USE CASE STUDY REPORT

Student Names: Siddhesh More

Topic: Merchandise Printing Press

Executive Summary: The primary objective of the study was to design and implement a relational database that is industrial ready for the application in the customized printing industry for the use by the workers who have been finding difficulty to keep the track of the orders and the quantity of the supply to the retailers as they were facing a huge problem regarding data duplication. This relational database reduces data input process time by approx. 60% and the result in huge saving cost benefits across the industry. This database also implements a central analysis platform that has immense potential for analytics on the preferences of customers i.e., which product is customized, which city or the region the sales are greater, also the database gives the employees a monthly reminder to place an order for their respective products.

The database was modelled taking requirements of the data fields required by the industry regarding their sales with the retailers/customers, along with input from the employees of industry related to their sales. The EER and UML diagrams were modelled, 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 in MySQL by using the eight tables of the provided data for 300+ customers and then it was also implemented on the MongoDB NoSQL database to get the relationships and study the feasibility of this database in a NoSQL environment.

The created database is much more efficient to get the depths of the production line of the industry. Later connecting it to Python the analytics capabilities are immense, some of which

have been shown in the study. These queries can be very helpful for tracking the production line and getting the analysis for the improvement the industry requires in their sales. The scope of the project is to benefit the small-scale industry to later turn it into a large-scale industry and keep track of the production, further it may help the industry for marketing and advertising to expand their sales graph.

I. Introduction

The e-commerce business in the United States has grown so efficiently in past decades that the generation applies their creativity anywhere on just one tap. It is a noticeable change from the last five years, that people have started their own start-ups, mostly their own boutiques and many more rather than relying on the known fashionista companies to use the products with the same design that have the same original copies of around 1000+.
Since this small-scale industry is turning into a large-scale industry and due to which the relational database comes into picture. Having a relational database for recording all the data of sales and production of the industry solves the problem of Data duplication and can shorten the data entry process time by over 50% as the whole process of arranging the data is not required every single time. The industry can access relational databases to predict the behavior of the customer such as the likes and their dislikes. Another use of this database is to keep track on which part of the country the sales are more and on which product has given the most profit and this can help to increase the revenue of the industry.

This relational database includes a customer table which gives the insights of the customers details that includes the ID, address, the password and phone number, which will later give the details of from which region are the most customer to expand their sales more in that region also if the industry is tied up to a particular retailer store this data may give the monthly reminder to place orders for their respective products. Another addition to the database is the Template database which is correlated to the customer database which keeps the records of Customer ID like the customer database, the design pattern they have chosen, color, if it is customized or the old design. All these entities are combined, and a database is stored of every specific city with the number of productions done in a particular year. The database provides the industry with a simple yet powerful database that addresses and solves the key data entry process being faced by industry and by the employees in an orderly and efficient manner.

Impreso Prints Ltd. is a company that prints customized merchandise such as t-shirts, mugs, badges, etc. They require a detailed database that can help them improve operations, understand customer needs better, maximize profit and improve the efficiency of the supply chain. Their business model works on a push-pull mechanism. They obtain plain, blank and colorful merchandise from certain suppliers over the country, which are stored in the inventory. In the next step, these merchandises are printed as per templates. The templates are pre-existing, designed by graphic designers employed by the company. To customize, customers can create their own templates or upload their own images to the database. They can later place an order with their choice of template and quantity. The order details go to the printing department, where the merchandise is printed, and sent to the customer via shipping that has been outsourced.

II. Conceptual Data Modeling

Templates are updated by both designer (employee) and customer. Both entities can add multiple designs.
Customer can place more than one order, any quantities and mixed product types.
Printing times and scheduling, resources available are all important with the production timings, however, there is always enough buffer available to overcome shortages.
Customer shipping is outsourced, but customers can track their orders on the company website.
Forward (push) orders are placed to the suppliers after considering demand and shortages.
Inventory is stacked with the nearest suppliers, in case material is available.
Inventory cannot go empty in any other cases than supplier shortages, in which case, supplies from suppliers farer are order.

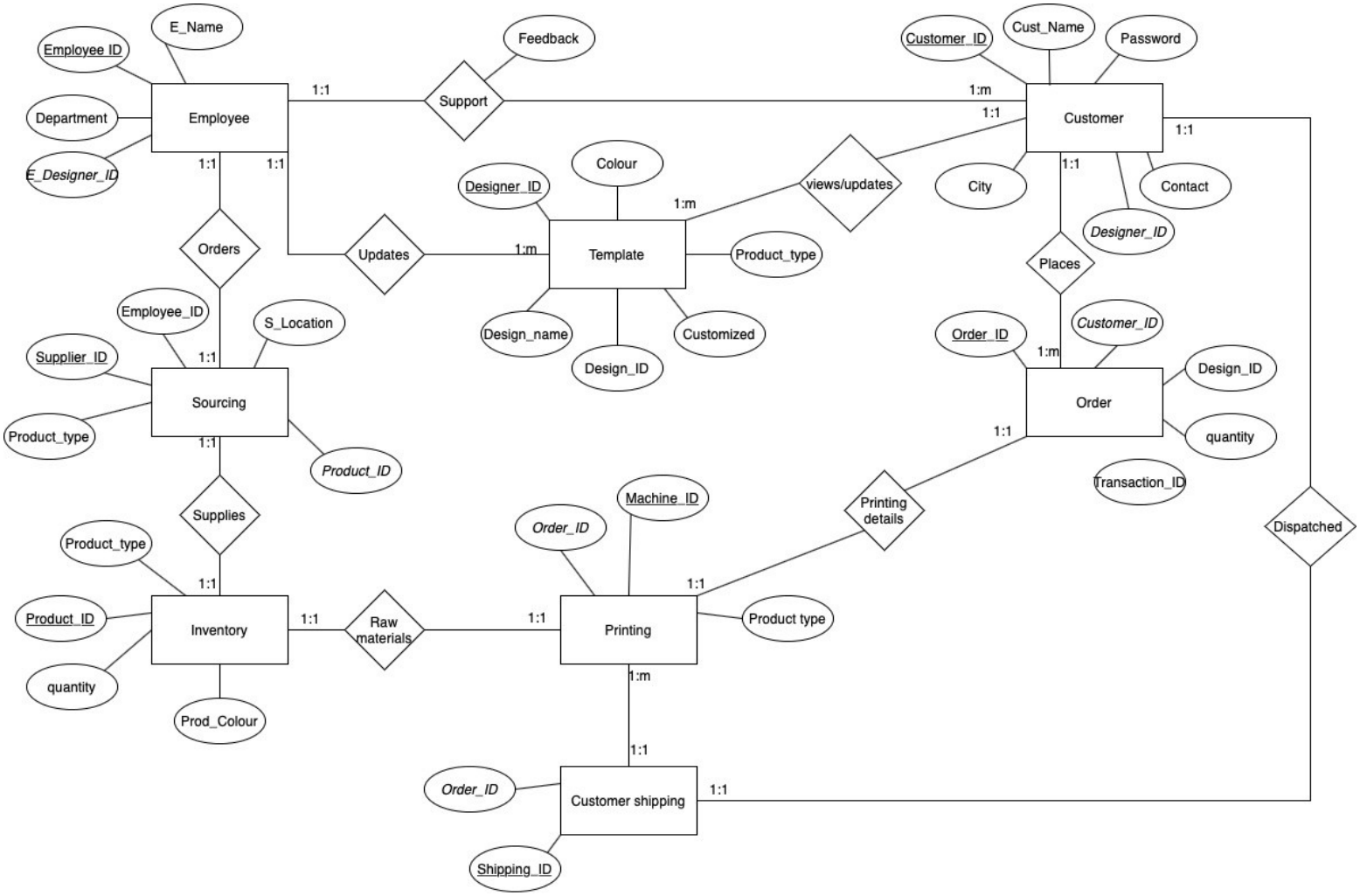

Figure 1: Entity-Relationship Diagram

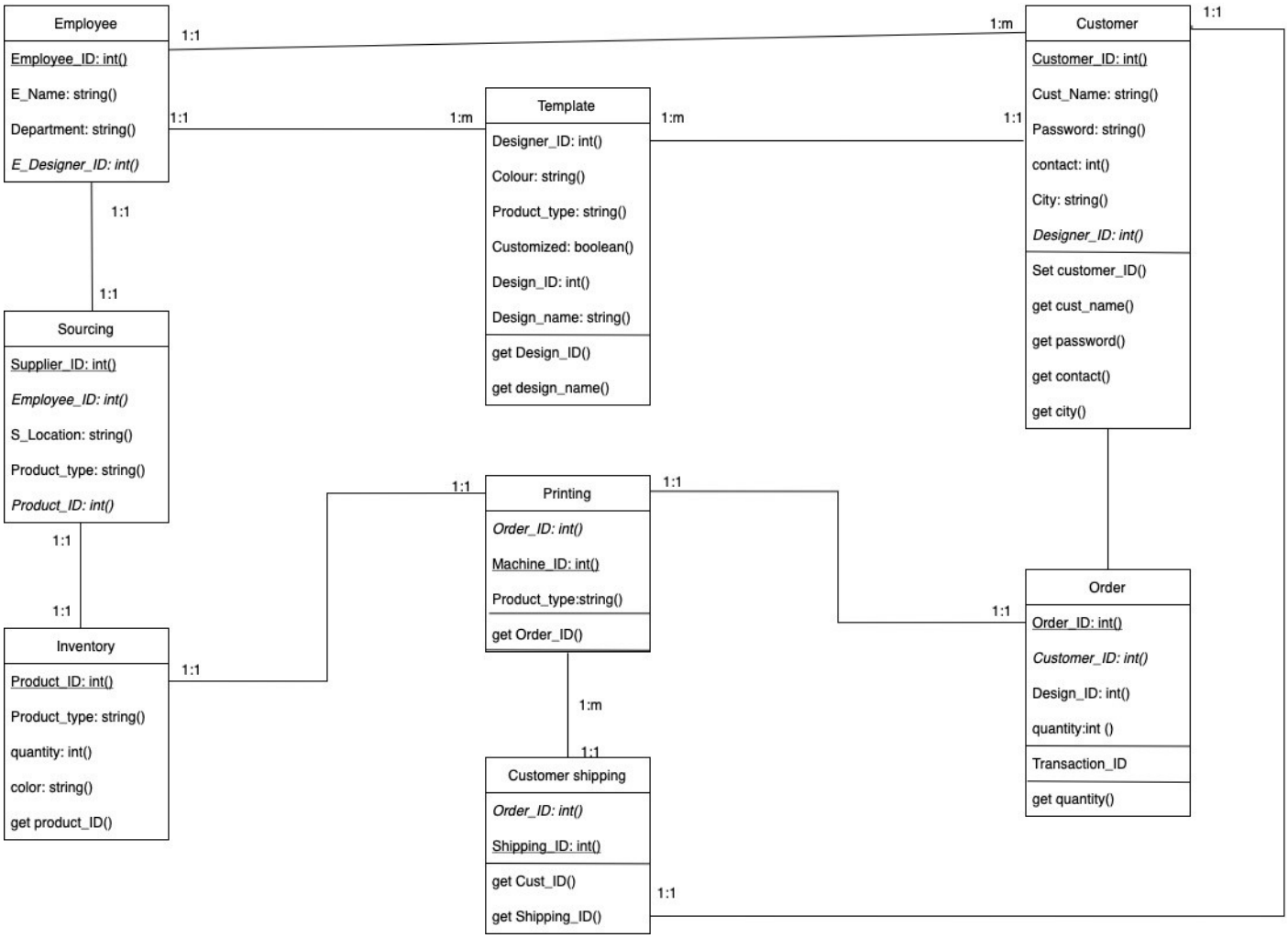

Figure 2: UML Diagram

III. Mapping Conceptual Model to Relational Model

Nomenclature: Primary key is underlined; *foreign key* is italicized.

Employee (Employee_ID, E_name, Department, *E_Designer_ID*)
Customer (Customer_ID, cust_name, password, contact, city, *Designer_ID*)
Orders (Order_ID, *Customer_ID,* Design_ID, quantity)
Template (Designer_ID, color, product_type, customized, Design_ID, Design_name)
Printing (*Order_ID*, Machine_ID, Product_type)
Customer_Shipping (*Order_ID,* Shipping_ID)
Sourcing (Supplier_ID, *Employee_ID*, S_location, Product_type, *Product_ID*)
Inventory (Product_ID, *Product_type*, quantity, color)

IV. Implementation of Relation Model via MySQL and NoSQL

Data was available in both structured and unstructured forms. It was cleansed, tabulated and implemented in MySQL to establish a working Database Management System. This section contains examples on how to handle various scenarios.

Scenario 1: The company has decided to award 10 customers who ordered the most quantity, and 5 of them who placed an order maximum time.

```
select b.Cust_Name
from orders as a, Customer as b
where b.Customer_ID = a.Customer_ID
order by a.quantity desc
limit 10;
```

Cust_Name
Alverta
Cherry
Clifford
Dee
Doshia
Lillis
Olie
Arrie
Merle
Onie

```
select customer_ID, count(*) from orders
group by customer_ID
order by count(*) desc
limit 5;
```

customer_ID	count(*)
100145	6
100158	5
100076	5
100167	5
100035	5

Scenario 2: The HR department needs a separate table with sourcing details of employees and suppliers in order to remind them if stock runs out. They also want to reward employees who sourced the most products.

```
Create view supplies as
(
select s.Employee_ID, s.Supplier_ID, i.Product_ID
from sourcing s
left join inventory i
on s.Product_ID = i.Product_ID
);
Select * from supplies;
```

Employee_ID	Supplier_ID	Product_ID
10006	200001	1
10018	200002	2
10030	200003	3
10042	200004	4
10061	200005	5
10006	200006	6
10018	200007	7
10030	200008	8
10042	200009	9

```
select Employee_ID, count(*) from supplies
group by Employee_ID
order by count(*) desc;
```

Employee_ID	count(*)
10061	9
10030	7
10042	7
10006	6
10018	6

Scenario 3: A new animated movie involving animals ‘The Lion King’ is going to be released. Find the customers who ordered similar designs to send personalized ads and offers.

```
select distinct c.customer_ID, c.Cust_name
from template as t, customer as c
where t.Design_name = 'animals'
or t.Design_name = 'movie'
or t.Design_name = 'cartoon'
and t.designer_ID = c.designer_ID
```

customer_ID	Cust_name
100001	Magdalen
100002	Maymie
100003	Minervia
100004	Muriel
100005	Neppie
100006	Olie
100007	Onie

Scenario 4: An old customer wants to reorder his design. Return his ordered design details.

```
select design_ID, transaction_ID
from orders
where customer_ID = 100086
```

design_ID	transaction_ID
359	731r175
547	430s944

Scenario 5: Company wants to expand. List out necessary details required for the same.

```
select city, count(*) from customer
group by city
order by count(*) desc
```

city	count(*)
New York	9
Boston	9
Nashua	9
Stowe	9
Rochestor	9
New Jersey	9

```
select S_location, count(*) from sourcing
group by S_location
order by count(*) desc
```

S_location	count(*)
Nashua	3
Stowe	3
Rochestor	3
New York	2
Boston	2
Los Angeles	2

Scenario 6: For tax paying purposes, company needs all the transactions for quantity > 20 items.

```
select o.transaction_ID, c.Customer_ID, c.cust_name
from orders o, customer c
where o.quantity > 20
and c.customer_ID = o.customer_Id
order by quantity desc
```

transaction_ID	Customer_ID	cust_name
657u252	100028	Alverta
746z792	100144	Cherry
933t400	100149	Clifford
742d703	100153	Dee
645a118	100055	Doshia
430s944	100086	Lillis

Scenario 7: A complaint is raised by a customer who ordered mugs of getting a manufacturing defect. Figure out the machine in need of repairs.

```
select distinct o.Customer_ID, p.Machine_ID
from orders o, printing p
where Customer_Id = 100218
and p.product_type = 'mug'
order by Machine_ID asc
```

Customer_ID	Machine_ID
100218	5
100218	6

Scenario 8: IT department want to set an automated system that updates items as ‘out of stock’ in the system; also displays the respective Employee ID.

```
select distinct i.product_ID, s.Supplier_ID, s.Employee_ID
from inventory i, sourcing s
where i.quantity = 0
```

product_ID	Supplier_ID	Employee_ID
31	200001	10006
14	200001	10006
7	200001	10006
31	200002	10018
14	200002	10018

V. Implementation in NoSQL using MongoDB

Scenario 1: The Manager needs the data of the product ‘Banner’ that have been produced in Machine with ID ‘10’

```
db.printing.find(
  {$and: [{ Machine_ID: 10},{ Product_type: 'Banner'}]}
)
```

	_id ObjectId	Order_ID String	Machine_ID Int32	Product_type String
1	61b3af63f37e8f0e65619e76	"2"	10	"Banner"
2	61b3af63f37e8f0e65619e77	"3"	10	"Banner"
3	61b3af63f37e8f0e65619e78	"4"	10	"Banner"
4	61b3af63f37e8f0e65619ea5	"49"	10	"Banner"
5	61b3af63f37e8f0e65619ed7	"99"	10	"Banner"
6	61b3af63f37e8f0e65619edf	"107"	10	"Banner"
7	61b3af63f37e8f0e65619ee4	"112"	10	"Banner"
8	61b3af63f37e8f0e65619ee5	"113"	10	"Banner"
9	61b3af63f37e8f0e65619ee6	"114"	10	"Banner"
10	61b3af63f37e8f0e65619f13	"159"	10	"Banner"
11	61b3af63f37e8f0e65619f45	"209"	10	"Banner"
12	61b3af63f37e8f0e65619f4d	"217"	10	"Banner"
13	61b3af63f37e8f0e65619f82	"270"	10	"Banner"
14	61b3af63f37e8f0e65619f8a	"278"	10	"Banner"
15	61b3af63f37e8f0e65619fbf	"331"	10	"Banner"
16	61b3af63f37e8f0e65619fc7	"339"	10	"Banner"
17	61b3af63f37e8f0e65619ffc	"392"	10	"Banner"
18	61b3af63f37e8f0e6561a004	"400"	10	"Banner"
19	61b3af63f37e8f0e6561a031	"445"	10	"Banner"

Scenario 2: The Marketing department needs the overview of sales in the city ‘New York’ to get insights and may try to apply more marketing strategies to expand the business in New York.

```
db.customer.find(
  {Address: 'New York'}
)
```

	_id ObjectId	Sr Object	Customer_ID Int32	Cust_Name String	Designer_ID Int32	Address String
1	61b3aefff37e8f0e65619ae7	{} 1 fields	100001	"Magdalen"	1200	"New York"
2	61b3aefff37e8f0e65619b12	{} 1 fields	100044	"Cherry"	1286	"New York"
3	61b3aefff37e8f0e65619b3d	{} 1 fields	100087	"Louetta"	1372	"New York"
4	61b3aefff37e8f0e65619b4b	{} 1 fields	100101	"Magdalen"	1400	"New York"
5	61b3aefff37e8f0e65619b76	{} 1 fields	100144	"Cherry"	1486	"New York"
6	61b3aefff37e8f0e65619ba1	{} 1 fields	100187	"Louetta"	1572	"New York"
7	61b3aefff37e8f0e65619baf	{} 1 fields	100201	"Magdalen"	1600	"New York"
8	61b3aefff37e8f0e65619bda	{} 1 fields	100244	"Cherry"	1686	"New York"
9	61b3aefff37e8f0e65619c05	{} 1 fields	100287	"Louetta"	1772	"New York"

Scenario 3: The production department suddenly got a hint that the T-shirts printed in violet colors have less production rate compared to other colors; so the departments needs a data to cross check regarding the issue simultaneously they also need the the quantity of customised shirts in their.

```
db.templates.find(
  {$and: [{ Customized: 'Y'},{ Color: 'Violet'},{ Product_Type: 'T-Shirt'}]}
)
```

	_id ObjectId	Design_Id Int32	Color String	Product_Type String	Customized String	Design_Name String
1	61b3c288f37e8f0e6561a3bd	157	"Violet"	"T-Shirt"	"Y"	"Abstract"
2	61b3c288f37e8f0e6561a415	245	"Violet"	"T-Shirt"	"Y"	"Abstract"
3	61b3c288f37e8f0e6561a46d	333	"Violet"	"T-Shirt"	"Y"	"Abstract"

VI. Database Access via R or 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 cursor.execute to run and fetchall from the query, followed by converting the list into a dataframe using pandas library and using matplotlib to plot the graphs for the analytics.

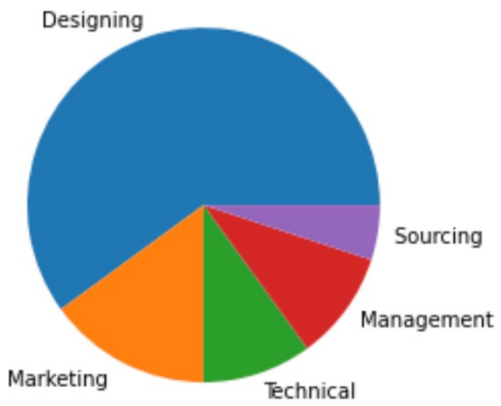

Figure 3: Pie chart representing diversification of 20 latest recruits

s

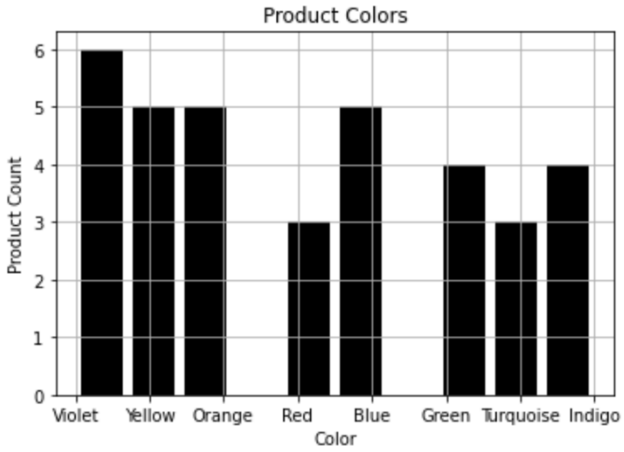

Figure 4: Histogram representing the popularity of colors amongst customers

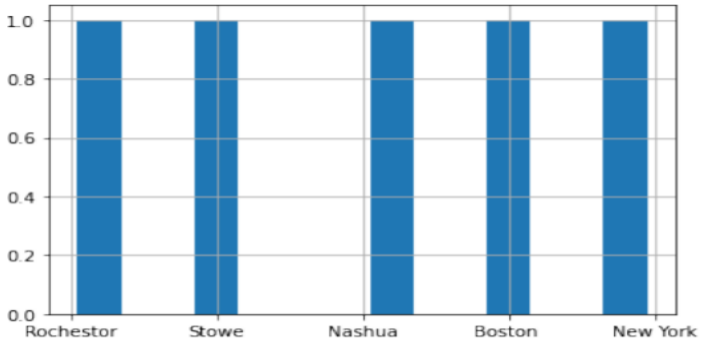

Figure 5: Bar chart representing cities from which maximum customers hail

VII. Summary and recommendation

The Merchandise Printing Database designed on MySQL is an industry ready relational database that can be implemented for the industry called Impreso Prints Ltd. It will result in great cost savings in the customer data input process and provides great analytics capabilities, a small part of which is shown in this report utilizing Python.

The further improvement on the database would be it will give an overview to the industry to expand their small-scale industry into the large-scale industry by analyzing the sales of the production of each city they have been supplying and can implement new marketing strategies by ads or by banners to make it more visual to people every-day.

The future recommendation of the database would be, instead of using MySQL for data input process or for data management purpose, it can be done much more easily and with fewer lines of queries on NoSQL. Also, the tables that are mentioned can be reduced around 4-5 by combining the similar contents from the dataset to reduce the storage and hardships while implementation.