Retail Store Management

Course Title:

Course Leader:

Project By:

Aniket Nimbalkar

Index

1. Project Inception:

Retail shops help people to fulfil the daily needs of groceries. Tracking such a huge amount of data related to products, customers, employees, orders, payments, and suppliers is essential for retail shops. Managing all these records on paper is untrustworthy, susceptible and sloppy. The motto of our project is to create a rational database to access, manage, modify, and update data as per the requirement.

2. Database Analysis:

I have worked on a database to support a retail grocery shop to manage their services with ease of technology and minimize the errors which can be caused by paper work.

2.1. Business Situation:

Retail shops get numerous products (prod_id, name, price, sale_price, desc, mnf_date, exp_date, aisle_row_no) from various suppliers (supplier_id, name, phone, email, address). These products are displayed at different aisles (aisle_id, aisle_no, desc) and stock of all products being tracked in the inventories (inventory_id, prod_id, qty, in_stock, supplier_id) . A customer can pick up the products from these aisles and simply walk to the billing counter where Employee (emp_id, name, email, phone, position, Job_role, joining_date, leaving_date) of a retail shop can assist a customer (cust_id, name, email, phone, dob, address, postcode, age) to process the order (order_id, cust_id, emp_id, total_qty, price, tax, total_price, order_date, order_time). After billing all the order products (order_prod_id, price, qty, prod_id, order_id) customer can process a payment (payment_id, order_id, price, desc, status, method, updated_at)

2.2. Business Rules:

- Aisle may display many products.
- A customer can add multiple products in a single order.
- A single employee can assist various customers for the order billing process.
- There will be one payment instance for the order
- The stock quantity of all products will be tracked at the inventory i.e. Inventory will have collection of numerous products
- A Supplier will supply various products to a retail shop
- A customer can have multiple orders

2.3 List of Entity/Attributes:

1. Entity: CUSTOMERS:
    a. Attributes: cust_id, name, email, phone, dob, address, postcode, age
2. Entity: PRODUCTS:
    a. Attributes: prod_id, inventory_id, name, price, sale_price, desc, mnf_date, exp_date, aisle_row_no
3. Entity: EMPLOYEES:
    a. Attributes: emp_id, name, email, phone, position, job_role, joining_date, leaving_date
4. Entity: ORDERS:
    a. Attributes: order_id, cust_id, emp_id, total_qty, price, tax, total_price, order_date, order_time
5. Entity: ORDER_PRODUCTS:
    a. Attributes: order_prod_id, price, qty, prod_id, order_id
6. Entity: PAYMENTS:
    a. Attributes: payment_id, order_id, price, desc, status, method, updated_at
7. Entity: SUPPLIER:
    a. Attributes: supplier_id, name, phone, email, address
8. Entity: INVENTORIES:
    a. Attributes: inventory_id, qty, in_stock, supplier_id
9. Entity: AISLE_NO:
    a. Attributes: aisle_id, aisle_no, desc

2.4. Simple Relationships:

[CUSTOMERS] **1** HAS → **N** [ORDERS]

[PRODUCTS] **1** LISTS → **1** [INVENTORIES]

[EMPLOYEES] **1** ASSIST → **N** [ORDERS]

[ORDERS] **1** GENERATES → **1** [PAYMENTS]

[ORDERS] **1** HAS → **N** [ORDERS_PRODUCTS]

[SUPPLIERS] **M** SUPPLY → **N** [INVENTORIES]

[AISLE] **M** DISPLAYS → **N** [PRODUCTS]

2.5. Connectivity, Cardinalities and Participation:

A CUSTOMER can have minimum of __1__ ORDERS A

CUSTOMER can have maximum of __N__ ORDERS

Reverse:

ORDERS connect with minimum of ___1___ CUSTOMERS

ORDERS connect with maximum of __1__ CUSTOMERS

A SUPPLIER supply products minimum of __1__ INVENTORY

A SUPPLIER supply products maximum of __N__ INVENTORIES

Reverse:

INVENTORY will be filled by minimum of __1__ Supplier

INVENTORY will be filled by maximum of __N__ SUPPLIERIES

.

An EMPLOYEE managing minimum of __1__ ORDER An

EMPLOYEE managing maximum of __N__ ORDERS

Reverse:

An ORDER is managed by minimum of __1__ EMPLOYEE An

ORDER is managed by maximum of __1__ EMPLOYEE

An ORDER generating minimum of __1__ PAYMENT An

ORDER generating maximum of __1__ PAYMENT

Reverse:

A PAYMENT is generated by minimum of __1__ ORDER A

PAYMENT is generated by maximum of __1__ ORDER

A PRODUCT is listed at minimum of _1_ INVENTORY A

PRODUCT is listed at Maximum of _1_ INVENTORY

Reverse:

An INVENTORY list minimum of _1_ PRODUCT

An INVENTORY list Maximum of _N_PRODUCT

An ORDER has minimum of _1_ ORDER_PRODUCTS An

ORDER has Maximum of _M_ ORDER_PRODUCTS

Reverse:

An ORDER_PRODUCTS has minimum of _1_ ORDER An

ORDER_PRODUCTS has Maximum of _M_ ORDER

An AISLE displays minimum of _1_ PRODUCTS An

AISLE displays Maximum of _N_ PRODUCTS

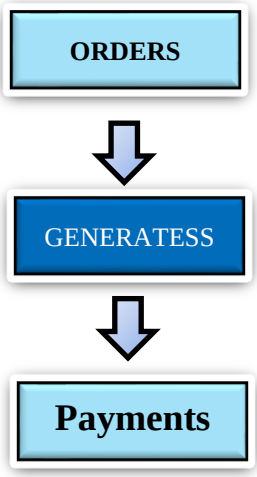
Reverse:

A PRODUCTS displayed at minimum of _1_ AISLE A

PRODUCTS displayed at Maximum of _N_ AISLE

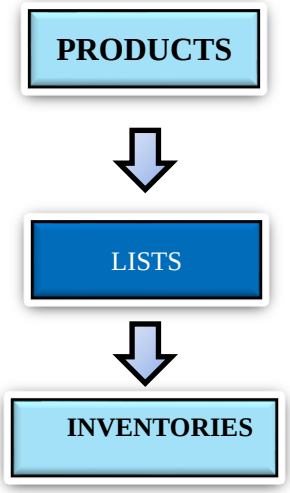
2.6 ERD Mapping:

2.6.1. Mapping 1: 1 Relationships:



order_id	cust_id	emp_id	qty	price	Tax	total_price	order_date	order_time
1	001	234	3	678.0	0.0	678.0	27-10-2021	10:10
2	008	345	4	549.0	0.0	549.0	28-10-2021	12:00
3	009	321	1	610.0	0.0	610.0	29.10-2021	16:00

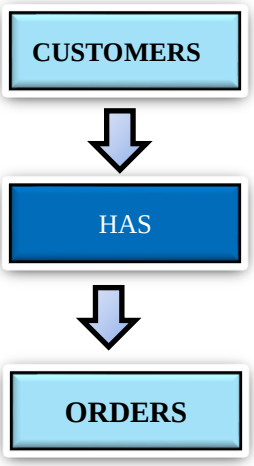
Payment_id	order_id	price	status	method	updated_at
987	1	678.0	Completed	COC	29-10-2021 16 :01
876	2	549.0	Completed	Card	28-12-2021 12 :01
678	3	610.0	Pending	Card	27-10-2021 10 :11



prod_id	name	price	Sale_price	Desc	Mnf_date	exp_date	Aisle_row_no
765	Rice	126.0	99.0	Kohinoor Rice	01/01/2022	01/01/2025	1
234	Cheese	310.0	248.0	Go Cheese	01/01/2021	15/01/2021	2
546	Milk	74.0	70.0	Gokul Milk	15/12/2021	20/12/2021	2

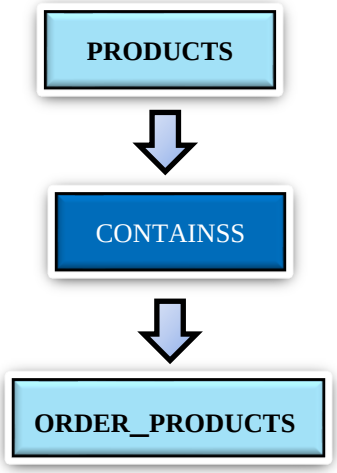
Inventory_id	Prod_id	Qty	In_stock	Supplier_id
788	765	230	TRUE	1
987	234	120	TRUE	2
546	546	421	TRUE	3

2.6.2. Mapping 1: N Relationships:



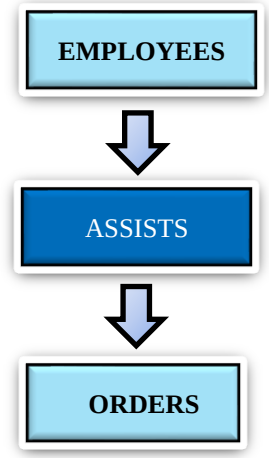
cust_id	name	Email	Phone	Dob	Address	Postcode	age
001	Rohan	rohan@mail.com	09925647412	10-04-1990	Sec-10 Panvel	410206	15
008	Darshan	darsh@mail.com	0776545678	18-06-1992	Sec-45 Panvel	410206	16
009	Akshay	aksh@mail.com	0876515678	03-03-1993	Sec-16 Panvel	410206	17

order_id	cust_id	emp_id	qty	price	Tax	total_price	order_date	order_time
1	001	234	3	678.0	0.0	678.0	27-10-2021	10:10
2	008	345	4	549.0	0.0	549.0	28-10-2021	12:00
3	009	321	1	610.0	0.0	610.0	29.10-2021	16:00



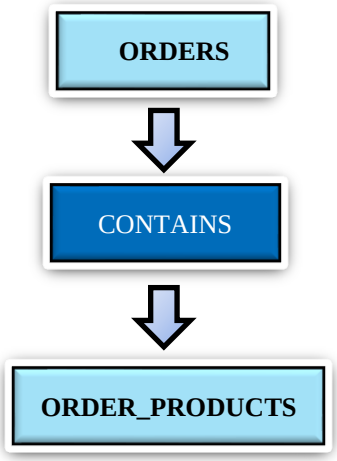
prod_id	name	price	Sale_price	Desc	Mnf_date	exp_date	Aisle_row_no
765	Rice	126.0	99.0	Kohinoor Rice	01/01/2022	01/01/2025	1
234	Cheese	310.0	248.0	Go Cheese	01/01/2021	15/01/2021	2
546	Milk	74.0	70.0	Gokul Milk	15/12/2021	20/12/2021	2

order_prod_id	Price	Qty	Prod_id	Order_id
456	126.0	3	765	001
879	310.0	4	234	002
986	74.0	1	546	003



Emp_id	name	Email	Phone	Position	Job_role	Joining_date	Leaving_date
234	Sam	sam@test.com	0987654567	Staff	Retail Assistant	27/10/2020	27/11/2020
345	Sanj	sanj@test.com	0875567787	Staff	Cashier	28/10/2020	28/11/2020
321	Rock	rock@test.com	0345678766	Manager	Staff Management	29/10/2020	29/11/2020

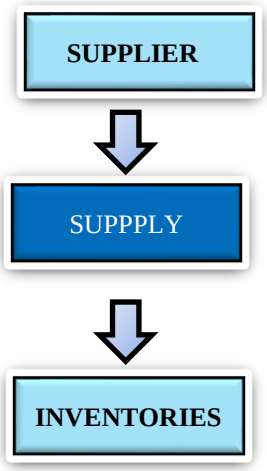
order_id	cust_id	emp_id	qty	price	Tax	total_price	order_date	order_time
1	001	234	3	678.0	0.0	678.0	27-10-2021	10:10
2	008	345	4	549.0	0.0	549.0	28-10-2021	12:00
3	009	321	1	610.0	0.0	610.0	29.10-2021	16:00



order_id	cust_id	emp_id	qty	price	Tax	total_price	order_date	order_time
1	001	234	3	678.0	0.0	678.0	27-10-2021	10:10
2	008	345	4	549.0	0.0	549.0	28-10-2021	12:00
3	009	321	1	610.0	0.0	610.0	29.10-2021	16:00

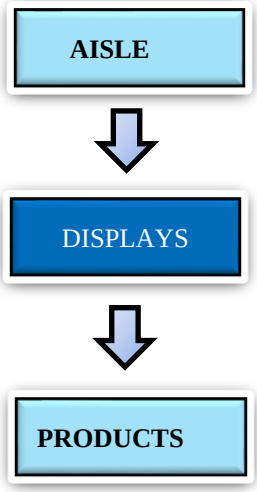
order_prod_id	Price	Qty	Prod_id	Order_id
456	126.0	3	765	001
879	310.0	4	234	002
986	74.0	1	546	003

2.6.3. Mapping M: N Relationships:



supplier_id	name	Phone	Email	address
1	SUPP1	0987687678	Supp1@test.com	Supp1 add
2	SUPP2	0987658778	Supp2@test.com	Supp2 add
3	SUPP3	0453434534	Supp3@test.com	Supp3 add

Inventory_id	Prod_id	Qty	In_stock	Supplier_id
788	765	230	TRUE	1
987	234	120	TRUE	2
546	546	421	TRUE	3



aisle_id	aisle_no	Desc
098	1	Rice Section
076	2	Cheese Section
061	3	Oats Section

prod_id	name	price	Sale_price	Desc	Mnf_date	exp_date	Aisle_row_no
765	Rice	126.0	99.0	Kohinoor Rice	01/01/2022	01/01/2025	1
234	Cheese	310.0	248.0	Go Cheese	01/01/2021	15/01/2021	2
546	Milk	74.0	70.0	Gokul Milk	15/12/2021	20/12/2021	2

3. Database Design:

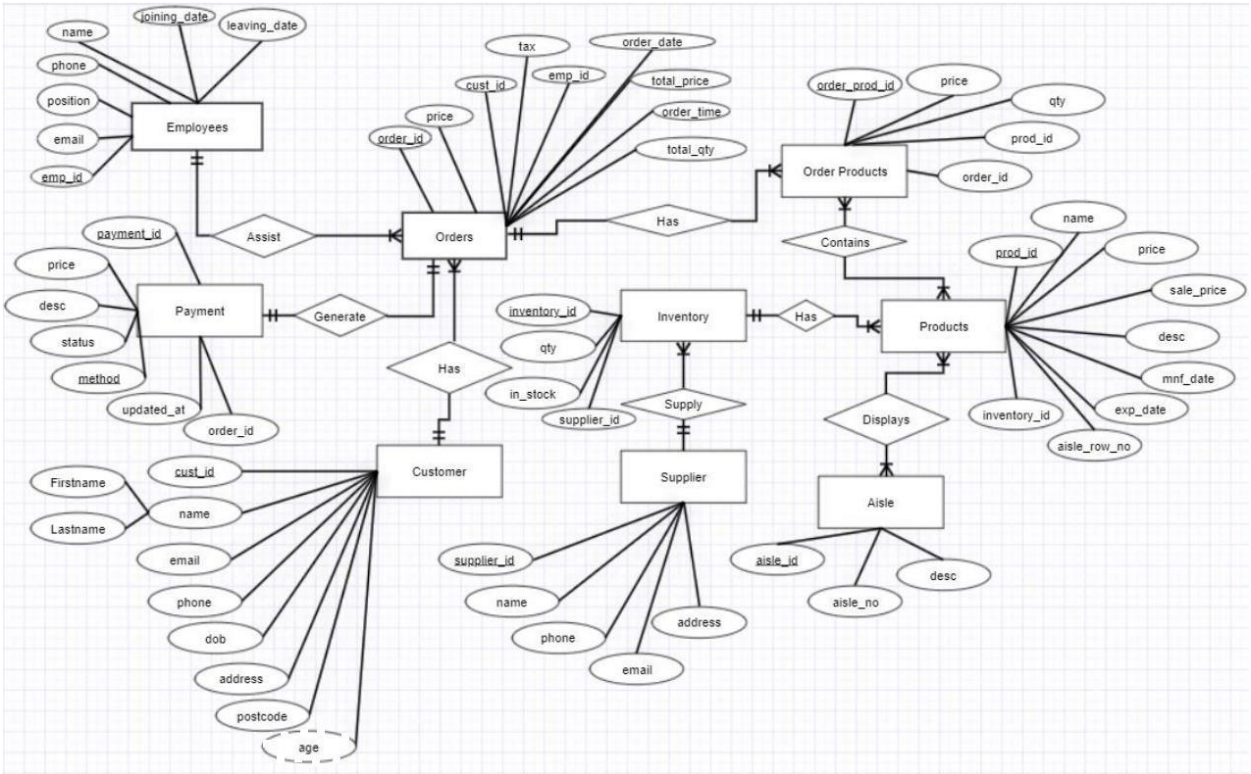


Fig 1: ER Diagram

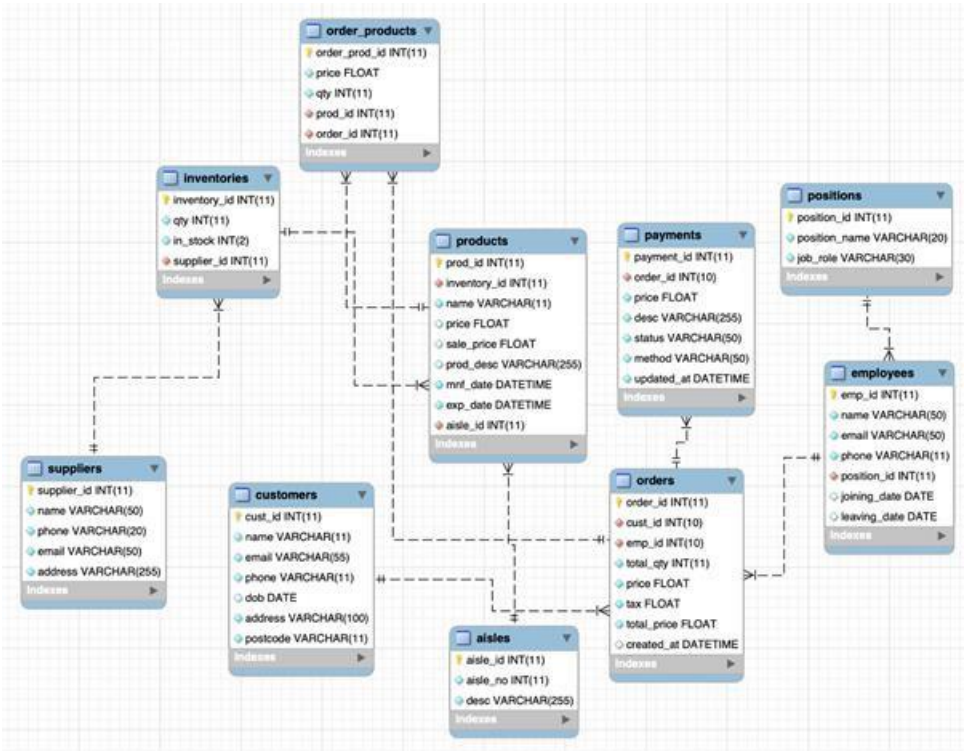


Fig 2: Class Diagram after normalization

4. Normalization

Whenever any table really not properly normalized as well as has duplicate information, this would not only consume more disk space, but this will be tough to organize as well as update into the DB without experiencing data loss.

4.1Customers

cust_id	name	Email	Phone	Dob	Address	Postcode	age
001	Rohan	rohan@mail.com	09925647412	10-04-1990	Sec-10 Panvel	410206	15
008	Darshan	darsh@mail.com	0776545678	18-06-1992	Sec-45 Panvel	410206	16
009	Akshay	aksh@mail.com	0876515678	03-03-1993	Sec-16 Panvel	410206	17

4.2 Products

prod_id	name	price	Sale_price	Desc	Mnf_date	exp_date	Aisle_row_no
765	Rice	126.0	99.0	Kohinoor Rice	01/01/2022	01/01/2025	1
234	Cheese	310.0	248.0	Go Cheese	01/01/2021	15/01/2021	2
546	Milk	74.0	70.0	Gokul Milk	15/12/2021	20/12/2021	2

4.3 Inventories:

Inventory_id	Prod_id	Qty	In_stock	Supplier_id
788	765	230	TRUE	1
987	234	120	TRUE	2
546	546	421	TRUE	3

4.4 Order_Products

order_prod_id	Price	Qty	Prod_id	Order_id
456	126.0	3	765	001
879	310.0	4	234	002
986	74.0	1	546	003

4.5 Suppliers:

supplier_id	name	Phone	Email	address
1	SUPP1	0987687678	Supp1@test.com	Supp1 add
2	SUPP2	0987658778	Supp2@test.com	Supp2 add
3	SUPP3	0453434534	Supp3@test.com	Supp3 add

4.6 Payments:

Payment_id	order_id	price	status	method	updated_at
987	1	678.0	Completed	COC	29-10-2021 16 :01
876	2	549.0	Completed	Card	28-12-2021 12 :01
678	3	610.0	Pending	Card	27-10-2021 10 :11

4.7 Aisle_No

aisle_id	aisle_no	Desc
098	1	Rice Section
076	2	Cheese Section
061	3	Oats Section

4.8 Orders:

order_id	cust_id	emp_id	qty	price	Tax	total_price	order_date	order_time
1	001	234	3	678.0	0.0	678.0	27-10-2021	10:10
2	008	345	4	549.0	0.0	549.0	28-10-2021	12:00
3	009	321	1	610.0	0.0	610.0	29.10-2021	16:00

The above tables have been in the 3NF because each column has just the same type of one value, every column has a distinct name, as well as the sequence in which the data is stored is not important. In addition, the main key (PK) has been determined, and all non-key properties are totally entirely reliant mostly on PK. Therefore, there seem to be no transitive or partial dependencies.

4.9 Employees

Emp_id	name	Email	Phone	Position	Job_role	Joining_date	Leaving_date
234	Sam	sam@test.com	0987654567	Staff	Retail Assistant	27/10/2020	27/11/2020
345	Sanj	sanj@test.com	0875567787	Staff	Cashier	28/10/2020	28/11/2020
321	Rock	rock@test.com	0345678766	Manager	Staff Management	29/10/2020	29/11/2020

The employee table has a partial dependency, to solve this position column and job role has to take out in the new separate table.

Emp_id	name	Email	Phone	Position_id	Joining_date	Leaving_date
234	Sam	sam@test.com	0987654567	1	27/10/2020	27/11/2020
345	Sanj	sanj@test.com	0875567787	2	28/10/2020	28/11/2020
321	Rock	rock@test.com	0345678766	3	29/10/2020	29/11/2020

Position_id	Position_Name	Job_role
1	staff	Retail_Assistant
2	staff	Cashier
3	manager	Staff_Management

5. Database Implementation:

5.1. Table Creation:

Customers Table:

```
CREATE TABLE Customers (cust_id INT  NOT NULL Identity(1,1), name VARCHAR(11) NOT NULL ,
email VARCHAR(11) NOT NULL ,
phone VARCHAR(12) NOT NULL , dob DATE NULL DEFAULT NULL , address VARCHAR(20) NOT NULL ,
postcode VARCHAR(15) NOT NULL, PRIMARY KEY (cust_id));
ALTER TABLE customers ADD UNIQUE(email);
```

Products Table:

```
CREATE TABLE Products ( prod_id INT  NOT NULL Identity(1,1) , inventory_id INT NOT NULL , name VARCHAR(11) NOT NULL ,
price FLOAT(24) NULL , sale_price FLOAT(24) NULL , prod_desc VARCHAR(255) NULL DEFAULT NULL ,
mnf_date DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP , exp_date DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP ,
aisle_id INT  NOT NULL , PRIMARY KEY (prod_id)) ;

ALTER TABLE products ADD CONSTRAINT aisles_has_products FOREIGN KEY (aisle_id)
REFERENCES aisles(aisle_id) ON DELETE NO ACTION ON UPDATE NO ACTION;

ALTER TABLE products CHANGE inventory_id inventory_id INT  NOT NULL;

ALTER TABLE products ADD CONSTRAINT inventory_products FOREIGN KEY
(inventory_id) REFERENCES inventories(inventory_id) ON DELETE NO ACTION ON UPDATE NO ACTION;
```

Employees Table:

```
CREATE TABLE Employees ( emp_id INT  NOT NULL Identity(1,1) , name VARCHAR(50) NOT NULL , email VARCHAR(50) NOT NULL ,
phone VARCHAR(11) NOT NULL , position_id INT(25) NOT NULL , `joining_date` DATE NULL ,
leaving_date DATE NULL , PRIMARY KEY (emp_id));

ALTER TABLE employees ADD CONSTRAINT employee_position_id FOREIGN KEY
(position_id) REFERENCES positions(position_id) ON DELETE NO ACTION ON UPDATE NO ACTION;
```

Positions Table:

```
CREATE TABLE Positions(Position_id int(11) NOT NULL Identity(1,1),
Position_Name varchar(20) NOT NULL,Job_role varchar(30) NOT NULL,
PRIMARY KEY (Position_id));
```

Orders Table:

```
CREATE TABLE Orders ( order_id INT  NOT NULL Identity(1,1), cust_id INT  NOT NULL
emp_id INT NOT NULL , total_qty INT NOT NULL , price FLOAT NOT NULL ,
tax FLOAT NOT NULL , total_price FLOAT NOT NULL , order_datetime DATETIME NOT NULL
DEFAULT CURRENT_TIMESTAMP,
PRIMARY KEY (order_id)) ;

ALTER TABLE orders ADD CONSTRAINT order_customers FOREIGN KEY (cust_id)
REFERENCES customers(cust_id) ON DELETE NO ACTION ON UPDATE NO ACTION;

ALTER TABLE orders ADD CONSTRAINT order_employees FOREIGN KEY (emp_id)
REFERENCES employees(emp_id) ON DELETE NO ACTION ON UPDATE NO ACTION;
```

Order Products Table:

```
CREATE TABLE Order_Products ( order_prod_id INT NOT NULL Identity(1,1),
price FLOAT NOT NULL , qty INT NOT NULL , prod_id INT UNSIGNED NOT NULL ,
order_id INT  NOT NULL , PRIMARY KEY (order_prod_id));

ALTER TABLE order_products ADD CONSTRAINT op_product FOREIGN KEY (prod_id)
REFERENCES products(prod_id) ON DELETE NO ACTION ON UPDATE NO ACTION;

ALTER  TABLE  order_products  ADD  CONSTRAINT  op_order  FOREIGN  KEY  (order_id)
REFERENCES
orders(order_id) ON DELETE NO ACTION ON UPDATE NO ACTION;
```

Payments Table:

```
CREATE TABLE Payments ( payment_id INT NOT NULL Identity(1,1), order_id INT  NOT
NULL , price FLOAT NOT NULL ,
descr VARCHAR(255) NOT NULL ,
status VARCHAR(50) NOT NULL , method VARCHAR(50) NOT NULL , updated_at DATETIME
NOT NULL DEFAULT CURRENT_TIMESTAMP , PRIMARY KEY (payment_id));


ALTER  TABLE  payments  ADD  CONSTRAINT  order_payments  FOREIGN  KEY  (order_id)
REFERENCES orders(order_id)
ON DELETE NO ACTION ON UPDATE NO ACTION;
```

Suppliers Table:

```
CREATE TABLE Suppliers ( supplier_id  INT NOT NULL Identity(1,1), name VARCHAR(50)
NOT NULL , phone INT(11)NOT NULL ,
email VARCHAR(50) NOT NULL , address VARCHAR(255) NOT NULL ,
PRIMARY KEY (supplier_id)) ;

ALTER TABLE suppliers ADD CONSTRAINT supplier_inventories FOREIGN KEY
(supplier_id) REFERENCES inventories(supplier_id) ON DELETE NO ACTION ON UPDATE
NO ACTION;
```

Inventories Table:

```
CREATE TABLE Inventories ( inventory_id INT NOT NULL AUTO_INCREMENT ,
qty INT NOT NULL , in_stock INT(2) NOT NULL COMMENT '0 for out of stock, 1 for in stock',
supplier_id INT  NOT NULL , PRIMARY KEY (inventory_id));

ALTER TABLE inventories CHANGE inventory_id inventory_id INT(11) UNSIGNED NOT NULL AUTO_INCREMENT;

ALTER TABLE inventories ADD CONSTRAINT supplier_inventories FOREIGN KEY (supplier_id)
REFERENCES suppliers(supplier_id) ON DELETE NO ACTION ON UPDATE NO ACTION;
```

Aisles Table:

```
CREATE TABLE Aisles ( aisle_id INT NOT NULL Identity(1,1), aisle_no INT(11)  NOT NULL ,
descr VARCHAR(255) NOT NULL , PRIMARY KEY (aisle_id)) ;

ALTER TABLE aisles CHANGE aisle_id aisle_id INT  NOT NULL Identity(1,1);
```

5.2. Data entry in tables:

Customers:

```
INSERT INTO customers

(cust_id,name,email,phone,dob,address,postcode)

VALUES

(NULL,'Rohan','rohan@mail.com',09925647412,'1997-04-25','sec-
10,Panvel','410206'),

(NULL,'Nikhil','nikhil@mail.com',08564712360,'1991-07-09','sec-
11,Panvel','410206'),

(NULL,'Shankar','shankar@mail.com',06945782014,'1996-06-04','sec-
09,Panvel','410206'),

(NULL,'Sandesh','sandesh@mail.com',08542369714,'1999-08-05','sec-
10,Panvel','410206'),

(NULL,'Divesh','divesh@mail.com',07145856932,'1999-01-02','sec-
10,Panvel','410206'),

(NULL,'Sumit','sumit@mail.com',0123654740,'1993-02-07','sec-
08,Panvel','410206'),

(NULL,'Prashant','prashant@mail.com',0878545678,'1994-06-03','sec-
11,Panvel','410206'),

(NULL,'Darsh','darshan@mail.com',0776545678,'1998-12-02','sec-
12,Panvel','410206'),

(NULL,'Akshay','akshay@mail.com',0876515678,'1990-07-07','sec-
16,Panvel','410206'),

(NULL,'Paresh','paresh@mail.com',7456788237,'1990-04-17','sec-01,Panvel',
'410206');
```

Results | Messages

	cust_id	name	email	phone	dob	address	postcode
1	1	Rohan	rohan@mail.com	9925647412	1997-04-25	sec-10,Panvel	410206
2	2	Nikhil	nikhil@mail.com	8564712360	1991-07-09	sec-11,Panvel	410206
3	3	Shankar	shankar@mail.com	6945782014	1996-06-04	sec-09,Panvel	410206
4	4	Sandesh	sandesh@mail.com	8542369714	1999-08-05	sec-10,Panvel	410206
5	5	Divesh	divesh@mail.com	7145856932	1999-01-02	sec-10,Panvel	410206
6	6	Sumit	sumit@mail.com	123654740	1993-02-07	sec-08,Panvel	410206
7	7	Prashant	prashant@mail.com	878545678	1994-06-03	sec-11,Panvel	410206
8	8	Darsh	darshan@mail.com	776545678	1998-12-02	sec-12,Panvel	410206
9	9	Akshay	akshay@mail.com	876515678	1990-07-07	sec-16,Panvel	410206
10	10	Paresh	paresh@mail.com	7456788237	1990-04-17	sec-01,Panvel	410206

Products:

```
Products:
INSERT INTO products
(prod_id,inventory_id,name,price,sale_price,prod_desc,mnf_date,exp_date,aisle_id
)
VALUES
(NULL, 1, 'Rice',126.0,99.0,'Kohinoor Rice','2022-01-01','2022-01-21', 1),
(NULL, 2, 'Cheese',310.0,248.0,'GO Cheese','2022-01-01','2022-01-21', 2),
(NULL, 3, 'Biscuits',99.0,68.0,'Parle G Biscuits','2022-01-01','2022-01-21', 3),
(NULL, 4, 'Milk',74.0,70.0,'Gokul Milk','2022-01-02','2022-01-11', 2),
(NULL,5,'Chocolate',65.0,61.0,'Dairy    Milk    chocolate','2022-01-02','2022-01-
12',3),
(NULL, 6, 'Butter',136.0,112.0,'Amul butter','2022-01-03','2022-01-13', 3),
(NULL, 7, 'Tea',367.0,299.0,'10 tea bags','2022-01-03','2022-01-14', 3),
(NULL, 8, 'Coffee',112.0,89.0,'Nescafe coffee','2022-01-04','2022-01-15', 3),
(NULL, 9, 'Oats',349.0,320.0,'Kellogs Oats','2022-01-04','2022-01-16', 3),
(NULL, 10, 'Curd',49.0,45.0,'Amul Curd','2022-01-04','2022-01-11',2)
```

100 %

Results | Messages

	prod_id	inventory_id	name	price	sale_price	prod_desc	mnf_date	exp_date	aisle_id
1	124	6	Butter	136	112	Amul butter	2022-01-03 00:00:00.000	2022-01-13 00:00:00.000	3
2	147	3	Biscuits	99	68	Parle G Biscuits	2022-01-01 00:00:00.000	2022-01-21 00:00:00.000	3
3	234	2	Cheese	310	248	GO Cheese	2022-01-01 00:00:00.000	2022-01-21 00:00:00.000	2
4	258	9	Oats	349	320	Kellogs Oats	2022-01-04 00:00:00.000	2022-01-16 00:00:00.000	3
5	367	7	Tea	367	299	10 tea bags	2022-01-03 00:00:00.000	2022-01-14 00:00:00.000	3
6	369	10	Curd	49	45	Amul Curd	2022-01-04 00:00:00.000	2022-01-11 00:00:00.000	2
7	421	8	Coffee	112	89	Nescafe coffee	2022-01-04 00:00:00.000	2022-01-15 00:00:00.000	3
8	546	4	Milk	74	70	Gokul Milk	2022-01-02 00:00:00.000	2022-01-11 00:00:00.000	2
9	765	1	Rice	126	99	Kohinoor Rice	2022-01-01 00:00:00.000	2022-01-21 00:00:00.000	1
10	789	5	Chocolate	65	61	Dairy Milk chocolate	2022-01-02 00:00:00.000	2022-01-12 00:00:00.000	3

Employees:

```
Employees:
INSERT INTO employees
(emp_id,name,email,phone,position_id,joining_date,leaving_date)
VALUES
(NULL, 'Employee 1', 'employee1@test.com', 0123456789, 1, '2010-01-01', '2021-
01-01'),
(NULL, 'Employee 2', 'employee2@test.com', 0123236789, 2, '2014-01-01', '2019-
04-12'),
(NULL, 'Employee 3', 'employee3@test.com', 0127656789, 3, '2012-01-01', '2016-
06-09');
```

100 %

Results | Messages

	emp_id	name	email	phone	position_id	joining_date	leaving_date
1	234	Employee 1	employee1@test.com	123456789	1	2010-01-01	2021-01-01
2	321	Employee 3	employee3@test.com	127656789	3	2012-01-01	2016-06-09
3	345	Employee 2	employee2@test.com	123236789	2	2014-01-01	2019-04-12

Orders:

```
INSERT INTO orders
 (order_id,cust_id,emp_id,total_qty,price,tax,
 total_price,created_at)
 VALUES
 (NULL,  7,  1,  12,  678.0,  0.00,  678.0,  '2021-01-01
 12:56:59'),
 (NULL,  8,  2,  9,  549.0,  0.00,  549.0,  '2019-04-12
 10:16:10'),
 (NULL,  9,  2,  5,  225.0,  0.00,  225.0,  '2016-06-09
 16:45:20'),
 (NULL,  8,  1,  4,1196.0  ,  0.00,  1196.0,  '2016-06-09
 16:45:20'),
 (NULL,  11,  1,  10,  610.0,  0.00,  610.0,  '2016-06-09
 16:45:20'),
 (NULL,  8,  2,  11,  1239.0,  0.00,  1239.0,  '2016-06-09
 16:45:20'),
 (NULL,  8,  2,  7,673.0  ,  0.00,  673.0,  '2016-06-09
 16:45:20'),
 (NULL,  8,  1,  10,  1201.0,  0.00,  1201.0,  '2016-06-09
 16:45:20'),
 (NULL,  15,  2,  6,429.0,  0.00,  429.0,  '2016-06-09
 16:45:20'),
 (NULL,16,1,5,1121.0,            0.00,1121.0,'2016-06-09
 16:45:20'),
 (NULL,  8,  1,  10,  650.0,  0.00,  650.0,  '2016-06-09
 16:45:20');
```

100 %

Results | Messages

	order_id	cust_id	emp_id	total_qty	price	tax	total_price	order_datetime
1	1	1	234	12	678	0	678	2021-01-01 12:56:59.000
2	2	8	345	9	549	0	549	2019-04-12 10:16:10.000
3	3	9	234	10	610	0	610	2016-06-09 16:45:20.000
4	4	2	345	5	225	0	225	2016-06-09 16:45:20.000
5	5	8	234	4	1196	0	1196	2016-06-09 16:45:20.000
6	6	4	345	11	1239	0	1239	2016-06-09 16:45:20.000
7	7	6	345	7	673	0	673	2016-06-09 16:45:20.000
8	8	5	234	10	1201	0	1201	2016-06-09 16:45:20.000
9	9	3	345	6	429	0	429	2016-06-09 16:45:20.000
10	10	2	234	5	1121	0	1121	2016-06-09 16:45:20.000
11	11	10	234	10	650	0	650	2016-06-09 16:45:20.000

Order Products:

```
INSERT INTO order_products
(order_prod_id,price,qty,prod_id,order_id)
VALUES
(NULL, 367.0, 10, 7, 1),
(NULL, 112.0, 8, 8, 2),
(NULL, 349.0, 9, 9, 3),
(NULL,126.0 , 4, 1, 1),
(NULL, 49.0, 2, 10, 1),
(NULL, 349.0, 3, 9, 1),
(NULL, 112.0, 8, 8, 2),
(NULL, 349.0, 9, 9, 2),
(NULL, 112.0, 8, 8, 3),
(NULL, 349.0, 9, 9, 3);
```

100 %

Results | Messages

	order_prod_id	price	qty	prod_id	order_id
1	147	112	2	421	1
2	159	65	3	789	1
3	160	49	9	369	3
4	357	112	8	421	3
5	456	126	3	765	1
6	743	136	8	124	2
7	761	74	9	546	3
8	852	99	9	147	2
9	879	310	4	234	1
10	986	74	1	546	2

Payment:

```
INSERT INTO payments
(payment_id,order_id,price,descr,status,method,updated_at)
VALUES
(NULL, 1, 678.0, 'pending from bank', 'pending', 'Debit/Credit Card', '2021-01-01 12:56:59'),
(NULL, 2, 549.0, 'completed from bank', 'completed', 'Debit/Credit Card', '2021-01-01 12:56:59'),
(NULL, 3, 225.0, 'mismatched cash', 'pending', 'Cash', '2021-01-01 12:56:59'),
(NULL, 12, 1196.0, 'mismatched cash', 'pending', 'Cash', '2021-01-01 12:56:59'),
(NULL, 13, 610.0, 'completed from bank', 'completed', 'Debit/Credit Card', '2021-01-01 12:56:59'),
(NULL, 14, 1239.0, 'completed from bank', 'completed', 'Debit/Credit Card', '2021-01-01 12:56:59'),
(NULL, 15, 673.0, 'pending from bank', 'pending', 'Debit/Credit Card', '2021-01-01 12:56:59'),
(NULL, 16, 1201.0, 'mismatched cash', 'pending', 'Cash', '2021-01-01 12:56:59'),
(NULL, 17, 429.0, 'pending from bank', 'pending', 'Debit/Credit Card', '2021-01-01 12:56:59'),
(NULL, 18, 1121.0, 'mismatched cash', 'pending', 'Cash', '2021-01-01 12:56:59'),
(NULL, 19, 650.0, 'pending from bank', 'pending', 'Debit/Credit Card', '2021-01-01 12:56:59');
```

100 %

Results | Messages

	payment_id	order_id	price	descr	status	method	updated_at
1	142	7	673	pending from bank	pending	Debit/Credit Card	2021-01-01 12:56:59.000
2	159	4	225	mismatched cash	pending	Cash	2021-01-01 12:56:59.000
3	168	8	1201	mismatched cash	pending	Cash	2021-01-01 12:56:59.000
4	324	5	1196	mismatched cash	pending	Cash	2021-01-01 12:56:59.000
5	357	9	429	pending from bank	pending	Debit/Credit Card	2021-01-01 12:56:59.000
6	569	10	1121	mismatched cash	pending	Cash	2021-01-01 12:56:59.000
7	678	3	610	completed from bank	completed	Debit/Credit Card	2021-01-01 12:56:59.000
8	852	11	650	pending from bank	pending	Debit/Credit Card	2021-01-01 12:56:59.000
9	876	2	549	completed from bank	completed	Debit/Credit Card	2021-01-01 12:56:59.000
10	879	6	1239	completed from bank	completed	Debit/Credit Card	2021-01-01 12:56:59.000
11	987	1	678	pending from bank	pending	Debit/Credit Card	2021-01-01 12:56:59.000

Supplier:

```
INSERT INTO suppliers
(supplier_id,name,phone,email,address)
VALUES
(1, 'Supplier 1','0123567876', 'supplier1@test.com', 'Navi Mumbai'),
(2, 'Supplier 2','9765678988', 'supplier2@test.com', 'Navi Mumbai'),
(3, 'Supplier 3','4675678769', 'supplier3@test.com', 'Panvel'),
(4, 'Supplier 4','4675678769', 'supplier4@test.com', 'New Panvel'),
(5, 'Supplier 5','4675678769', 'supplier5@test.com', 'Mumbai'),
(6, 'Supplier 6','4675678769', 'supplier6@test.com', 'Panvel'),
(7, 'Supplier 7','4675678769', 'supplier7@test.com', 'Panvel'),
(8, 'Supplier 8','4675678769', 'supplier8@test.com', 'Navi Mumbai'),
(9, 'Supplier 9','4675678769', 'supplier9@test.com', 'Panvel'),
(10, 'Supplier 10','4675678769', 'supplier10@test.com', 'Mumbai');
```

100 %

Results | Messages

	supplier_id	name	phone	email	address
1	1	Supplier 1	0123567876	supplier1@test.com	Navi Mumbai
2	2	Supplier 2	9765678988	supplier2@test.com	Navi Mumbai
3	3	Supplier 3	4675678769	supplier3@test.com	Panvel
4	4	Supplier 4	4675678769	supplier4@test.com	New Panvel
5	5	Supplier 5	4675678769	supplier5@test.com	Mumbai
6	6	Supplier 6	4675678769	supplier6@test.com	Panvel
7	7	Supplier 7	4675678769	supplier7@test.com	Panvel
8	8	Supplier 8	4675678769	supplier8@test.com	Navi Mumbai
9	9	Supplier 9	4675678769	supplier9@test.com	Panvel
10	10	Supplier 10	4675678769	supplier10@test.com	Mumbai

Inventory:

```
INSERT          INTO
inventories
(inventory_id,qty,in_stock,supplier_id)
     VALUES
     (1, 200, 1, 1),
     (2, 160, 1, 2),
     (3, 0, 0, 3),
     (4,1000, 1, 4),
     (5, 800,1, 5),
     (6, 900,1, 6),
     (7, 650,1, 7),
     (8, 430,1, 8),
     (9, 959,1, 9),
     (10,642,1,10);
```

100 %

Results | Messages

	inventory_id	qty	in_stock	supplier_id
1	359	800	1	5
2	457	1000	1	4
3	546	421	0	3
4	614	900	1	6
5	710	642	1	10
6	736	650	1	7
7	788	230	1	1
8	852	430	1	8
9	913	959	1	9
10	987	120	1	2

Aisles:

```
INSERT     INTO     aisles
(aisle_id,aisle_no,descr)
VALUES
(1, 21,'Aisle 1'),
(2, 22,'Aisle 2'),
(3, 23,'Aisle 3'),
(4, 24,'Aisle 4'),
(5, 25,'Aisle 5'),
(6, 26,'Aisle 6'),
(7, 27,'Aisle 7'),
(8, 28,'Aisle 8'),
(9, 29,'Aisle 9'),
(10, 30,'Aisle 10');
```

100 %

Results | Messages

	aisle_id	aisle_no	descr
1	14	4	Aisle 4
2	32	5	Aisle 5
3	41	9	Aisle 9
4	47	7	Aisle 7
5	61	3	Aisle 3
6	67	10	Aisle 10
7	76	2	Aisle 2
8	85	8	Aisle 8
9	96	6	Aisle 6
10	98	1	Aisle 1

Positions:

```
INSERT INTO positions (position_id, position_name, job_role) VALUES
(1, 'staff', 'Retail_Assistance'),
(2, 'staff', 'Cashier'),
(3, 'Manager', 'Staff_Management');
```

100 %

Results | Messages

	Position_id	Position_Name	Job_role
1	1	staff	Retail_Assistance
2	2	staff	Cashier
3	3	Manager	Staff_Management

5.3. SQL Queries Testing:

1. In SQL queries testing, it is showing three queries to retrieve data from database based on user's need.

Below is the usage of join query, to get all the customers information with order and order product details. Here I am using join which will help us to connect each table with another based on the same customer id and order id with using order by name.

select * from customers Inner JOIN orders on customers.cust_id=orders.cust_id Inner JOIN order_products on orders.order_id=order_products.order_id ORDER BY name DESC

Results | Messages

	cust_id	name	email	phone	dob	address	postcode	order_id	cust_id	emp_id	total_qty	price	tax	total_price	order_datetime	order_prod_id	price	qty	prod_id	order_id
1	1	Rohan	rohan@mail.com	9925647412	1997-04-25	sec-10,Panvel	410206	1	1	234	12	678	0	678	2021-01-01 12:56:59.000	147	112	2	421	1
2	1	Rohan	rohan@mail.com	9925647412	1997-04-25	sec-10,Panvel	410206	1	1	234	12	678	0	678	2021-01-01 12:56:59.000	159	65	3	789	1
3	1	Rohan	rohan@mail.com	9925647412	1997-04-25	sec-10,Panvel	410206	1	1	234	12	678	0	678	2021-01-01 12:56:59.000	456	126	3	765	1
4	1	Rohan	rohan@mail.com	9925647412	1997-04-25	sec-10,Panvel	410206	1	1	234	12	678	0	678	2021-01-01 12:56:59.000	879	310	4	234	1
5	8	Darsh	darshan@mail.com	776545678	1998-12-02	sec-12,Panvel	410206	2	8	345	9	549	0	549	2019-04-12 10:16:10.000	986	74	1	546	2
6	8	Darsh	darshan@mail.com	776545678	1998-12-02	sec-12,Panvel	410206	2	8	345	9	549	0	549	2019-04-12 10:16:10.000	743	136	8	124	2
7	8	Darsh	darshan@mail.com	776545678	1998-12-02	sec-12,Panvel	410206	2	8	345	9	549	0	549	2019-04-12 10:16:10.000	852	99	9	147	2
8	9	Akshay	akshay@mail.com	876515678	1990-07-07	sec-16,Panvel	410206	3	9	234	10	610	0	610	2016-06-09 16:45:20.000	761	74	9	546	3
9	9	Akshay	akshay@mail.com	876515678	1990-07-07	sec-16,Panvel	410206	3	9	234	10	610	0	610	2016-06-09 16:45:20.000	160	49	9	369	3
10	9	Akshay	akshay@mail.com	876515678	1990-07-07	sec-16,Panvel	410206	3	9	234	10	610	0	610	2016-06-09 16:45:20.000	357	112	8	421	3

2. Second one is for the usage of distinct. In this example I am using distinct to get all the types of employees position.

select DISTINCT `position` from employees;

Results | Messages

	position_name
1	Manager
2	staff

3. Third one is to use one aggregate function. Here I am using sum function to check the total amount of sale from orders table using sum function in ms sql query.

SELECT SUM(`total_price`) FROM orders;

100 %

Results | Messages

	(No column name)
1	8571

4. For the query optimization I are using index in our table. Indexing helps for the faster data retrieval. In large database indexing is very helpful to get the fast result.

```
CREATE UNIQUE INDEX random_index_name ON customers (name);
```

5. In this query I will test the foreign key constraint and primary key constraint added in employees table, We tested this constraints with inserting a position_id which is not present in positions table and inserting a duplicate entry but it shows below error

```
Employees:
INSERT INTO employees
(emp_id,name,email,phone,position_id,joining_date,leaving_date)
VALUES
(234, 'Employee 4', 'employee1@test.com', 0123456789, 1, '2010-01-01', '2021-01-01')
```

```
Messages
Msg 2627, Level 14, State 1, Line 306
Violation of PRIMARY KEY constraint 'PK__Employee__1299A8614600322C'. Cannot insert duplicate key in object 'dbo.Employees'. The duplicate key value is (234).
The statement has been terminated.

Completion time: 2024-11-10T16:40:36.1231715+05:30
```

6. In this query I will try to fetch customer details who ordered highest amount of order using aggregate function and a subquery

```
SELECT c.cust_id,c.name,c.email,c.phone FROM customers c inner join orders o on
c.cust_id= o.cust_id WHERE o.total_price = (SELECT MAX(total_price) FROM orders);
```

100 %

Results | Messages

	cust_id	name	email	phone
1	4	Sandesh	sandesh@mail.com	8542369714

7. In this query I will try to fetch customers details in descending order of ordered total price.

SELECT c.cust_id, c.name, c.email, c.phone, o.order_id, o.total_price from customers c inner join orders o on c.cust_id = o.cust_id order by o.total_price desc;

Results | Messages

	cust_id	name	email	phone	order_id	total_price
1	4	Sandesh	sandesh@mail.com	8542369714	6	1239
2	5	Divesh	divesh@mail.com	7145856932	8	1201
3	8	Darsh	darshan@mail.com	776545678	5	1196
4	2	Nikhil	nikhil@mail.com	8564712360	10	1121
5	1	Rohan	rohan@mail.com	9925647412	1	678
6	6	Sumit	sumit@mail.com	123654740	7	673
7	10	Paresh	paresh@mail.com	7456788237	11	650
8	9	Akshay	akshay@mail.com	876515678	3	610
9	8	Darsh	darshan@mail.com	776545678	2	549
10	3	Shankar	shankar@mail.com	6945782014	9	429
11	2	Nikhil	nikhil@mail.com	8564712360	4	225

8. In this query I will try to average of orders total price.

SELECT AVG(total_price) FROM order

Results | Messages

	(No column name)
1	779.181818181818

6.Conclusion

This database includes everything that a retail shop may require, and integrating them would help their organization function even more efficiently. since all necessary information is entered into the system that may be retrieved at any moment. Our database maintains data integrity as well as data security.

This database verifies that all information has been provided, but if any information has been provided incorrectly, this database should display an alert and refuse to save it in the database. We acquired cognitive information of this database prior to establishing it. Moreover, normalization was required to ensure that now the tables are now in the 3rd NF. Ultimately. I stayed on track and feel I created a really well database.