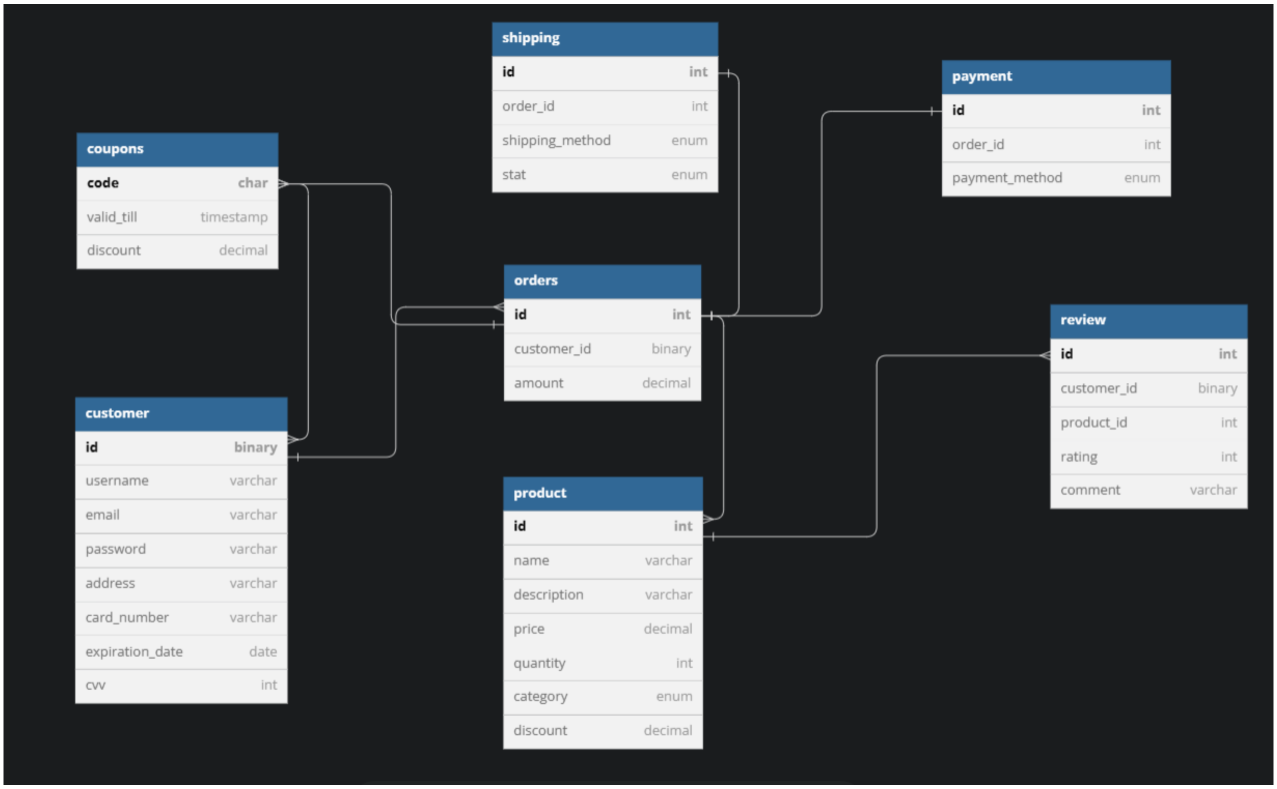
Database Systems Project Report

Project Number 17
e-Commerce

Name	Id
Sparsh Goenka	2021A7PS2413P
Hardik Gupta	2021A7PS2421P

ER Diagram

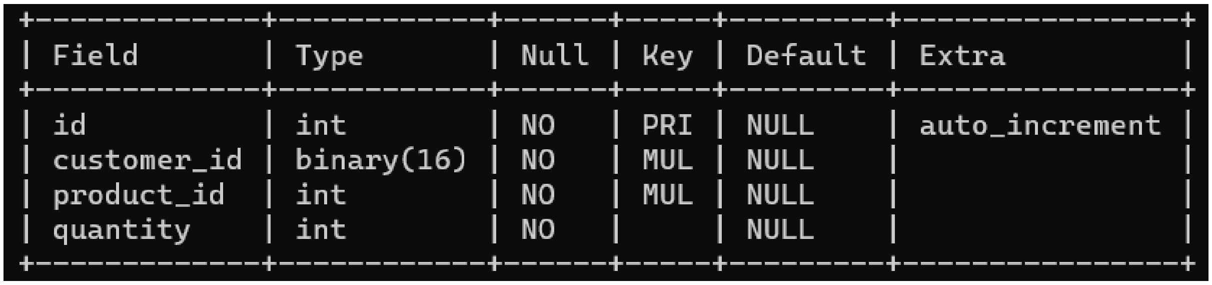


The relationship between the order and product entity types needs to be expanded. This can be done by creating another relationship to store order details, which can be used to store the product and its quantities. A temporary table is also needed to store the details of the products which need to be ordered. This can be achieved via the addition of a cart table to temporarily store data before its addition to the order table.

It is visible in the customer entity that the address and payment details (Card number, Expiry date and CVV) attributes are nested relations. This means that the given entity relationship diagram does not satisfy the first normal form. We can solve this by creating new relationships for the address and the payment details of a customer.Doing this solves the problem, and the third normal form is also found to be satisfied.

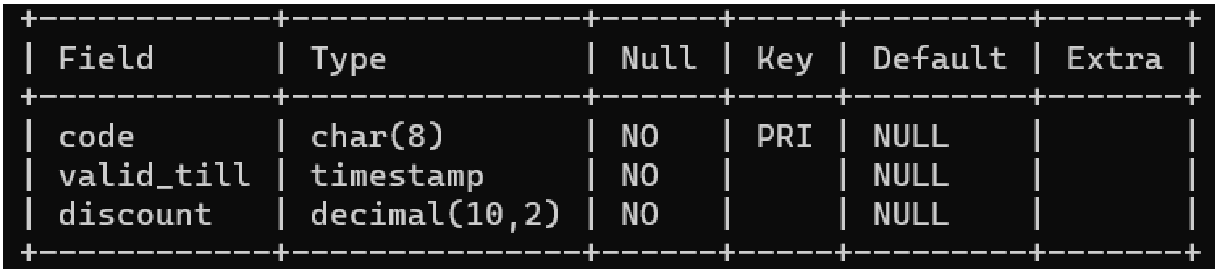
Relational Schema

The following tables show the schema of the relations used to represent the normalized form of the relational diagram:



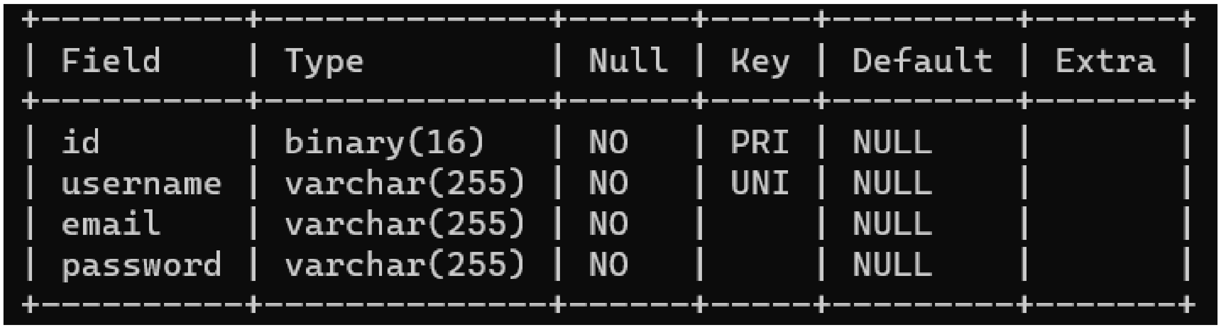
```
+-------------+------------+------+-----+---------+----------------+
| Field       | Type       | Null | Key | Default | Extra          |
+-------------+------------+------+-----+---------+----------------+
| id          | int        | NO   | PRI | NULL    | auto_increment |
| customer_id | binary(16) | NO   | MUL | NULL    |                |
| product_id  | int        | NO   | MUL | NULL    |                |
| quantity    | int        | NO   |     | NULL    |                |
+-------------+------------+------+-----+---------+----------------+
```

Cart Table Schema



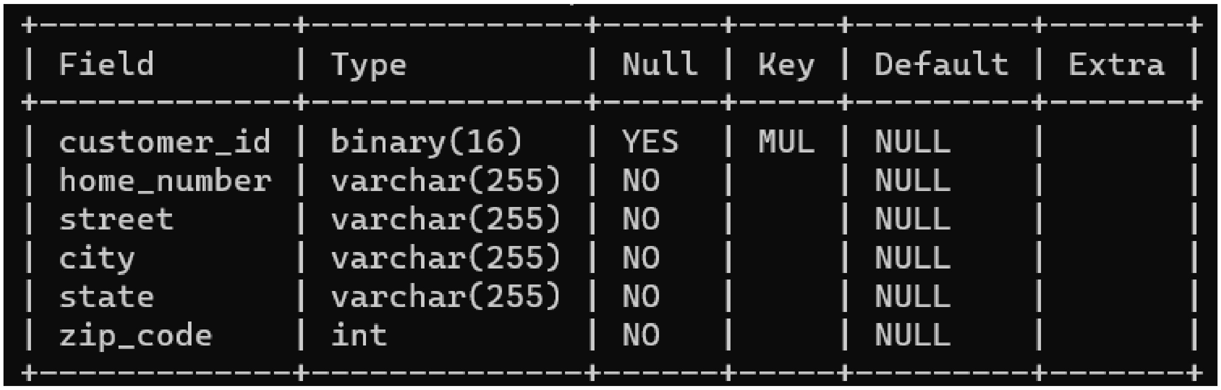
```
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| code       | char(8)       | NO   | PRI | NULL    |       |
| valid_till | timestamp     | NO   |     | NULL    |       |
| discount   | decimal(10,2) | NO   |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
```

Coupons Table Schema



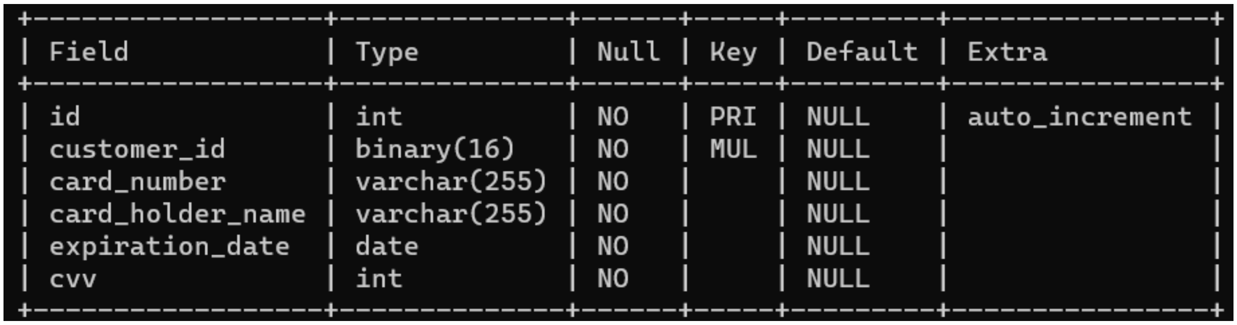
```
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| id       | binary(16)   | NO   | PRI | NULL    |       |
| username | varchar(255) | NO   | UNI | NULL    |       |
| email    | varchar(255) | NO   |     | NULL    |       |
| password | varchar(255) | NO   |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
```

Customer Table Schema



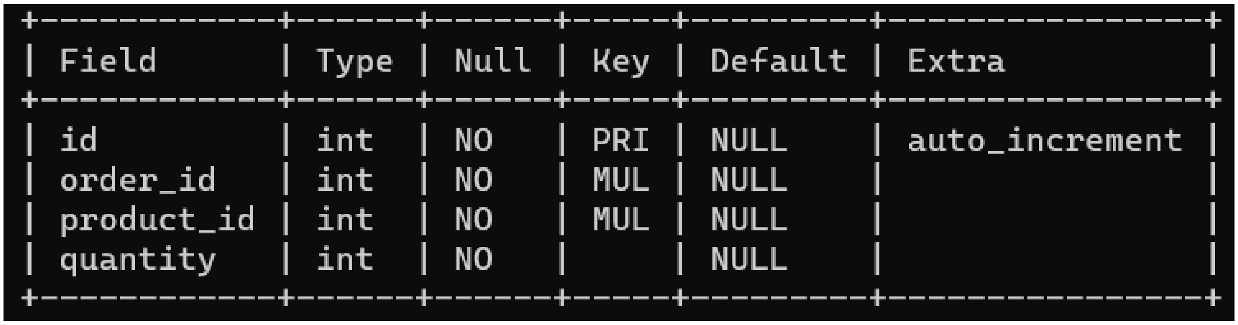
```
+-------------+--------------+------+-----+---------+-------+
| Field       | Type         | Null | Key | Default | Extra |
+-------------+--------------+------+-----+---------+-------+
| customer_id | binary(16)   | YES  | MUL | NULL    |       |
| home_number | varchar(255) | NO   |     | NULL    |       |
| street      | varchar(255) | NO   |     | NULL    |       |
| city        | varchar(255) | NO   |     | NULL    |       |
| state       | varchar(255) | NO   |     | NULL    |       |
| zip_code    | int          | NO   |     | NULL    |       |
+-------------+--------------+------+-----+---------+-------+
```

Customer address Table Schema



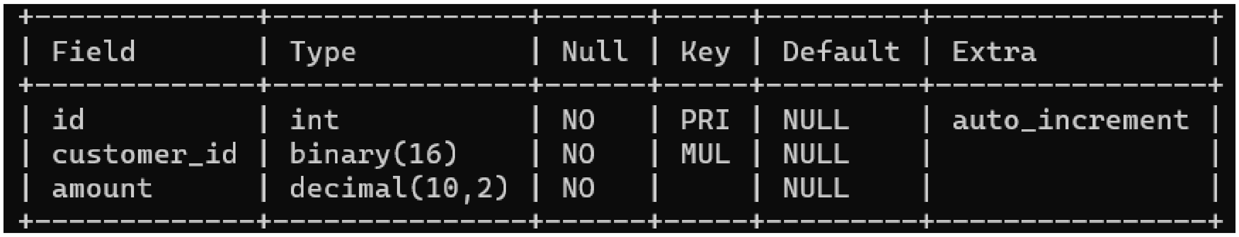
```
+------------------+--------------+------+-----+---------+----------------+
| Field            | Type         | Null | Key | Default | Extra          |
+------------------+--------------+------+-----+---------+----------------+
| id               | int          | NO   | PRI | NULL    | auto_increment |
| customer_id      | binary(16)   | NO   | MUL | NULL    |                |
| card_number      | varchar(255) | NO   |     | NULL    |                |
| card_holder_name | varchar(255) | NO   |     | NULL    |                |
| expiration_date  | date         | NO   |     | NULL    |                |
| cvv              | int          | NO   |     | NULL    |                |
+------------------+--------------+------+-----+---------+----------------+
```

Customer payment information Table Schema



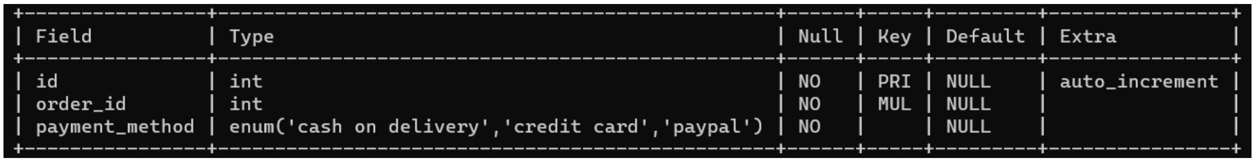
```
+------------+------+------+-----+---------+----------------+
| Field      | Type | Null | Key | Default | Extra          |
+------------+------+------+-----+---------+----------------+
| id         | int  | NO   | PRI | NULL    | auto_increment |
| order_id   | int  | NO   | MUL | NULL    |                |
| product_id | int  | NO   | MUL | NULL    |                |
| quantity   | int  | NO   |     | NULL    |                |
+------------+------+------+-----+---------+----------------+
```

Order details Table Schema



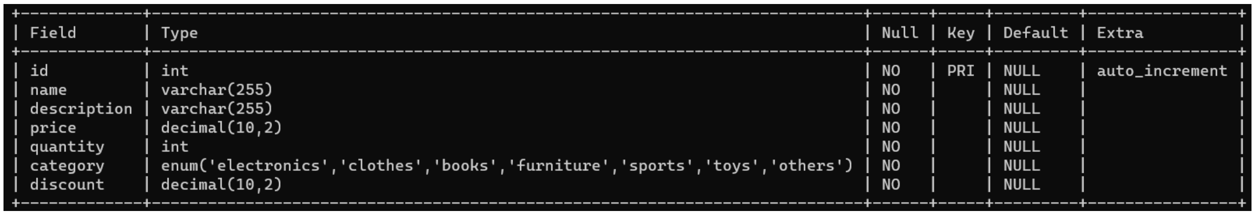
```
+-------------+---------------+------+-----+---------+----------------+
| Field       | Type          | Null | Key | Default | Extra          |
+-------------+---------------+------+-----+---------+----------------+
| id          | int           | NO   | PRI | NULL    | auto_increment |
| customer_id | binary(16)    | NO   | MUL | NULL    |                |
| amount      | decimal(10,2) | NO   |     | NULL    |                |
+-------------+---------------+------+-----+---------+----------------+
```

Orders Table Schema



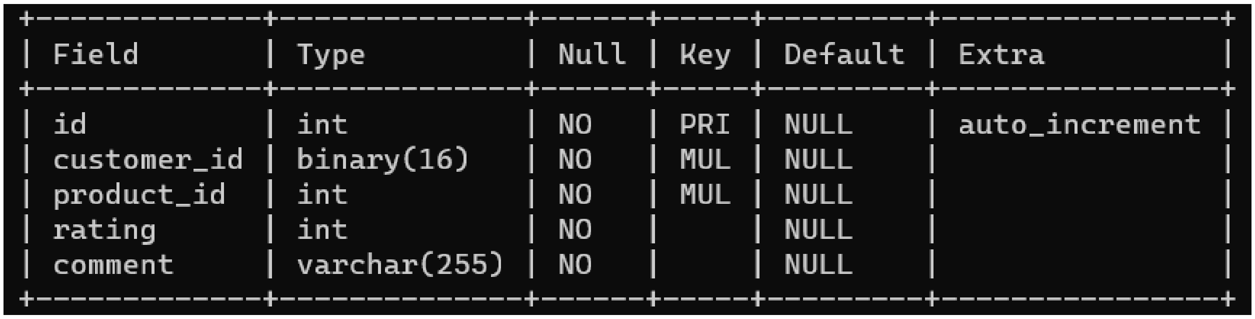
```
+----------------+-------------------------------------------------+------+-----+---------+----------------+
| Field          | Type                                            | Null | Key | Default | Extra          |
+----------------+-------------------------------------------------+------+-----+---------+----------------+
| id             | int                                             | NO   | PRI | NULL    | auto_increment |
| order_id       | int                                             | NO   | MUL | NULL    |                |
| payment_method | enum('cash on delivery','credit card','paypal') | NO   |     | NULL    |                |
+----------------+-------------------------------------------------+------+-----+---------+----------------+
```

Payment Table Schema



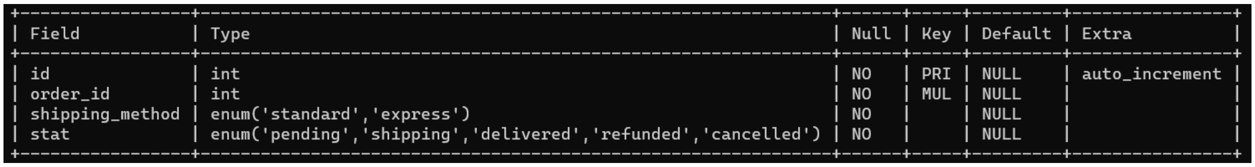
```
+-------------+----------------------------------------------------------------------------+------+-----+---------+----------------+
| Field       | Type                                                                       | Null | Key | Default | Extra          |
+-------------+----------------------------------------------------------------------------+------+-----+---------+----------------+
| id          | int                                                                        | NO   | PRI | NULL    | auto_increment |
| name        | varchar(255)                                                               | NO   |     | NULL    |                |
| description | varchar(255)                                                               | NO   |     | NULL    |                |
| price       | decimal(10,2)                                                              | NO   |     | NULL    |                |
| quantity    | int                                                                        | NO   |     | NULL    |                |
| category    | enum('electronics','clothes','books','furniture','sports','toys','others') | NO   |     | NULL    |                |
| discount    | decimal(10,2)                                                              | NO   |     | NULL    |                |
+-------------+----------------------------------------------------------------------------+------+-----+---------+----------------+
```

Product Table Schema



```
+-------------+--------------+------+-----+---------+----------------+
| Field       | Type         | Null | Key | Default | Extra          |
+-------------+--------------+------+-----+---------+----------------+
| id          | int          | NO   | PRI | NULL    | auto_increment |
| customer_id | binary(16)   | NO   | MUL | NULL    |                |
| product_id  | int          | NO   | MUL | NULL    |                |
| rating      | int          | NO   |     | NULL    |                |
| comment     | varchar(255) | NO   |     | NULL    |                |
+-------------+--------------+------+-----+---------+----------------+
```

Review Table Schema



```
+-----------------+---------------------------------------------------------------+------+-----+---------+----------------+
| Field           | Type                                                          | Null | Key | Default | Extra          |
+-----------------+---------------------------------------------------------------+------+-----+---------+----------------+
| id              | int                                                           | NO   | PRI | NULL    | auto_increment |
| order_id        | int                                                           | NO   | MUL | NULL    |                |
| shipping_method | enum('standard','express')                                    | NO   |     | NULL    |                |
| stat            | enum('pending','shipping','delivered','refunded','cancelled') | NO   |     | NULL    |                |
+-----------------+---------------------------------------------------------------+------+-----+---------+----------------+
```

Shipping Table Schema

Conversion and Normalization

The schema provided appears to be in Third Normal Form (3NF) based on the following explanations:

1. No Duplicate Data: In 3NF, there should be no duplicate data within each table. Duplicates can lead to data inconsistency and update anomalies. Upon examining the schema, there are no repeating groups or redundant data that could result in duplicate data within any of the tables.

2. All Fields are Functionally Dependent on Primary Keys: In 3NF, all fields in a table should be functionally dependent on the primary key, and there should be no partial dependencies. Partial dependencies occur when a non-primary key field is dependent on only part of the composite primary key. In the provided schema, all the tables have a primary key, and all the non-primary key fields are fully dependent on the primary key of their respective tables. For example, in the "customer_address" table, the fields "home_number", "street", "city", "state", and "zip_code" are all fully dependent on the primary key "customer_id".

3. Elimination of Transitive Dependencies: In 3NF, there should be no transitive dependencies, which means that non-primary key fields should not be dependent on other non-primary key fields. Transitive dependencies can cause data redundancy and update anomalies. Upon examining the schema, there are no transitive dependencies present.

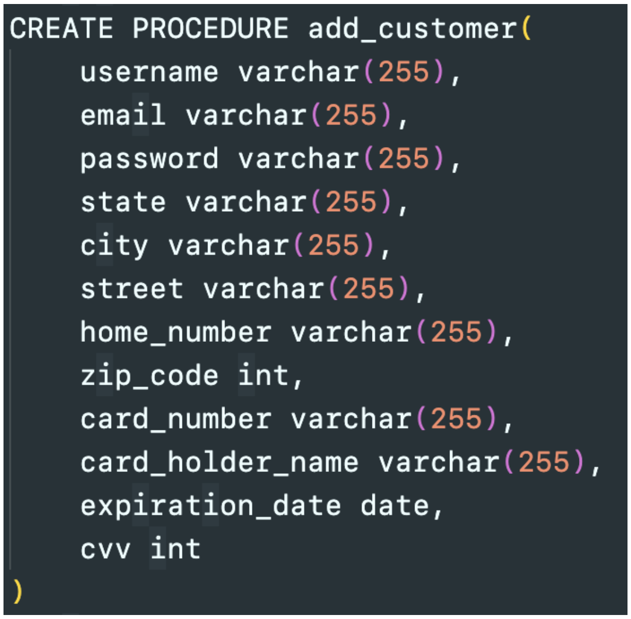
4. Use of Foreign Keys: In 3NF, foreign keys are used to establish relationships between tables. Foreign keys are used to refer to the primary key of another table and create relationships, which help in maintaining data integrity and consistency. In the provided schema, foreign keys are used to establish relationships between tables such as "customer_address", "customer_payment_info", "cart", "orders", "order_details", "shipping", "payment", and "review", by referring to the primary key "customer_id" in the "customer" table and the primary key "id" in the "product" table.
5. No Derived Data: In 3NF, there should be no derived data, which means that all data should be stored directly in the table and not derived from other data. Upon examining the schema, there are no derived data fields present.

SQL Queries

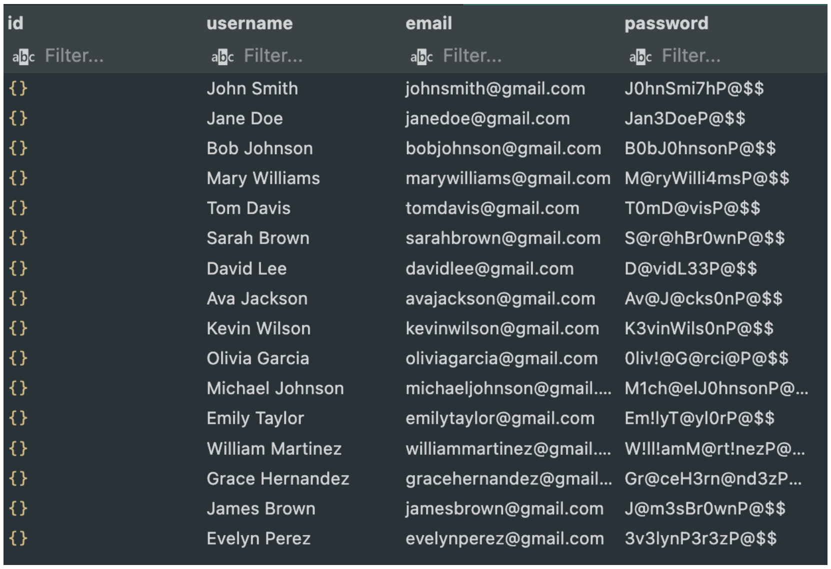
Various procedures have been made to easily run the queries provided by the user. Following is the documentation of these various procedures.

1. Creating new account / adding customer information

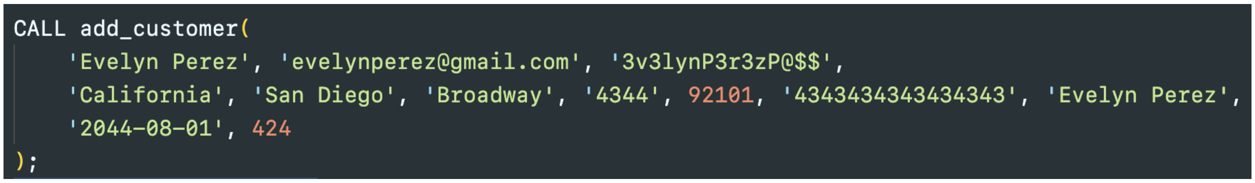
   This procedure takes in relevant inputs and fills then into the required tables. The id of the customer is a universally unique identifier (UUID) stored in binary format in the database. (which is why it is just shown as {} in screenshot below as it is binary encoded).



```
CREATE PROCEDURE add_customer(
    username varchar(255),
    email varchar(255),
    password varchar(255),
    state varchar(255),
    city varchar(255),
    street varchar(255),
    home_number varchar(255),
    zip_code int,
    card_number varchar(255),
    card_holder_name varchar(255),
    expiration_date date,
    cvv int
)
```

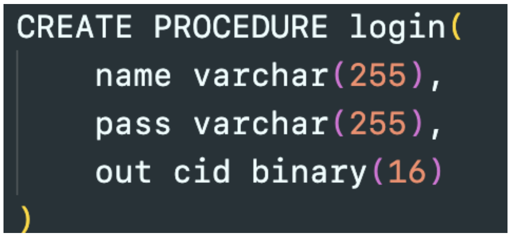


id	username	email	password
abc Filter...	abc Filter...	abc Filter...	abc Filter...
{}	John Smith	johnsmith@gmail.com	J0hnSmi7hP@$$
{}	Jane Doe	janedoe@gmail.com	Jan3DoeP@$$
{}	Bob Johnson	bobjohnson@gmail.com	B0bJ0hnsonP@$$
{}	Mary Williams	marywilliams@gmail.com	M@ryWilli4msP@$$
{}	Tom Davis	tomdavis@gmail.com	T0mD@visP@$$
{}	Sarah Brown	sarahbrown@gmail.com	S@r@hBr0wnP@$$
{}	David Lee	davidlee@gmail.com	D@vidL33P@$$
{}	Ava Jackson	avajackson@gmail.com	Av@J@cks0nP@$$
{}	Kevin Wilson	kevinwilson@gmail.com	K3vinWils0nP@$$
{}	Olivia Garcia	oliviagarcia@gmail.com	0liv!@G@rci@P@$$
{}	Michael Johnson	michaeljohnson@gmail....	M1ch@elJ0hnsonP@...
{}	Emily Taylor	emilytaylor@gmail.com	Em!lyT@yl0rP@$$
{}	William Martinez	williammartinez@gmail....	W!ll!amM@rt!nezP@...
{}	Grace Hernandez	gracehernandez@gmail...	Gr@ceH3rn@nd3zP...
{}	James Brown	jamesbrown@gmail.com	J@m3sBr0wnP@$$
{}	Evelyn Perez	evelynperez@gmail.com	3v3lynP3r3zP@$$

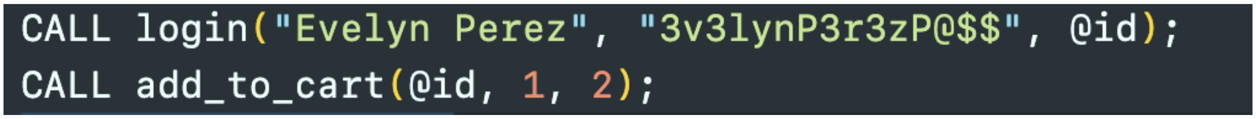


```
CALL add_customer(
    'Evelyn Perez', 'evelynperez@gmail.com', '3v3lynP3r3zP@$$',
    'California', 'San Diego', 'Broadway', '4344', 92101, '4343434343434343', 'Evelyn Perez',
    '2044-08-01', 424
);
```

The login procedure helps to fetch customer id through the unique username and password

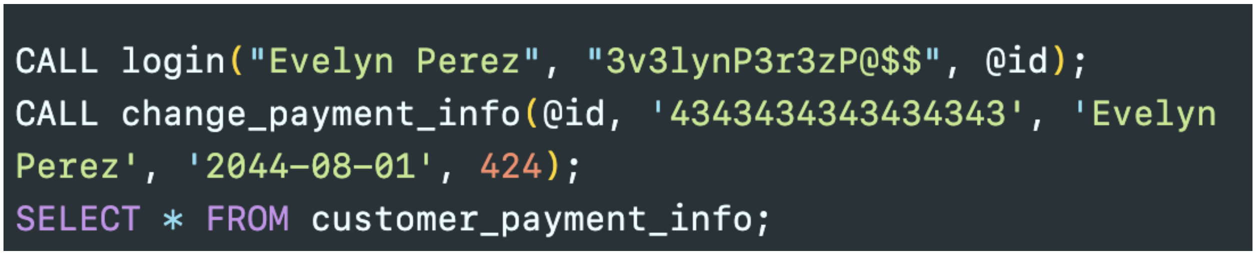


```
CREATE PROCEDURE login(
    name varchar(255),
    pass varchar(255),
    out cid binary(16)
)
```

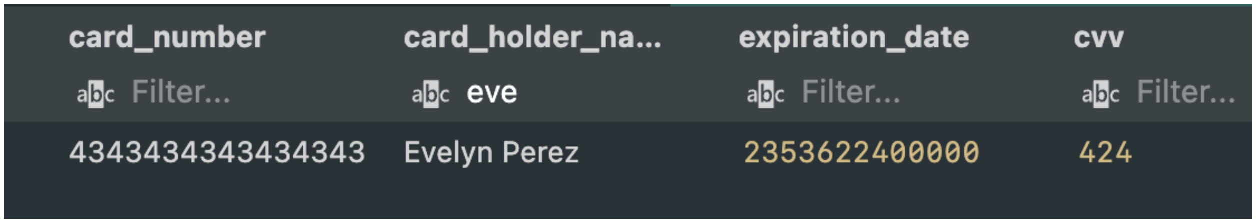


```
CALL login("Evelyn Perez", "3v3lynP3r3zP@$$", @id);
CALL add_to_cart(@id, 1, 2);
```

2. Change customer payment information

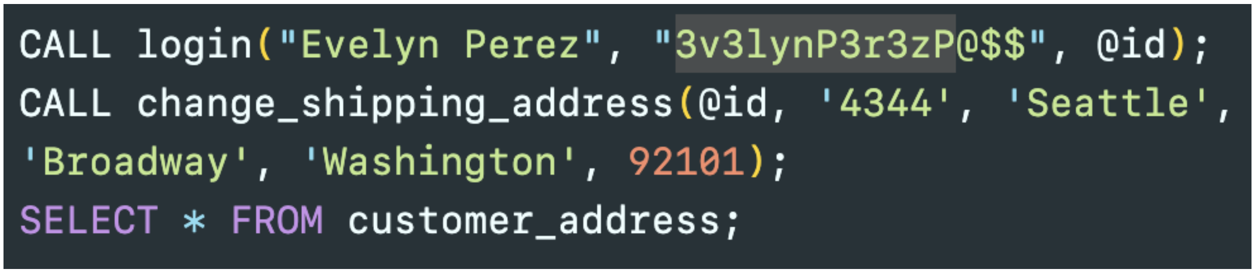


```
CALL login("Evelyn Perez", "3v3lynP3r3zP@$$", @id);
CALL change_payment_info(@id, '4343434343434343', 'Evelyn
Perez', '2044-08-01', 424);
SELECT * FROM customer_payment_info;
```

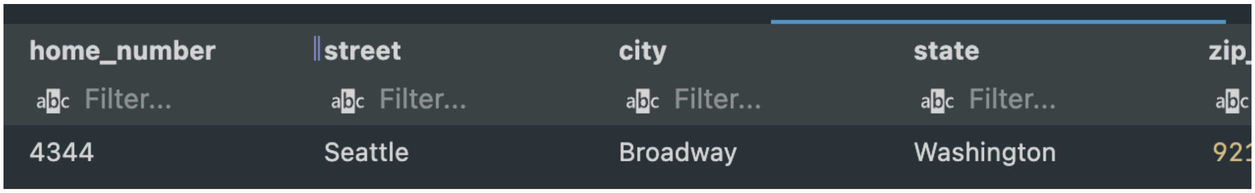


card_number	card_holder_na...	expiration_date	cvv
abc Filter...	abc eve	abc Filter...	abc Filter...
4343434343434343	Evelyn Perez	2353622400000	424

3. Change customer shipping address



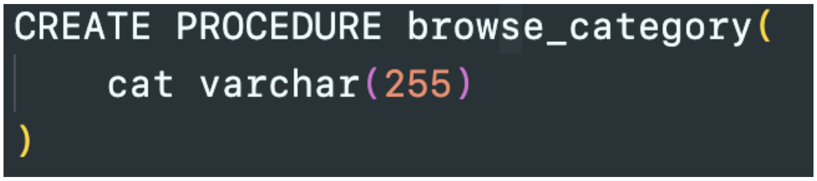
```
CALL login("Evelyn Perez", "3v3lynP3r3zP@$$", @id);
CALL change_shipping_address(@id, '4344', 'Seattle',
'Broadway', 'Washington', 92101);
SELECT * FROM customer_address;
```



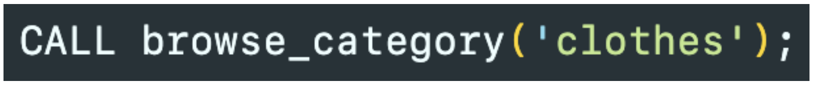
home_number	street	city	state	zip_
abc Filter...	abc Filter...	abc Filter...	abc Filter...	abc
4344	Seattle	Broadway	Washington	92

4. Browse for products in a specific category

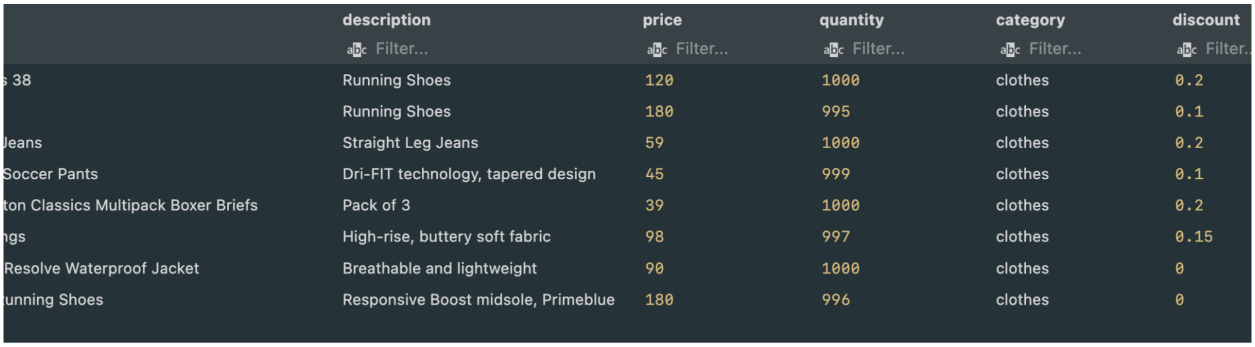
This procedure allows to see the products in a particular category.



```
CREATE PROCEDURE browse_category(
    cat varchar(255)
)
```



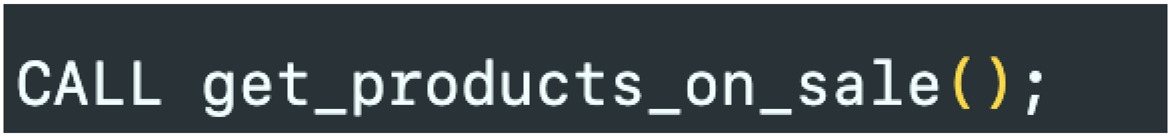
```
CALL browse_category('clothes');
```



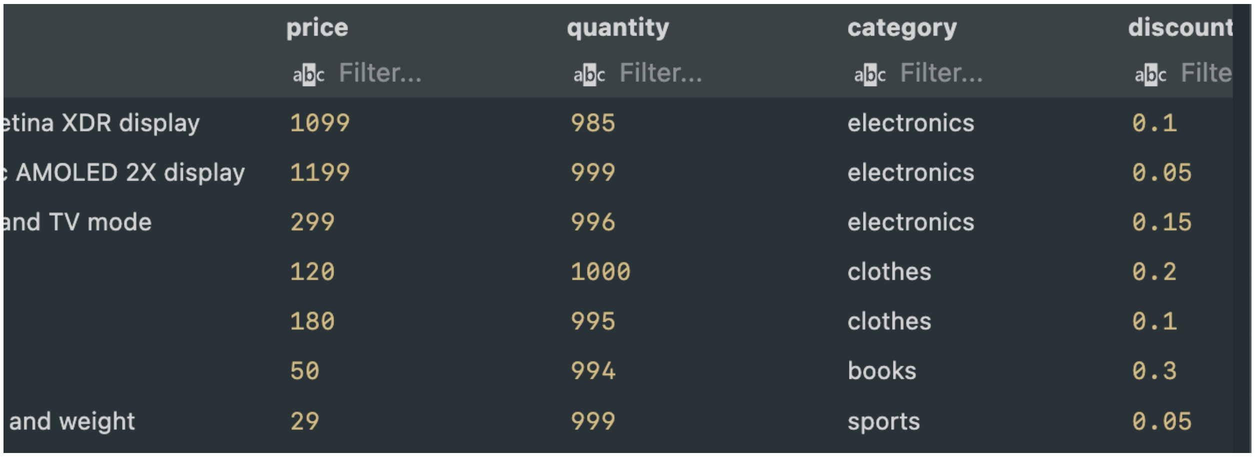
	description	price	quantity	category	discount
	Filter...	Filter...	Filter...	Filter...	Filter..
s 38	Running Shoes	120	1000	clothes	0.2
	Running Shoes	180	995	clothes	0.1
Jeans	Straight Leg Jeans	59	1000	clothes	0.2
Soccer Pants	Dri-FIT technology, tapered design	45	999	clothes	0.1
ton Classics Multipack Boxer Briefs	Pack of 3	39	1000	clothes	0.2
ngs	High-rise, buttery soft fabric	98	997	clothes	0.15
Resolve Waterproof Jacket	Breathable and lightweight	90	1000	clothes	0
unning Shoes	Responsive Boost midsole, Primeblue	180	996	clothes	0

5. Get products on sale

Returns the products which have some discount available.



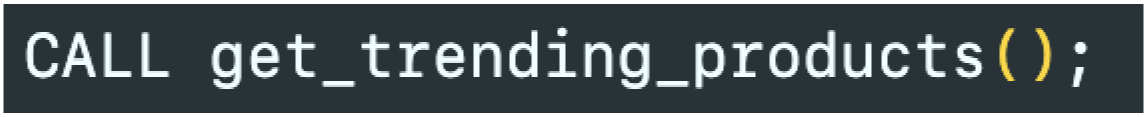
```
CALL get_products_on_sale();
```



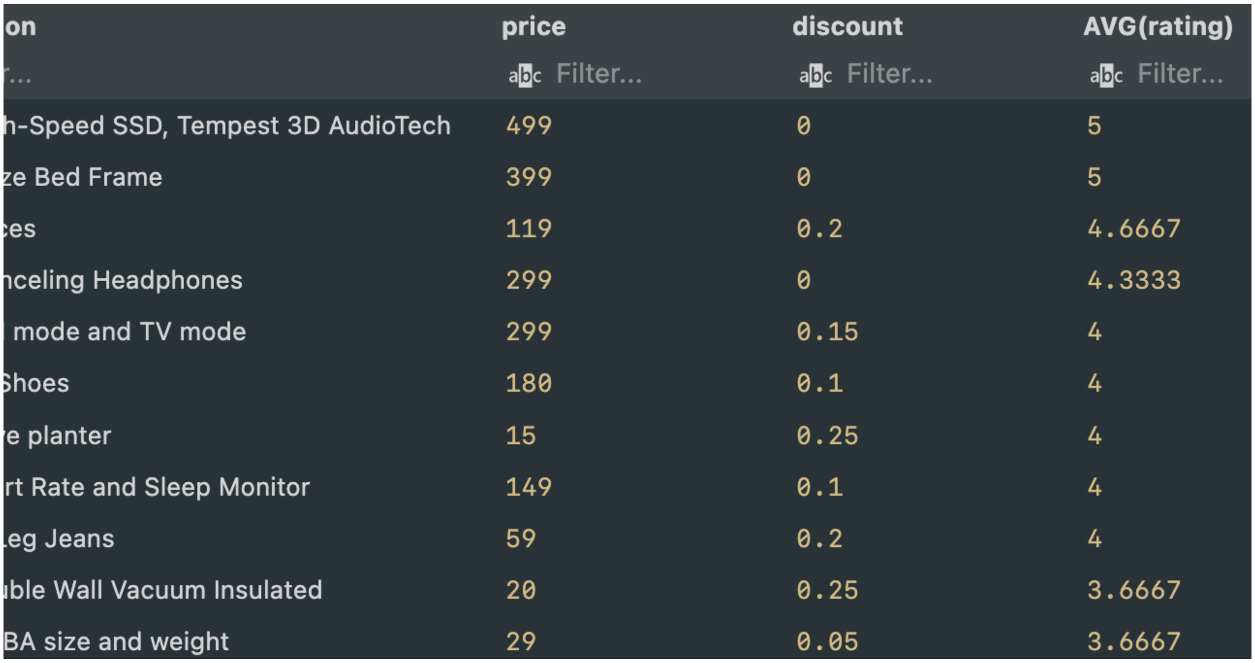
	price	quantity	category	discount
	Filter...	Filter...	Filter...	Filte
etina XDR display	1099	985	electronics	0.1
: AMOLED 2X display	1199	999	electronics	0.05
and TV mode	299	996	electronics	0.15
	120	1000	clothes	0.2
	180	995	clothes	0.1
	50	994	books	0.3
and weight	29	999	sports	0.05

6. Get trending products

   Returns the products with high average ratings.



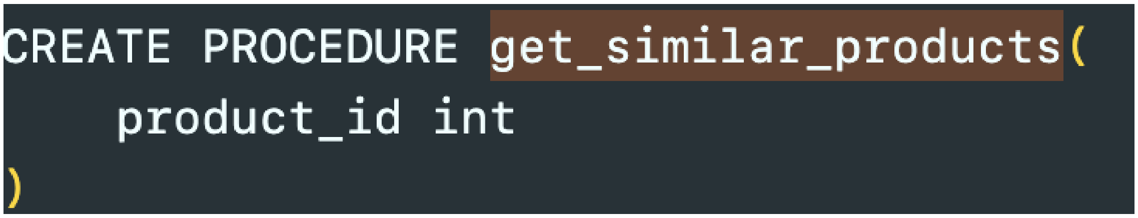
```
CALL get_trending_products();
```



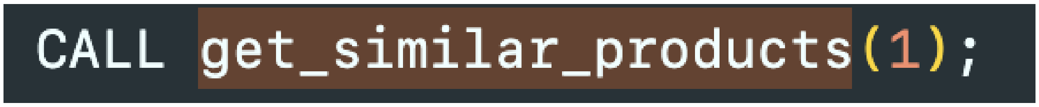
on	price	discount	AVG(rating)
r...	abc Filter...	abc Filter...	abc Filter...
h-Speed SSD, Tempest 3D AudioTech	499	0	5
ze Bed Frame	399	0	5
ces	119	0.2	4.6667
nceling Headphones	299	0	4.3333
mode and TV mode	299	0.15	4
Shoes	180	0.1	4
e planter	15	0.25	4
rt Rate and Sleep Monitor	149	0.1	4
eg Jeans	59	0.2	4
ble Wall Vacuum Insulated	20	0.25	3.6667
BA size and weight	29	0.05	3.6667

7. Get similar product to the one you purchase

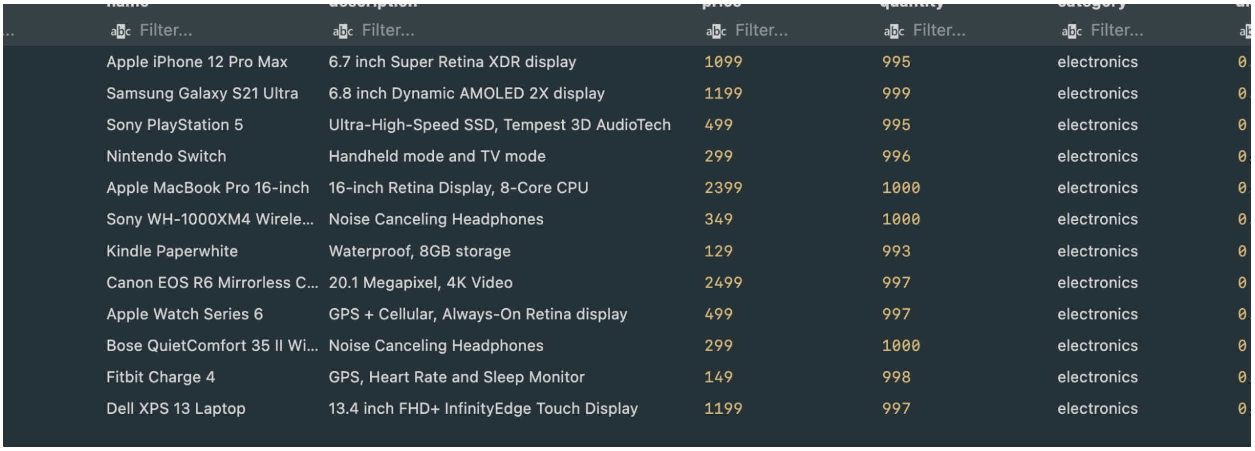
   This procedure takes in the id of the product and gets similar products by showing other products of same category.



```
CREATE PROCEDURE get_similar_products(
    product_id int
)
```



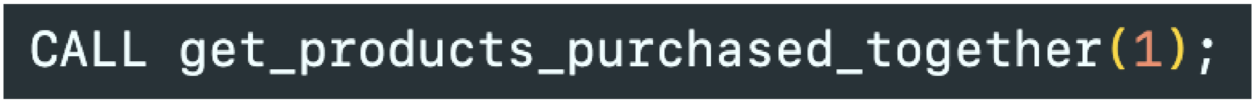
```
 CALL get_similar_products(1);
```



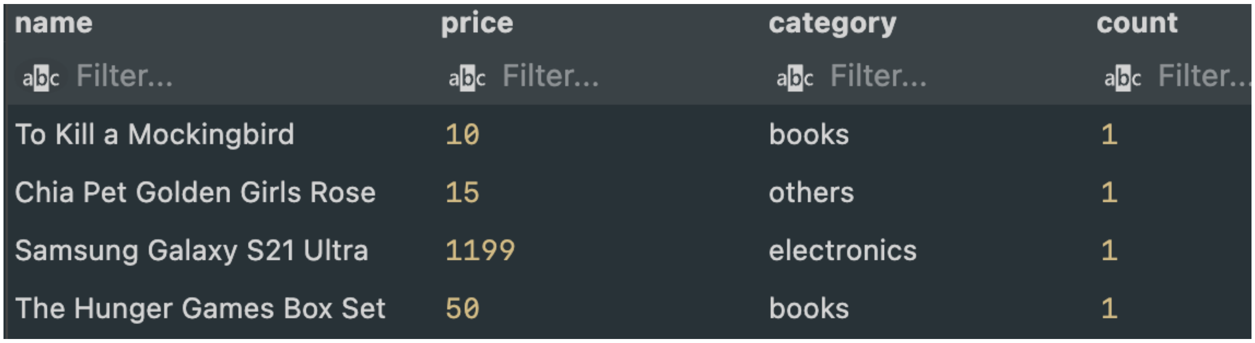
name	description	price	quantity	category	
Apple iPhone 12 Pro Max	6.7 inch Super Retina XDR display	1099	995	electronics	0.
Samsung Galaxy S21 Ultra	6.8 inch Dynamic AMOLED 2X display	1199	999	electronics	0.
Sony PlayStation 5	Ultra-High-Speed SSD, Tempest 3D AudioTech	499	995	electronics	0
Nintendo Switch	Handheld mode and TV mode	299	996	electronics	0.
Apple MacBook Pro 16-inch	16-inch Retina Display, 8-Core CPU	2399	1000	electronics	0.
Sony WH-1000XM4 Wirele...	Noise Canceling Headphones	349	1000	electronics	0.
Kindle Paperwhite	Waterproof, 8GB storage	129	993	electronics	0
Canon EOS R6 Mirrorless C...	20.1 Megapixel, 4K Video	2499	997	electronics	0
Apple Watch Series 6	GPS + Cellular, Always-On Retina display	499	997	electronics	0.
Bose QuietComfort 35 II Wi...	Noise Canceling Headphones	299	1000	electronics	0
Fitbit Charge 4	GPS, Heart Rate and Sleep Monitor	149	998	electronics	0.
Dell XPS 13 Laptop	13.4 inch FHD+ InfinityEdge Touch Display	1199	997	electronics	0.

8. <u>Get frequently purchased product with the one you purchase</u>

This procedure takes in the id of the product and gets products that are frequently purchased with that product



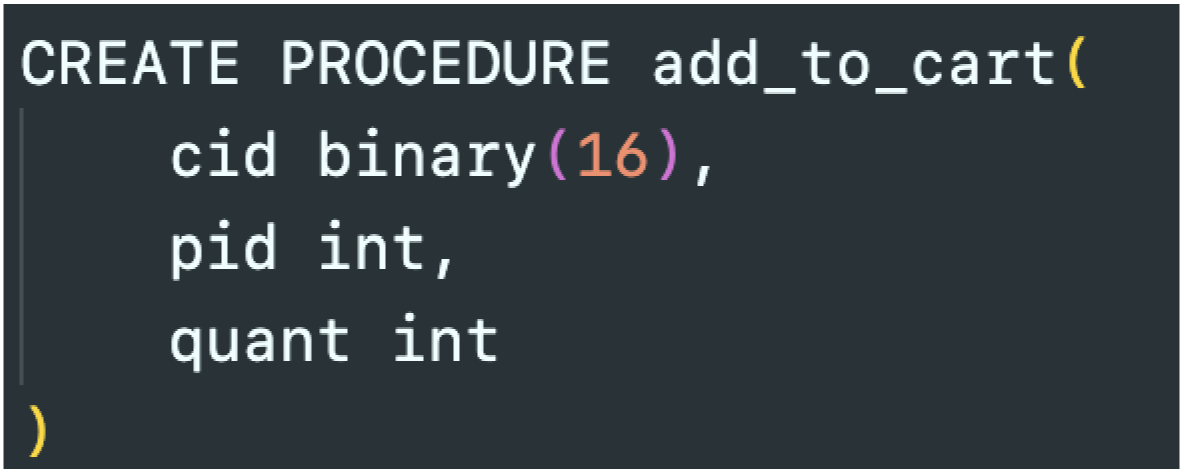
```
 CALL get_products_purchased_together(1);
```



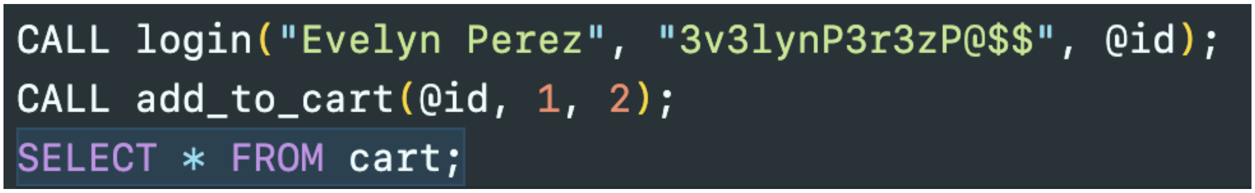
name	price	category	count
To Kill a Mockingbird	10	books	1
Chia Pet Golden Girls Rose	15	others	1
Samsung Galaxy S21 Ultra	1199	electronics	1
The Hunger Games Box Set	50	books	1

9. <u>Add products to cart</u>

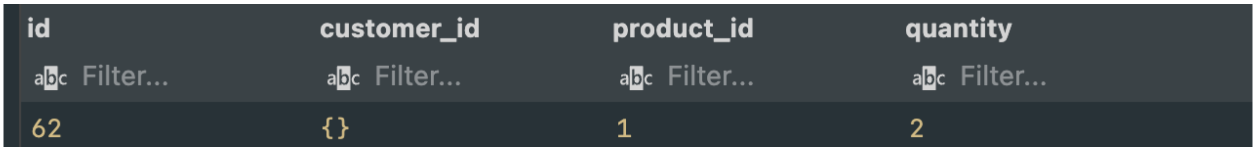
Add the products to cart so that you can place order, while placing order all the items in cart are ordered and user’s cart gets emptied. It takes in customer’s id, procut’s id and quantity of product as input.



```
CREATE PROCEDURE add_to_cart(
    cid binary(16),
    pid int,
    quant int
)
```



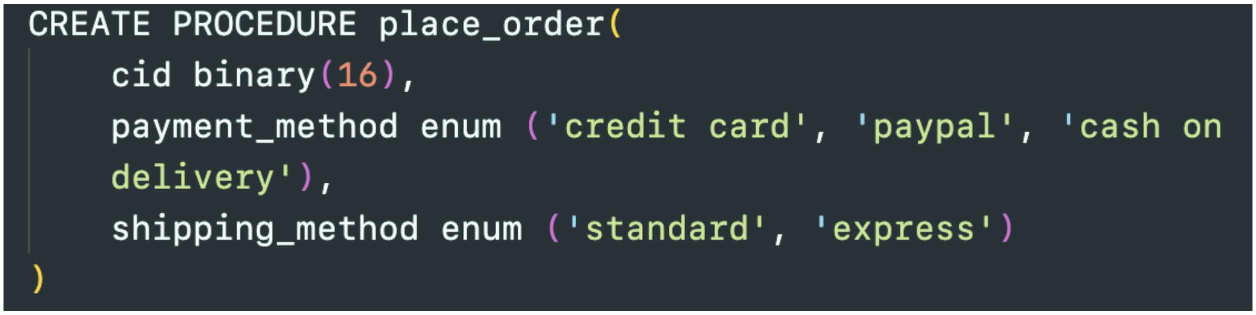
```
CALL login("Evelyn Perez", "3v3lynP3r3zP@$$", @id);
CALL add_to_cart(@id, 1, 2);
SELECT * FROM cart;
```



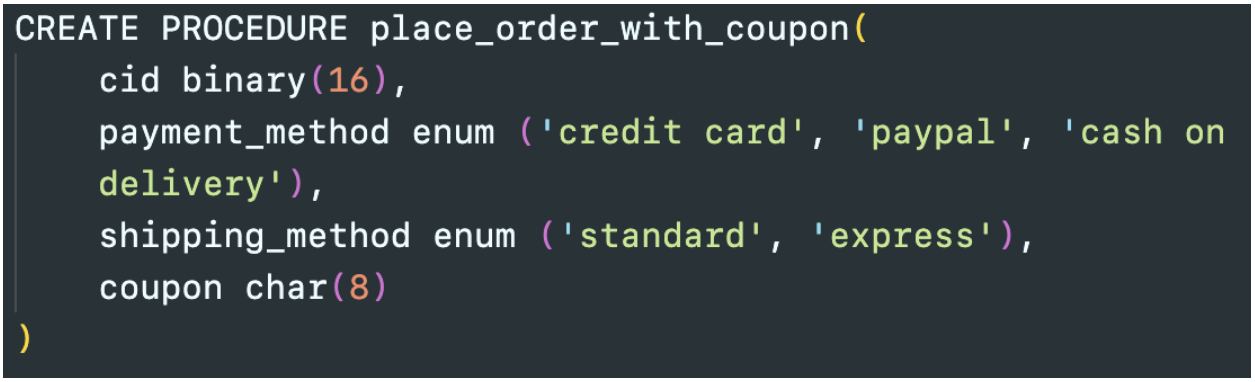
id	customer_id	product_id	quantity
Filter...	Filter...	Filter...	Filter...
62	{}	1	2

10. Place Order / Place Order with coupon

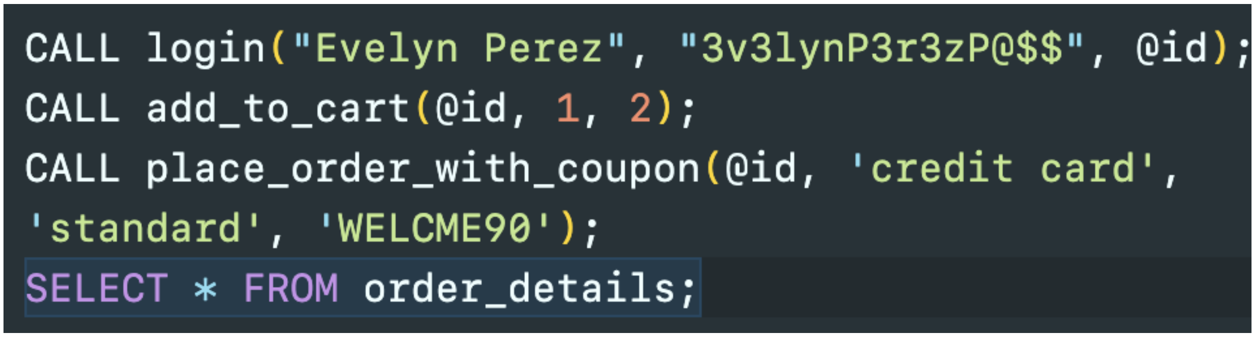
Places order with relevant discount and empties the cart. Adds data to payment and shipping tables. Also removes relevant quantities from product tables. Using coupon gives the discount that the coupon has.



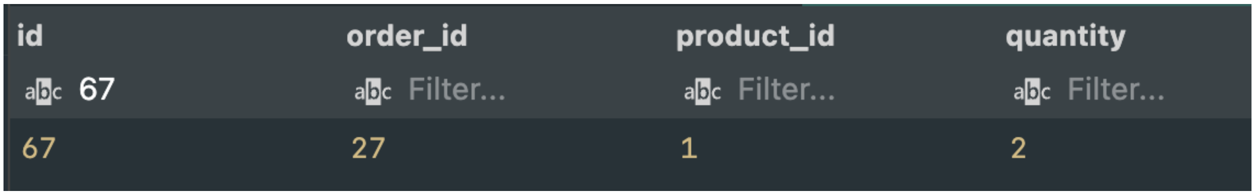
```
CREATE PROCEDURE place_order(
    cid binary(16),
    payment_method enum ('credit card', 'paypal', 'cash on
    delivery'),
    shipping_method enum ('standard', 'express')
)
```



```
CREATE PROCEDURE place_order_with_coupon(
    cid binary(16),
    payment_method enum ('credit card', 'paypal', 'cash on
    delivery'),
    shipping_method enum ('standard', 'express'),
    coupon char(8)
)
```



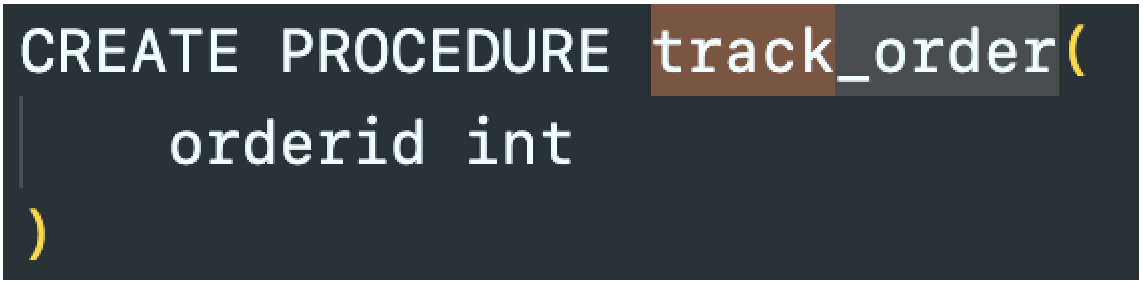
```
CALL login("Evelyn Perez", "3v3lynP3r3zP@$$", @id);
CALL add_to_cart(@id, 1, 2);
CALL place_order_with_coupon(@id, 'credit card',
'standard', 'WELCME90');
SELECT * FROM order_details;
```



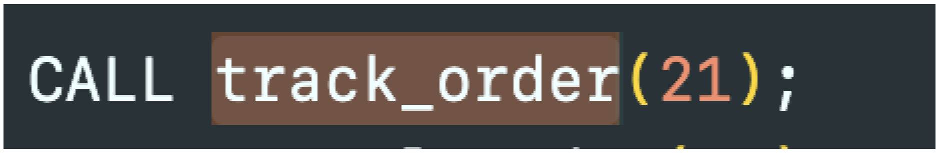
id	order_id	product_id	quantity
67	Filter...	Filter...	Filter...
67	27	1	2

11. Track your order

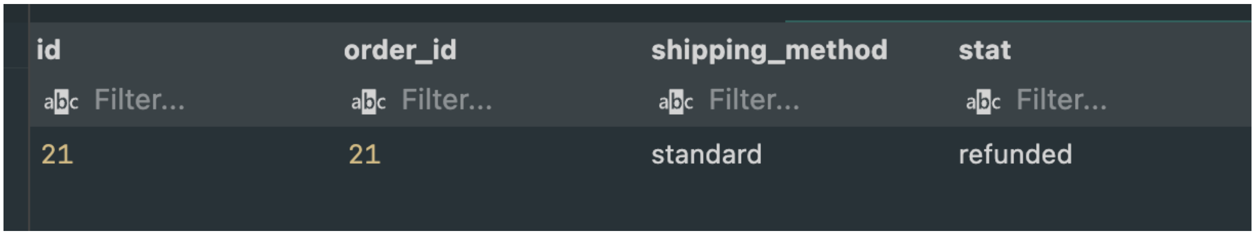
This procedure takes in the id of the order you want to track, which you can get after placing the order through the place_order procedure



```
CREATE PROCEDURE track_order(
    orderid int
)
```



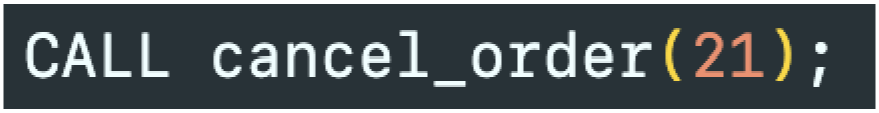
```
CALL track_order(21);
```



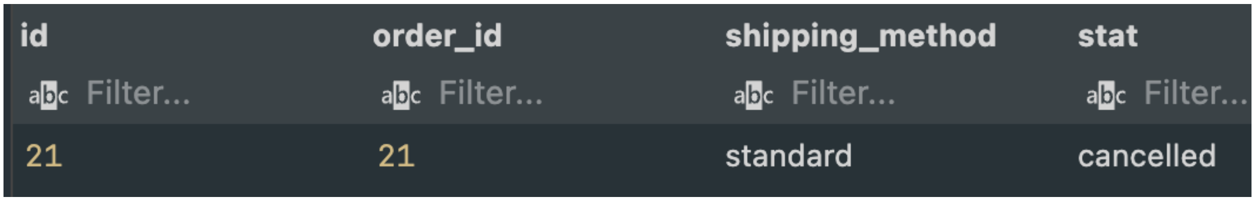
id	order_id	shipping_method	stat
Filter...	Filter...	Filter...	Filter...
21	21	standard	refunded

12. Cancel your order

This procedure takes in the id of the order you want to track, which you can get after placing the order through the place_order procedure



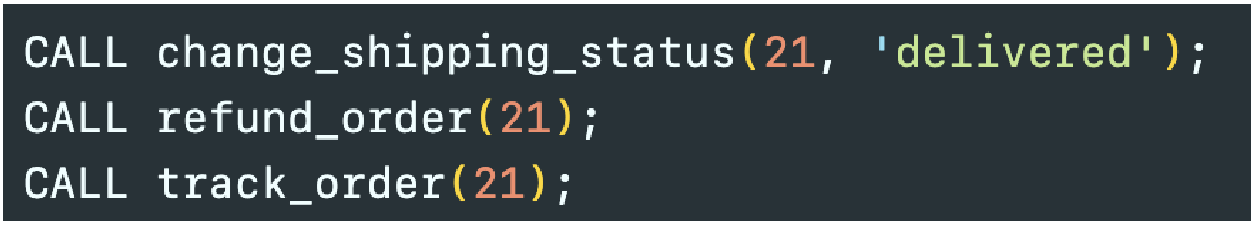
```
CALL cancel_order(21);
```



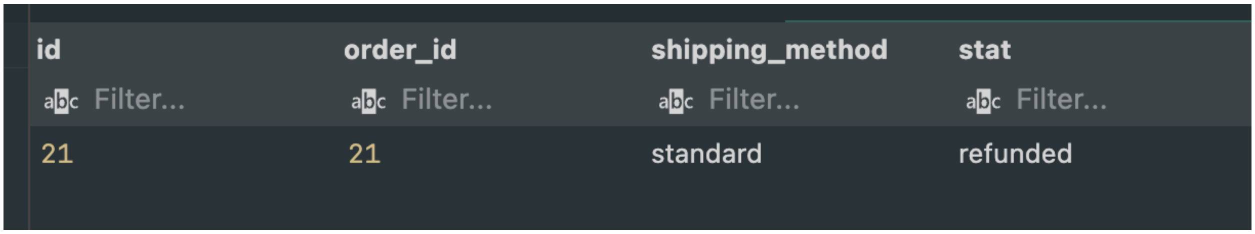
id	order_id	shipping_method	stat
Filter...	Filter...	Filter...	Filter...
21	21	standard	cancelled

13. Refund your order

This procedure takes in the id of the order you want to track, which you can get after placing the order through the place_order procedure



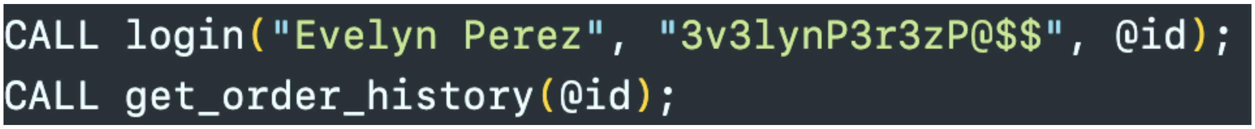
```
CALL change_shipping_status(21, 'delivered');
CALL refund_order(21);
CALL track_order(21);
```



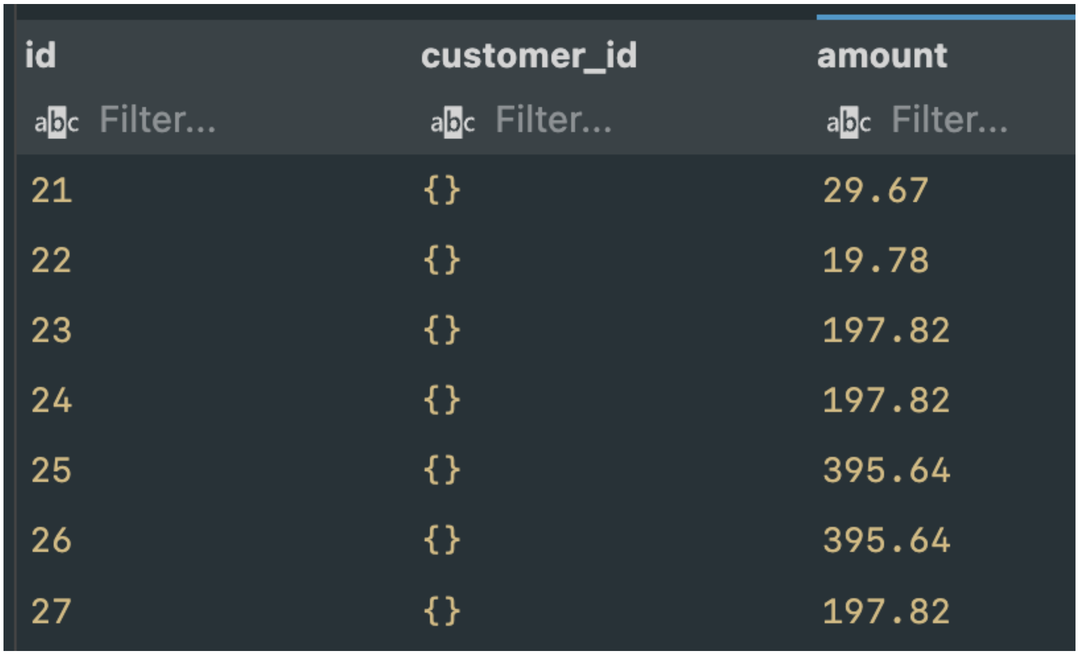
id	order_id	shipping_method	stat
Filter...	Filter...	Filter...	Filter...
21	21	standard	refunded

14. See orders history

Takes in customer id as input



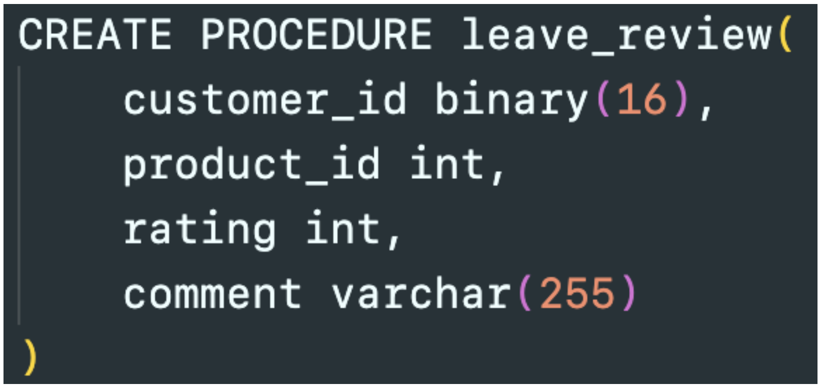
```
CALL login("Evelyn Perez", "3v3lynP3r3zP@$$", @id);
CALL get_order_history(@id);
```



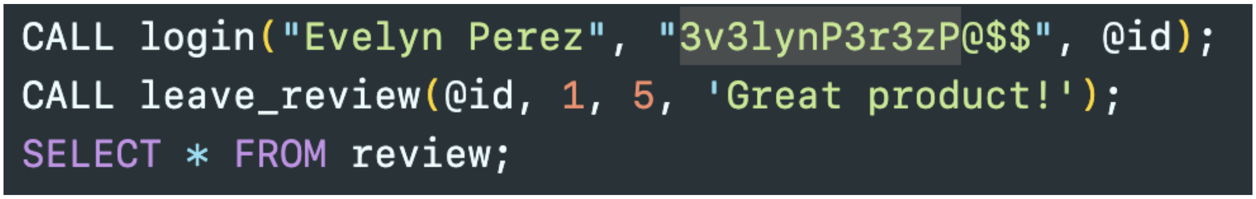
id	customer_id	amount
Filter...	Filter...	Filter...
21	{}	29.67
22	{}	19.78
23	{}	197.82
24	{}	197.82
25	{}	395.64
26	{}	395.64
27	{}	197.82

15. Leave Review

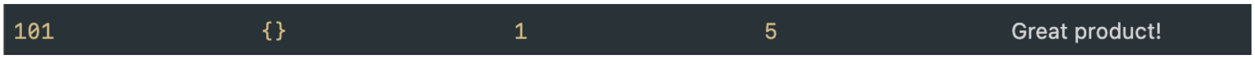
Takes in customer id, product id, rating and comment as input



```
CREATE PROCEDURE leave_review(
    customer_id binary(16),
    product_id int,
    rating int,
    comment varchar(255)
)
```

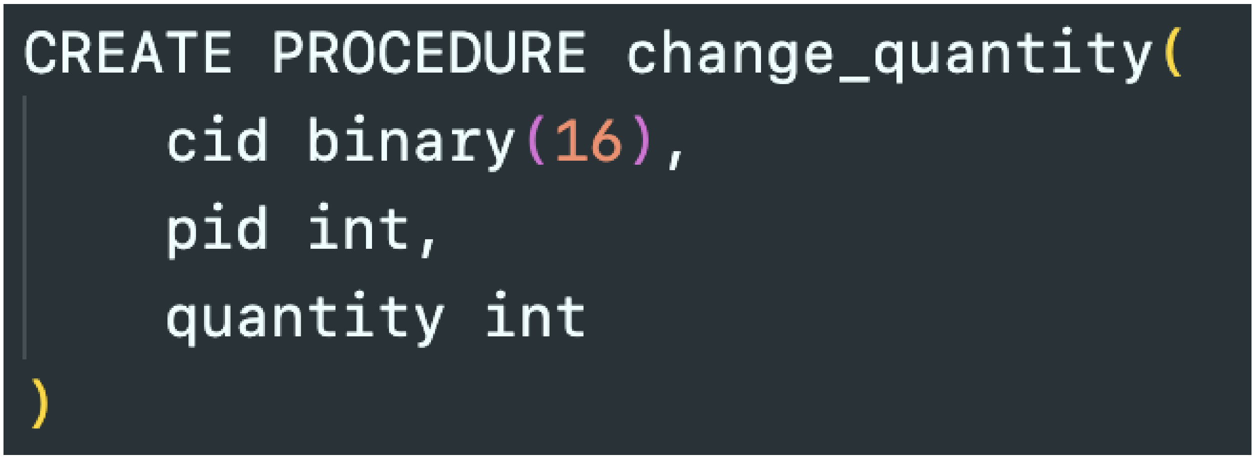


```
CALL login("Evelyn Perez", "3v3lynP3r3zP@$$", @id);
CALL leave_review(@id, 1, 5, 'Great product!');
SELECT * FROM review;
```

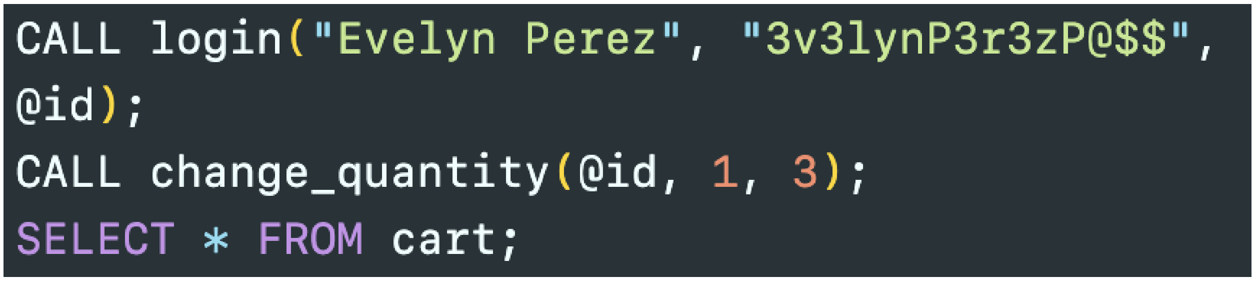


```
101          {}          1          5          Great product!
```

16. Change product quantity in cart



```
CREATE PROCEDURE change_quantity(
    cid binary(16),
    pid int,
    quantity int
)
```



```
CALL login("Evelyn Perez", "3v3lynP3r3zP@$$",
@id);
CALL change_quantity(@id, 1, 3);
SELECT * FROM cart;
```

There are some other helper procedures that can be used like `change_shipping_status`, `generate_random_reviews`, `generate_random_orders`

Video Explanation

https://drive.google.com/drive/folders/1nJD9Zbz89HZnZOtKT4hpE2xCXFc1Qfp4?usp=sharing