USE CASE STUDY REPORT

Student Names: Prajwal Gowda

Executive Summary:

The primary objective of this study was to design and implement an ERP module that is industry-ready software. Software that stores master and transaction details. This model covers all the five gold modules that are needed in a manufacturing company- sales module, purchase module, finance module, service module, and production module. The basic idea here is to understand the flow of business that is Procure to pay: here, we need master data of supplier, employee, inventory, and transaction data of purchase order, receive entry, inspection, ap invoicing, and payment entry. Quote to Cash: we need master data of the customer, part, quality control(production), and transaction data of sales orders, shipment, AR invoice, and payment entries. The software will keep the company having tracker of all their profit and loss, costs, efficiency, etc.

The database was designed by taking all the input from the company along with the master and some transaction history. The EER and UML diagrams were modeled, followed by the mapping of the conceptual model to a relational model with the required primary and foreign keys. This database was then implemented fully MySQL and a prototype with two tables was implemented on Humongous, Mongo DB to study the feasibility of this database in a NoSQL environment. The created database is a great success, and by connecting it to Python, the analytics capabilities are immense, some of which have been shown in the study. Feather the data was uploaded on Tableau and a connection was established between a few tables through flowcharts and some visualizations were performed, some of which have been shown in the study.

These queries can be very helpful to track all its records from the database and can get accurate records. Adding the finance database will make this software ready to go live.

I. Introduction

PML is a metal company planning to open a new manufacturing plant in India. The company needs a database system to store all its master and transaction data.

Procure to pay:

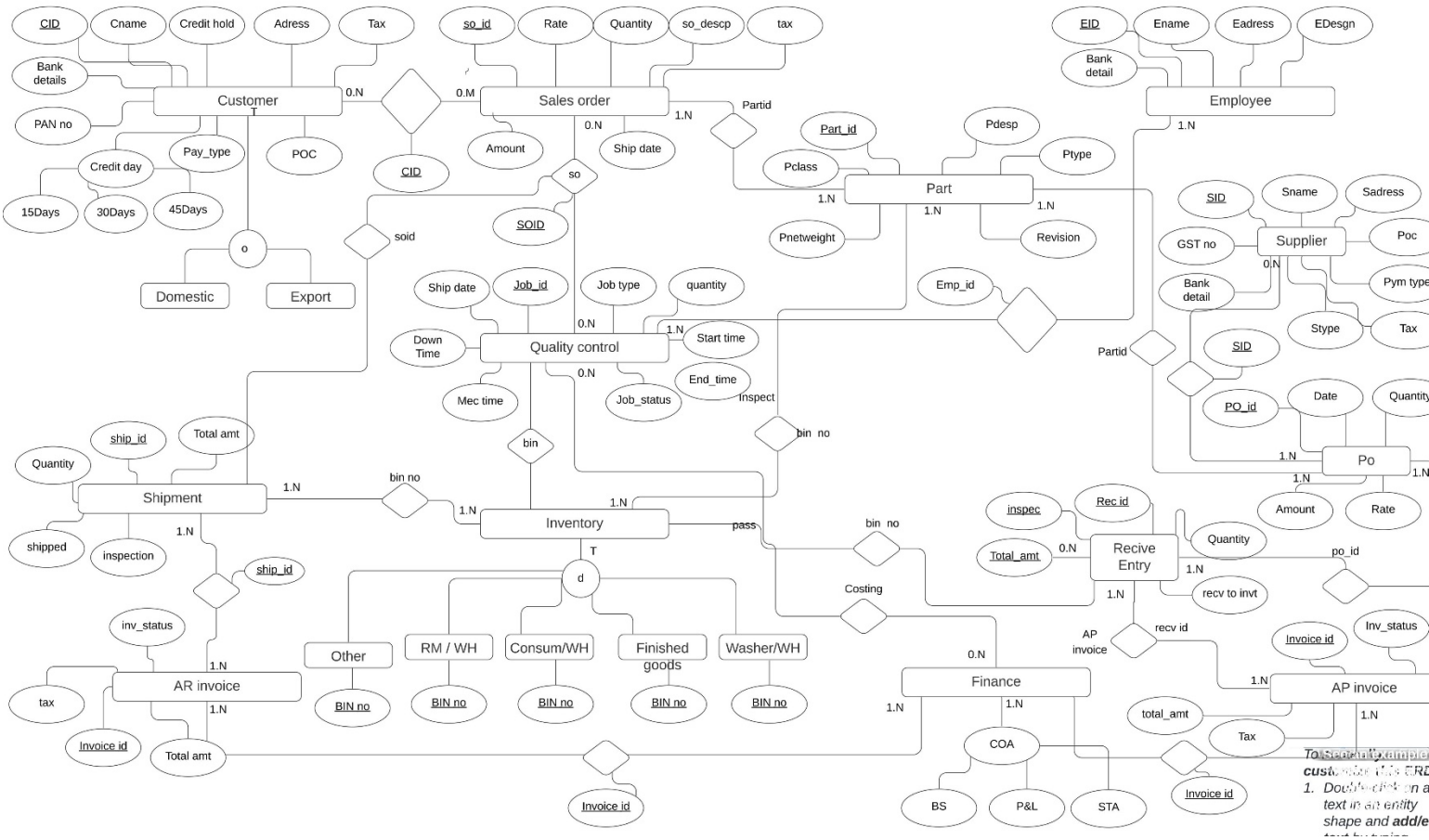
- Employees inform the purchase manager about the particulate part that is required for production, and the Purchase manager finds the suppliers who give the best price and material for that part.
- Purchase managers send the mail to the supplier, and after the confirmation from the supplier, the PO is created.
- The supplier sends the Parts, and a receive entry is created once the parts are on the floor; parts are inspected if they pass, they are added to the inventory; otherwise, if they fail, a DMR is created.
- The inventory manager again checks the failed parts, where he can accept, retrieve, or reject them, for rejected parts, a debit memo is created.
- AP invoice is created, and payment entry is done.

Quote to Cash:

- When a customer inquiries and requests a quotation, a case entry is created where all the details of the parts are mentioned with rate, this case entry is sent as a quotation to the customer.
- Once the customer confirms the order, we convert the case entry to SO.
- SO is seen by the production department: the department checks the ship date and plans their production accordingly.
- Production starts when a job is entered (job entry) where the method of manufacture is mentioned, what materials are needed, the bill of material is present, and time and expense. Once the production is completed, the product is inspected.
- Once the inspection is done, the job received to inventory is done.
- A customer shipment entry record is created, an AR invoice is created, the product is delivered to the customer via transport, and the customer makes the payment per credit period.

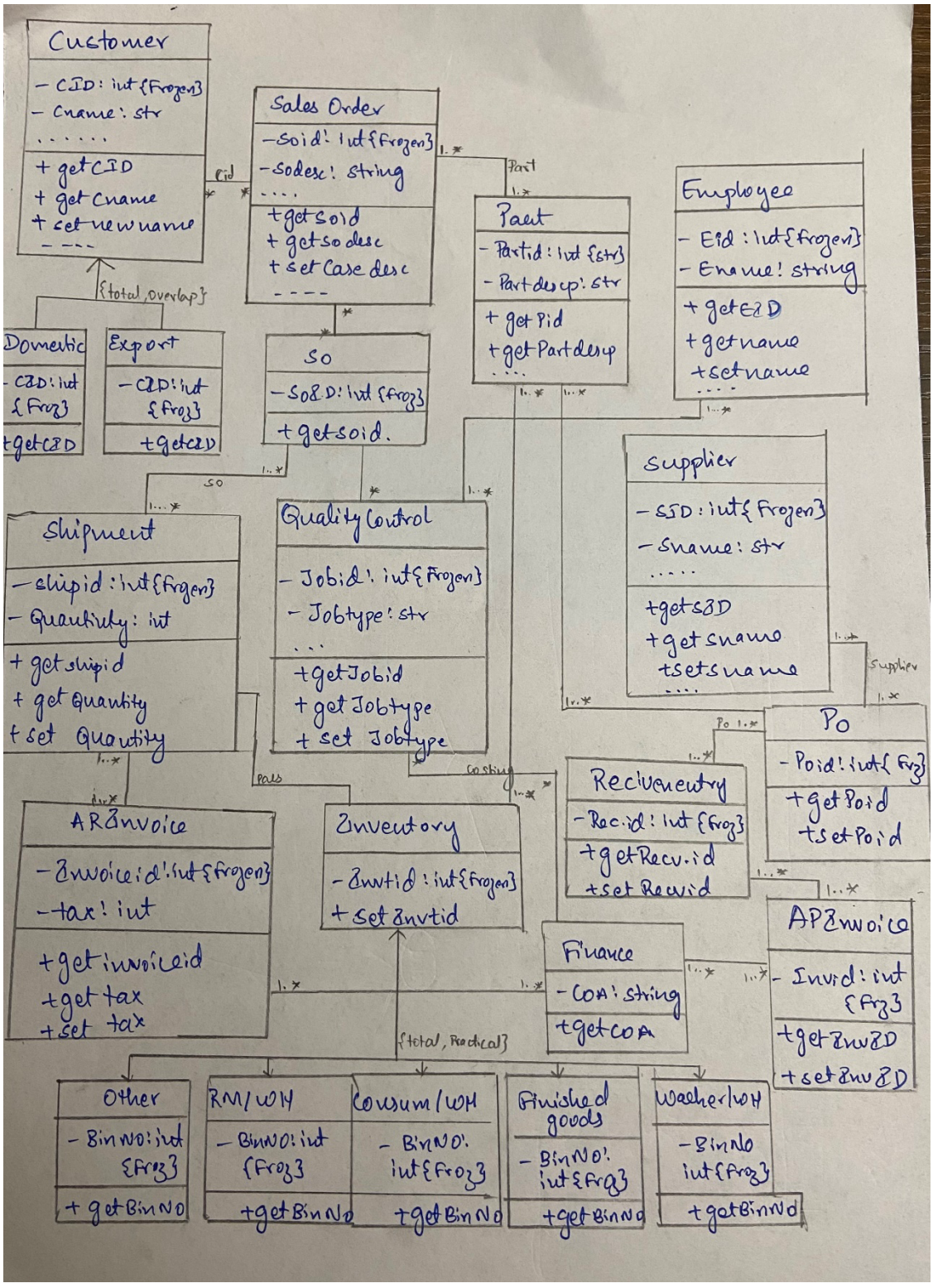
II. Conceptual Data Modeling

1. EER Diagram



-

2. UML Diagram



III. Mapping Conceptual Model to Relational Model

Primary Key- Underlined **Foreign Key- *Italicized***

Customer (Customer_Id, Customer_name, Customer_Type ,Address, Point_of_contact, Credit_hold, Credit_Days, Payment_Type, GST_Number, Pan_Number, Tax, Bank_Details)

Part (ParT_ID, Part_name, Part_description, Part_Type, Part_Class, Part_net_weight, Revision)

Sales_Order_to_Part (*Part_ID*)

Sales_Order (SO_ID, *Customer_ID*, *Part_ID*, SO_Description, Ship_Date, Quantity, Rate, Tax, Amount)

Inventory (Bin_NO, Warehouse_Name)

Shipment (Ship_ID, *SO_ID*, Quantity, Total_Amount, Shipped, Inspection, *Bin_NO*)

SO_to_Shipment (*SO_ID*)

Employee (Employee_ID, Employee_Name, Employee_Designation , Employee_Address , Bank_Deatils)

QualityControl (Job_ID, *SO_ID,* Job_type, Start_time, End_time, Machine_time, Down_time, Job_Status, Inspection, *Employee_ID*, *Bin_No*)

SO_to_QualityControl (*SO_ID,* Ship_Date)

Employee (Emp_ID,Emp_Name,Emp_Address, E_Desgn, Age, Joining_date, Bank_deatils)
On_Floor_manager (*Emp_ID*, *Job_ID*)

Inventory_to_Shipment (*Bin_No*)

Shipment_to_AR_invoice (*CS_ID*,)

AR_Invoice(Invoice_ID, *Ship_ID*, Tax, Total_Amount, InvoiceStatus)

Inventory_to_part (*Bin_ID)*

Supplier (Supplier ID, Supplier_Name, Address, Supplier_Type, Credit_Days, Payment_Type, Point_of_contact, GST_number , Pan_number, Tax, Bank_Detail)

Purchase_order (PO_ID, *Supplier_ID*, *Part_ID,* Date, Quantity, Rate, Tax, Amount)

Supplier_ Purchase_order (Supplier_*ID*, Date)

Part_to_PO (*Part_ID)*

Receipt_Entry(Rec_ID, *PO_ID*, Quantity, Total_Amount, ReceivedtoInventory, Inspection, *Bin_No*)

PO_to_Receip_Entry (*PO_ID)*

Receip_Entry_to_AP_Invoice (*Rec_ID*)

AP_Invoice (Invoice_ID, *Rec_ID*, Tax, Total_Amount, InvoiceStatus)

IV. Implementation of Relation Model via MySQL and NoSQL

MySQL Implementation:

The database was created in MySQL and the following queries were performed:

Query 1: Find sales order with distinct customer_id with their rate, quantity and amount'.

```
use metal; {Schema}
select distinct(Customer_ID),Rate,Quantity,Amount from salesorder
group by customer_id
order by customer_id asc;
```

Output:

Customer_ID	Rate	Quantity	Amount
1000	500	45	22500
1001	2000	67	134000
1002	3000	3	9000
1003	1000	10	10000
1009	2000	30	60000
1010	1000	20	20000
1011	10000	5	50000
1012	10000	6	60000
1013	1000	21	21000
1014	2500	56	140000

Customer_ID	Rate	Quantity	Amount
1015	1200	3	3600
1016	500	100	50000
1017	3000	23	69000
1019	10000	3	30000
1020	3000	4	12000
1021	3500	2	7000
1022	1200	45	54000
1023	3000	2	6000
1024	20000	6	120000
1025	3000	7	21000

Customer_ID	Rate	Quantity	Amount
1029	3500	12	42000
1030	2300	45	103500
1031	5400	6	32400
1034	2300	8	18400
1035	3000	9	27000
1045	2000	7	14000
1046	2000	10	20000
1047	3400	82	278800
1048	2500	23	57500
1049	500	34	17000

Query 2: Find job id, sales id, job type, inspection and bin no from qualitycontrol whose job type starts with "F%".

```
use metal; {Schema}
select job_id, so_id,job_type,inspection,bin_no
from qualitycontrol where job_type Like 'F%'
limit 10;
```

Output:

Job_ID	SO_ID	Job_Type	Inspection	Bin_No
502	102	Forging	No	FGSF01
504	104	Forging	No	FGSF09
506	106	Forging	No	FGSM04
510	110	Forging	No	FGSM05
515	115	Forging	No	FL06
519	119	Forging	Yes	SF02
522	122	Forging	Yes	CM03
525	125	Forging	Yes	CM05
529	129	Forging	Yes	FGSM04
531	131	Forging	Yes	FGSM06

Query 3: Find Purchase order whose invoice status are open, with receipt entry id, quantity and total amount.

```
use metal; {Schema}
select r.PO_id,r.rec_id, i.invoice_id,  r.quantity, i.total_amount, i.invoicestatus
from receiptentry r
join apinvoice i
on r.rec_id=i.rec_id
having i.Total_amount < 10000;
```

Output:

PO_id	rec_id	invoice_id	quantity	total_amount	invoicestatus
107	7	10007	20	6720	Open
128	28	10028	7	6160	Open
129	29	10029	5	2625	Open
130	30	10030	5	4725	Open

Query 4: Find sales order id, customer id, part id, ship id, invoice id and total amount with sales description whose letter or number start with "1%".

```
use metal; {Schema}
select s.so_id, so.customer_id, so.part_id, s.ship_id,i.invoice_id, so.so_description,
i.total_amount
from arinvoice as i
join shipment as s
using (ship_id)
join salesorder as so
using(so_id)
where so.so_description like '1%';
```

Output:

so_id	customer_id	part_id	ship_id	invoice_id	so_description	total_amount
107	1014	500007	7	10007	1/2" FLANGE 300#	144200
112	1023	600007	12	10012	1/2" SORF 150 #	6300
113	1024	600010	13	10013	1 1/2" SORF 150 #	132000
114	1025	600013	14	10014	1 1/2" SORF 150 #	24780
129	1046	500007	29	10029	1/2" FLANGE 300#	25600

Query 5: Find supplier id from supplier and purchase order using union.

```
use metal; {Schema}
select supplier_id from purchaseorder
union
select supplier_id from suppliers
having count(supplier_id)
limit 10;
```

Output:

supplier_id
1005
1045
1046
1050
1047
1033
1036
1037
1026
1027

Query 6: Find PO ID, Part ID and Quantity using purchase order and describe the duration of quantity that should be completed in one to five days using case when.

```
use metal; {Schema}
select  PO_id, Part_ID, quantity,
case
   when  quantity < 10 then 'One Day '
   when  quantity >=10 and quantity <20 then'Two Days '
   when  quantity >=20 and quantity <30 then'Three Days'
   when quantity >=30 and quantity <40 then'Four Days'
   when quantity >=40 and quantity <=50 then'Five Days'
end as Duration
FROM purchaseorder
order by quantity asc;
```

Output:

PO_id	Part_ID	quantity	Duration
113	700003	3	One Day
105	300001	5	One Day
130	400022	5	One Day
129	500023	5	One Day
128	400022	7	One Day
112	700002	10	Two Days
127	400021	10	Two Days
119	400013	11	Two Days
106	300002	15	Two Days
110	400004	18	Two Days
104	200022	20	Three Days
107	400001	20	Three Days
111	700001	20	Three Days
109	400003	22	Three Days
108	400002	23	Three Days
118	400012	23	Three Days
134	500004	23	Three Days
141	500007	25	Three Days
148	900006	28	Three Days

PO_id	Part_ID	quantity	Duration
103	200021	30	Four Days
146	900004	33	Four Days
142	500008	33	Four Days
116	400010	34	Four Days
150	900008	36	Four Days
120	400014	36	Four Days
144	500010	37	Four Days
131	400023	37	Four Days
121	400015	39	Four Days
137	500007	39	Four Days
122	400016	40	Five Days
145	900003	40	Five Days
102	200020	40	Five Days
133	400025	43	Five Days
123	400017	44	Five Days
136	500006	44	Five Days
149	900007	44	Five Days
135	500005	45	Five Days
117	400011	45	Five Days
114	700004	45	Five Days
143	500009	46	Five Days

Query 7: Find warehouse name from inventory, only those wasrehouse name from quality control whose inspection status is "Yes".

```
use metal; {Schema}
select distinct Warehouse_Name
from Inventory
where  Bin_No in
  (select Bin_No
   from qualitycontrol
   where inspection = 'Yes')
order by Warehouse_Name;
```

Output:

Warehouse_Name
Capital Machinary
Consumable
Finished Goods
Other
Ram Material

Query 8: Find supplier id, supplier name, payment type and tax whose purchase order rate is less than equal to 500

```
use metal; {Schema}
select s.supplier_id,s.supplier_name, s.payment_type, s.Tax
from suppliers  as s
where supplier_id = any
(select supplier_id
from purchaseorder as p
where p.rate <= 500);
```

Output:

supplier_id	supplier_name	payment_type	Tax
1046	Uday	Cheque	18
1036	Prajath	Cheque	12
1033	Kunti	Cheque	12

NoSQL Implementation:

Two tables(Customers, Invoice) have been created in Humongos online mongo db platform. The following mongo db queries were done:

Query 1: Find customer those tax is 28 and ship id is 4

```
db.collection.find(
({
    $or: [
   {Tax: 28}, {Ship_ID:4}
    ]
 })
```

Output | Database content


```
    },
    {
        "_id": 10009,
        "Total_Amount": 59000
    },
    {
        "_id": 10005,
        "Total_Amount": 63000
    },
    {
        "_id": 10004,
        "Total_Amount": 64000
    },
    {
        "_id": 10002,
        "Total_Amount": 76800
    },
    {
        "_id": 10010,
        "Total_Amount": 88320
    },
    {
        "_id": 10007,
        "Total_Amount": 144200
    }
]
```

Query 2: Find customers with total amount more than 50000

```
db.collection.find(
{"Total_Amount":{$gt:50000}
```

Output | Database content


```
[
    {
        "_id": "61b3cfeda3fc820015adfd32",
        "Invoice_ID": 10002,
        "Ship_ID": 2,
        "Tax": 28,
        "Total_Amount": 76800,
        "InvoiceStatus": "Open"
    },
    {
        "_id": "61b3cfeda3fc820015adfd34",
        "Invoice_ID": 10004,
        "Ship_ID": 4,
        "Tax": 28,
        "Total_Amount": 64000,
        "InvoiceStatus": "Open"
    },
    {
        "_id": "61b3cfeda3fc820015adfd35",
        "Invoice_ID": 10005,
        "Ship_ID": 5,
        "Tax": 5,
        "Total_Amount": 63000,
        "InvoiceStatus": "Open"
    },
```


Query 3: Using Aggregate function find the invoice and total amount which is sorted by total amount = 1.

```
db.invoice.aggregate([
{$group: {_id: "$Invoice_ID", Total_Amount: {$avg:"$Total_Amount"}}},
{$sort: {Total_Amount: 1}}])
```

Output | Database content

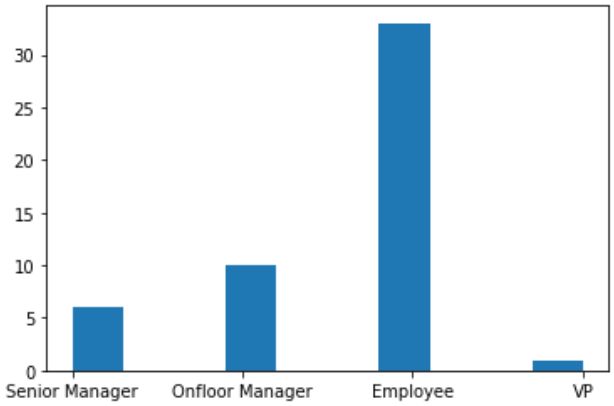

```
    },
    {
        "_id": 10009,
        "Total_Amount": 59000
    },
    {
        "_id": 10005,
        "Total_Amount": 63000
    },
    {
        "_id": 10004,
        "Total_Amount": 64000
    },
    {
        "_id": 10002,
        "Total_Amount": 76800
    },
    {
        "_id": 10010,
        "Total_Amount": 88320
    },
    {
        "_id": 10007,
        "Total_Amount": 144200
    }
]
```

```
[
    {
        "_id": 10008,
        "Total_Amount": 4032
    },
    {
        "_id": 10001,
        "Total_Amount": 11800
    },
    {
        "_id": 10006,
        "Total_Amount": 23100
    },
    {
        "_id": 10003,
        "Total_Amount": 23600
    },
    {
        "_id": 10009,
        "Total_Amount": 59000
    },
    {
        "_id": 10005,
        "Total_Amount": 63000
    },
```

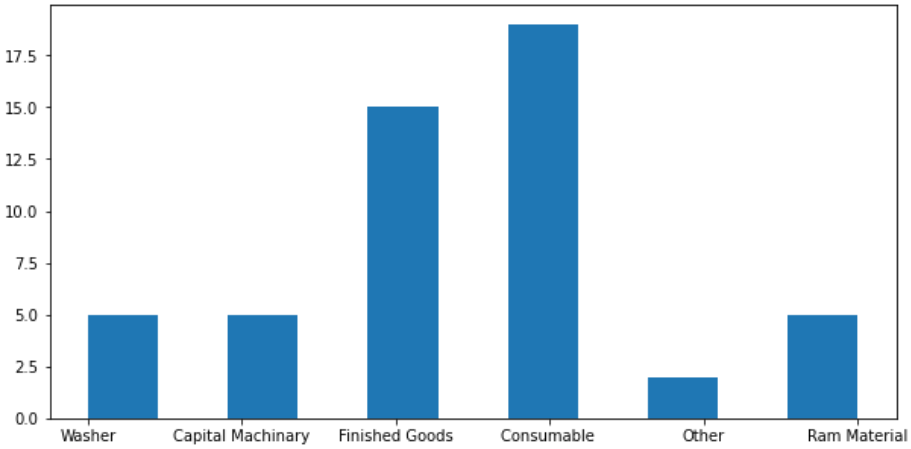
IV Database Access via Python

The database is accessed using Python and visualization of analyzed data is shown below. The connection of MySQL to Python is done using mysql.connector, , followed by converting the list into a dataframe using pandas library and using matplotlib, using numpay to plot the graphs for the analytics.

Graph1: Employees and their Designation

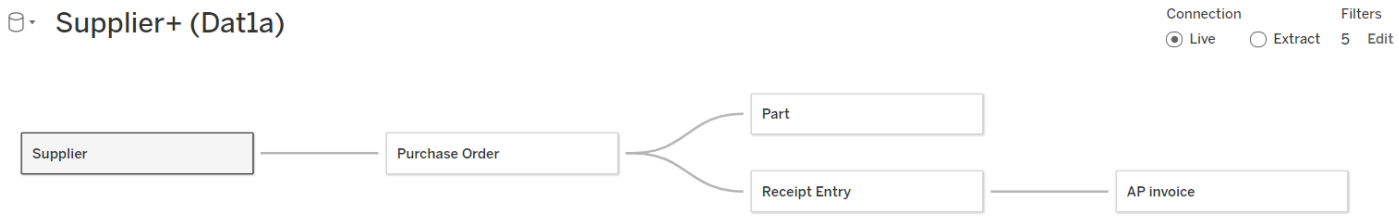


Graph3: Different and number of warehouses in the company



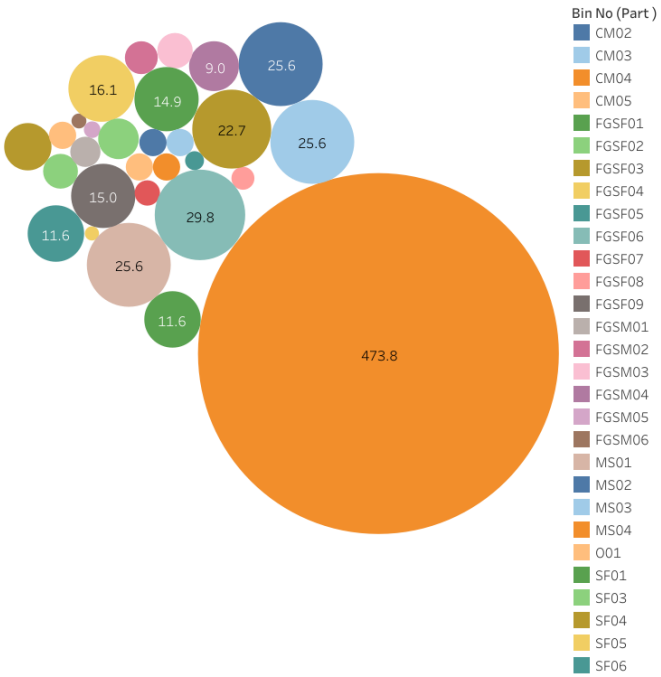
The database was also linked to Tableau and following are some visualizations in table.

Connected procure to pay that is Supplier to AP invoice.



<Supplier with quantity, rate, tax and amount>

<Part net weight >



VII. Summary and Recommendation

The focus was on a metal company but it's a good ERP software for all the manufacture company, who are planning to start a new manufacturing plant.

Adding finance module to the software which is currently not mapped but are designed in the eer and uml mode will bring a light to the software. Specially mapping the accounting system like chart of account, balance sheet, GL control and profit and loss statements etc. will enrich this ERP software can be used by any manufacturing company.