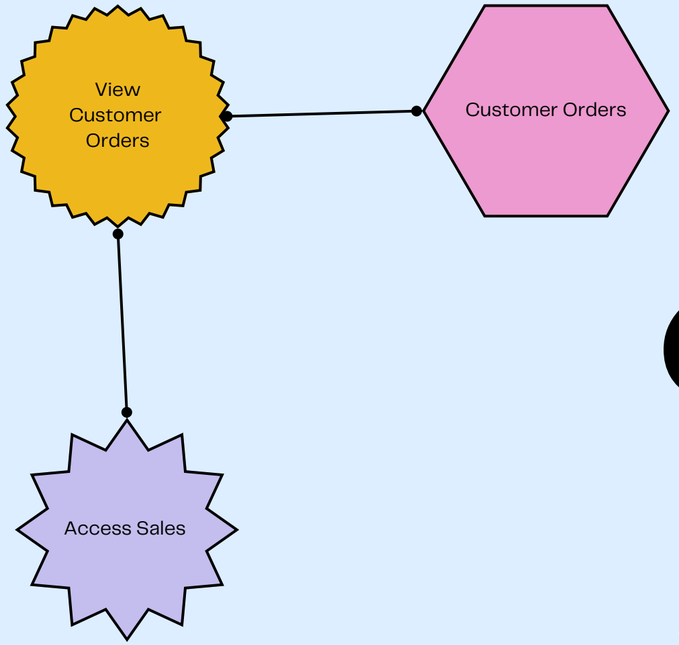
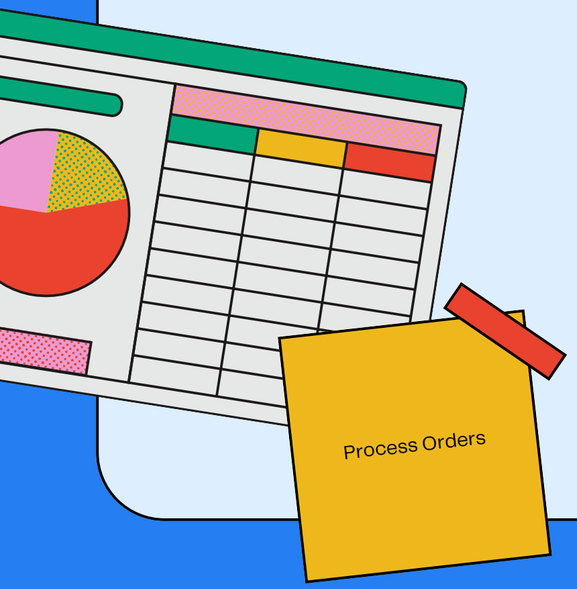
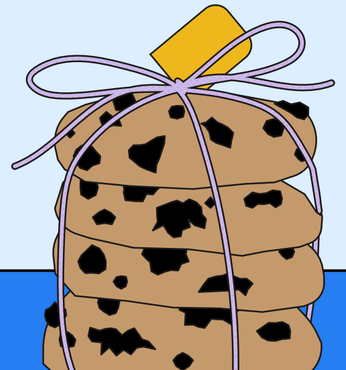
December 5, 2025

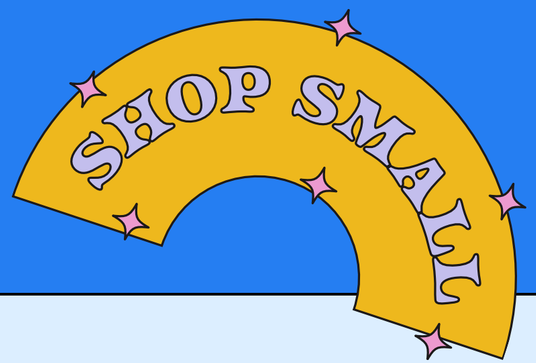


Grocery Store Inventory Management

By: Alejandro Garcia, Arpita Misal, Bhavya Preethika Vatsavayi, Jasleen Narang, Blake Montgomery







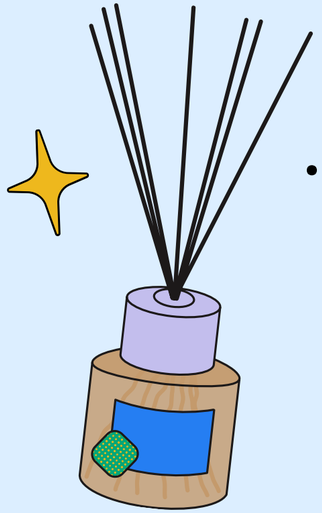
Problem Description

Challenges :

•– Inefficient tracking of inventory, customer orders, and supplier relationships.

•– Manual or outdated systems lead to errors and delays.

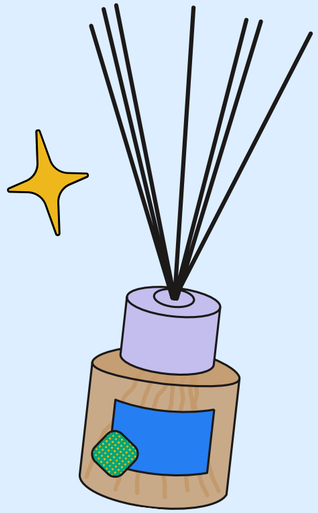
•– Lack of real-time updates hinders decision-making and operational efficiency.



Application Overview

Purpose:

- Web-based solution using MySQL for data management.
- Real-time tracking and decision-making support.



Core Features:

- Inventory control, order processing, supplier management, and reporting.
- Employee and Product management.

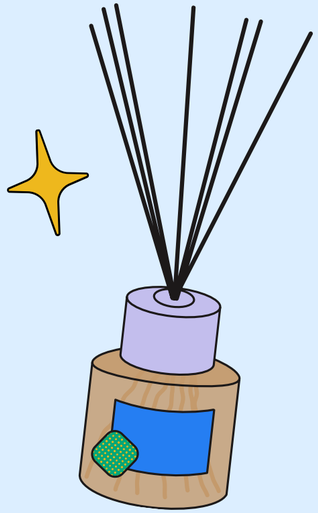
System Functionality

Functionality:

•– Product and inventory management.

•– Order processing and Employee management.

•– Supplier management and report generation.



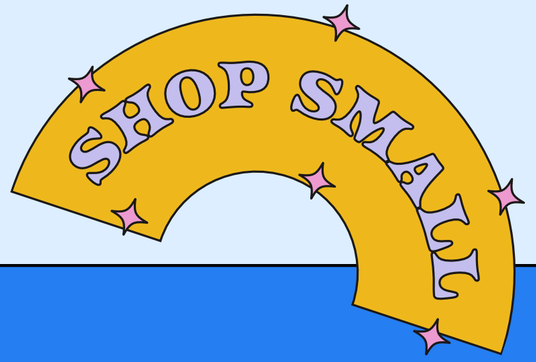
Architecture:

•– Two-tier design with Presentation Tier and Application/Data Tier.

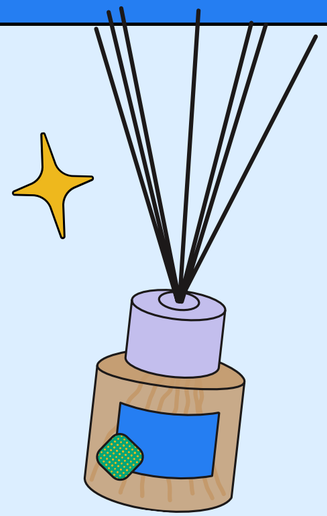
•– Secure API endpoints for data interaction.

Major Design Decisions

- Buttons in column format for easier navigation: Allows for cleaner layout, useful for grocery store employees, makes it quick and efficient to use
- Tableview to show data clearly: Allows for more structured data for inventory, orders, and customer information
- Simple design for enhanced usability: Straightforward and minimalist design ensures easy accessibility for all users
- Real-time updates: The design incorporates real-time synchronization for inventory levels and order statuses
- Error-handling and validation: User-friendly error messages and data validation for more accuracy and reduce mistakes.



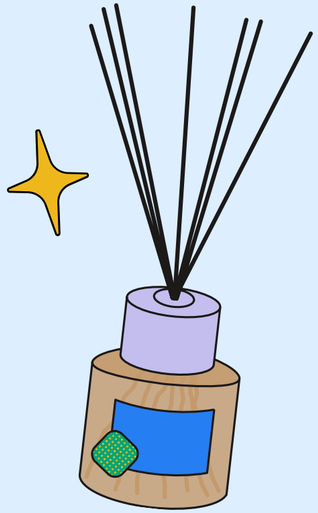
Entities & Attributes



Product

Attributes:

- product_id (Primary Key, single-valued, atomic, stored)
- product_name (single-valued, atomic, stored)
- category_id (Foreign Key to Category, single-valued, atomic, stored)
- price (single-valued, atomic, stored)Z
- stock_quantity (single-valued, atomic, derived from Inventory.stock_level)
- expiration_date (single-valued, atomic, stored)



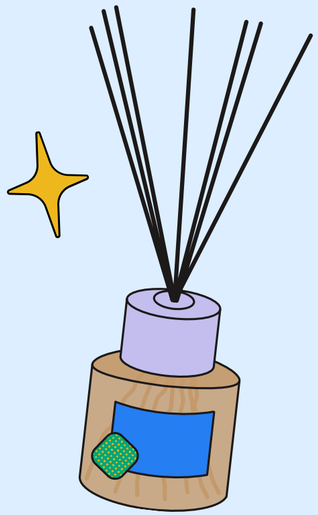
Category & Supplier

Category Attributes:

- category_id (Primary Key, single-valued, atomic, stored)
- category_name (single-valued, atomic, stored)
- description (single-valued, atomic, stored)

Supplier Attributes:

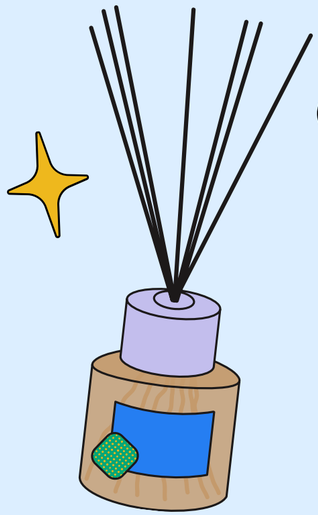
- supplier_id (Primary Key, single-valued, atomic, stored)
- supplier_name (single-valued, atomic, stored)
- street (single-valued, atomic, stored)
- city (single-valued, atomic, stored)
- state (single-valued, atomic, stored)
- zip (single-valued, atomic, stored)
- phone (single-valued, atomic, stored)
- email (single-valued, atomic, stored)



Customer & Order

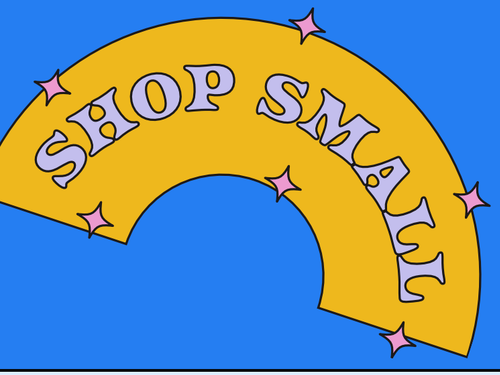
Customer Attributes:

- customer_id (Primary Key, single-valued, atomic, stored)
- first_name (single-valued, atomic, stored)
- last_name (single-valued, atomic, stored)
- email (single-valued, atomic, stored)
- phone (single-valued, atomic, stored)



Order Attributes:

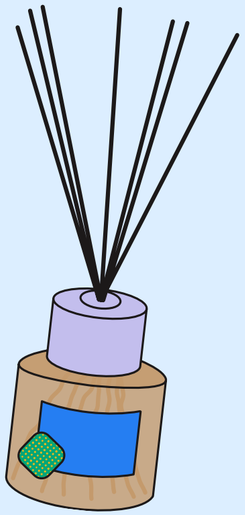
- order_id (Primary Key, single-valued, atomic, stored)
- customer_id (Foreign Key to Customer, single-valued, atomic, stored)
- order_date (single-valued, atomic, stored)
- order_status (single-valued, atomic, stored)



Order_item & Inventory

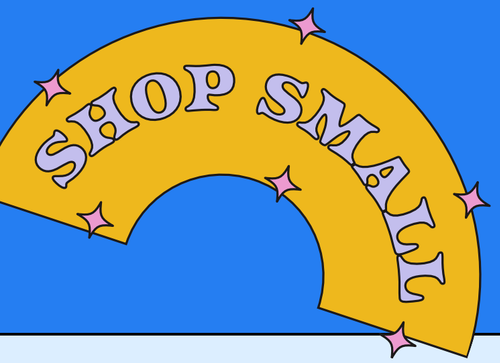
Order_item Attributes:

- order_item_id (Primary Key, single-valued, atomic, stored)
- order_id (Foreign Key to Order, single-valued, atomic, stored)
- product_id (Foreign Key to Product, single-valued, atomic, stored)
- quantity (single-valued, atomic, stored)
- price (single-valued, atomic, derived: quantity * Product.price)



Inventory Attributes:

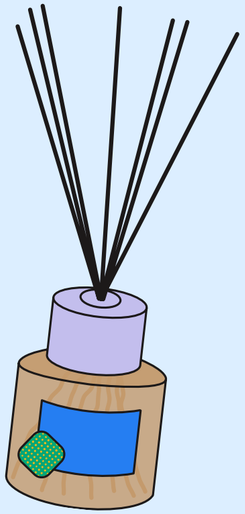
- inventory_id (Primary Key, single-valued, atomic, stored)
- product_id (Foreign Key to Product, single-valued, atomic, stored)
- is_reorder_needed (single-valued, atomic, stored)



Employee & Discount

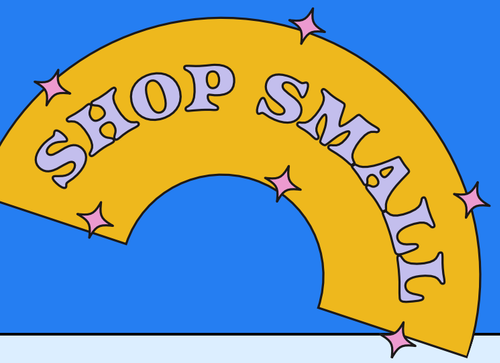
Employee Attributes:

- employee_id (Primary Key, single-valued, atomic, stored)
- first_name (single-valued, atomic, stored)
- last_name (single-valued, atomic, stored)
- position (single-valued, atomic, stored)
- salary (single-valued, atomic, stored)
- hire_date (single-valued, atomic, stored)



discount Attributes:

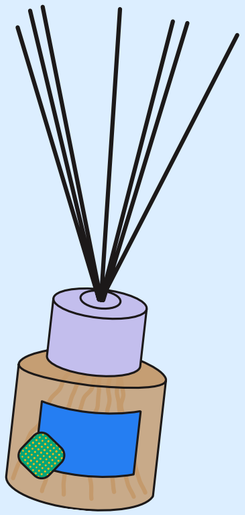
- discount_id (Primary Key, single-valued, atomic, stored)
- product_id (Foreign Key to Product, single-valued, atomic, stored)
- discount_percentage (single-valued, atomic, stored)
- start_date (single-valued, atomic, stored)
- end_date (single-valued, atomic, stored)



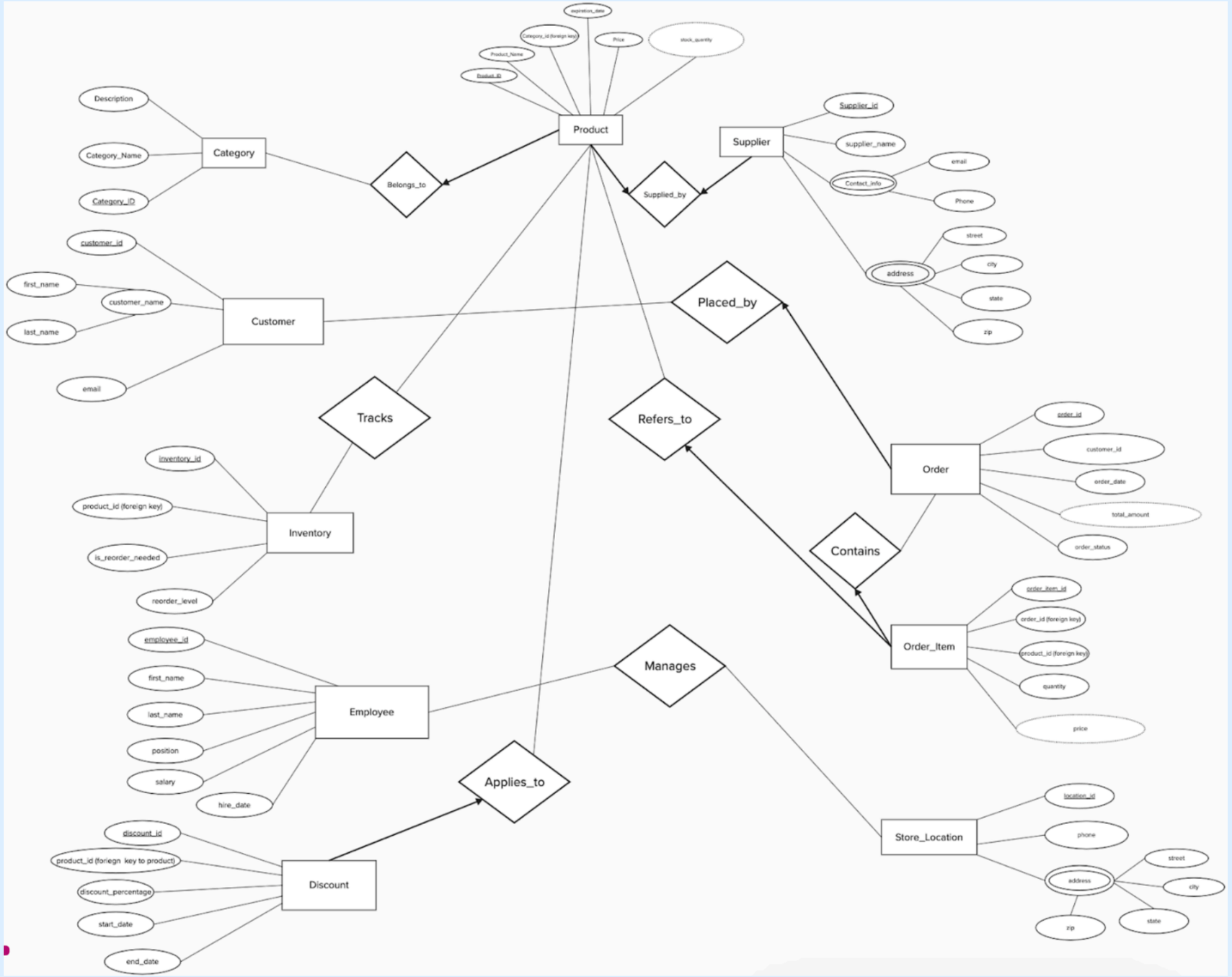
Store Location

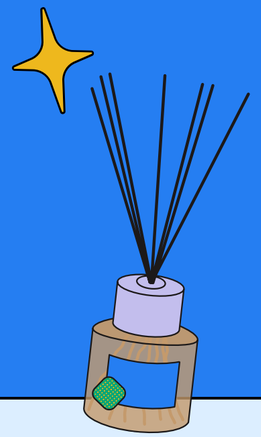
store_location Attributes:

- location_id (Primary Key, single-valued, atomic, stored)
- store_name (single-valued, atomic, stored)
- street (single-valued, atomic, stored)
- city (single-valued, atomic, stored)
- state (single-valued, atomic, stored)
- zip (single-valued, atomic, stored)
- phone (single-valued, atomic, stored)



ER DIAGRAM


Product
Product_ID
Product_Name
Category_id (foreign key)
Price
stock_quantity
Belongs_to
Category
Description
Category_Name
Category_ID
Supplied_by
Supplier
Supplier_id
supplier_name
Contact_info
email
Phone
address
street
city
state
zip
Customer
customer_id
first_name
customer_name
last_name
email
Placed_by
Tracks
Refers_to
Order
order_id
customer_id
order_date
total_amount
order_status
Contains
Inventory
inventory_id
product_id (foreign key)
is_reorder_needed
reorder_level
Order_Item
order_item_id
order_id (foreign key)
product_id (foreign key)
quantity
price
Manages
Employee
employee_id
first_name
last_name
position
salary
hire_date
Applies_to
Store_Location
location_id
phone
address
street
city
state
zip
Discount
discount_id
product_id (foriegn key to product)
discount_percentage
start_date
end_date



Index for Product

```
1   describe Product;
2
```

100% 18:1

Result Grid Filter Rows: Search Export:

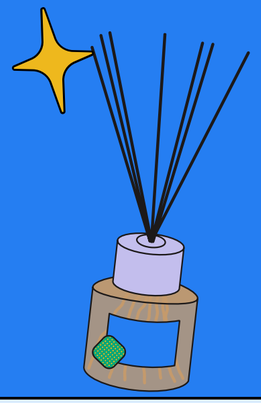
Field	Type	Null	Key	Default	Extra
product_id	int	NO	PRI	NULL	auto_increment
product_name	varchar(255)	NO		NULL	
category_id	int	NO	MUL	NULL	
price	decimal(10,2)	NO		NULL	
expiration_date	date	NO		NULL	
stock_quantity	int	YES		NULL	

```
1 •  show index from Product;
```

100% 24:1

Result Grid Filter Rows: Search Export:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
Product	0	PRIMARY	1	product_id	A	149	NULL	NULL		BTREE			YES	NULL
Product	1	fk_category	1	category_id	A	71	NULL	NULL		BTREE			YES	NULL
Product	1	idx_product_name	1	product_name	A	148	NULL	NULL		BTREE			YES	NULL



Index for Product

```
1   EXPLAIN SELECT * FROM Product;
2
```

100% 30:1

Result Grid Filter Rows: Search Export:

id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Product	NULL	ALL	NULL	NULL	NULL	NULL	149	100.00	NULL

<- THIS IS QUERY 1

THIS IS QUERY 2 ->

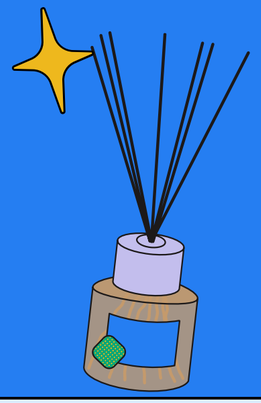
Limit to 5000 rows

```
1   EXPLAIN SELECT * FROM Product WHERE product_name LIKE 'Some%';
2
```

100% 1:2

Result Grid Filter Rows: Search Export:

id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Product	NULL	range	idx_product_name	idx_product_name	1022	NULL	1	100.00	Using index condition



Index for Order

```
1   describe `Order`;
```

100% 18:1

Result Grid Filter Rows: Search Export:

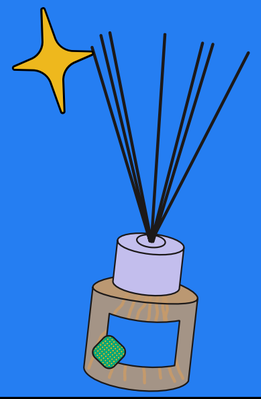
Field	Type	Null	Key	Default	Extra
order_id	int	NO	PRI	NULL	auto_increment
customer_id	int	NO	MUL	NULL	
order_date	date	NO		NULL	
order_status	enum('Pending','Shipped','Delivered','Cancelled')	NO		NULL	

```
1 •  show index from `Order`;
```

100% 25:1

Result Grid Filter Rows: Search Export:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
Order	0	PRIMARY	1	order_id	A	50	NULL	NULL		BTREE			YES	NULL
Order	1	customer_id	1	customer_id	A	50	NULL	NULL		BTREE			YES	NULL
Order	1	idx_order_status	1	order_status	A	3	NULL	NULL		BTREE			YES	NULL



Index for Order

```
1   EXPLAIN SELECT * FROM `Order`;
2
```

100% 30:1

Result Grid Filter Rows: Search Export:

id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Order	NULL	ALL	NULL	NULL	NULL	NULL	50	100.00	NULL

<- THIS IS QUERY 1

THIS IS QUERY 2 ->

```
1   EXPLAIN SELECT * FROM `Order` WHERE order_status = 'Delivered';
2
```

100% 9:1

Result Grid Filter Rows: Search Export:

id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Order	NULL	ref	idx_order_status	idx_order_status	1	const	21	100.00	Using index condition

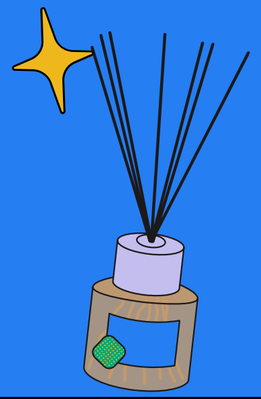
Index for Discount

```
1 •  describe Discount;
```

100% 19:1

Result Grid | Filter Rows: Search | Export:

Field	Type	Null	Key	Default	Extra
discount_id	int	NO	PRI	NULL	auto_increment
product_id	int	NO	MUL	NULL	
discount_percentage	decimal(5,2)	NO	MUL	NULL	
start_date	date	NO	MUL	NULL	
end_date	date	NO		NULL	



Index for Discount

```
1  EXPLAIN SELECT *
2  FROM Discount d;
3
```

100% 17:1

Result Grid Filter Rows: Search Export:

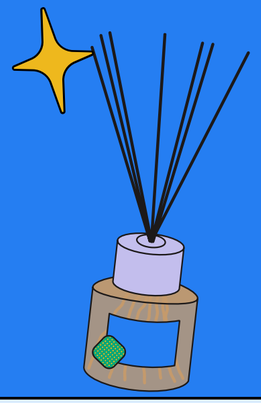
id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	d	NULL	ALL	NULL	NULL	NULL	NULL	4	100.00	NULL

```
1  EXPLAIN SELECT d.discount_id, p.product_id, p.product_name, d.discount_percentage, d.start_date, d.end_date
2  FROM Discount d
3  JOIN Product p ON d.product_id = p.product_id
4  WHERE d.start_date = '2024-12-05';
5
```

100% 9:1

Result Grid Filter Rows: Search Export:

id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	d	NULL	ref	idx_product_start_date,idx_date_range	idx_date_range	3	const	1	100.00	NULL
1	SIMPLE	p	NULL	eq_ref	PRIMARY	PRIMARY	4	grocerystore.d.product_id	1	100.00	NULL



Index for Customer

Query 1 | Product_Insert_Statements_150* | Grocery_Product_Insert_Statements* | customer_entries_with_phone*

Limit to 5000 rows

```
1  show index from Customer;
```

100% 27:1

Result Grid Filter Rows: Search Export:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
Customer	0	PRIMARY	1	customer_id	A	0	NULL	NULL		BTREE			YES	NULL
Customer	0	phone	1	phone	A	0	NULL	NULL		BTREE			YES	NULL
Customer	0	email	1	email	A	0	NULL	NULL		BTREE			YES	NULL
Customer	1	idx_customer_name	1	first_name	A	156	NULL	NULL		BTREE			YES	NULL
Customer	1	idx_customer_name	2	last_name	A	350	NULL	NULL		BTREE			YES	NULL

```
1  EXPLAIN SELECT customer_id, first_name, last_name, phone, email
2  FROM Customer
3  WHERE first_name LIKE 'John%' OR last_name LIKE 'Doe%';
4
```

100% 1:4

Result Grid Filter Rows: Search Export:

id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Customer	NULL	ALL	idx_customer_name	NULL	NULL	NULL	3250	20.99	Using where

Index for Category

```
describe Category;
```

100% 19:1

Result Grid Filter Rows: Search Export:

Field	Type	Null	Key	Default	Extra
category_id	int	NO	PRI	NULL	
category_name	varchar(255)	NO	UNI	NULL	
description	text	NO		NULL	

Query 1 | Product_Insert_Statements_150* | Grocery_Product_Insert_Statements* | customer_entries_with_phone

Limit to 5000 rows

```
EXPLAIN SELECT category_id, category_name, description
FROM Category
WHERE category_name LIKE 'Craft%';
```

100% 9:1

Result Grid Filter Rows: Search Export:

id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Category	NULL	range	category_name	category_name	1022	NULL	1	100.00	Using index condition

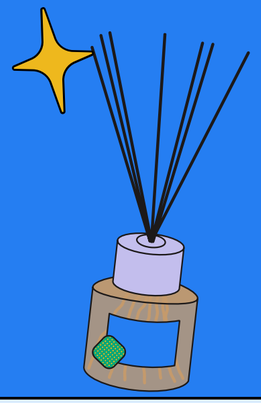
Limit to 5000 rows

```
EXPLAIN SELECT * FROM Category;
```

100% 31:1

Result Grid Filter Rows: Search Export:

id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Category	NULL	ALL	NULL	NULL	NULL	NULL	83	100.00	NULL



Index for Store Location

```
1   show index from Store_Location;
```

100% 16:1

Result Grid Filter Rows: Search Export:

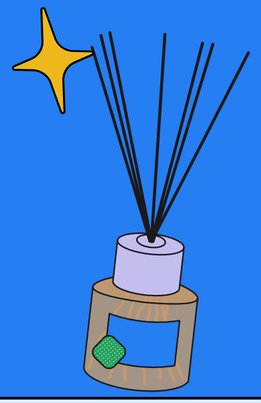
Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
Store_Location	0	PRIMARY	1	location_id	A	85	NULL	NULL		BTREE			YES	NULL

```
1   describe Store_Location;
```

100% 24:1

Result Grid Filter Rows: Search Export:

Field	Type	Null	Key	Default	Extra
location_id	int	NO	PRI	NULL	auto_increment
store_name	varchar(255)	NO		NULL	
street	varchar(255)	NO		NULL	
city	varchar(100)	NO		NULL	
state	varchar(50)	NO		NULL	
zip	varchar(10)	NO		NULL	
phone	varchar(15)	NO		NULL	



Index for Store Location

```
1 •  EXPLAIN SELECT * FROM Store_Location;
2
```

100% 37:1

Result Grid Filter Rows: Search Export:

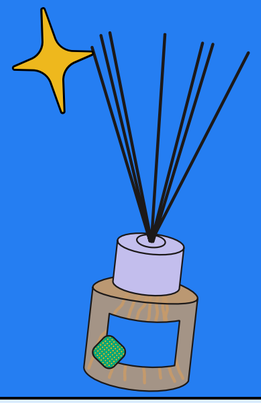
id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Store_Location	NULL	ALL	NULL	NULL	NULL	NULL	85	100.00	NULL

```
1 •  EXPLAIN SELECT * FROM Store_Location WHERE store_name LIKE 'Walmart%';
2
```

100% 1:2

Result Grid Filter Rows: Search Export:

id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Store_Location	NULL	range	idx_store_name	idx_store_name	1022	NULL	1	100.00	Using index condition



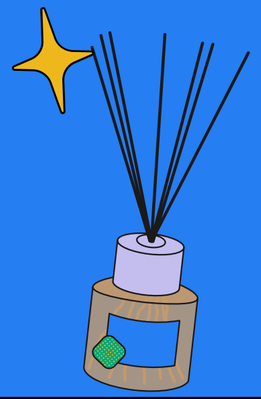
Index for Inventory

```
1 • show index from Inventory;
```

100% 27:1

Result Grid Filter Rows: Search Export:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
Inventory	0	PRIMARY	1	inventory_id	A	100	NULL	NULL		BTREE			YES	NULL
Inventory	1	product_id	1	product_id	A	100	NULL	NULL		BTREE			YES	NULL
Inventory	1	idx_reorder_needed	1	is_reorder_needed	A	2	NULL	NULL		BTREE			YES	NULL
Inventory	1	idx_reorder_product	1	is_reorder_needed	A	2	NULL	NULL		BTREE			YES	NULL
Inventory	1	idx_reorder_product	2	product_id	A	100	NULL	NULL		BTREE			YES	NULL



Index for Inventory

```
1  EXPLAIN SELECT * FROM Inventory WHERE is_reorder_needed = 1;
2
```

100% 1:2

Result Grid Filter Rows: Search Export:

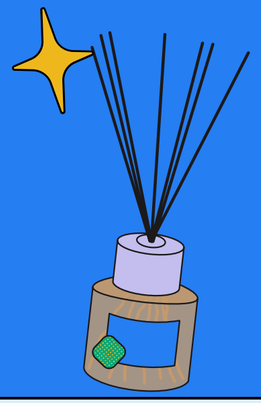
id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Inventory	NULL	ALL	NULL	NULL	NULL	NULL	100	10.00	Using where

```
1  EXPLAIN SELECT * FROM Inventory WHERE is_reorder_needed = 1;
2
```

100% 1:2

Result Grid Filter Rows: Search Export:

id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Inventory	NULL	ref	idx_reorder_needed,idx_reorder_product	idx_reorder_product	1	const	54	100.00	Using index



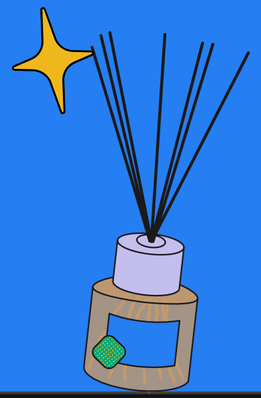
Index for Employee

```
1 •  show index from Employee;
```

100% 26:1

Result Grid Filter Rows: Search Export:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
Employee	0	PRIMARY	1	employee_id	A	2837	NULL	NULL		BTREE			YES	NULL
Employee	1	idx_first_name	1	first_name	A	17	NULL	NULL		BTREE			YES	NULL
Employee	1	idx_last_name	1	last_name	A	14	NULL	NULL		BTREE			YES	NULL
Employee	1	idx_employee_search	1	first_name	A	17	NULL	NULL		BTREE			YES	NULL
Employee	1	idx_employee_search	2	last_name	A	108	NULL	NULL		BTREE			YES	NULL
Employee	1	idx_employee_search	3	position	A	790	NULL	NULL		BTREE			YES	NULL



Index for Employee

```
1  EXPLAIN
2  SELECT employee_id, first_name, last_name, position, salary, hire_date
3  FROM Employee;
4
```

100% 14:3

Result Grid Filter Rows: Search Export:

id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Employee	NULL	ALL	NULL	NULL	NULL	NULL	2837	100.00	NULL

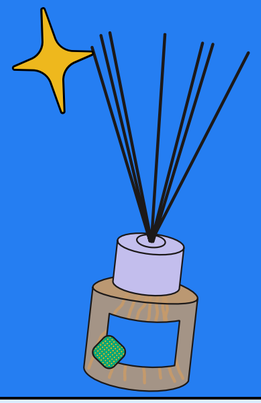
```
1  EXPLAIN
2  SELECT employee_id, first_name, last_name, position, salary, hire_date
3  FROM Employee
4  WHERE first_name LIKE 'John%';
5
```

100% 8:1

Result Grid Filter Rows: Search Export:

id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Employee	NULL	range	idx_first_name,idx_employee_search	idx_first_name	202	NULL	280	100.00	Using index condition

Index for Supplier



Query 1 | Product_Insert_Statements_150* | Grocery_Product_Insert_Statements* | customer_entries* | customer_entries_with_phone* | updated_customer_entries_with_valid_p

Limit to 5000 rows

```
1  EXPLAIN SELECT supplie
2  FROM Supplier
3  WHERE supplier_name LI
4
```

100% 1:4

Result Grid Filter Rows: Searc

id	select_ty...	table	partitions
1	SIMPLE	Supplier	NULL

```
1  EXPLAIN
2  SELECT employee_id, first_name, last_name, position, salary, hire_date
3  FROM Employee;
4
```

100% 14:3

Result Grid Filter Rows: Search Export:

id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Employee	NULL	ALL	NULL	NULL	NULL	NULL	2837	100.00	NULL

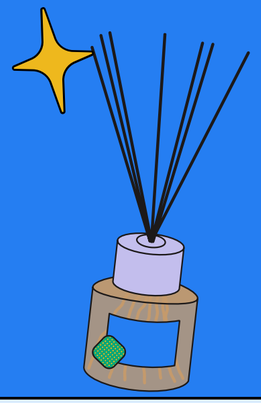
100% 1:3

Result Grid Filter Rows: Search Export:

id	select_ty...	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Supplier	NULL	ALL	NULL	NULL	NULL	NULL	6584	100.00	NULL

This is the optimized query

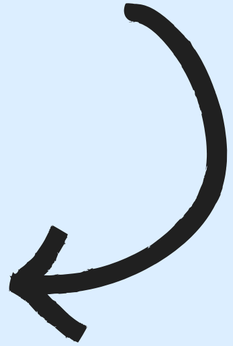
This is the non-optimized query



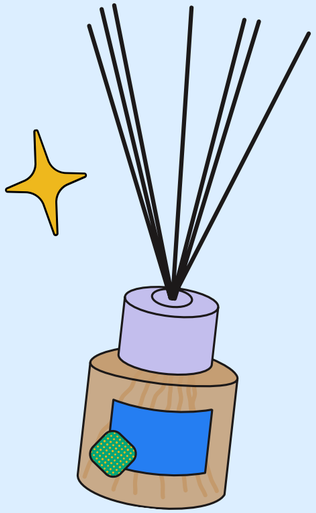
Index for Supplier

Field	Type	Null	Key	Default	Extra
supplier_id	int	NO	PRI	NULL	auto_increment
supplier_name	varchar(255)	NO	MUL	NULL	
phone	varchar(20)	NO		NULL	
email	varchar(255)	NO	MUL	NULL	
street	varchar(255)	NO		NULL	
city	varchar(100)	NO	MUL	NULL	
state	varchar(50)	NO	MUL	NULL	
zip	varchar(10)	NO		NULL	

This is the Structure of Supplier



Demo



Challenges Encountered

Team Challenges:

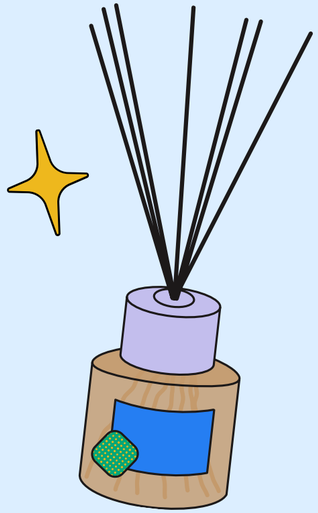
•– Coordinating database and application integration.

Technical Issues:

•– Ensuring real-time database updates and concurrency control.

•– Managing role-based access and secure authentication.

•– Integrating different functional modules seamlessly.



Solutions:

•– Implemented ACID properties and rigorous testing.

•– Modular design for scalability and maintainability.

Next Steps

Planned Enhancements:

•– Introduce role-based access control for employees and secure payment processing mechanisms.

•– Introduce feedback mechanisms to collect customer input.

•–Forecasting demand trends and optimizing stock levels.