerName
Patricia Curtis

Query 9: (EXISTS) Write a SQL query using 'EXISTS' that returns the supplierName, supplierID who delivered the producttype 'Instrumentation'

```
SELECT s.supplierName, s.supplierID
```

```
FROM supplier s
WHERE EXISTS (
SELECT 1
FROM delivery d
JOIN product p ON d.d_productID = p.productID
WHERE d.d_supplierID = s.supplierID
AND p.producttype = Instrumentation);
```

	supplierName	supplierID
▶	Winters-Fritz	2
	Hurst-Casey	9
	Stark, Alvarado and Miles	11
	Marshall and Sons	12
	Hickman LLC	13
	Barker-Potter	14
	Reyes LLC	15

Query 10: (SUBQUERY)Retrieve all the customer names, customerID where the differenece between maximum and minimum orders is strictly above 1 (use subquery)

```
SELECT customerID, customerName
FROM (
SELECT c.customerID, c.customerName,
MAX(o.orderID) - MIN(o.orderID) AS orderDifference
FROM customer c
JOIN orders o ON c.customerID = o.o_customerID
GROUP BY c.customerID, c.customerName
) subquery
WHERE orderDifference &amp;gt; 1;
```

	customerID	customerName
▶	4	Brian Gomez
	6	Katelyn Martinez
	7	Douglas Browning
	8	Kimberly Weaver
	9	Ebony Knight
	10	Benjamin Jones
	11	Brianna Mcdonald

Query 11: (UNION)Write a SQL query that returns the customer id and address whose name is Johnson LLC; or have ordered the productID ‘50’

```
SELECT DISTINCT c.customerID, c.customerAddress
FROM customer c WHERE c.customerName = &;Douglas Browning&;
UNION
SELECT DISTINCT c.customerID, c.customerAddress
FROM customer c
LEFT JOIN orders o ON c.customerID = o.o_customerID
LEFT JOIN product p ON o.o_plantID = p.p_manufacturingID
WHERE p.productID = 50;
```

	customerID	customerAddress
▶	7	496 Brooks Station, Katelynville, AK 04919

Query 12: (WITHOUT LIMIT) Retrieve names and total count of top 5 productNames which were ordered by customers (without LIMIT)

```
SELECT productName, orderCount
FROM (
SELECT p.productName, COUNT(*) as orderCount,
ROW_NUMBER() OVER (ORDER BY COUNT(*) DESC) as rn
FROM product p
JOIN delivery d ON p.productID = d.d_productID
GROUP BY p.productName
) subquery
WHERE rn &lt;= 5;
```

	productName	orderCount
▶	FluidicForce Mechanism	11
	HydraForce Actuator	10
	FluidicCeramic Component	10
	NanoProbe Analyzer	9
	NanoGlide Mechanism	9

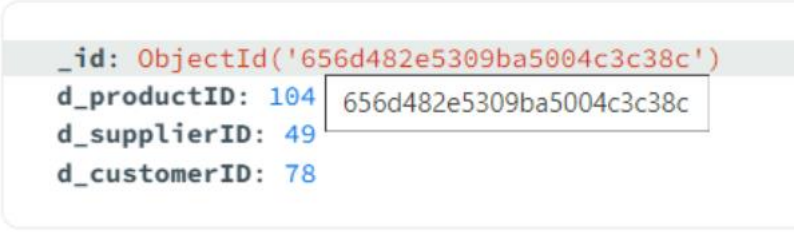
Query 13: (CTE) Retrieve top 5 plants which manufacture maximum products without limit

```
WITH RankedPlants AS (
SELECT p.plantID, p.plantCity, p.plantZipcode, COUNT(pr.productID) AS
totalProductsManufactured,
ROW_NUMBER() OVER (ORDER BY COUNT(pr.productID) DESC) AS
plant_rank
FROM manufacturing m
JOIN product pr ON m.manufacturingID = pr.p_manufacturingID
JOIN plant p ON m.m_plantID = p.plantID
GROUP BY p.plantID, p.plantCity, p.plantZipcode
)
SELECT plantID, plantCity, plantZipcode, totalProductsManufactured
FROM RankedPlants
WHERE plant_rank &lt;= 5;
```

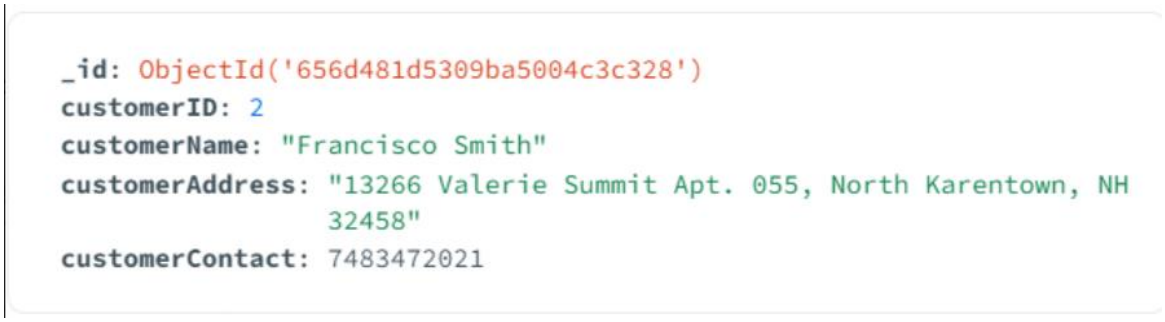
	plantID	plantCity	plantZipcode	totalProductsManufactured
▶	12	Chambersville	42431	17
	27	Lauratown	47688	16
	28	Lake Brandifurt	27630	16
	30	Kingside	25226	15
	4	North Scottbury	50402	14

V. IMPLEMENTATION OF RELATIONAL MODEL NoSQL

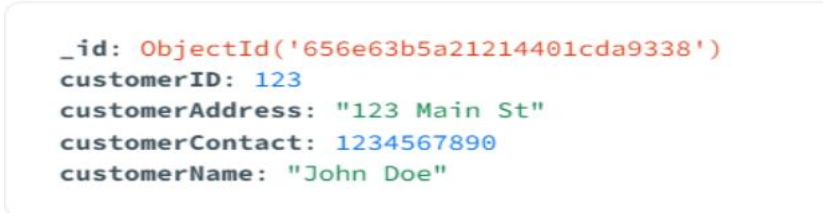
Query 1: Retrieve deliveries made to a specific customer for a particular product type
[{$match: {d_customerID: 78,d_productID: 104}}]



Query 2: Retreive details of customer with customerContact: 7483472021
[{$match: {customerContact: 7483472021}},{$limit: 1}]

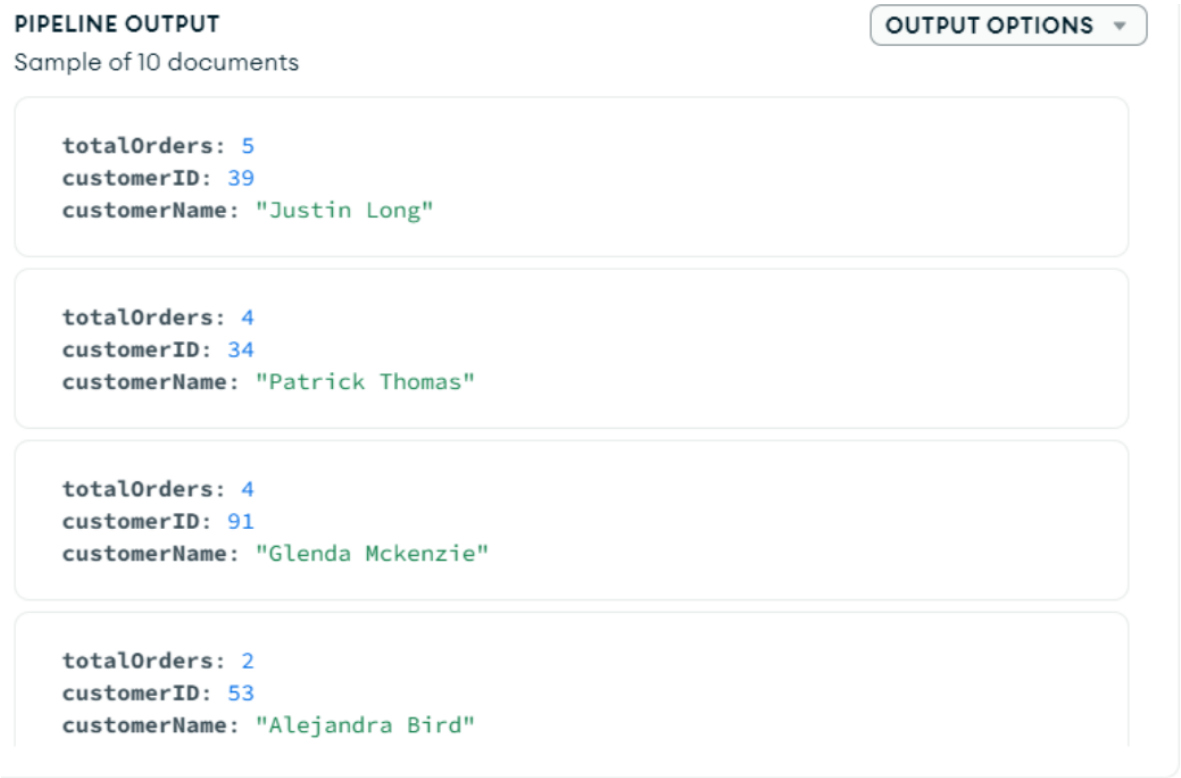


Query 3: insert customer details into customer table
[{$match: {customerID: 123}},
{$set: {customerName: John Doe,customerAddress: 123 Main St,customerContact: 1234567890}},{$merge: {into: customer, whenMatched: merge, whenNotMatched: insert}}]



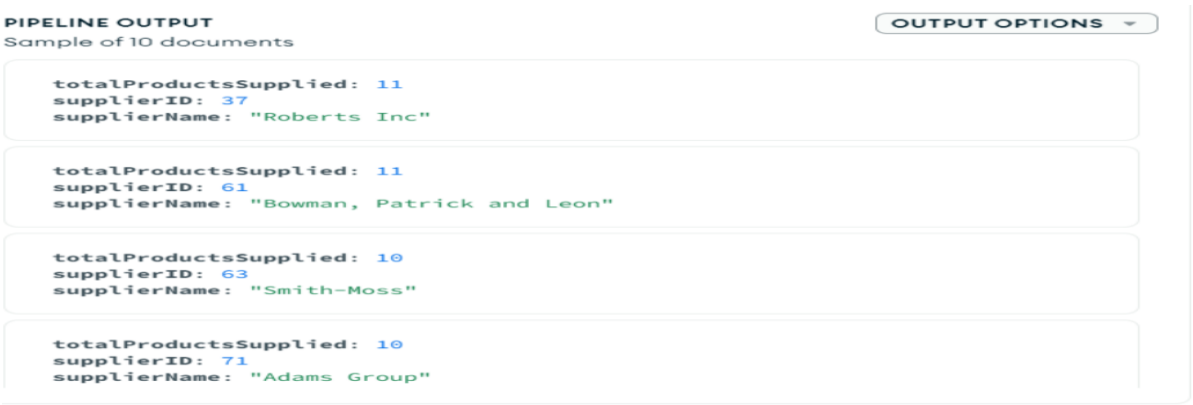
Below queries are performed using complete 12 tables of Manufacturing database

Query 4: Retrieve the count of orders placed by each customer along with their names.
[{$group: {
_id: $o_customerID, customerName: {$first: $customerName},totalOrders: {$sum: 1}
}},{$lookup: {from: customer, localField: _id, foreignField: customerID, as: customerDetails}},
{$unwind: $customerDetails},
{$project: {_id: 0,customerID: $_id, customerName: $customerDetails.customerName, totalOrders: 1}}]



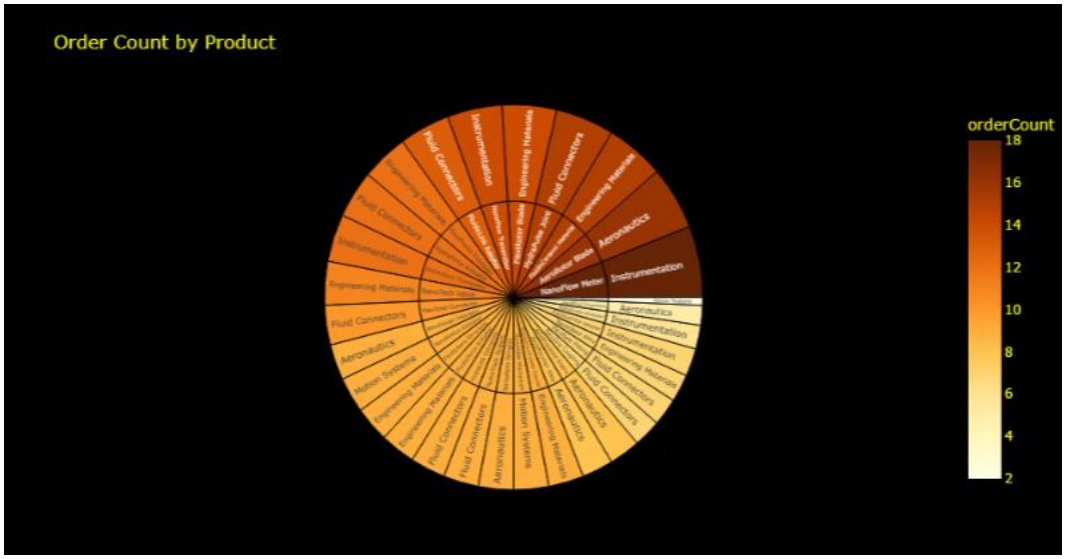
Query 5: Find the total count of products supplied by each supplier and sort the result in descending order.

```
[{$group: {_id: $d_supplierID,supplierName: {$first: $supplierName},totalProductsSupplied:
{$sum: 1}}},{$lookup: {from: supplier,localField: _id,foreignField: supplierID, as:
 supplierDetails}},{$unwind: $supplierDetails},{$project: {_id: 0, supplierID: $_id,
supplierName: $supplierDetails.supplierName,totalProductsSupplied: 1}},{
$sort: {totalProductsSupplied: -1}}]
```

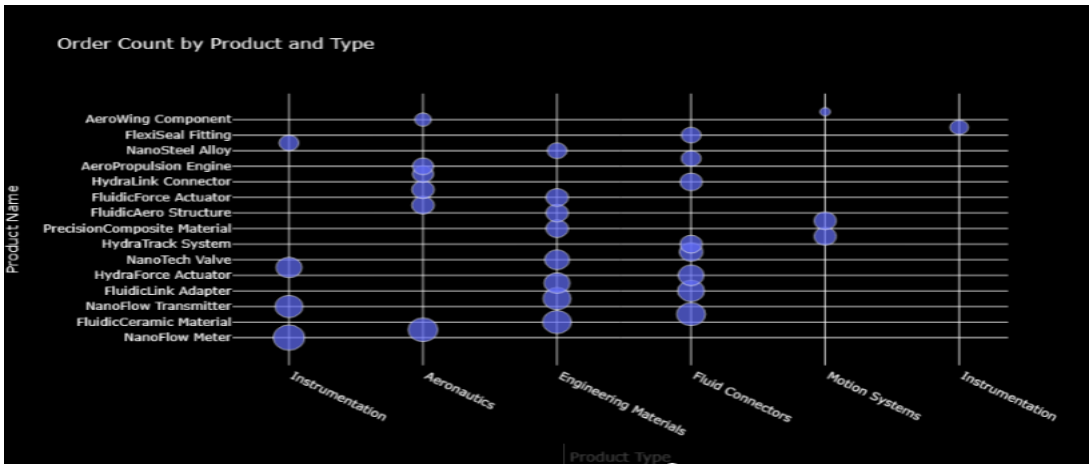


VI. DATABASE ACCESS VIA PYTHON

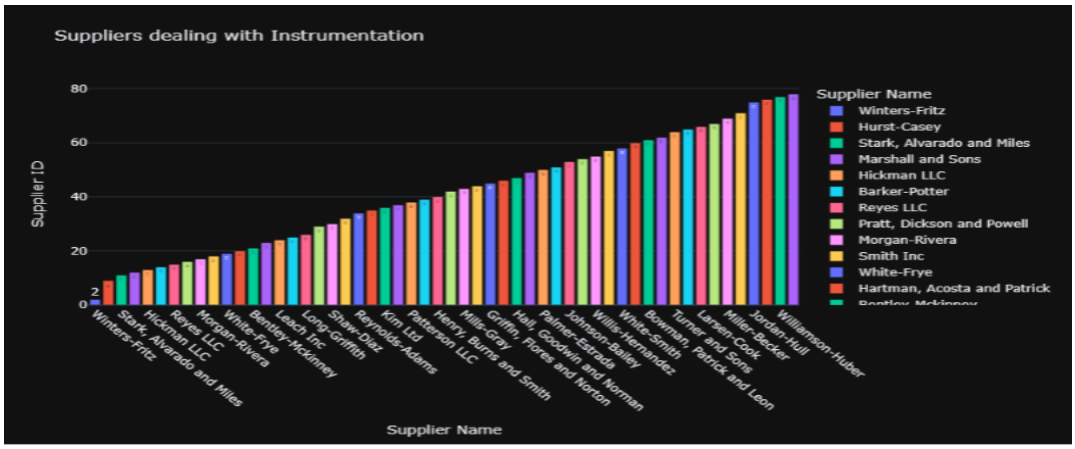
Graph 1: Ordercount per product



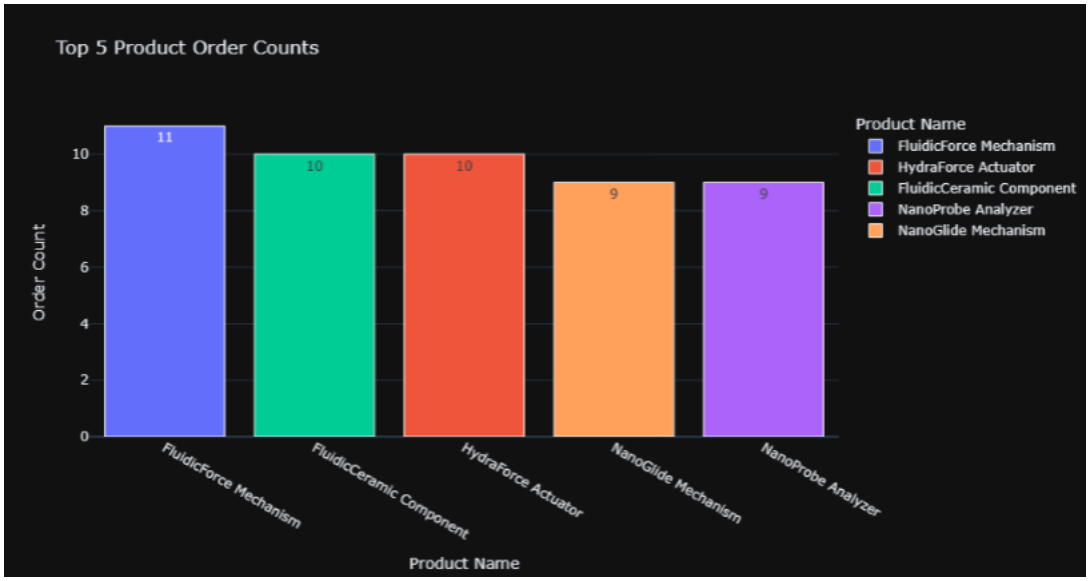
Graph 2: Ordercount per productname and producttype



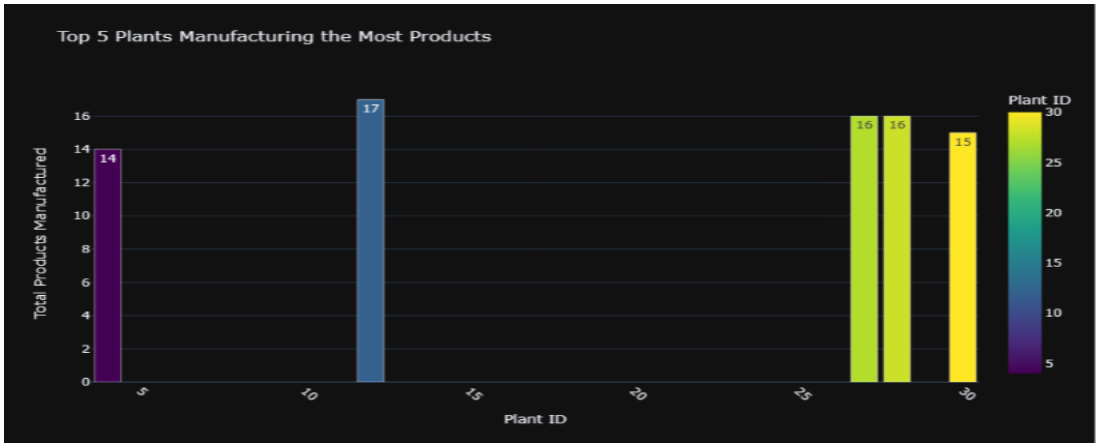
Graph 3: suppliers who deal with Instrumentation



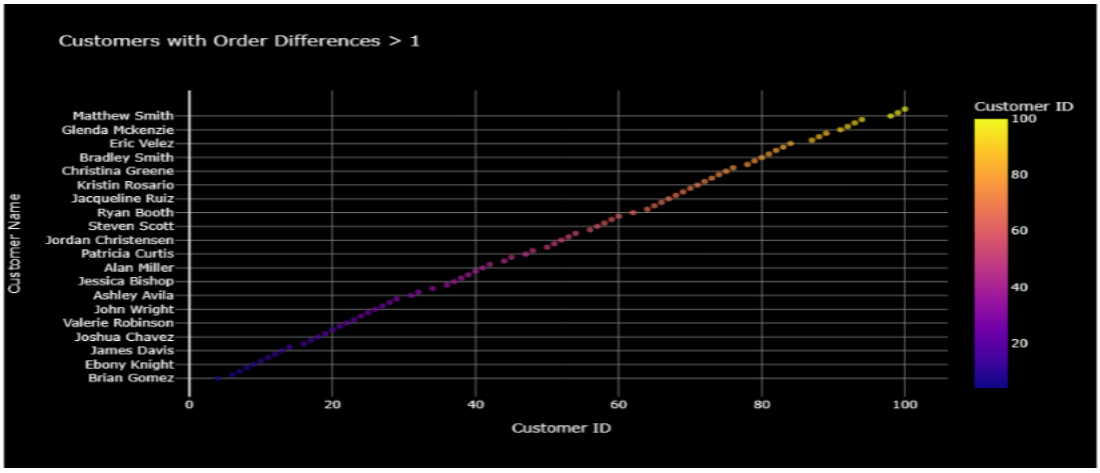
Graph 4: Top 5 products ordered



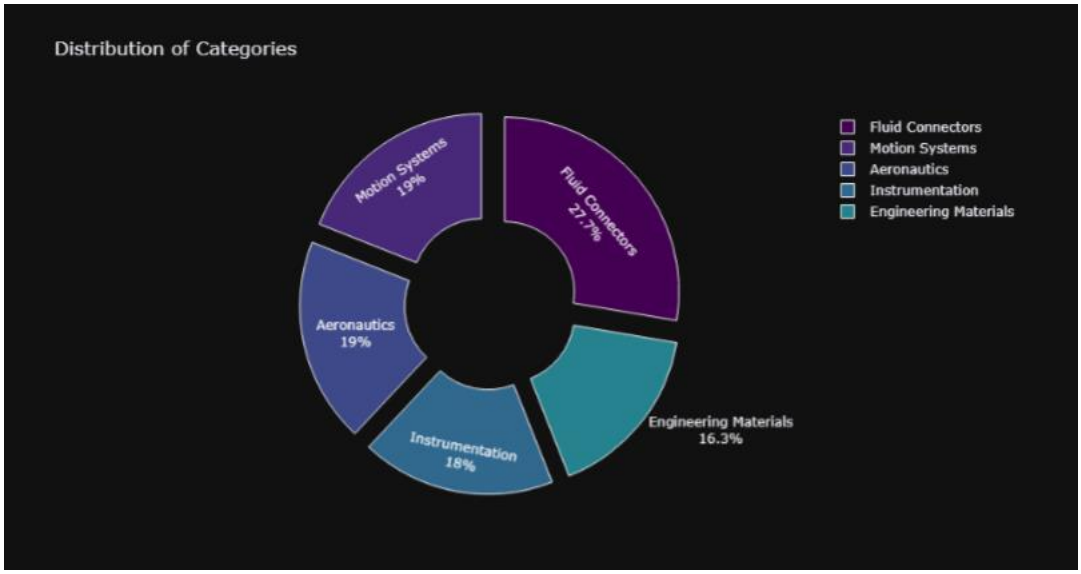
Graph 5: Top 5 manufacturing plants



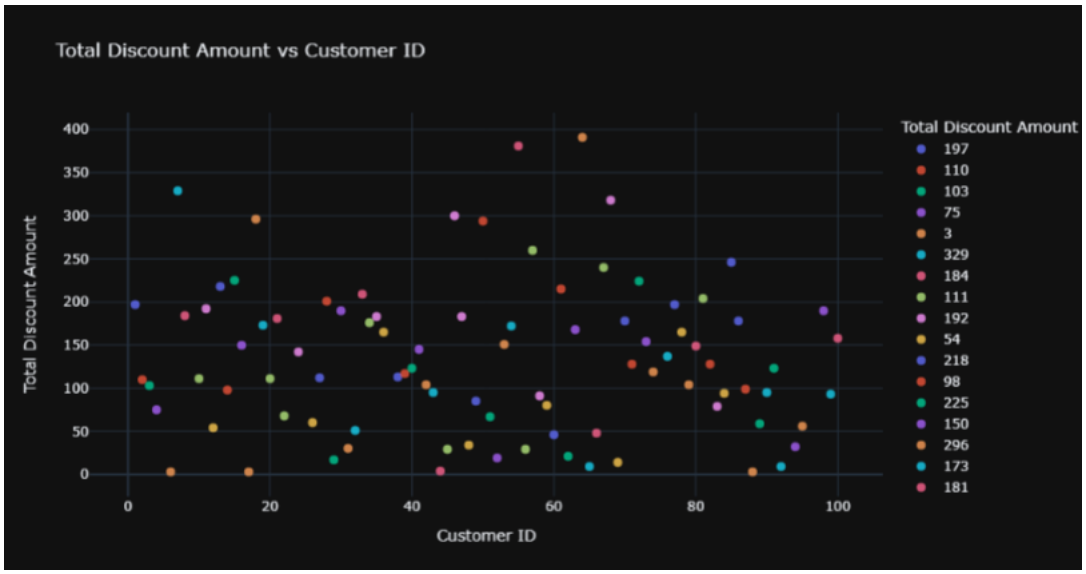
Graph 6: customers max and min order difference is more than 1



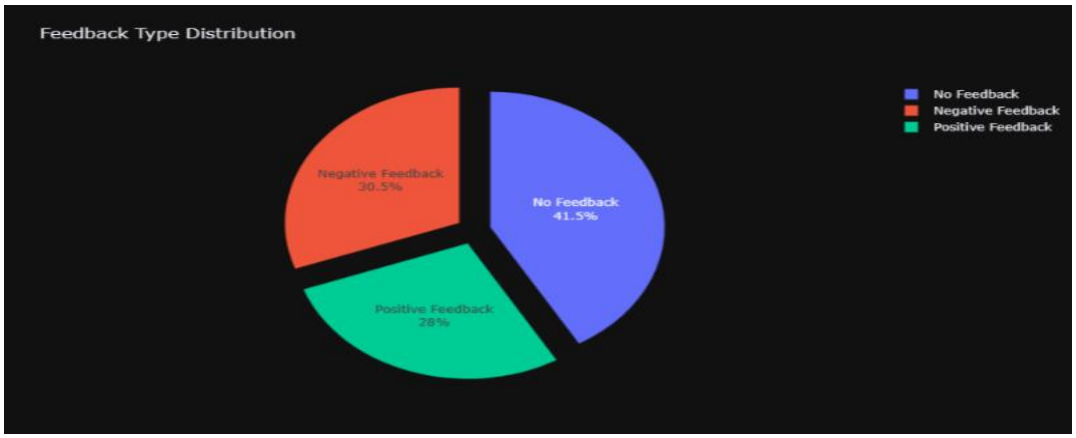
Graph 8: orders distribution of productTypes



Graph 9: DiscountAmount scatterplot of customers

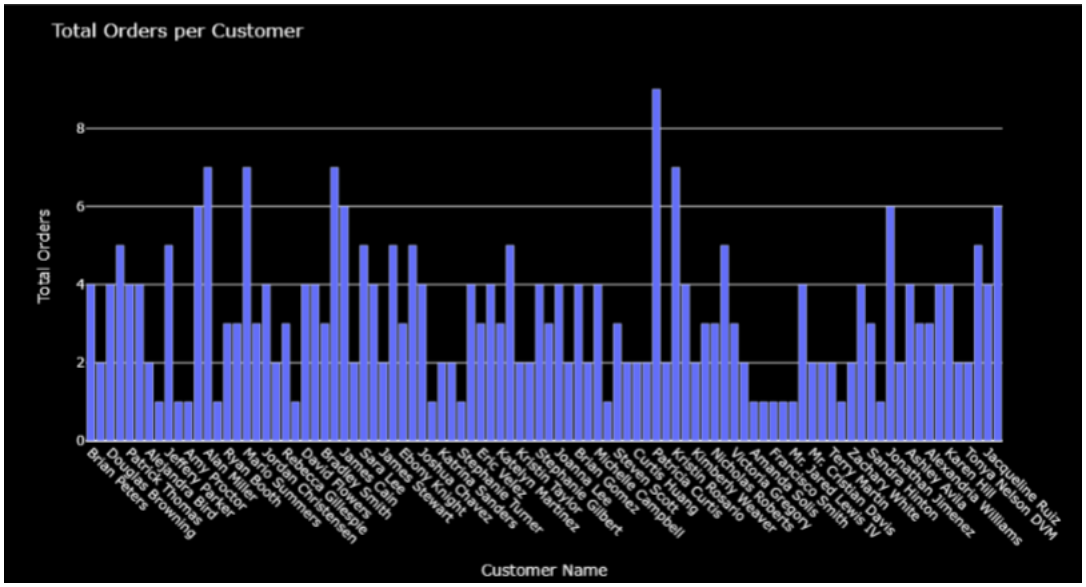


Graph 10: Feedback distribution

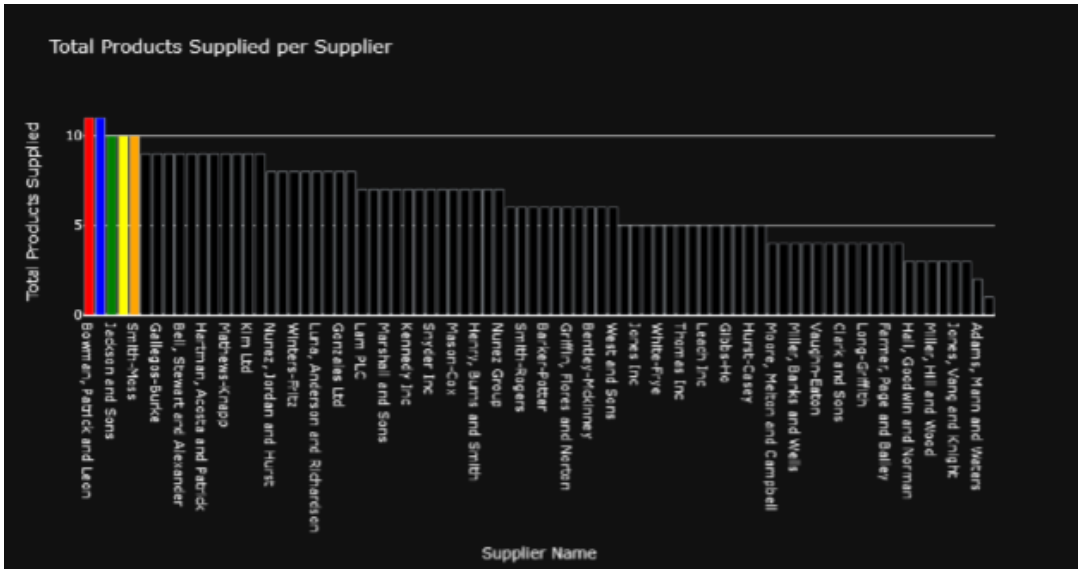


NoSQL- PYTHON CONNECTION

Graph 1:customername with their ordercount



Graph 2: Total products supplied by supplier



VII. SUMMARY AND RECOMMENDATIONS

Blue Manufacturers helped us to get the real time insights for our business model that enhanced the company performance to understand customer demand and our supply chain. Ours is mainly an Industry oriented database which helped us to improve transportation facilities as the customers were facing very bad experience as per our feedback.

The visualizations derived from our data analysis distinctly highlight the substantial prevalence of Fluid Connectors in the supply chain, closely followed by Aeronautics and Motion Systems, both sharing a comparable market share. Noteworthy is the observation that only a modest percentage, specifically 5-10% of our clientele, expressed satisfaction with the current discount offerings, while the majority perceived these incentives as minimally advantageous.In response to these insights, it is imperative for our company to strategically enhance customer acquisition and bolster profit margins. A targeted approach involves the development of a diverse array of referral offers designed to attract a substantial customer base. Emphasis should be placed on creating compelling incentives that resonate with potential customers, thereby positioning our company as a preferred choice in the market. Furthermore, a proactive strategy involves directing referral programs from the company itself, rather than relying solely on recommendations from existing customers. This approach ensures a more controlled and directed outreach, potentially yielding a more significant impact on customer acquisition and overall profitability. As we plan ahead, aligning these initiatives with a forward-thinking perspective will be instrumental in securing a competitive edge in the market.

Our ongoing efforts to enhance our database infrastructure primarily focus on elevating data quality and performance metrics in alignment with current market demands. Recognizing the significance of addressing these key parameters, we are dedicated to implementing refinements that ensure the reliability and efficiency of our data assets. In parallel, the adoption of MongoDB, a prominent NoSQL database, is a strategic move aimed at handling extensive volumes of unstructured and unprocessed data in the form of documents. This decision is underpinned by the recognition of MongoDB's efficacy in managing dynamic and diverse datasets. The proficiency gained through our experience with NoSQL tables imparts a distinct advantage, particularly in the realms of real-time applications and business analysis. This expertise positions us strategically to navigate the complexities inherent in processing and deriving actionable insights from contemporary data landscapes, contributing significantly to our operational effectiveness and decision-making capabilities.