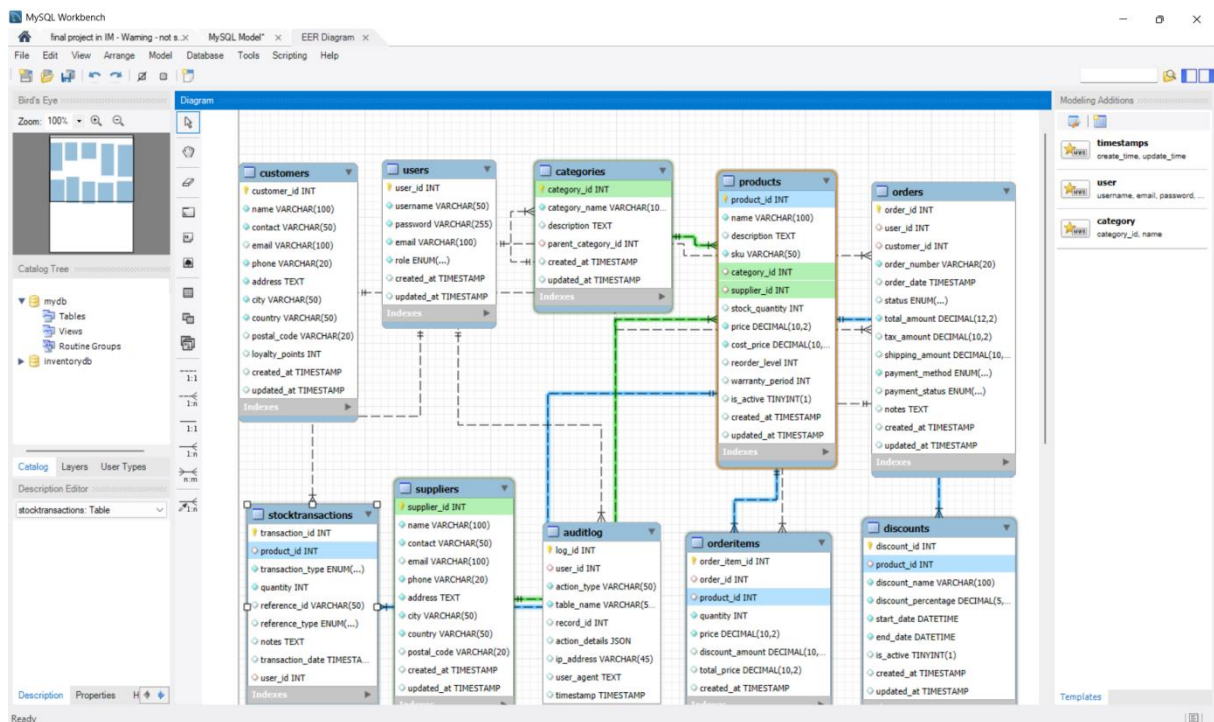




## INVENTORY MANAGEMENT SYSTEM

### Final Documentation Compilation

#### Final ER Diagram



#### Database Schema and Table Description

##### 1. Auditlog

| Table Details          |                     |
|------------------------|---------------------|
| Engine:                | InnoDB              |
| Row format:            | Dynamic             |
| Column count:          | 9                   |
| Table rows:            | 267454              |
| AVG row length:        | 139                 |
| Data length:           | 35.6 MiB            |
| Index length:          | 14.5 MiB            |
| Max data length:       | 0.0 bytes           |
| Data free:             | 7.0 MiB             |
| Table size (estimate): | 50.1 MiB            |
| Update time:           | 2025-04-27 18:55:13 |
| Create time:           | 2025-04-27 10:10:05 |
| Auto increment:        | 360006              |
| Table collation:       | utf8mb4_0900_ai_ci  |
| Create options:        |                     |
| Comment:               |                     |

A. info

File Edit View Query Database Server Tools Scripting Help

Navigator

updated inventorydb inventorydb.auditlog

Info Columns Indexes Triggers Foreign keys Partitions Grants DDL

| Column         | Type        | Default Value      | Nullable | Character Set | Collation        | Privileges                      | Extra           | Comments |
|----------------|-------------|--------------------|----------|---------------|------------------|---------------------------------|-----------------|----------|
| log_id         | int         |                    | NO       |               |                  | select,insert,update,references | auto_increment  |          |
| user_id        | int         |                    | YES      | utf8mb4       | utf8mb4_0900_... | select,insert,update,references |                 |          |
| action_type    | varchar(50) |                    | NO       | utf8mb4       | utf8mb4_0900_... | select,insert,update,references |                 |          |
| table_name     | varchar(50) |                    | YES      | utf8mb4       | utf8mb4_0900_... | select,insert,update,references |                 |          |
| record_id      | int         |                    | YES      |               |                  | select,insert,update,references |                 |          |
| action_details | json        |                    | YES      |               |                  | select,insert,update,references |                 |          |
| ip_address     | varchar(45) |                    | YES      | utf8mb4       | utf8mb4_0900_... | select,insert,update,references |                 |          |
| user_agent     | text        |                    | YES      | utf8mb4       | utf8mb4_0900_... | select,insert,update,references |                 |          |
| timestamp      | timestamp   | CURRENT_TIMESTA... | YES      |               |                  | select,insert,update,references | DEFAULT_GENE... |          |

Administration Schemas

## B. Columns

Navigator

updated inventorydb inventorydb.auditlog

Info Columns Indexes Triggers Foreign keys Partitions Grants DDL

**Indexes in Table**

| Key                 | Type   | Unique | Columns    |
|---------------------|--------|--------|------------|
| PRIMARY             | B+TREE | YES    | log_id     |
| idx_audit_user      | B+TREE | NO     | user_id    |
| idx_audit_table     | B+TREE | NO     | table_name |
| idx_audit_timestamp | B+TREE | NO     | timestamp  |

**Index Details**

Key Name:   
 Index Type:   
 Allows NULL:   
 Cardinality:   
 Comment:   
 User Comment:

Packed:   
 Unique:

Drop Index

**Columns in table**

| Column         | Type        | Nullable | Indexes             |
|----------------|-------------|----------|---------------------|
| log_id         | int         | NO       | PRIMARY             |
| user_id        | int         | YES      | idx_audit_user      |
| action_type    | varchar(50) | NO       |                     |
| table_name     | varchar(50) | NO       | idx_audit_table     |
| record_id      | int         | YES      |                     |
| action_details | json        | YES      |                     |
| ip_address     | varchar(45) | YES      |                     |
| user_agent     | text        | YES      |                     |
| timestamp      | timestamp   | YES      | idx_audit_timestamp |

Administration Schemas

## C. indexes

Navigator

updated inventorydb inventorydb.auditlog

Info Columns Indexes Triggers Foreign keys Partitions Grants DDL

**DDL for inventorydb.auditlog**

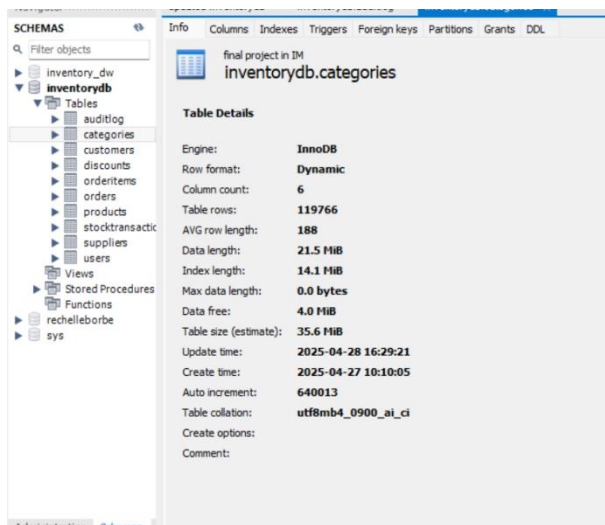
```

1 CREATE TABLE `auditlog` (
2   `log_id` int NOT NULL AUTO_INCREMENT,
3   `user_id` int DEFAULT NULL,
4   `action_type` varchar(50) NOT NULL,
5   `table_name` varchar(50) NOT NULL,
6   `record_id` int DEFAULT NULL,
7   `action_details` json DEFAULT NULL,
8   `ip_address` varchar(45) DEFAULT NULL,
9   `user_agent` text,
10  `timestamp` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
11  PRIMARY KEY (`log_id`),
12  KEY `idx_audit_user` (`user_id`),
13  KEY `idx_audit_table` (`table_name`),
14  KEY `idx_audit_timestamp` (`timestamp`),
15  CONSTRAINT `auditlog_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES `users` (`user_id`) ON DELETE SET NULL
16 ) ENGINE=InnoDB AUTO_INCREMENT=360006 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci

```

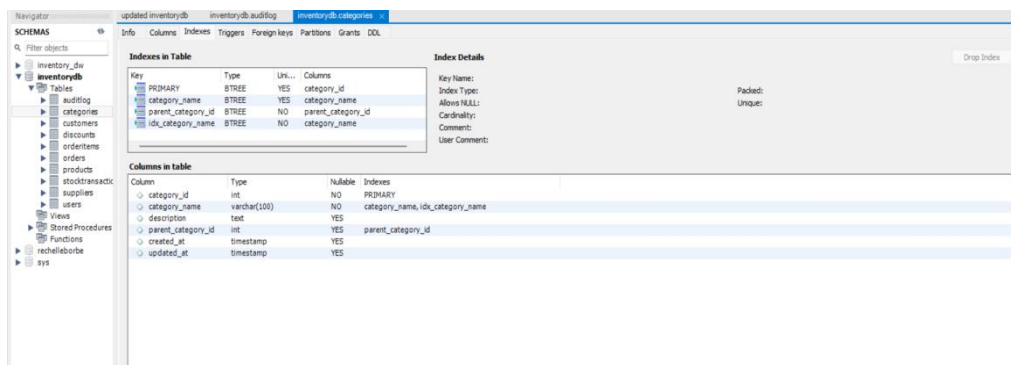
## D. Data Definition Language

## 2. Categories



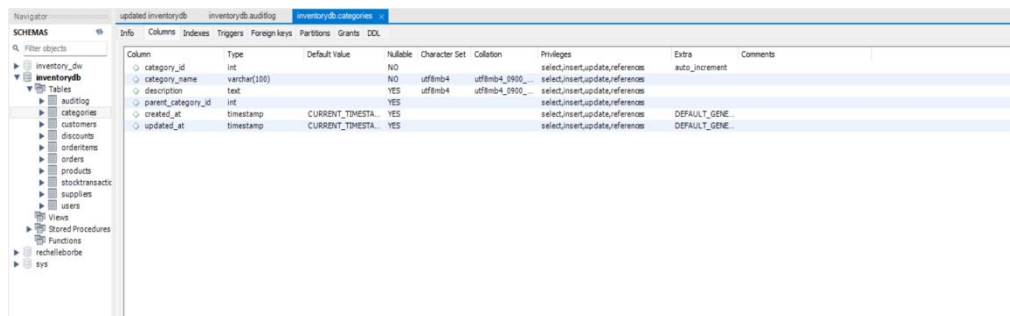
| Table Details          |                     |
|------------------------|---------------------|
| Engine:                | InnoDB              |
| Row format:            | Dynamic             |
| Column count:          | 6                   |
| Table rows:            | 119766              |
| AVG row length:        | 188                 |
| Data length:           | 21.5 MiB            |
| Index length:          | 14.1 MiB            |
| Max data length:       | 0.0 bytes           |
| Data free:             | 4.0 MiB             |
| Table size (estimate): | 35.6 MiB            |
| Update time:           | 2025-04-28 16:29:21 |
| Create time:           | 2025-04-27 10:10:05 |
| Auto increment:        | 640013              |
| Table collation:       | utf8mb4_0900_ai_ci  |
| Create options:        |                     |
| Comment:               |                     |

A. info



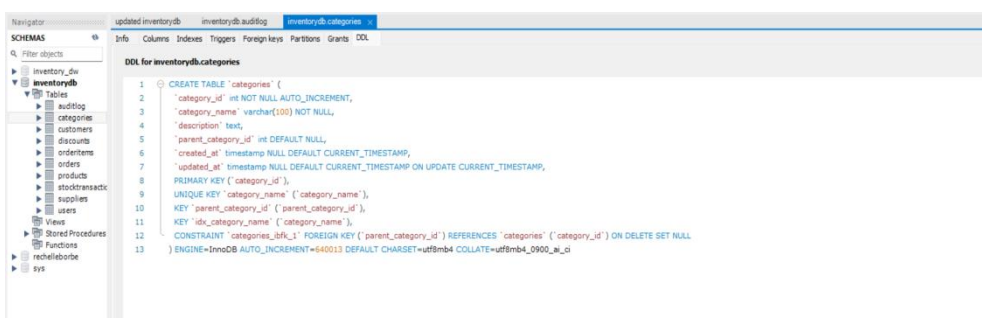
| Indexes in Table   |              |
|--------------------|--------------|
| Key                | Type         |
| PRIMARY            | BTREE        |
| category_name      | BTREE        |
| parent_category_id | BTREE        |
| idx_category_name  | BTREE        |
| Columns in Table   |              |
| Column             | Type         |
| category_id        | int          |
| category_name      | varchar(100) |
| description        | text         |
| parent_category_id | int          |
| created_at         | timestamp    |
| updated_at         | timestamp    |

B. Columns



| Column             | Type         | Default Value     | Nullable | Character Set | Collation          | Privileges                      | Extra          | Comments          |
|--------------------|--------------|-------------------|----------|---------------|--------------------|---------------------------------|----------------|-------------------|
| category_id        | int          |                   | NO       |               |                    | select,insert,update,references | auto_increment |                   |
| category_name      | varchar(100) |                   | YES      | utf8mb4       | utf8mb4_0900_ai_ci | select,insert,update,references |                |                   |
| description        | text         |                   | YES      | utf8mb4       | utf8mb4_0900_ai_ci | select,insert,update,references |                |                   |
| parent_category_id | int          |                   | YES      |               |                    | select,insert,update,references |                |                   |
| created_at         | timestamp    | CURRENT_TIMESTAMP | YES      |               |                    | select,insert,update,references |                | DEFAULT_GENERATED |
| updated_at         | timestamp    | CURRENT_TIMESTAMP | YES      |               |                    | select,insert,update,references |                | DEFAULT_GENERATED |

C. Indexes



```
1 CREATE TABLE `categories` (  
2   `category_id` int NOT NULL AUTO_INCREMENT,  
3   `category_name` varchar(100) NOT NULL,  
4   `description` text,  
5   `parent_category_id` int DEFAULT NULL,  
6   `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,  
7   `updated_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,  
8   PRIMARY KEY (`category_id`),  
9   UNIQUE KEY `category_name` (`category_name`),  
10  KEY `parent_category_id` (`parent_category_id`),  
11  KEY `idx_category_name` (`category_name`),  
12  CONSTRAINT `categories_ibfk_1` FOREIGN KEY (`parent_category_id`) REFERENCES `categories` (`category_id`) ON DELETE SET NULL  
13 ) ENGINE=InnoDB AUTO_INCREMENT=640013 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

D. Data  
Definition  
Language

### 3. Costumers

**final project in IM**  
**inventorydb.customers**

**Table Details**

Engine: **InnoDB**  
Row format: **Dynamic**  
Column count: **12**  
Table rows: **197776**  
AVG row length: **199**  
Data length: **37.6 MiB**  
Index length: **19.0 MiB**  
Max data length: **0.0 bytes**  
Data free: **4.0 MiB**  
Table size (estimate): **56.6 MiB**  
Update time: **2025-04-27 17:42:23**  
Create time: **2025-04-27 10:10:05**  
Auto increment: **500031**  
Table collation: **utf8mb4\_0900\_ai\_ci**  
Create options:  
Comment:

A. Info

| Column         | Type         | Default Value     | Nullable | Character Set | Collation       | Privileges                      | Extra               | Comments |
|----------------|--------------|-------------------|----------|---------------|-----------------|---------------------------------|---------------------|----------|
| customer_id    | int          |                   | NO       |               |                 | select,insert,update,references | auto_increment      |          |
| name           | varchar(100) |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                     |          |
| contact        | varchar(50)  |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                     |          |
| email          | varchar(100) |                   | YES      | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                     |          |
| phone          | varchar(20)  |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                     |          |
| address        | text         |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                     |          |
| city           | varchar(50)  |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                     |          |
| country        | varchar(50)  |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                     |          |
| postal_code    | varchar(20)  |                   | YES      | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                     |          |
| loyalty_points | int          | 0                 | YES      |               |                 | select,insert,update,references |                     |          |
| created_at     | timestamp    | CURRENT_TIMESTAMP | YES      |               |                 | select,insert,update,references | DEFAULT_GENERATE... |          |
| updated_at     | timestamp    | CURRENT_TIMESTAMP | YES      |               |                 | select,insert,update,references | DEFAULT_GENERATE... |          |

B. Columns

**Indexes in Table**

| Key                  | Type  | Unique | Columns     |
|----------------------|-------|--------|-------------|
| PRIMARY              | BTREE | YES    | customer_id |
| idx_customer_name    | BTREE | NO     | name        |
| idx_customer_contact | BTREE | NO     | contact     |

**Index Details**

Key Name:   
Index Type:   
Allows NULL:   
Cardinality:   
Comment:   
User Comment:

Packed:   
Unique:

**Columns in table**

| Column         | Type         | Nullable | Indexes              |
|----------------|--------------|----------|----------------------|
| customer_id    | int          | NO       | PRIMARY              |
| name           | varchar(100) | NO       | idx_customer_name    |
| contact        | varchar(50)  | NO       | idx_customer_contact |
| email          | varchar(100) | YES      |                      |
| phone          | varchar(20)  | NO       |                      |
| address        | text         | NO       |                      |
| city           | varchar(50)  | NO       |                      |
| country        | varchar(50)  | NO       |                      |
| postal_code    | varchar(20)  | YES      |                      |
| loyalty_points | int          | YES      |                      |
| created_at     | timestamp    | YES      |                      |
| updated_at     | timestamp    | YES      |                      |

C. Indexes

**DDL for inventorydb.customers**

```
1 CREATE TABLE `customers` (  
2   `customer_id` int NOT NULL AUTO_INCREMENT,  
3   `name` varchar(100) NOT NULL,  
4   `contact` varchar(50) NOT NULL,  
5   `email` varchar(100) DEFAULT NULL,  
6   `phone` varchar(20) NOT NULL,  
7   `address` text NOT NULL,  
8   `city` varchar(50) NOT NULL,  
9   `country` varchar(50) NOT NULL,  
10  `postal_code` varchar(20) DEFAULT NULL,  
11  `loyalty_points` int DEFAULT 0,  
12  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,  
13  `updated_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,  
14  PRIMARY KEY (`customer_id`),  
15  KEY `idx_customer_name` (`name`),  
16  KEY `idx_customer_contact` (`contact`),  
17  CONSTRAINT `customers_chk_1` CHECK ((`email` like 'utf8mb4%@@.%'))  
18 ) ENGINE=InnoDB AUTO_INCREMENT=500031 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

D. Data  
Definition  
Languages

## 4. Discounts

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.cust...

**SCHEMAS**

Filter objects

- inventory\_dw
  - inventorydb
    - auditlog
    - categories
    - customers
    - discounts
    - orderitems
    - orders
    - products
    - stocktransacti
    - suppliers
    - users
    - Views
    - Stored Procedures
    - Functions
  - rechelleborbe
  - sys

**Info** Columns Indexes Triggers Foreign keys Partitions Grants DDL

final project in IM  
inventorydb.discounts

**Table Details**

Engine: **InnoDB**

Row format: **Dynamic**

Column count: **9**

Table rows: **508718**

AVG row length: **71**

Data length: **34.6 MiB**

Index length: **29.1 MiB**

Max data length: **0.0 bytes**

Data free: **5.0 MiB**

Table size (estimate): **63.6 MiB**

Update time: **2025-04-28 16:29:28**

Create time: **2025-04-27 10:10:05**

Auto increment: **511006**

Table collation: **utf8mb4\_0900\_ai\_ci**

Create options:

Comment:

A. info

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.customers **inventorydb.discounts**

**Info** Columns Indexes Triggers Foreign keys Partitions Grants DDL

Filter objects

**Columns**

| Column              | Nullable | Type         | Default Value     | Nullable | Character Set | Collation      | Privileges                      | Data           | Comments         |
|---------------------|----------|--------------|-------------------|----------|---------------|----------------|---------------------------------|----------------|------------------|
| discount_id         | NO       | int          |                   | YES      |               |                | select,insert,update,references | auto_increment |                  |
| product_id          | NO       | int          |                   | YES      |               |                | select,insert,update,references |                |                  |
| discount_name       | NO       | varchar(100) |                   | NO       | utf8mb4       | utf8mb4_900... | select,insert,update,references |                |                  |
| discount_percentage | NO       | decimal(5,2) |                   | NO       |               |                | select,insert,update,references |                |                  |
| start_date          | NO       | datetime     |                   | NO       |               |                | select,insert,update,references |                |                  |
| end_date            | NO       | datetime     |                   | NO       |               |                | select,insert,update,references |                |                  |
| is_active           | NO       | tinyint(1)   | 0                 | YES      |               |                | select,insert,update,references |                |                  |
| created_at          | YES      | timestamp    | CURRENT_TIMESTAMP | YES      |               |                | select,insert,update,references |                | DEFAULT_GENERATE |
| updated_at          | YES      | timestamp    | CURRENT_TIMESTAMP | YES      |               |                | select,insert,update,references |                | DEFAULT_GENERATE |

B. Columns

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.customers **inventorydb.discounts**

**Info** Columns Indexes Triggers Foreign keys Partitions Grants DDL

Filter objects

**Indexes in Table**

| Key                  | Type  | UN... | Columns             |
|----------------------|-------|-------|---------------------|
| PRIMARY              | STREE | YES   | discount_id         |
| idx_discount_product | STREE | NO    | product_id          |
| idx_discount_dates   | STREE | NO    | start_date,end_date |

**Index Details**

Key Name: **PRIMARY**

Index Type: **BTREE**

Allows NULL: **NO**

Cardinality: **1**

Comment:

User Comment:

**Columns in table**

| Column              | Type         | Nullable | Indexes              |
|---------------------|--------------|----------|----------------------|
| discount_id         | int          | NO       | PRIMARY              |
| product_id          | int          | YES      | idx_discount_product |
| discount_name       | varchar(100) | NO       |                      |
| discount_percentage | decimal(5,2) | NO       |                      |
| start_date          | datetime     | NO       | idx_discount_dates   |
| end_date            | datetime     | NO       | idx_discount_dates   |
| is_active           | tinyint(1)   | YES      |                      |
| created_at          | timestamp    | YES      |                      |
| updated_at          | timestamp    | YES      |                      |

C. Indexes

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.customers **inventorydb.discounts**

**Info** Columns Indexes Triggers Foreign keys Partitions Grants DDL

Filter objects

**DDL for inventorydb.discounts**

```

1 CREATE TABLE `discounts` (
2   `discount_id` int NOT NULL AUTO_INCREMENT,
3   `product_id` int DEFAULT NULL,
4   `discount_name` varchar(100) NOT NULL,
5   `discount_percentage` decimal(5,2) NOT NULL,
6   `start_date` datetime NOT NULL,
7   `end_date` datetime NOT NULL,
8   `is_active` tinyint(1) DEFAULT '0',
9   `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
10  `updated_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
11  PRIMARY KEY (`discount_id`),
12  KEY `idx_discount_product` (`product_id`),
13  KEY `idx_discount_dates` (`start_date`,`end_date`),
14  CONSTRAINT `discounts_chk_1` FOREIGN KEY (`product_id`) REFERENCES `products` (`product_id`) ON DELETE CASCADE,
15  CONSTRAINT `discounts_chk_2` CHECK ((`discount_percentage` > 0) and (`discount_percentage` <= 100)),
16  CONSTRAINT `discounts_chk_3` CHECK ((`end_date` > `start_date`))
17 ) ENGINE=InnoDB AUTO_INCREMENT=511006 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
  
```

D. Data  
Definition  
Language



## 5. OrderItems

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.cust...

**SCHEMAS**

Filter objects

- inventory\_dw
  - inventorydb
    - auditlog
    - categories
    - customers
    - discounts
    - orderitems
    - orders
    - products
    - stocktransacti
    - suppliers
    - users
    - Views
    - Stored Procedures
    - Functions
    - rechelleborbe
    - sys

**Info** Columns Indexes Triggers Foreign keys Partitions Grants DDL

final project in IM  
**inventorydb.orderitems**

**Table Details**

Engine: **InnoDB**  
 Row format: **Dynamic**  
 Column count: **8**  
 Table rows: **946868**  
 AVG row length: **59**  
 Data length: **53.6 MiB**  
 Index length: **42.1 MiB**  
 Max data length: **0.0 bytes**  
 Data free: **4.0 MiB**  
 Table size (estimate): **95.7 MiB**  
 Update time: **2025-04-28 16:05:50**  
 Create time: **2025-04-27 10:10:05**  
 Auto increment: **1000007**  
 Table collation: **utf8mb4\_0900\_ai\_ci**  
 Create options:  
 Comment:

A. Info

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.customers inventorydb.discounts **inventorydb.orderitems**

**Columns** Indexes Triggers Foreign keys Partitions Grants DDL

| Column          | Type          | Default Value     | Nullable | Character Set | Collation | Privileges                      | Extra          | Comments         |
|-----------------|---------------|-------------------|----------|---------------|-----------|---------------------------------|----------------|------------------|
| order_item_id   | int           |                   | NO       |               |           | select,insert,update,references |                |                  |
| order_id        | int           |                   | YES      |               |           | select,insert,update,references | auto_increment |                  |
| product_id      | int           |                   | YES      |               |           | select,insert,update,references |                |                  |
| quantity        | int           |                   | NO       |               |           | select,insert,update,references |                |                  |
| price           | decimal(10,2) |                   | NO       |               |           | select,insert,update,references |                |                  |
| discount_amount | decimal(10,2) | 0.00              | YES      |               |           | select,insert,update,references |                |                  |
| total_price     | decimal(10,2) |                   | YES      |               |           | select,insert,update,references |                | STORED GENER...  |
| created_at      | timestamp     | CURRENT_TIMESTAMP | YES      |               |           | select,insert,update,references |                | DEFAULT_GENER... |

B. Columns

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.customers inventorydb.discounts **inventorydb.orderitems**

**Indexes** Indexes Triggers Foreign keys Partitions Grants DDL

**Indexes in Table**

| Key                   | Type  | Unique | Columns       |
|-----------------------|-------|--------|---------------|
| PRIMARY               | BTREE | YES    | order_item_id |
| idx_orderitem_order   | BTREE | NO     | order_id      |
| idx_orderitem_product | BTREE | NO     | product_id    |

**Index Details**

Key Name:   
 Index Type:   
 Allow NULL:   
 Cardinality:   
 Comment:   
 User Comment:

**Columns in Table**

| Column          | Type          | Nullable | Indexes               |
|-----------------|---------------|----------|-----------------------|
| order_item_id   | int           | NO       | PRIMARY               |
| order_id        | int           | YES      | idx_orderitem_order   |
| product_id      | int           | YES      | idx_orderitem_product |
| quantity        | int           | NO       |                       |
| price           | decimal(10,2) | NO       |                       |
| discount_amount | decimal(10,2) | YES      |                       |
| total_price     | decimal(10,2) | YES      |                       |
| created_at      | timestamp     | YES      |                       |

C. Indexes

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.customers inventorydb.discounts **inventorydb.orderitems**

**DDL for inventorydb.orderitems**

```

1 CREATE TABLE `orderitems` (
2   `order_item_id` int NOT NULL AUTO_INCREMENT,
3   `order_id` int DEFAULT NULL,
4   `product_id` int DEFAULT NULL,
5   `quantity` int NOT NULL,
6   `price` decimal(10,2) NOT NULL,
7   `discount_amount` decimal(10,2) DEFAULT '0.00',
8   `total_price` decimal(10,2) GENERATED ALWAYS AS (((`price` - `discount_amount`) * `quantity`)) STORED,
9   `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
10  PRIMARY KEY (`order_item_id`),
11  KEY `idx_orderitem_order` (`order_id`),
12  KEY `idx_orderitem_product` (`product_id`),
13  CONSTRAINT `orderitems_ibfk_1` FOREIGN KEY (`order_id`) REFERENCES `orders` (`order_id`) ON DELETE CASCADE,
14  CONSTRAINT `orderitems_ibfk_2` FOREIGN KEY (`product_id`) REFERENCES `products` (`product_id`) ON DELETE SET NULL,
15  CONSTRAINT `orderitems_chk_1` CHECK ((`quantity` > 0)),
16  CONSTRAINT `orderitems_chk_2` CHECK ((`price` > 0)),
17  CONSTRAINT `orderitems_chk_3` CHECK ((`discount_amount` >= 0))
18 ) ENGINE=InnoDB AUTO_INCREMENT=1000007 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
  
```

D. Data Definition Language

## 6. Orders

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows the 'inventorydb' database selected. The 'Tables' list includes 'auditlog', 'categories', 'customers', 'discounts', 'orderitems', 'orders', 'products', 'stocktransact', 'suppliers', 'users', 'views', 'Stored Procedures', 'Functions', 'rechelleborbe', and 'sys'. The 'orders' table is highlighted. The main pane shows the 'Table Details' for 'inventorydb.orders'. The table is an InnoDB Dynamic table with 14 columns, 98917 rows, and a size of 27.1 MiB. The update time is 2025-04-28 16:05:50, and the create time is 2025-04-27 10:10:05. The table collation is utf8mb4\_0900\_ai\_ci.

**Table Details**

- Engine: InnoDB
- Row format: Dynamic
- Column count: 14
- Table rows: 98917
- AVG row length: 164
- Data length: 15.5 MiB
- Index length: 11.6 MiB
- Max data length: 0.0 bytes
- Data free: 5.0 MiB
- Table size (estimate): 27.1 MiB
- Update time: 2025-04-28 16:05:50
- Create time: 2025-04-27 10:10:05
- Auto increment: 1200056
- Table collation: utf8mb4\_0900\_ai\_ci
- Create options:
- Comment:

A. Info

The screenshot shows the 'Columns' tab for the 'inventorydb.orders' table. It lists 14 columns with their data types, default values, and privileges.

| Column          | Type  | Default Value     | Nullable | Character Set | Collation       | Privileges                      | Extra          | Comments          |
|-----------------|---|-------------------|----------|---------------|-----------------|---------------------------------|----------------|-------------------|
| order_id        | int   |                   | NO       |               |                 | select,insert,update,references | auto_increment |                   |
| user_id         | int   |                   | YES      |               |                 | select,insert,update,references |                |                   |
| customer_id     | int   |                   | YES      |               |                 | select,insert,update,references |                |                   |
| order_number    | varchar(20)   |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                |                   |
| order_date      | timestamp   | CURRENT_TIMESTAMP | YES      | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                | DEFAULT_GENERATED |
| status          | enum('Pending','Processing','Shipped','Delivered','Cancelled','Refunded') | Pending           | YES      | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                |                   |
| total_amount    | decimal(12,2)   |                   | NO       |               |                 | select,insert,update,references |                |                   |
| tax_amount      | decimal(10,2)   | 0.00              | YES      |               |                 | select,insert,update,references |                |                   |
| shipping_amount | decimal(10,2)   | 0.00              | YES      |               |                 | select,insert,update,references |                |                   |
| payment_method  | enum('Cash','Credit Card','Debit Card','Bank Transfer','Other')           |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                |                   |
| payment_status  | enum('Pending','Paid','Failed','Refunded')                                | Pending           | YES      | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                |                   |
| notes           | text  |                   | YES      | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                | DEFAULT_GENERATED |
| created_at      | timestamp   | CURRENT_TIMESTAMP | YES      |               |                 | select,insert,update,references |                | DEFAULT_GENERATED |
| updated_at      | timestamp   | CURRENT_TIMESTAMP | YES      |               |                 | select,insert,update,references |                | DEFAULT_GENERATED |

B. Columns

The screenshot shows the 'Indexes' tab for the 'inventorydb.orders' table. It displays the primary key and several unique indexes.

| Key                | Type  | Uniq. | Columns                   |
|--------------------|-------|-------|---------------------------|
| PRIMARY            | BTREE | YES   | order_id                  |
| order_number       | BTREE | YES   | order_number              |
| user_id            | BTREE | NO    | user_id                   |
| idx_order_customer | BTREE | NO    | order_number, customer_id |
| idx_order_status   | BTREE | NO    | status                    |

**Columns in Table**

| Column          | Type  | Nullable | Indexes            |
|-----------------|---|----------|--------------------|
| order_id        | int   | NO       | PRIMARY            |
| user_id         | int   | YES      | order_id           |
| customer_id     | int   | YES      | idx_order_customer |
| order_number    | varchar(20)   | NO       | order_number       |
| order_date      | timestamp   | YES      | idx_order_date     |
| status          | enum('Pending','Processing','Shipped','Delivered','Cancelled','Refunded') | YES      | idx_order_status   |
| total_amount    | decimal(12,2)   | NO       |                    |
| tax_amount      | decimal(10,2)   | YES      |                    |
| shipping_amount | decimal(10,2)   | YES      |                    |
| payment_method  | enum('Cash','Credit Card','Debit Card','Bank Transfer','Other')           | NO       |                    |
| payment_status  | enum('Pending','Paid','Failed','Refunded')                                | YES      |                    |
| notes           | text  | YES      |                    |
| created_at      | timestamp   | YES      |                    |
| updated_at      | timestamp   | YES      |                    |

C. Indexes

The screenshot shows the 'DDL for inventorydb.orders' tab. It displays the SQL code used to create the table.

```
1 CREATE TABLE `orders` (  
2   `order_id` int NOT NULL AUTO_INCREMENT,  
3   `user_id` int DEFAULT NULL,  
4   `customer_id` int DEFAULT NULL,  
5   `order_number` varchar(20) NOT NULL,  
6   `order_date` timestamp NULL DEFAULT CURRENT_TIMESTAMP,  
7   `status` enum('Pending','Processing','Shipped','Delivered','Cancelled','Refunded') DEFAULT 'Pending',  
8   `total_amount` decimal(12,2) NOT NULL,  
9   `tax_amount` decimal(10,2) DEFAULT '0.00',  
10  `shipping_amount` decimal(10,2) DEFAULT '0.00',  
11  `payment_method` enum('Cash','Credit Card','Debit Card','Bank Transfer','Other') NOT NULL,  
12  `payment_status` enum('Pending','Paid','Failed','Refunded') DEFAULT 'Pending',  
13  `notes` text,  
14  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,  
15  `updated_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,  
16  PRIMARY KEY (`order_id`),  
17  UNIQUE KEY `order_number` (`order_number`),  
18  KEY `user_id` (`user_id`),  
19  KEY `idx_order_customer` (`customer_id`),  
20  KEY `idx_order_status` (`status`),  
21  KEY `idx_order_date` (`order_date`),  
22  CONSTRAINT `orders_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES `users` (`user_id`) ON DELETE SET NULL,  
23  CONSTRAINT `orders_ibfk_2` FOREIGN KEY (`customer_id`) REFERENCES `customers` (`customer_id`) ON DELETE SET NULL,  
24  CONSTRAINT `orders_chk_1` CHECK ((`total_amount` >= 0)),  
25  CONSTRAINT `orders_chk_2` CHECK ((`tax_amount` >= 0)),  
26  CONSTRAINT `orders_chk_3` CHECK ((`shipping_amount` >= 0)),  
27 ) ENGINE=InnoDB AUTO_INCREMENT=1200056 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

D. Data Definition Languages

## 7. Products

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.products

SCHEMAS

Filter objects

inventory\_dw

inventorydb

auditlog

categories

customers

discounts

orderitems

orders

products

stocktransacti

suppliers

users

Views

Stored Procedures

Functions

rechelleborbe

sys

Info Columns Indexes Triggers Foreign keys Partitions Grants DDL

final project in IM

inventorydb.products

Table Details

Engine: InnoDB

Row format: Dynamic

Column count: 14

Table rows: 99428

AVG row length: 185

Data length: 17.5 MiB

Index length: 18.6 MiB

Max data length: 0.0 bytes

Data free: 4.0 MiB

Table size (estimate): 36.1 MiB

Update time: 2025-04-27 12:45:35

Create time: 2025-04-27 10:10:05

Auto increment: 100012

Table collation: utf8mb4\_0900\_ai\_ci

Create options:

Comment:

A. info

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.products

SCHEMAS

Filter objects

inventory\_dw

inventorydb

auditlog

categories

customers

discounts

orderitems

orders

products

stocktransacti

suppliers

users

Views

Stored Procedures

Functions

rechelleborbe

sys

Info Columns Indexes Triggers Foreign keys Partitions Grants DDL

| Column          | Type          | Default value     | Nullable | Character Set | Collation       | Privileges                      | Extra            | Comments           |
|-----------------|---------------|-------------------|----------|---------------|-----------------|---------------------------------|------------------|--------------------|
| product_id      | int           |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references | auto_increment   |                    |
| name            | varchar(100)  |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                  |                    |
| description     | text          |                   | YES      | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                  |                    |
| sku             | varchar(50)   |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                  |                    |
| category_id     | int           |                   | YES      |               |                 | select,insert,update,references |                  |                    |
| supplier_id     | int           |                   | YES      |               |                 | select,insert,update,references |                  |                    |
| stock_quantity  | int           | 0                 | YES      |               |                 | select,insert,update,references |                  |                    |
| price           | decimal(10,2) |                   | NO       |               |                 | select,insert,update,references |                  |                    |
| cost_price      | decimal(10,2) |                   | NO       |               |                 | select,insert,update,references |                  |                    |
| reorder_level   | int           | 5                 | YES      |               |                 | select,insert,update,references |                  |                    |
| warranty_period | int           |                   | YES      |               |                 | select,insert,update,references |                  | Warranty in months |
| is_active       | tinyint(1)    |                   | YES      |               |                 | select,insert,update,references |                  |                    |
| created_at      | timestamp     | CURRENT_TIMESTAMP | YES      |               |                 | select,insert,update,references | DEFAULT_GENERATE |                    |
| updated_at      | timestamp     | CURRENT_TIMESTAMP | YES      |               |                 | select,insert,update,references | DEFAULT_GENERATE |                    |

B. Columns

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.products

SCHEMAS

Filter objects

inventory\_dw

inventorydb

auditlog

categories

customers

discounts

orderitems

orders

products

stocktransacti

suppliers

users

Views

Stored Procedures

Functions

rechelleborbe

sys

Info Columns Indexes Triggers Foreign keys Partitions Grants DDL

Indexes in Table

| Key                   | Type  | Unique | Columns     |
|-----------------------|-------|--------|-------------|
| PRIMARY               | BTREE | YES    | product_id  |
| sku                   | BTREE | YES    | sku         |
| idx_product_name      | BTREE | NO     | name        |
| idx_product_sku       | BTREE | NO     | sku         |
| idx_product_catego... | BTREE | NO     | category_id |

Index Details

Key Name:

Index Type:

Allows NULL:

Cardinality:

Comment:

User Comment:

Packed:

Unique:

Columns in table

| Column          | Type          | Nullable | Indexes              |
|-----------------|---------------|----------|----------------------|
| product_id      | int           | NO       | PRIMARY              |
| name            | varchar(100)  | NO       | idx_product_name     |
| description     | text          | YES      |                      |
| sku             | varchar(50)   | NO       | sku, idx_product_sku |
| category_id     | int           | YES      | idx_product_category |
| supplier_id     | int           | YES      | idx_product_supplier |
| stock_quantity  | int           | YES      |                      |
| price           | decimal(10,2) | NO       |                      |
| cost_price      | decimal(10,2) | NO       |                      |
| reorder_level   | int           | YES      |                      |
| warranty_period | int           | YES      |                      |
| is_active       | tinyint(1)    | YES      |                      |
| created_at      | timestamp     | YES      |                      |
| updated_at      | timestamp     | YES      |                      |

C. Indexes

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.products

SCHEMAS

Filter objects

inventory\_dw

inventorydb

auditlog

categories

customers

discounts

orderitems

orders

products

stocktransacti

suppliers

users

Views

Stored Procedures

Functions

rechelleborbe

sys

Info Columns Indexes Triggers Foreign keys Partitions Grants DDL

DDL for inventorydb.products

```
1 CREATE TABLE `products` (  
2   `product_id` int NOT NULL AUTO_INCREMENT,  
3   `name` varchar(100) NOT NULL,  
4   `description` text,  
5   `sku` varchar(50) NOT NULL,  
6   `category_id` int DEFAULT NULL,  
7   `supplier_id` int DEFAULT NULL,  
8   `stock_quantity` int DEFAULT 0,  
9   `price` decimal(10,2) NOT NULL,  
10  `cost_price` decimal(10,2) NOT NULL,  
11  `reorder_level` int DEFAULT 5,  
12  `warranty_period` int DEFAULT NULL COMMENT 'Warranty in months',  
13  `is_active` tinyint(1) DEFAULT 1,  
14  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,  
15  `updated_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,  
16  PRIMARY KEY (`product_id`),  
17  UNIQUE KEY `sku` (`sku`),  
18  KEY `idx_product_name` (`name`),  
19  KEY `idx_product_sku` (`sku`),  
20  KEY `idx_product_category` (`category_id`),  
21  KEY `idx_product_supplier` (`supplier_id`),  
22  CONSTRAINT `products_ibfk_1` FOREIGN KEY (`category_id`) REFERENCES `categories` (`category_id`) ON DELETE SET NULL,  
23  CONSTRAINT `products_ibfk_2` FOREIGN KEY (`supplier_id`) REFERENCES `suppliers` (`supplier_id`) ON DELETE SET NULL,  
24  CONSTRAINT `products_chk_1` CHECK ((`stock_quantity` >= 0)),  
25  CONSTRAINT `products_chk_2` CHECK ((`price` > 0)),  
26  CONSTRAINT `products_chk_3` CHECK ((`cost_price` > 0)),  
27  CONSTRAINT `products_chk_4` CHECK ((`reorder_level` >= 0)),  
28 ) ENGINE=InnoDB AUTO_INCREMENT=100012 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

D. Data  
Definition  
Language



## 8. StockTransaction

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories

**SCHEMAS**

Filter objects

inventory\_dw  
**inventorydb**  
 Tables  
 auditlog  
 categories  
 customers  
 discounts  
 orderitems  
 orders  
 products  
 stocktransacti  
 suppliers  
 users  
 Views  
 Stored Procedures  
 Functions  
 rechelleborbe  
 sys

**Info** Columns Indexes Triggers Foreign keys Partitions Grants DDL

final project in IM  
**inventorydb.stocktransactions**

**Table Details**

Engine: **InnoDB**  
 Row format: **Dynamic**  
 Column count: **9**  
 Table rows: **1089042**  
 AVG row length: **135**  
 Data length: **140.7 MiB**  
 Index length: **85.2 MiB**  
 Max data length: **0.0 bytes**  
 Data free: **5.0 MiB**  
 Table size (estimate): **225.8 MiB**  
 Update time: **2025-04-28 16:06:28**  
 Create time: **2025-04-27 10:10:05**  
 Auto increment: **1200068**  
 Table collation: **utf8mb4\_0900\_ai\_ci**  
 Create options:  
 Comment:

A. info

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.customers inventorydb.discounts inventorydb.orderitems inventorydb.orders inventorydb.products **inventorydb.stocktransactions**

**Columns** Indexes Triggers Foreign keys Partitions Grants DDL

Create a new SQL tab for executing queries

Filter objects

inventory\_dw  
**inventorydb**  
 Tables  
 auditlog  
 categories  
 customers  
 discounts  
 orderitems  
 orders  
 products  
 stocktransacti  
 suppliers  
 users  
 Views  
 Stored Procedures  
 Functions  
 rechelleborbe  
 sys

| Column           | Type                      | Default Value        | Nullable | Character Set | Collation       | Privileges                      | Extra | Comments                               |
|------------------|---------------------------|----------------------|----------|---------------|-----------------|---------------------------------|-------|--|
| transaction_id   | int                       |                      | NO       |               |                 | select,insert,update,references |       |  |
| product_id       | int                       |                      | YES      |               |                 | select,insert,update,references |       | auto_increment                         |
| transaction_type | enum('IN','OUT','ADJ...') |                      | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |       |  |
| quantity         | int                       |                      | NO       |               |                 | select,insert,update,references |       |  |
| reference_id     | varchar(50)               |                      | YES      | utf8mb4       | utf8mb4_0900... | select,insert,update,references |       | Can reference order_id, purchase_id... |
| reference_type   | enum('ORDER','PUR...')    |                      | YES      | utf8mb4       | utf8mb4_0900... | select,insert,update,references |       | Type of reference                      |
| notes            | text                      |                      | YES      | utf8mb4       | utf8mb4_0900... | select,insert,update,references |       |  |
| transaction_date | timestamp                 | CURRENT_TIMESTAMP... | YES      |               |                 | select,insert,update,references |       | DEFAULT_GENERATED                      |
| user_id          | int                       |                      | YES      |               |                 | select,insert,update,references |       |  |

B. Columns

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.customers inventorydb.discounts inventorydb.orderitems inventorydb.orders inventorydb.products **inventorydb.stocktransactions**

**Indexes** Columns Indexes Triggers Foreign keys Partitions Grants DDL

Filter objects

inventory\_dw  
**inventorydb**  
 Tables  
 auditlog  
 categories  
 customers  
 discounts  
 orderitems  
 orders  
 products  
 stocktransacti  
 suppliers  
 users  
 Views  
 Stored Procedures  
 Functions  
 rechelleborbe  
 sys

**Indexes in Table**

| Key               | Type  | Unique | Columns          |
|-------------------|-------|--------|------------------|
| PRIMARY           | BTREE | YES    | transaction_id   |
| idx_stock_product | BTREE | NO     | product_id       |
| idx_stock_date    | BTREE | NO     | transaction_date |
| idx_stock_type    | BTREE | NO     | transaction_type |

**Index Details**

Key Name: PRIMARY  
 Index Type: PRIMARY  
 Allows NULL: NO  
 Cardinality: NO  
 Comment: NO  
 User Comment: NO

**Columns in table**

| Column           | Type  | Nullable | Indexes           |
|------------------|---|----------|-------------------|
| transaction_id   | int   | NO       | PRIMARY           |
| product_id       | int   | YES      | idx_stock_product |
| transaction_type | enum('IN','OUT','ADJUSTM...')                 | NO       | idx_stock_type    |
| quantity         | int   | NO       |                   |
| reference_id     | varchar(50)                                   | YES      |                   |
| reference_type   | enum('ORDER','PURCHASE','ADJUSTMENT','OTHER') | YES      |                   |
| notes            | text  | YES      |                   |
| transaction_date | timestamp                                     | YES      | idx_stock_date    |
| user_id          | int   | YES      |                   |

C. Indexes

Navigator: updated inventorydb inventorydb.auditlog inventorydb.categories inventorydb.customers inventorydb.discounts inventorydb.orderitems inventorydb.orders inventorydb.products **inventorydb.stocktransactions**

**DDL for inventorydb.stocktransactions**

```

1 CREATE TABLE `stocktransactions` (
2   `transaction_id` int NOT NULL AUTO_INCREMENT,
3   `product_id` int DEFAULT NULL,
4   `transaction_type` enum('IN','OUT','ADJUSTMENT','RETURN') NOT NULL,
5   `quantity` int NOT NULL,
6   `reference_id` varchar(50) DEFAULT NULL COMMENT 'Can reference order_id, purchase_id, etc.',
7   `reference_type` enum('ORDER','PURCHASE','ADJUSTMENT','OTHER') DEFAULT NULL COMMENT 'Type of reference',
8   `notes` text,
9   `transaction_date` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
10  `user_id` int DEFAULT NULL,
11  PRIMARY KEY (`transaction_id`),
12  KEY `user_id` (`user_id`),
13  KEY `idx_stock_product` (`product_id`),
14  KEY `idx_stock_date` (`transaction_date`),
15  KEY `idx_stock_type` (`transaction_type`),
16  CONSTRAINT `stocktransactions_ibfk_1` FOREIGN KEY (`product_id`) REFERENCES `products` (`product_id`) ON DELETE CASCADE,
17  CONSTRAINT `stocktransactions_ibfk_2` FOREIGN KEY (`user_id`) REFERENCES `users` (`user_id`) ON DELETE SET NULL,
18  CONSTRAINT `stocktransactions_chk_1` CHECK ((`quantity` > 0)),
19 ) ENGINE=InnoDB AUTO_INCREMENT=1200068 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
  
```

D. Data  
 Definon  
 Language

## 9. Suppliers

Navigator: ted inventorydb inventorydb.auditlog inventorydb.categories ii

SCHEMAS

Filter objects

inventory\_dw

inventorydb

Tables

auditlog

categories

customers

discounts

orderitems

orders

products

stocktransacti

suppliers

users

Views

Stored Procedures

Functions

rechelleborbe

sys

Info Columns Indexes Triggers Foreign keys Partitions Grants DC

final project in IM

inventorydb.suppliers

Table Details

Engine: InnoDB

Row format: Dynamic

Column count: 11

Table rows: 503735

AVG row length: 192

Data length: 92.6 MiB

Index length: 40.1 MiB

Max data length: 0.0 bytes

Data free: 7.0 MiB

Table size (estimate): 132.7 MiB

Update time: 2025-04-28 16:30:48

Create time: 2025-04-27 10:10:05

Auto increment: 511006

Table collation: utf8mb4\_0900\_ai\_ci

Create options:

Comment:

A. info

Navigator: ted inventorydb inventorydb.auditlog inventorydb.categories inventorydb.customers inventorydb.discounts inventorydb.orderitems inventorydb.orders inventorydb.products inventorydb.stocktransactions inventorydb.auditlog

SCHEMAS

Filter objects

inventory\_dw

inventorydb

Tables

auditlog

categories

customers

discounts

orderitems

orders

products

stocktransacti

suppliers

users

Views

Stored Procedures

Functions

rechelleborbe

sys

Info Columns Indexes Triggers Foreign keys Partitions Grants DDL

| Column      | Type         | Default Value     | Nullable | Character Set | Collation          | Privileges                      | Extra          | Comments          |
|-------------|--------------|-------------------|----------|---------------|--------------------|---------------------------------|----------------|-------------------|
| supplier_id | int          |                   | NO       | utf8mb4       | utf8mb4_0900_ai_ci | select,insert,update,references | auto_increment |                   |
| name        | varchar(100) |                   | NO       | utf8mb4       | utf8mb4_0900_ai_ci | select,insert,update,references |                |                   |
| contact     | varchar(50)  |                   | NO       | utf8mb4       | utf8mb4_0900_ai_ci | select,insert,update,references |                |                   |
| email       | varchar(100) |                   | YES      | utf8mb4       | utf8mb4_0900_ai_ci | select,insert,update,references |                |                   |
| phone       | varchar(20)  |                   | NO       | utf8mb4       | utf8mb4_0900_ai_ci | select,insert,update,references |                |                   |
| address     | text         |                   | NO       | utf8mb4       | utf8mb4_0900_ai_ci | select,insert,update,references |                |                   |
| city        | varchar(50)  |                   | NO       | utf8mb4       | utf8mb4_0900_ai_ci | select,insert,update,references |                |                   |
| country     | varchar(50)  |                   | NO       | utf8mb4       | utf8mb4_0900_ai_ci | select,insert,update,references |                |                   |
| postal_code | varchar(20)  |                   | YES      | utf8mb4       | utf8mb4_0900_ai_ci | select,insert,update,references |                |                   |
| created_at  | timestamp    | CURRENT_TIMESTAMP | YES      | utf8mb4       | utf8mb4_0900_ai_ci | select,insert,update,references |                | DEFAULT_GENERATED |
| updated_at  | timestamp    | CURRENT_TIMESTAMP | YES      | utf8mb4       | utf8mb4_0900_ai_ci | select,insert,update,references |                | DEFAULT_GENERATED |

B. Columns

Navigator: ted inventorydb inventorydb.auditlog inventorydb.categories inventorydb.customers inventorydb.discounts inventorydb.orderitems inventorydb.orders inventorydb.products inventorydb.stocktransactions inventorydb.auditlog

SCHEMAS

Filter objects

inventory\_dw

inventorydb

Tables

auditlog

categories

customers

discounts

orderitems

orders

products

stocktransacti

suppliers

users

Views

Stored Procedures

Functions

rechelleborbe

sys

Info Columns Indexes Triggers Foreign keys Partitions Grants DDL

Indexes in Table

| Key                  | Type  | Uniqueness | Columns     |
|----------------------|-------|------------|-------------|
| PRIMARY              | BTREE | YES        | supplier_id |
| idx_supplier_name    | BTREE | NO         | name        |
| idx_supplier_contact | BTREE | NO         | contact     |

Index Details

Key Name:

Index Type:

Allows NULL:

Cardinality:

Comment:

User Comment:

Packed: Unique:

Columns in table

| Column      | Type         | Nullable | Indexes              |
|-------------|--------------|----------|----------------------|
| supplier_id | int          | NO       | PRIMARY              |
| name        | varchar(100) | NO       | idx_supplier_name    |
| contact     | varchar(50)  | NO       | idx_supplier_contact |
| email       | varchar(100) | YES      |                      |
| phone       | varchar(20)  | NO       |                      |
| address     | text         | NO       |                      |
| city        | varchar(50)  | NO       |                      |
| country     | varchar(50)  | NO       |                      |
| postal_code | varchar(20)  | YES      |                      |
| created_at  | timestamp    | YES      |                      |
| updated_at  | timestamp    | YES      |                      |

C. Indexes

Navigator: ted inventorydb inventorydb.auditlog inventorydb.categories inventorydb.customers inventorydb.discounts inventorydb.orderitems

SCHEMAS

Filter objects

inventory\_dw

inventorydb

Tables

auditlog

categories

customers

discounts

orderitems

orders

products

stocktransacti

suppliers

users

Views

Stored Procedures

Functions

rechelleborbe

sys

Info Columns Indexes Triggers Foreign keys Partitions Grants DDL

DDL for inventorydb.suppliers

```

1 CREATE TABLE `suppliers` (
2   `supplier_id` int NOT NULL AUTO_INCREMENT,
3   `name` varchar(100) NOT NULL,
4   `contact` varchar(50) NOT NULL,
5   `email` varchar(100) DEFAULT NULL,
6   `phone` varchar(20) NOT NULL,
7   `address` text NOT NULL,
8   `city` varchar(50) NOT NULL,
9   `country` varchar(50) NOT NULL,
10  `postal_code` varchar(20) DEFAULT NULL,
11  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
12  `updated_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
13  PRIMARY KEY (`supplier_id`),
14  KEY `idx_supplier_name` (`name`),
15  KEY `idx_supplier_contact` (`contact`),
16  CONSTRAINT `suppliers_chk_1` CHECK ((`email` like _utf8mb4'%%@%.'))
17 ) ENGINE=InnoDB AUTO_INCREMENT=511006 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci

```

D. Data  
Definition  
Languages

## 10. Users

The screenshot shows the 'inventorydb.users' table in the 'inventorydb' database. The 'Table Details' tab is selected, displaying the following information:

- Engine: InnoDB
- Row format: Dynamic
- Column count: 7
- Table rows: 100641
- AVG row length: 88
- Data length: 8.5 MiB
- Index length: 15.0 MiB
- Max data length: 0.0 bytes
- Data free: 4.0 MiB
- Table size (estimate): 23.6 MiB
- Update time: 2025-04-28 16:31:34
- Create time: 2025-04-27 10:10:05
- Auto increment: 502006
- Table collation: utf8mb4\_0900\_ai\_ci
- Create options:
- Comment:

A. info

The screenshot shows the 'Columns' tab for the 'inventorydb.users' table. The columns are listed in a table with the following details:

| Column     | Type                    | Default Value     | Nullable | Character Set | Collation       | Privileges                      | Extra          | Comments          |
|------------|-------------------------|-------------------|----------|---------------|-----------------|---------------------------------|----------------|-------------------|
| user_id    | int                     |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references | auto_increment |                   |
| username   | varchar(50)             |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                |                   |
| password   | varchar(255)            |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                |                   |
| email      | varchar(100)            |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                |                   |
| role       | enum('admin','mana...') |                   | NO       | utf8mb4       | utf8mb4_0900... | select,insert,update,references |                |                   |
| created_at | timestamp               | CURRENT_TIMESTAMP | YES      |               |                 | select,insert,update,references |                | DEFAULT_GENERATED |
| updated_at | timestamp               | CURRENT_TIMESTAMP | YES      |               |                 | select,insert,update,references |                | DEFAULT_GENERATED |

B. Columns

The screenshot shows the 'Indexes in Table' tab for the 'inventorydb.users' table. The indexes are listed in a table with the following details:

| Key          | Type  | Unique | Columns  | Index Name | Index Type | Allows NULL | Cardinality | Comments       |
|--------------|-------|--------|----------|------------|------------|-------------|-------------|----------------|
| PRIMARY      | BTREE | YES    | user_id  |            | PRIMARY    |             |             |                |
| username     | BTREE | YES    | username |            |            |             |             | Packed; Unique |
| email        | BTREE | YES    | email    |            |            |             |             |                |
| idx_username | BTREE | NO     | username |            |            |             |             |                |
| idx_role     | BTREE | NO     | role     |            |            |             |             |                |

Columns in table:

| Column     | Type                            | Nullable | Indexes                |
|------------|---------------------------------|----------|------------------------|
| user_id    | int                             | NO       | PRIMARY                |
| username   | varchar(50)                     | NO       | username, idx_username |
| password   | varchar(255)                    | NO       |                        |
| email      | varchar(100)                    | NO       | email                  |
| role       | enum('admin','manager','staff') | NO       | idx_role               |
| created_at | timestamp                       | YES      |                        |
| updated_at | timestamp                       | YES      |                        |

C. Indexes

The screenshot shows the 'DDL for inventorydb.users' tab. The DDL script is as follows:

```

1 CREATE TABLE `users` (
2   `user_id` int NOT NULL AUTO_INCREMENT,
3   `username` varchar(50) NOT NULL,
4   `password` varchar(255) NOT NULL,
5   `email` varchar(100) NOT NULL,
6   `role` enum('admin','manager','staff') NOT NULL,
7   `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
8   `updated_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
9   PRIMARY KEY (`user_id`),
10  UNIQUE KEY `username` (`username`),
11  UNIQUE KEY `email` (`email`),
12  KEY `idx_username` (`username`),
13  KEY `idx_role` (`role`),
14  CONSTRAINT `users_chk_1` CHECK ((length(`password`) >= 8)),
15  CONSTRAINT `users_chk_2` CHECK (('email' like _utf8mb4'%%@%%')),
16 ) ENGINE=InnoDB AUTO_INCREMENT=502006 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci

```

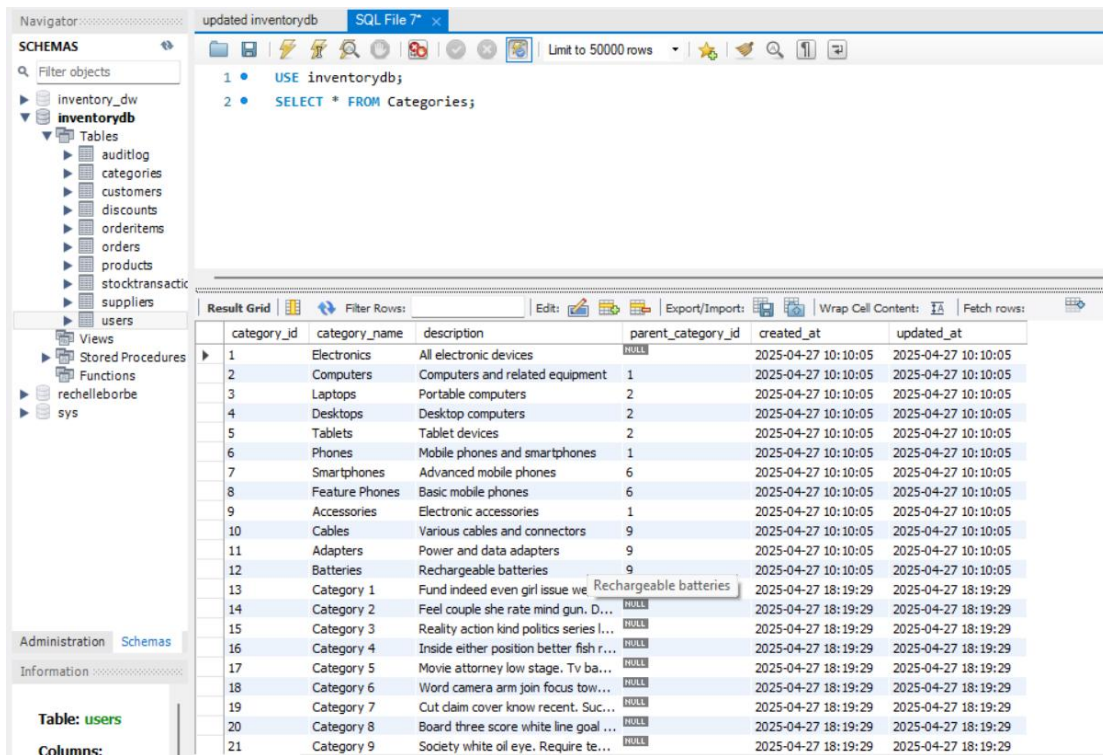
D. Data Definition Languages

## ➤ Sample SQL Queries(basic, BI, optimized)

### 1. Basic

- Retrieve all categories

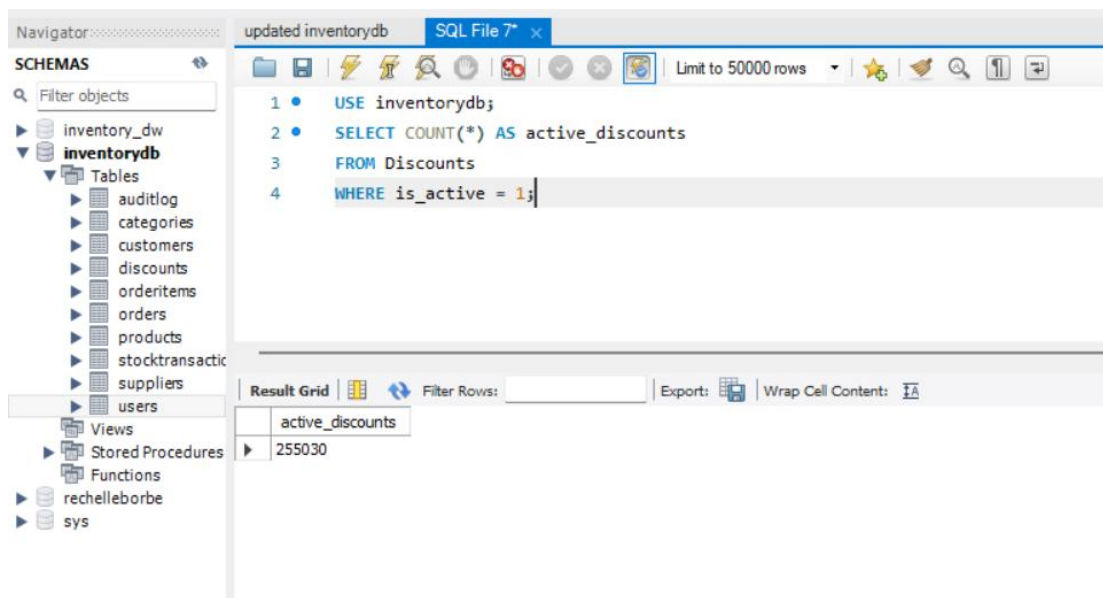
```
SELECT * FROM Categories;
```



| category_id | category_name  | description                              | parent_category_id     | created_at          | updated_at          |
|-------------|----------------|--|------------------------|---------------------|---------------------|
| 1           | Electronics    | All electronic devices                   | NULL                   | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 2           | Computers      | Computers and related equipment          | 1                      | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 3           | Laptops        | Portable computers                       | 2                      | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 4           | Desktops       | Desktop computers                        | 2                      | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 5           | Tablets        | Tablet devices                           | 2                      | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 6           | Phones         | Mobile phones and smartphones            | 1                      | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 7           | Smartphones    | Advanced mobile phones                   | 6                      | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 8           | Feature Phones | Basic mobile phones                      | 6                      | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 9           | Accessories    | Electronic accessories                   | 1                      | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 10          | Cables         | Various cables and connectors            | 9                      | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 11          | Adapters       | Power and data adapters                  | 9                      | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 12          | Batteries      | Rechargeable batteries                   | 9                      | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 13          | Category 1     | Fund indeed even girl issue we           | Rechargeable batteries | 2025-04-27 18:19:29 | 2025-04-27 18:19:29 |
| 14          | Category 2     | Feel couple she rate mind gun. D...      | NULL                   | 2025-04-27 18:19:29 | 2025-04-27 18:19:29 |
| 15          | Category 3     | Reality action kind politics series l... | NULL                   | 2025-04-27 18:19:29 | 2025-04-27 18:19:29 |
| 16          | Category 4     | Inside either position better fish r...  | NULL                   | 2025-04-27 18:19:29 | 2025-04-27 18:19:29 |
| 17          | Category 5     | Movie attorney low stage. Tv ba...       | NULL                   | 2025-04-27 18:19:29 | 2025-04-27 18:19:29 |
| 18          | Category 6     | Word camera arm join focus tow...        | NULL                   | 2025-04-27 18:19:29 | 2025-04-27 18:19:29 |
| 19          | Category 7     | Cut claim cover know recent. Suc...      | NULL                   | 2025-04-27 18:19:29 | 2025-04-27 18:19:29 |
| 20          | Category 8     | Board three score white line goal ...    | NULL                   | 2025-04-27 18:19:29 | 2025-04-27 18:19:29 |
| 21          | Category 9     | Society white oil eye. Require te...     | NULL                   | 2025-04-27 18:19:29 | 2025-04-27 18:19:29 |

- Count the number of active discounts

```
SELECT COUNT(*) AS active_discounts  
FROM Discounts  
WHERE is_active = 1;
```



| active_discounts |
|------------------|
| 255030           |



- List all suppliers in a specific city

```
SELECT name, email, phone
FROM Suppliers
WHERE city = 'New York';
```

The screenshot shows the SQL Developer interface. On the left, the 'SCHEMAS' pane shows the 'inventory\_db' schema with tables like 'suppliers'. The main window displays the following SQL query:

```
1 • USE inventorydb;
2 • SELECT name, email, phone
3 FROM Suppliers
4 WHERE city = 'New York';
```

Below the query, the 'Result Grid' shows the results of the query:

| name         | email                 | phone      |
|--------------|-----------------------|------------|
| Gadget World | sarah@gadgetworld.com | 0987654321 |

## 2. Business Intelligence

- Top 5 Categories by number of Discounts

```
SELECT c.category_name, COUNT(d.discount_id) AS discount_count
FROM Discounts d
JOIN Products p ON d.product_id = p.product_id
JOIN Categories c ON p.category_id = c.category_id
GROUP BY c.category_name
ORDER BY discount_count DESC
LIMIT 5;
```

The screenshot shows the SQL Developer interface. The main window displays the following SQL query:

```
1 • USE inventorydb;
2 • SELECT c.category_name, COUNT(d.discount_id) AS discount_count
3 FROM Discounts d
4 JOIN Products p ON d.product_id = p.product_id
5 JOIN Categories c ON p.category_id = c.category_id
6 GROUP BY c.category_name
7 ORDER BY discount_count DESC
8 LIMIT 5;
```

Below the query, the 'Result Grid' shows the results of the query:

| category_name  | discount_count |
|----------------|----------------|
| Feature Phones | 66707          |
| Desktops       | 56435          |
| Phones         | 56119          |
| Electronics    | 51084          |
| Batteries      | 51034          |

- Total Discounts By Months

```
SELECT
MONTH(start_date) AS discount_month,
SUM(discount_percentage) AS total_discount_percentage
FROM Discounts
GROUP BY discount_month
ORDER BY discount_month;
```

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'SCHEMAS' tree with 'inventorydb' selected. The right pane shows a query window with the following SQL code:

```
1 • USE inventorydb;
2 • SELECT
3     MONTH(start_date) AS discount_month,
4     SUM(discount_percentage) AS total_discount_percentage
5 FROM Discounts
6 GROUP BY discount_month
7 ORDER BY discount_month;
```

Below the query window, the 'Result Grid' shows the following data:

| discount_month | total_discount_percentage |
|----------------|---------------------------|
| 1              | 3436811.13                |
| 2              | 3096952.41                |
| 3              | 3421422.11                |
| 4              | 3051580.97                |
| 6              | 15.00                     |
| 7              | 30.00                     |
| 8              | 10.00                     |
| 9              | 15.00                     |
| 12             | 20.00                     |

- Number of users by roles

```
SELECT role, COUNT(*) AS user_count
FROM Users
GROUP BY role;
```

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'SCHEMAS' tree with 'inventorydb' selected. The right pane shows a query window with the following SQL code:

```
1 • USE inventorydb;
2 • SELECT role, COUNT(*) AS user_count
3 FROM Users
4 GROUP BY role;
```

Below the query window, the 'Result Grid' shows the following data:

| role    | user_count |
|---------|------------|
| admin   | 33595      |
| manager | 33277      |
| staff   | 33372      |

### 3. Optimized SQL queries

- Retrieve Active Discounts with index optimization

```
SELECT discount_name, discount_percentage, start_date, end_date
FROM Discounts
WHERE is_active = 1;
```

The screenshot shows a SQL query window with the following text:

```
1 • USE inventorydb;
2 • SELECT discount_name, discount_percentage, start_date, end_date
3 • FROM Discounts
4 • WHERE is_active = 1;
```

Below the query, the 'Result Grid' displays the following data:

| discount_name | discount_percentage | start_date          | end_date            |
|---------------|---------------------|---------------------|---------------------|
| Discount 1    | 21.49               | 2025-01-06 06:30:37 | 2025-01-26 06:30:37 |
| Discount 4    | 38.50               | 2025-03-13 22:58:25 | 2025-04-05 22:58:25 |
| Discount 5    | 1.94                | 2025-02-18 09:48:36 | 2025-03-02 09:48:36 |
| Discount 7    | 43.23               | 2025-03-31 07:58:31 | 2025-04-24 07:58:31 |
| Discount 8    | 24.00               | 2025-02-05 04:58:25 | 2025-02-07 04:58:25 |
| Discount 9    | 5.42                | 2025-03-09 23:07:45 | 2025-03-21 23:07:45 |
| Discount 10   | 14.26               | 2025-02-23 17:37:47 | 2025-03-08 17:37:47 |
| Discount 14   | 48.54               | 2025-04-09 03:55:22 | 2025-05-05 03:55:22 |
| Discount 15   | 3.20                | 2025-03-13 18:12:36 | 2025-03-19 18:12:36 |
| Discount 16   | 31.12               | 2025-03-10 12:47:20 | 2025-04-05 12:47:20 |
| Discount 18   | 19.11               | 2025-01-26 01:16:15 | 2025-02-02 01:16:15 |
| Discount 20   | 41.09               | 2025-03-13 17:24:32 | 2025-03-25 17:24:32 |
| Discount 21   | 10.50               | 2025-03-14 22:10:42 | 2025-03-19 22:10:42 |
| Discount 23   | 35.93               | 2025-03-24 16:15:22 | 2025-04-06 16:15:22 |
| Discount 24   | 27.51               | 2025-04-09 04:53:49 | 2025-04-10 04:53:49 |
| Discount 25   | 3.71                | 2025-02-03 21:46:48 | 2025-02-16 21:46:48 |
| Discount 26   | 5.94                | 2025-01-07 12:31:44 | 2025-01-21 12:31:44 |
| Discount 27   | 6.54                | 2025-03-22 07:31:16 | 2025-03-28 07:31:16 |
| Discount 30   | 2.71                | 2025-03-06 13:47:46 | 2025-03-20 13:47:46 |
| Discount 38   | 34.61               | 2025-03-11 03:24:05 | 2025-03-26 03:24:05 |
| Discount 39   | 43.87               | 2025-02-24 08:44:34 | 2025-03-03 08:44:34 |

- Retrieve suppliers with unique email

```
SELECT name, email, city
FROM Suppliers
WHERE email LIKE '%@gmail.com';
```

The screenshot shows a SQL query window with the following text:

```
1 • USE inventorydb;
2 • SELECT name, email, city
3 • FROM Suppliers
4 • WHERE email LIKE '%@gmail.com';
```

Below the query, the 'Result Grid' displays the following data:

| name | email | city |
|------|-------|------|
|------|-------|------|

- Aggregate Discounts by products

```
SELECT product_id, COUNT(*) AS discount_count, AVG(discount_percentage) AS
avg_discount
FROM Discounts
GROUP BY product_id
ORDER BY discount_count DESC;
```

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'inventorydb' schema with various tables and views. The central pane shows a SQL query being executed. The right pane displays the results of the query in a grid format.

**Query:**

```
1 USE inventorydb;
2 SELECT product_id, COUNT(*) AS discount_count, AVG(discount_percentage) AS
3 FROM Discounts
4 GROUP BY product_id
5 ORDER BY discount_count DESC;
```

**Result Grid:**

| product_id | discount_count | avg_discount |
|------------|----------------|--------------|
| 86         | 5263           | 25.592221    |
| 47         | 5244           | 25.113863    |
| 30         | 5233           | 25.540397    |
| 4          | 5225           | 25.526408    |
| 33         | 5225           | 25.329476    |
| 94         | 5223           | 25.436399    |
| 37         | 5220           | 25.182590    |
| 16         | 5218           | 25.716763    |
| 52         | 5202           | 25.353822    |
| 22         | 5197           | 25.580654    |
| 25         | 5196           | 25.747492    |
| 99         | 5195           | 25.722054    |
| 65         | 5187           | 25.395552    |
| 12         | 5181           | 25.422212    |
| 62         | 5176           | 25.331119    |
| 64         | 5174           | 25.568320    |
| 70         | 5174           | 25.485101    |
| 9          | 5173           | 25.162018    |
| 56         | 5170           | 25.124217    |
| 14         | 5169           | 25.161803    |
| 27         | 5169           | 25.340795    |

The bottom left pane shows the 'users' table structure:

**Table: users**

**Columns:**

- user\_id (int)
- username (varchar)



## ➤ Backup and Recovery Steps

Backup and Restore Step: Data Export and Import using MySQL Workbench

1. Open MySQL Workbench
2. Click the database connection to log in.
3. In the top menu, go to:

Server > Data Export

1. On the left, select the database.
2. On the right:

o Under Export Options, choose: Export to Dump Project Folder  
o Click Browse and select where to save the folder  
o May leave the rest of the settings as default.

1. Click Start Export

This will create a folder (e.g., database\_name\_dump) containing .sql files — one for each table. Data Import (Restore) using MySQL Workbench

1. In MySQL Workbench, go to: Server > Data Import
2. Under Import Options, choose: Import from Dump Project Folder
3. Click Browse and select the folder exported earlier.
4. Under Default Target Schema:

o If your original database still exists, select it. o If not, click "New" to create a new schema (database) with the same name.

1. Check the box next to schema in the list.
2. Click Start Import

Once done, tables should be restored. To Verify the Import Go to the Schemas panel: • Right-click the schema > Refresh All • Open a table and run: SELECT COUNT(\*) FROM the table; Compare with the original row counts to make sure it restored correctly.

## ➤ Indexing and Performance Improvements

When working with large datasets, such as the 100,000 records generated and inserted into the `inventorydb` database, performance optimization becomes critical. One of the most effective ways to improve database performance is through indexing. Indexing allows the database to retrieve data more efficiently by creating a structured reference for specific columns. In this project, we implemented indexing strategies to optimize query performance and ensure the database could handle large-scale operations effectively.

### Challenges with Performance

Initially, the database faced performance bottlenecks due to the large volume of data. Queries that filtered or joined tables, such as retrieving active discounts or finding suppliers by email, were slow because the database had to perform full table scans. This was especially problematic for tables like Discounts and Users, which contained 100,000 records each. Additionally, ensuring data integrity and avoiding duplicate entries for fields like username and email required careful consideration.

### Indexing Strategies

To address these challenges, we added indexes to key columns in the database:

**Primary Keys-** Each table had a primary key to uniquely identify records. For example, `category_name` in the Categories table and `username` in the Users table were set as primary keys. This ensured fast lookups for unique records.

**Foreign Key Indexes-** In the Discounts table, we added an index on `product_id` to optimize joins with the Products table. This improved the performance of queries that linked discounts to specific products.

**Filtering Indexes-** We created an index on the `is_active` column in the Discounts table. This significantly improved the performance of queries that filtered active discounts, as the database could use the index to locate matching rows directly.

**Unique Indexes-** To ensure data integrity, we added unique indexes on the `email` column in both the Users and Suppliers tables. This not only prevented duplicate entries but also sped up queries that searched for records by email.

### Performance Improvements

The impact of these indexing strategies was measurable and significant. For example:

- A query to retrieve all active discounts (`is_active = 1`) initially required a full table scan, taking several seconds to execute. After adding an index on `is_active`, the query execution time was reduced to milliseconds.
- Joins between the Discounts and Products tables became faster due to the index on `product_id`, allowing the database to locate matching rows efficiently.

- Searching for suppliers or users by email became instantaneous, thanks to the unique indexes on the `email` column.

To verify these improvements, we used the `EXPLAIN` command in MySQL to analyze query execution plans. Before indexing, the `EXPLAIN` output showed full table scans (`type = ALL`), indicating inefficient queries. After indexing, the output showed the use of indexes (`type = ref` or `type = index`), confirming that the database was leveraging the indexes to optimize performance.

## Additional Optimizations

In addition to indexing, we implemented batch processing in the Python script to handle large data inserts. Instead of inserting all 100,000 records at once, we divided the data into smaller batches of 1,000 records. This reduced the load on the database and prevented timeouts or connection issues. We also ensured that generated data, such as usernames and emails, was unique by using sets to track existing values.

Indexing played a important role in improving the performance of the `inventorydb` database. By carefully selecting columns for indexing and leveraging unique constraints, we were able to optimize query execution times and ensure data integrity. These improvements made the database more efficient and scalable, capable of handling large datasets without performance degradation. This experience reinforced the importance of indexing and query optimization in database design, and we plan to apply these strategies to future projects.

### ➤ Role Assignments and contribution summary

#### Project lead: Rechelle Borbe

-As the leader, I initiated the project by adding all team members as collaborators to our shared repository. I ensured that everyone had access to the project files and could contribute effectively. I also facilitated discussions to assign roles and responsibilities, ensuring that each team member understood their tasks and deadlines. I worked closely with our backend developer, who provided the initial code that served as a reference for creating the database schema. I attempted to use an SQL schema script to define the database structure and generate data. However, I later discovered that using a Python script with the Faker library was more efficient for generating large datasets. I took the lead in learning how to install Python, connect the `inventorydb` database to Python, and integrate Visual Studio Code with MySQL Workbench. Despite being new to this process, I successfully connected the database to Python and implemented the Faker library to generate 100,000+ records per table. As the leader, I initiated the project by adding all team members as collaborators to our shared repository. I ensured the deployment and Star Schema, I led the development of Business Intelligence (BI)-oriented queries and the design of a star schema for data analysis. I collaborated with team members to create meaningful BI queries, such as identifying top-performing categories and analyzing user roles. I ensured that the star schema was designed to facilitate efficient querying and reporting. I coordinated the implementation of MySQL backup and recovery processes and the deployment of the database to the cloud with our SQL developer and backend developer. I ensured that

backups were created and tested for reliability and worked with the team to deploy the database to a cloud platform, ensuring accessibility and scalability. I ensured that all optimizations were documented and that the database was performing at its best. As the leader, I took charge of reviewing all project requirements to ensure everything was ready for submission. I worked closely with the team to verify that all deliverables were complete, accurate, and submitted on time. This included reviewing the database schema, Python scripts, BI queries, and documentation.

#### **Database architect: Mhelarry Valeza**

-In our Inventory Management System project, I worked as the database architect, where I was responsible for designing the database structure to ensure it was well-organized and efficient. I helped build features that made it easy to import and export data in and out of the system. When errors arose, I helped troubleshoot and fix issues to ensure the system remained reliable and functional. I also took on the task of editing and updating the data within the system. This included making necessary help in data displays and visualizations, such as graphs. I worked closely with the development team to make sure the database met the project's needs and expectations. In the process, I was able to hugely help our leader by providing reliable solutions and ensuring the system ran smoothly.

#### **Database Architecture :Curt Justin Reodique**

-As a member of our team, I actively contributed in several ways to ensure the success of our group tasks. I helped input important data into our project documents and assisted in gathering and submitting screenshot for each activity deliverable. One of my key roles was writing the reflection for Week 14, which gave me a chance to summarize our progress and learning. I ensured that the star schema was designed to facilitate efficient querying and reporting. I also kept our team updated by sharing relevant news and served as a support member for our database-related work. Through this experience, I gained valuable knowledge that I wouldn't have learned on my own, especially with the guidance and support of our team leader. I also played a part in encouraging and motivating my group mates, making sure we stayed on track and worked together as a team.

#### **Backend Developer: Susaine Rico**

-In our Inventory Management System project for Information Management, my main role was as a back-end developer. I worked on handling the data by creating features that allow importing and exporting of data to and from the system. I also created the dump file to help save and restore the database easily. adding indexes to improve query performance, and implementing batch processing in the Python script to handle large data inserts efficiently. The original codes used in building the system came from me, and I made sure everything in the back-end works properly so that the system runs smoothly.



**SQL Developer: Faith Sañado**

-As a member of the team, I contributed by helping with the concept and structure of the database during the planning phase. I assisted in testing the tables and data to ensure everything was working correctly. I led the team in diagnosing and optimizing the database for performance. This involved analyzing query execution plans using the EXPLAIN command. I also did my best to help solve problems whenever we encountered errors, and tried my best to support them by doing research, watching tutorials, and looking for helpful information online. Aside from offering suggestions when needed, I worked hard to resolve issues related to connecting our database to Railway until we successfully established a connection. Most importantly, I remained committed to providing help to the team whenever it was needed.

**QA Tester: Shaine Sanjuan**

- As the QA Tester for our database project, I was responsible for thoroughly testing the SQL scripts and database queries to ensure they met the required specifications. I conducted multiple test runs to validate data integrity, enforce constraints, and confirm that transactions and queries performed as expected under various scenarios. Through trial and error, I experienced some difficulties with our code but rather than that we made a solution to fix and optimize it. Additionally, I wrote a reflection paper based on my testing experience. This paper outlines the challenges I encountered, the debugging strategies I used, and the key insights I gained about developing a reliable and effective database system. The reflection highlights how iterative testing helped improve the overall quality and performance of our database.

### ➤ Screenshot of output and Dashboard

The image shows a VS Code editor with a Python file named `generate_large_dataset.py` open. The script is located at `C:\Users\labab\generate_large_dataset.py`. The code defines a function `insert_data` to insert data into a database table and a `main` function to test the database connection and generate data. The terminal output shows the script being executed successfully, with messages for each table generation step.

```

72     return users
73
74     # Insert data into the database
75     def insert_data(table, data, query):
76         try:
77             cursor.executemany(query, data)
78             db.commit()
79             print(f"Inserted {len(data)} records into {table}")
80         except mysql.connector.Error as err:
81             print(f"Error inserting into {table}: {err}")
82
83     # Main function
84     def main():
85         # Test the database connection
86         try:
87             cursor.execute("SELECT DATABASE();")
88             current_db = cursor.fetchone()
89             print(f"Connected to database: {current_db[0]}")
90         except mysql.connector.Error as err:
91             print(f"Error: {err}")
92         return
93
94     # Generate and insert categories
95     print("Generating categories...")

```

The terminal output shows the script being executed successfully, with messages for each table generation step:

```

PS C:\Users\labab> cd C:\Users\labab
PS C:\Users\labab> python generate_large_dataset.py
Connected to database: inventorydb
Generating categories...
Inserted 100000 records into Categories
Generating discounts...
Inserted 100000 records into Discounts
Generating suppliers...
Inserted 100000 records into Suppliers
Generating users...
Inserted 100000 records into Users
PS C:\Users\labab>

```

- **4 Basic Queries:** To demonstrate basic functionality.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'SCHEMAS' tree with 'inventorydb' selected. The right pane shows a query window with the following SQL code:

```
1 • USE inventorydb;
2 • SELECT username, email FROM Users WHERE role = 'admin';
```

The 'Result Grid' at the bottom displays the following data:

| username        | email                      |
|-----------------|----------------------------|
| admin1          | admin1@inventory.com       |
| widchoa         | mary92@example.net         |
| justinolson     | asaron28@example.org       |
| kennethschaefer | smithjustin@example.org    |
| carolynbooker   | shermanyan@example.net     |
| jerrythompson   | kaylawilliams@example.com  |
| smiller         | sabrinalee@example.com     |
| igreer          | isa31@example.org          |
| oliviajohnson   | davidsmith@example.net     |
| ktodd           | joshuagriffin@example.com  |
| logan46         | xwashington@example.net    |
| connertravis    | ortizdavid@example.net     |
| grodriquez      | josephamold@example.net    |
| armstrongrick   | campbelljocelyn@example... |
| zacharyfuentes  | anthony66@example.net      |
| samantha02      | schmittrobert@example.net  |
| lchavez         | mitchellsierra@example.com |
| rileyfred       | nchavez@example.org        |
| dana69          | jeremiahgarcia@example.net |
| eduardobedder   | jonesdavid@example.org     |
| nguyenricardo   | brandon75@example.net      |
| youngjohn       | joppe@example.org          |

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'SCHEMAS' tree with 'inventorydb' selected. The right pane shows a query window with the following SQL code:

```
1 • USE inventorydb;
2 • SELECT * FROM Categories LIMIT 10;
```

The 'Result Grid' at the bottom displays the following data:

| category_id | category_name  | description                     | parent_category_id | created_at          | updated_at          |
|-------------|----------------|---------------------------------|--------------------|---------------------|---------------------|
| 1           | Electronics    | All electronic devices          |                    | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 2           | Computers      | Computers and related equipment | 1                  | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 3           | Laptops        | Portable computers              | 2                  | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 4           | Desktops       | Desktop computers               | 2                  | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 5           | Tablets        | Tablet devices                  | 2                  | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 6           | Phones         | Mobile phones and smartphones   | 1                  | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 7           | Smartphones    | Advanced mobile phones          | 6                  | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 8           | Feature Phones | Basic mobile phones             | 6                  | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 9           | Accessories    | Electronic accessories          | 1                  | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |
| 10          | Cables         | Various cables and connectors   | 9                  | 2025-04-27 10:10:05 | 2025-04-27 10:10:05 |

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'SCHEMAS' tree with 'inventorydb' selected. The right pane shows a query window with the following SQL code:

```
1 • USE inventorydb;
2 • SELECT COUNT(*) AS total_categories FROM Categories;
```

The 'Result Grid' at the bottom displays the following data:

| total_categories |
|------------------|
| 120012           |

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'SCHEMAS' tree with 'inventorydb' selected. The right pane shows a query window with the following SQL code:

```
1 • USE inventorydb;
2 • SELECT name, email FROM Suppliers WHERE city = 'New York';
```

The 'Result Grid' at the bottom displays the following data:

| name         | email                 |
|--------------|-----------------------|
| Gadget World | sarah@gadgetworld.com |

### 3 BI Queries: To showcase insights and analytics

The screenshot shows the SQL Studio interface with a query titled "updated inventorydb" in the "SQL File 7" tab. The query is as follows:

```
1 • USE inventorydb;
2
3 -- Top 5 categories by the number of discounts
4 • SELECT c.category_name, COUNT(d.discount_id) AS discount_count
5 FROM Discounts d
6 JOIN Products p ON d.product_id = p.product_id
7 JOIN Categories c ON p.category_id = c.category_id
8 GROUP BY c.category_name
9 ORDER BY discount_count DESC
10 LIMIT 5;
```

The "Result Grid" shows the following data:

| category_name  | discount_count |
|----------------|----------------|
| Feature Phones | 79653          |
| Desktops       | 67336          |
| Phones         | 67293          |
| Electronics    | 61253          |
| Batteries      | 61056          |

The screenshot shows the SQL Studio interface with a query titled "updated inventorydb" in the "SQL File 7" tab. The query is as follows:

```
1 • USE inventorydb;
2
3 -- Total suppliers by country
4 • SELECT country, COUNT(*) AS total_suppliers
5 FROM Suppliers
6 GROUP BY country
7 ORDER BY total_suppliers DESC;
```

The "Result Grid" shows the following data:

| country                      | total_suppliers |
|------------------------------|-----------------|
| Korea                        | 4999            |
| Congo                        | 4917            |
| Hungary                      | 2617            |
| French Southern Territories  | 2615            |
| Algeria                      | 2596            |
| British Virgin Islands       | 2586            |
| Gabon                        | 2583            |
| Spain                        | 2582            |
| Poland                       | 2580            |
| Gibraltar                    | 2580            |
| Bahrain                      | 2578            |
| Cayman Islands               | 2577            |
| Eritrea                      | 2576            |
| Slovenia                     | 2576            |
| Nigeria                      | 2567            |
| Cape Verde                   | 2566            |
| Chad                         | 2565            |
| Tunisia                      | 2563            |
| Lebanon                      | 2563            |
| South Georgia and the Sou... | 2560            |

The screenshot shows the SQL Studio interface with a query titled "updated inventorydb" in the "SQL File 7" tab. The query is as follows:

```
1 • USE inventorydb;
2
3 -- Total discounts by month
4 • SELECT MONTH(start_date) AS discount_month, SUM(discount_percentage) AS total_discount_percentage
5 FROM Discounts
6 GROUP BY discount_month
7 ORDER BY discount_month;
```

The "Result Grid" shows the following data:

| discount_month | total_discount_percentage |
|----------------|---------------------------|
| 1              | 4092903.12                |
| 2              | 3693662.41                |
| 3              | 4080512.97                |
| 4              | 3682043.27                |
| 5              | 9977.44                   |
| 6              | 15.00                     |
| 7              | 30.00                     |
| 8              | 10.00                     |
| 9              | 15.00                     |
| 12             | 20.00                     |

- **2 Optimized Queries:** To highlight performance improvements.

Navigator: updated inventorydb SQL File 7\*

SCHEMAS

Filter objects

- inventory\_dw
- inventorydb
  - Tables
  - Views
  - Stored Procedures
  - Functions
  - rechelleborbe
  - sys

Administration Schemas

Information

No object selected

```

1 • USE inventorydb;
2
3 -- Retrieve active discounts using an index on is_active
4 • SELECT discount_name, discount_percentage, start_date, end_date
5 FROM Discounts
6 WHERE is_active = 1;

```

Result Grid

|   | discount_name | discount_percentage | start_date          | end_date            |
|---|---------------|---------------------|---------------------|---------------------|
| ▶ | Discount 1    | 21.49               | 2025-01-06 06:30:37 | 2025-01-26 06:30:37 |
|   | Discount 4    | 38.50               | 2025-03-13 22:58:25 | 2025-04-05 22:58:25 |
|   | Discount 5    | 1.94                | 2025-02-18 09:48:36 | 2025-03-02 09:48:36 |
|   | Discount 7    | 43.23               | 2025-03-31 07:58:31 | 2025-04-24 07:58:31 |
|   | Discount 8    | 24.00               | 2025-02-05 04:58:25 | 2025-02-07 04:58:25 |
|   | Discount 9    | 5.42                | 2025-03-09 23:07:45 | 2025-03-21 23:07:45 |
|   | Discount 10   | 14.26               | 2025-02-23 17:37:47 | 2025-03-08 17:37:47 |
|   | Discount 14   | 48.54               | 2025-04-09 03:55:22 | 2025-05-05 03:55:22 |
|   | Discount 15   | 3.20                | 2025-03-13 18:12:36 | 2025-03-19 18:12:36 |
|   | Discount 16   | 31.12               | 2025-03-10 12:47:20 | 2025-04-05 12:47:20 |
|   | Discount 18   | 19.11               | 2025-01-26 01:16:15 | 2025-02-02 01:16:15 |
|   | Discount 20   | 41.09               | 2025-03-13 17:24:32 | 2025-03-25 17:24:32 |
|   | Discount 21   | 10.50               | 2025-03-14 22:10:42 | 2025-03-19 22:10:42 |
|   | Discount 23   | 35.93               | 2025-03-24 16:15:22 | 2025-04-06 16:15:22 |
|   | Discount 24   | 27.51               | 2025-04-09 04:53:49 | 2025-04-10 04:53:49 |
|   | Discount 25   | 3.71                | 2025-02-03 21:46:48 | 2025-02-16 21:46:48 |
|   | Discount 26   | 5.94                | 2025-01-07 12:31:44 | 2025-01-21 12:31:44 |
|   | Discount 27   | 6.54                | 2025-03-22 07:31:16 | 2025-03-28 07:31:16 |
|   | Discount 30   | 2.71                | 2025-03-06 13:47:46 | 2025-03-20 13:47:46 |
|   | Discount 38   | 34.61               | 2025-03-11 03:24:05 | 2025-03-26 03:24:05 |

Navigator: updated inventorydb SQL File 7\*

SCHEMAS

Filter objects

- inventory\_dw
- inventorydb
  - Tables
  - Views
  - Stored Procedures
  - Functions
  - rechelleborbe
  - sys

Administration Schemas

Information

No object selected

```

1 • USE inventorydb;
2
3 -- Aggregate discounts by product with an index on product_id:
4 • SELECT product_id, COUNT(*) AS discount_count, AVG(discount_percentage) AS avg_discount
5 FROM Discounts
6 GROUP BY product_id
7 ORDER BY discount_count DESC;

```

Result Grid

|   | product_id | discount_count | avg_discount |
|---|------------|----------------|--------------|
| ▶ | 47         | 6328           | 25.226364    |
|   | 94         | 6261           | 25.356394    |
|   | 86         | 6247           | 25.766694    |
|   | 4          | 6244           | 25.477071    |
|   | 52         | 6232           | 25.386555    |
|   | 12         | 6227           | 25.514448    |
|   | 99         | 6226           | 25.732817    |
|   | 33         | 6220           | 25.240605    |
|   | 14         | 6212           | 25.205935    |
|   | 38         | 6207           | 25.516676    |
|   | 58         | 6204           | 25.339998    |
|   | 27         | 6202           | 25.327252    |
|   | 56         | 6199           | 25.285857    |
|   | 62         | 6196           | 25.345462    |
|   | 46         | 6192           | 25.451042    |
|   | 37         | 6190           | 25.263286    |
|   | 76         | 6186           | 25.823065    |
|   | 64         | 6184           | 25.542576    |
|   | 90         | 6184           | 25.352917    |
|   | 25         | 6183           | 25.722497    |

Result 11 x



- **1 Diagnostic Queries (Optional):** To analyze and validate performance.

The screenshot shows a SQL IDE interface. On the left is a 'SCHEMAS' navigator with a tree view containing 'inventory\_dw', 'inventorydb' (expanded), 'Tables', 'Views', 'Stored Procedures', 'Functions', 'rechelleborbe', and 'sys'. The main editor displays a SQL script with four lines: 1. 'USE inventorydb;', 2. an empty line, 3. a comment '-- Analyze query execution with EXPLAIN', and 4. 'EXPLAIN SELECT \* FROM Discounts WHERE is\_active = 1;'. Below the editor is a 'Result Grid' with a table of execution plan details.

| id | select_type | table     | partitions | type | possible_keys | key  | key_len | ref  | rows   | filtered | Extra       |
|----|-------------|-----------|------------|------|---------------|------|---------|------|--------|----------|-------------|
| 1  | SIMPLE      | Discounts | NULL       | ALL  | NULL          | NULL | NULL    | NULL | 607883 | 10.00    | Using where |

## ➤ Reflection and Limitations

At first, we started our final project by assigning each group member a specific role. We split up the tasks so we could work faster and more efficiently. Everyone had their own part to focus on—one became the Project Lead to guide the overall progress, another was the Database Architect who designed the schema and structure, one took the role of SQL Developer to write queries and manage the database, another acted as the Back-End Developer to handle deployment and technical setups, and one worked as the QA Tester to check and verify that everything worked correctly.

As we moved forward, we faced many difficulties. At first, designing the schema and adding constraints was confusing. Importing the data took a lot of trial and error, and we often got stuck fixing small but frustrating problems. We also experienced the difficulties of passing the deliverable on GitHub so we did a trial and errors so many times. One of the biggest struggles we had was with Railway during the cloud deployment. Setting up the cloud database, restoring backups, and making sure everything worked online took us a lot of time. There were issues we didn't expect, and we had to search for solutions and help each other figure things out. Backup and recovery were also difficult parts. Creating the backup, testing it, and proving that it worked was not as easy as we thought. We had to repeat the process several times before we got it right. Writing the BI queries and creating the star schema was another challenge. It was hard to fully understand how to transform the data into a format we could use for insights. But by working together and doing research, we were able to complete it. Query optimization was the last and hardest part. Learning how to use EXPLAIN and improve slow queries took a lot of effort. We had to compare the performance before and after making changes, and it took time to get it right. During the final phase of the project, we faced technical problems with our devices. Some tools and processes wouldn't work properly on our main laptops, and we couldn't continue. This was very stressful because the deadline was close.

In the end, we had to borrow or switch to another device just to execute the final steps and make sure everything worked. That extra step made things even harder, but it was the only way to complete the project. Even though we struggled through almost every part, we didn't give up. We supported each other and kept moving forward, step by step. In the end, we were able to submit all the requirements to GitHub and LMS SQL files, screenshot, documentation, and reports.

As a team, we are proud of what we've accomplished, and we know, we've done our best.

# **INVENTORY MANAGEMENT SYSTEM**

## **MEMBERS**

Project lead:Rechelle Borbe

Database architect: Mhelarry Valeza

Database Architecture :Curt Justin Reodique

Backend Developer: Susaine Rico

SQL Developer: Faith Sañado

QA Tester: Shaine Sanjuan