

Use Case Report

Group no: 17

Name: Jignasuben R. Vekariya (002981797)

Nupur Bhavesh Shah (001077247)

Executive Summary:

The primary objective of this study was to design and implement a relational database that is industry ready for application for Export companies for use by processing data from Agents , Manufactures and Distributors including goods and payments of the same. There is huge amounts of data generation, and data can be reused by the properties using a relational database. This relational database reduces data input process time by 75% and result in better processing of goods over the world benefits across the industry. This information also allows to track the export data and keep track of the revenue that is generated by the same. The database was inputed taking the data fields of a sector and agents and manufacturer and the products they distribute through the distributors.

The EER and UML diagrams were designed, 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 and two relationships were implemented on NoSQL graph database to study the feasibility of this database in a NoSQL environment.

The created database is a great success, and by connecting it to Python and Tableau, the analytics capabilities are immense, some of which have been shown in the study.

Business Definition and Requirements:

Exporting company deals with export goods from home country to various country all over the world. As, Company has many agents and each agents have contracts with various manufacturer and distributor from other countries. All

manufacturer supplies goods to distributors from all over the world. These products are shipped in multiple containers.

Company wants to store the data of agents, manufacturers, and distributors.

They want to manage details of goods (like price, quantity) that are available for export from different manufacturers.

Also, they want to manage quantity, price, tax per quantity, and list of goods, that are purchased by distributors for import for their country.

Apart from that, they want to keep track of all container that are used for shipping.

Requirements from the company are as follow:

1. The company has its own many containers for shipment
2. All container owns by only one company.
2. Each agent is responsible for contracts with multiple manufactures and distributors. Agent should make sure the conditions and time-period of the contract and commission will be given for the same.
3. The manufacturer can supply one to many numbers and type of goods for export purpose.
4. The distributor can import one to many for his/her country from same or different manufacturer.
5. The distributor can make one to many payments which are under on his/her ID.
6. Each payment can be made by only that one distributor.

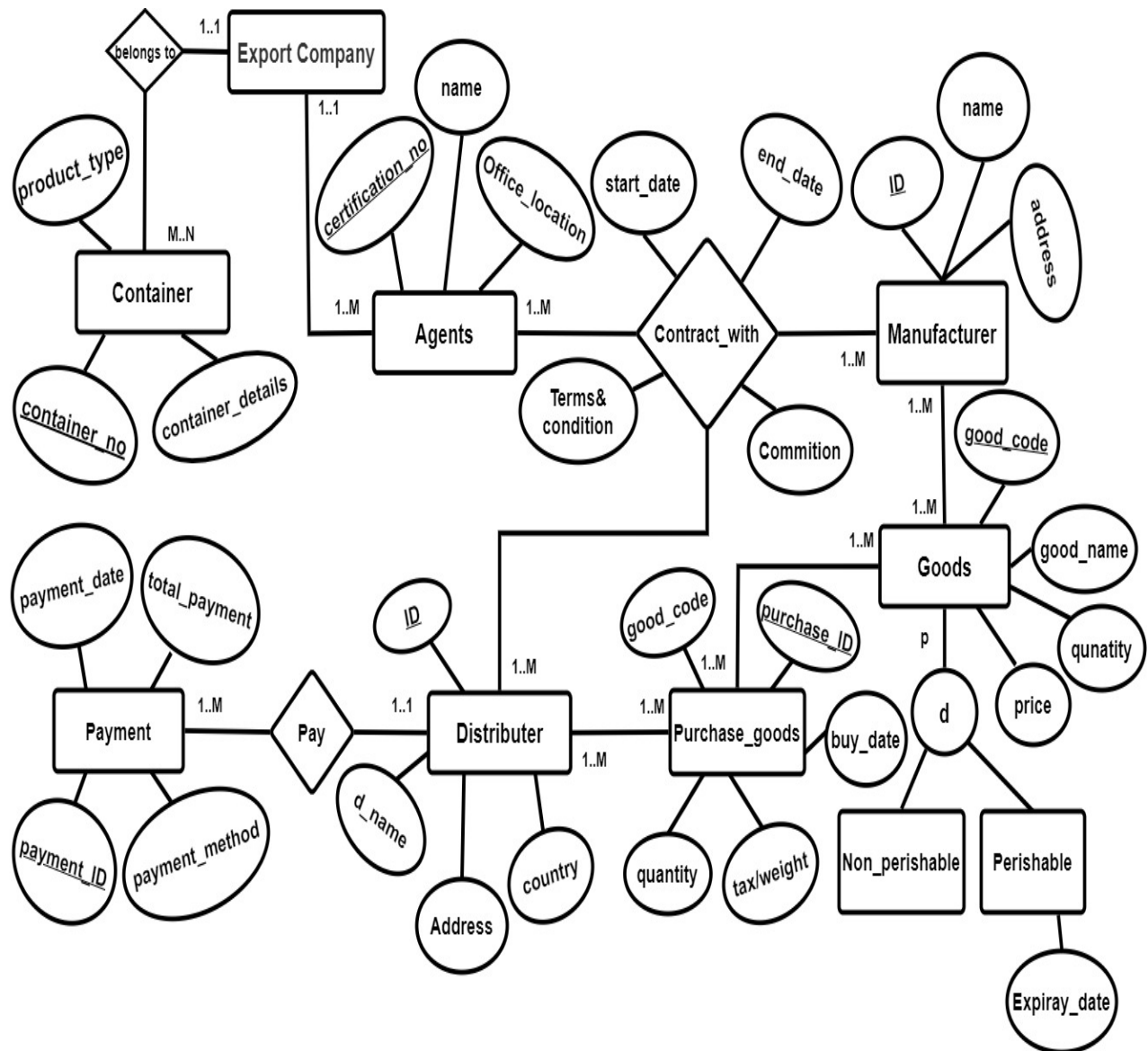
Conceptual Data Modeling

As company wants to store all data and make their management easier, we need to make database that fulfill their all requirements. For this, first we make tables for store data such as agents, manufacturers, distributors, and goods. Since, agent have contract with distributors and manufacturer, we make contract as a relation between them. For store data of goods that are purchased by the distributors and for make invoice for them, we make another table named purchase good. As multiple manufacturers supply their same good or same manufacturer supply different goods, we create another supply from table.

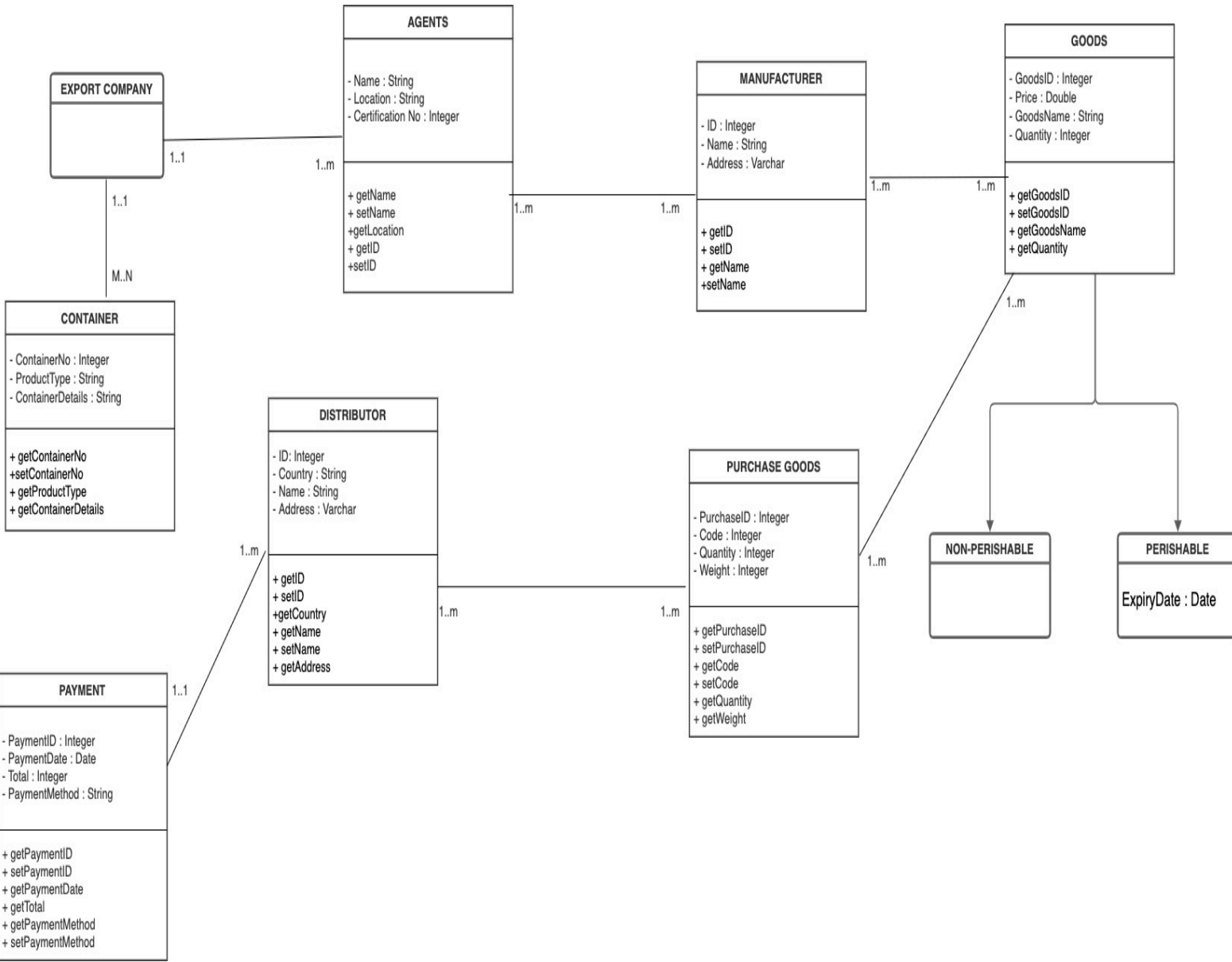
One or more agents can have contract with one or more distributors and manufacturers, we define one to many relationships between them.

And for relation between payment details and distributors, we map one to one relation from payment, as payment can be done by only one that particular distributor. As well as map one to many relation from distributor to payment, as may one distributor has to for pay one or more time importation.

ERR Diagram:



UML Design:



Mapping Conceptual to Relation Model:

Agents (certification_no, agent_name, office_location)

- Certification_no is primary key for agent table.
- Constraint: NOT NULL applied on certification_no

Manufacturer (ID, name, address)

- ID is primary key for manufacturer table.
- Constraint: NOT NULL applied on ID

Contract (certification_no, ID, start_date, end_date, commition, terms&condition)

- Certification_no is primary key. ID is foreign key form Manufacturer(ID) and Distributer(ID).
- Constraints: NOT NULL is applied on certification_no. ON UPDATE CASECADE is applied on ID.

Goods (good_code, good_name, quantity, price)

- Good_code is primary key
- Constraints: Not null applied on good_code.

Supply_from (good_code, manufacturer_ID)

- Good_code is foreign key from the Goods(good_code) table and manufacturer_ID is foreign key from the Manufacturer(ID) table.
- Constraints: ON UPDATE/ DELETE CASECADE is applied on both field

Perishable (perishable_code, expiray_date)

- perishable_code is super key from Goods(good_code) table

Unperishable (unperishable_code)

- unperishable_code is super key from Goods(good_code) table

Distributer (ID, d_name, address, country)

- ID is primary key for distributor table.
- Constraint: NOT NULL applied on ID

Purchase_goods (purchase_ID, good_code, buy_date, quantity, tax/weight, distributor_ID)

- Purchase_ID is primary key, good_code is foreign key from Goods table and distributor_ID is foreign key from the Distributer(ID) table.
- Constraint: NOT NULL is applied on all the keys and tax/weight. ON UPDATE CASCADE is applied on distributor_ID.

Payment (payment_ID, payment_date, total_payment, payment_method, distributor_ID)

- Payment_ID is primary key and distributor_ID is foreign key from the distributor table.
- Constraints: NOT NULL applied on both keys and total_payment. ON UPDATE CASCADE applied on distributor_ID.

Container (container_no, container_details, product_type)

- Container_no is primary key.
- Constraint: NOT NULL applied on container_no and product_type.

Implementation of Relation Model via MySQL and NoSQL



MySQL Implementation:

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

Query 1 : Retrieve the details of agents who has office location in Massachusetts

use tradedata

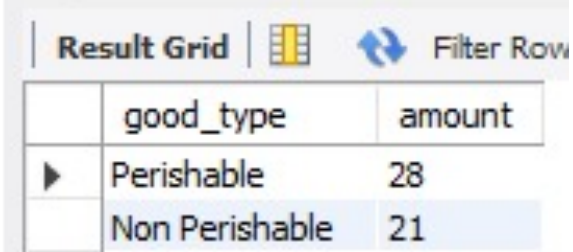
select * from agents

Result Grid   Filter Rows: <input type="text"/>			
	certification_no	agent_name	office_location
	8	Echo.	Boston,MA
	27	Jody	Waltham,MA
	28	Anne	Boston,MA
▶▶	NULL	NULL	NULL

where office_location like '%MA';

Query 2 : Calculate the Number of Perishable Goods and Nonperishable Goods

```
select good_type,  
count(*) as amount  
from goods  
group by good_type;
```



The screenshot shows a 'Result Grid' with two columns: 'good_type' and 'amount'. There are two rows: 'Perishable' with an amount of 28, and 'Non Perishable' with an amount of 21. The 'Non Perishable' row is highlighted in blue. Above the grid are icons for 'Filter Row' and a refresh symbol.

	good_type	amount
▶	Perishable	28
	Non Perishable	21

Query 3 : Retrieve the purchase details, count total amount without tax, agents and goods names from the database

```
select  
    p.purchase_id,  
    g.good_name,  
    p.quantity_in_tons,  
    g.price_in$_perTons*p.quantity_in_tons AS Total_Price,  
    p.buy_date,  
    d.dis_name,  
    a.agent_name  
from purchase_good p  
join agents a  
    ON p.agent_id=a.certification_no  
join goods g  
    ON p.good_code=g.good_code
```


join distributer d

ON p.dis_id=d.Id;

Result Grid		Filter Rows:		Export:	Wrap Cell Content:		
	purchase_id	good_name	quantity_in_tons	Total_Price	buy_date	dis_name	agent_name
▶	2	Crude oil	43	2058324	2021-02-03	Jayden	Ainsley
	4	chocolate	8	3629168	2022-12-15	John	Anne
	5	car tires	44	153340	2021-02-23	Jordan	Blair
	10	nondairy milk	11	9665436	2022-01-12	Denalia	Devon
	12	Pharmaceuticals	27	882252	2022-07-14	Jody	Ainsley
	14	Cottage cheese	45	20520	2022-05-19	Rajgopalam	Ainsley
	15	Beverage	28	907844	2022-01-01	Averill	Averill.
	18	Dairy Products	1	4345	2021-02-03	Drew	Dakota
	19	Poultry	17	740044	2022-02-02	Harley	Campbell.
	24	cooked leftovers	39	1697826	2022-02-12	Baldwini	Casey
	28	Syrup	29	132675	2021-02-03	Jaydena	Jackie
	29	Meat	9	310788	2021-02-03	Hayden	Baldwin.
	31	Fruits	20	1440	2022-08-02	Asa	Asa.
	32	Jelly	30	13950	2022-12-04	Natalie	Chase
	34	dried soups	22	753610	2021-02-03	Dunei	Dylan
	36	Cake Mixes	46	200376	2022-12-15	Asasa	Drew
	38	seafood	23	10436986	2021-02-03	Anne	Blair
	39	Rice	7	324471	2021-02-23	Jody	Cameron
	40	Wheat	40	2129800	2022-12-15	Jackie	Charlie
	41	Corn	16	56544	2021-02-23	Edena	Harley
	42	Baby Food & Ce...	21	1147545	2022-01-12	Drewi	Eden
	44	Feed	42	148428	2022-07-14	Devon	Drew
	46	Vegetables	32	1394016	2022-05-19	Jackie	Echo.
	47	Lunch meats	25	136600	2022-01-01	Dylan	Hayden
	48	Auto parts	34	154088	2022-02-10	Dylan	Devon

Query 4 : Retrieve the top 10 distributors and their agents names with purchased goods having a tax higher than 5%

Select d.dis_name, a.agent_name, g.good_name, p.tax_perTons AS Tax

from purchase_good p

join distributer d

ON p.dis_id=d.Id

join agents a

ON p.agent_id=a.certification_no



join goods g

ON p.good_code=g.good_code

Where p.tax_perTons >5

Order by p.tax_perTons desc

Limit 10;

Result Grid   Filter Rows: <input type="text"/> Exp				
	dis_name	agent_name	good_name	Tax
▶	Denalia	Devon	nondairy milk	9
	Drewi	Eden	Baby Food & Cereal	9
	Devon	Drew	Feed	9
	Jody	Ainsley	Pharmaceuticals	8
	Rajgopalam	Ainsley	Cottage cheese	8
	Asa	Asa.	Fruits	8
	Baldwini	Casey	cooked leftovers	7
	Anne	Blair	seafood	7
	Jody	Cameron	Rice	7
	Jackie	Echo.	Vegetables	7

Query 5 : Company want to give reminder those manufacturer whose contract with their agent will be end with in 15 months

So, they wants to view all the details of manufacturer and their agents as well as contract details

create view reminder AS

select a.agent_name,m.man_name,

c.start_date,

c.end_date,ceiling(datediff(c.end_date, curdate())/30) as
duration_in_months,




c.commission_in_per

from contract c

join agents a ON a.certification_no=c.certification_no

join manufacturer m ON m.Id=c.man_id

where datediff(c.end_date, curdate()) < 460 ;

Result Grid  Filter Rows: <input type="text"/> Export:  Wrap Cell Content: 						
	agent_name	man_name	start_date	end_date	duration_in_months	commission_in_per
▶	Blair	Volkswagen Group	2021-04-02	2023-02-09	15	4
	Jody	Samsung Electronics	2021-02-09	2023-02-01	14	1
	Jo	Ford Automotive	2022-06-17	2023-02-09	15	23
	Echo.	Industry Reactor	2021-04-02	2022-01-04	1	2
	Casey	Industry Forum	2021-11-12	2023-03-16	16	3
	Anne	Tengent Solution	2021-02-09	2022-10-20	11	3
	Charlie	ViaWave	2022-06-17	2023-03-16	16	1
	Baldwin.	Industry Central	2021-03-16	2022-01-11	1	2

Query 6 : Company wants to send notice to those agents who take more than 25% commission from their manufacturer

create view notice as

select * from agents

where certification_no In (

select distinct certification_no

from contract

where commission_in_per>25);

Result Grid			
Filter Rows:			
	certification_no	agent_name	office_location
▶	2	Averill.	Seattle, WA
	4	Campbell.	Austin, TX
	7	Dune.	Denver, CO
	19	Eden	Honolulu, HI
	29	Natalie	New York, NY

Query 7 : Management wants to check the stock of goods and status. make a procedure for this question

DELIMITER \$\$

```
CREATE PROCEDURE GetGoodStock(
)
```

BEGIN

create temporary table stocks

```
select quantity_in_tons as quantity,
IF(quantity_in_tons<50,'Out of stock','Stock Available')
AS Stock
```

from goods;

```
SELECT * FROM stocks;
```

END\$\$

DELIMITER ;

```
CALL GetGoodStock()
```

Result Grid		
Filter Rows:		
	quantity	Stock
▶	43	Out of stock
	43	Out of stock
	224	Stock Available
	42	Out of stock
	232	Stock Available
	867	Stock Available
	657	Stock Available
	76	Stock Available
	87	Stock Available
	98	Stock Available
	98	Stock Available
	576	Stock Available
	56	Stock Available
	45	Out of stock
	43	Out of stock
	23	Out of stock
	78	Stock Available
	96	Stock Available
	46	Out of stock
	86	Stock Available
	456	Stock Available
	56	Stock Available
	75	Stock Available
	45	Out of stock
	56	Stock Available
	75	Stock Available
	54	Stock Available
	567	Stock Available
	243	Stock Available
	53	Stock Available
	34	Out of stock
	87	Stock Available
	54	Stock Available
	856	Stock Available
	45	Out of stock
	87	Stock Available
	867	Stock Available
	87	Stock Available

Result 1 ×

NoSQL Implementation:

Database Access via MongoDB :

Tables are imported and queries are created in MongoDB. The following queries were done:

Query 1 : Details of the distributors who are from London, Uk



The screenshot shows the MongoDB Compass interface. At the top, a filter bar contains the query: `{ $and: [{ Country: "UK", Address: "London" }] }`. Below the filter bar, there are buttons for "ADD DATA", "VIEW", and icons for list, JSON, and grid views. The main area displays a table of distributor data with the following columns: `_id` (ObjectId), `ID String`, `Name String`, `Address String`, and `Country String`. The table contains three rows of data.

	<code>_id</code> ObjectId	ID String	Name String	Address String	Country String
1	61b64f2558789f3419deb7d1	"998"	"Asa "	"London"	"UK"
2	61b64f2558789f3419deb800	"345"	"Edena"	"London"	"UK"
3	61b64f2558789f3419deb805	"43"	"Jordana"	"London"	"UK"

Query 2 : Retrieve the goods details with a price greater than 50000

```
[{$match: {  
  Price: {  
    $gt: 50000  
  }  
}}, {$project: {  
  _id: 0,  
  'Goods name': 1,  
  'Quantity ': 1,  
  Price: 1  
}}]
```

Output after `$project` stage (Sample of 3 documents)

Goods name: "seafood"
Quantity: 657
Price: 453782

Goods name: "wheat"
Quantity: 856
Price: 53245

Goods name: "Breads"
Quantity: 98
Price: 64365

Query 3 : Retrieve the average quantity and price according to their goods division

```
[{
  $group: {
    _id: '$Goods division',
    avg_goods: {
      $avg: '$Quantity'
    },
    avg_price: {
      $avg: '$Price'
    }
  }
}]
```

Output after `$group` stage ⓘ (Sample of 2 documents)

```
_id: "Perishable"
avg_goods: 224.27272727272728
avg_price: 68407.09090909091
```

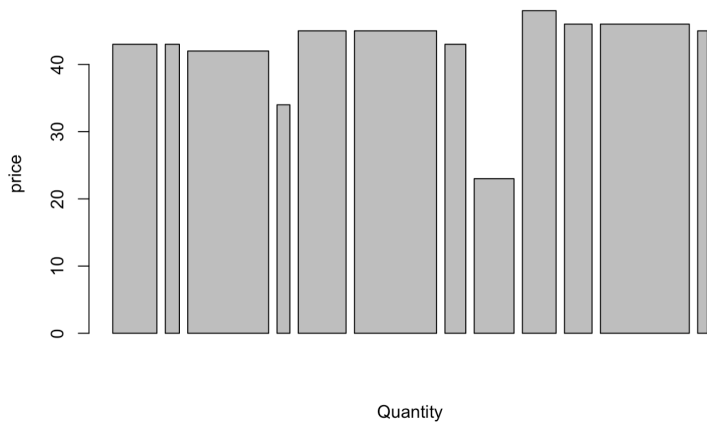
```
_id: "Non Perishable"
avg_goods: 258.44444444444446
avg_price: 19242.333333333332
```

Database Access via Python

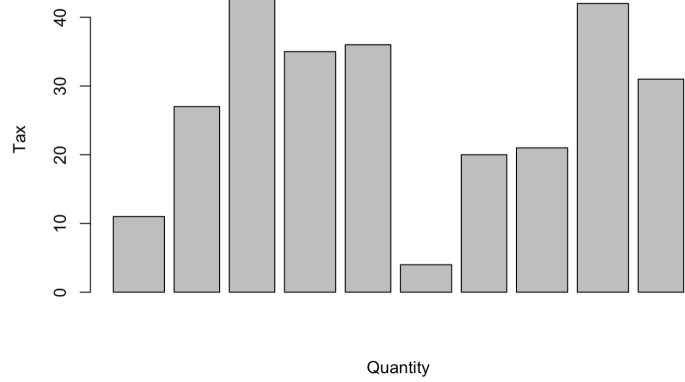
The database is accessed using R programming and visualization of analyzed data is shown below.

The connection of MySQL to R is done using ODBC, followed by connection to run and fetch all from query to plot the bar graph for the analytics

Graph 1 : Change of price with quantity

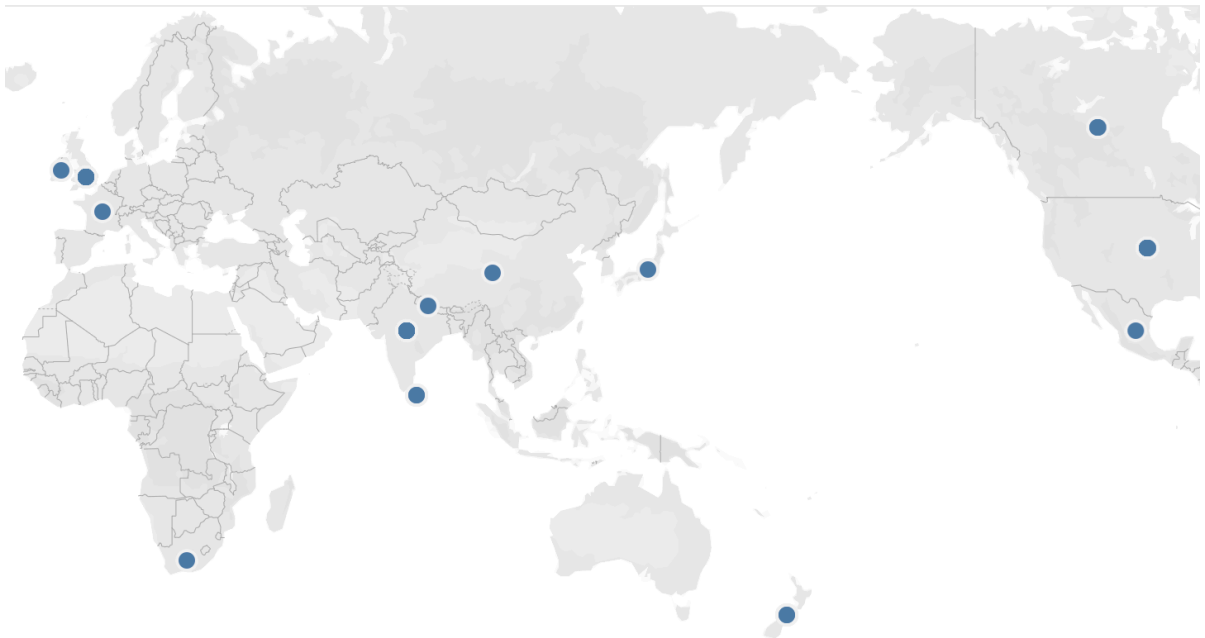


Graph 2 : Export amount of products tax and quantity



The database was also linked to Tableau for geo-mapping of location of distributors by running custom SQL Query on Database via Tableau

Location of distributors



Summary and Recommendation

The Export database designed on MySQL Workbench is an industry with huge data that can be defined with all the cities over the world and the database implementation is done for the data of the products, agents, manufacturers and containers. This implementation of the data will help with organizing the data that is processed to be analyzed and makes the structure easier to handle. A part of this database is also shown in the report using R programming to show a defined graph of the details involved.

In this dataset the improvement that we can add would be to define trends for the export process accordingly and place orders automatically when required by the agents and define the warehouse products for the manufacturers.