

|  |
| --- |
| Business Template  A database of household appliances store |
| **Logo / Image** |

Contents

[1 Business Description 3](#_Toc62212630)

[1.1 Business background 3](#_Toc62212631)

[1.2 Problems. Current Situation 3](#_Toc62212632)

[1.3 The benefits of implementing a database. Project Vision 3](#_Toc62212633)

[2 Model description 3](#_Toc62212634)

[2.1 Definitions & Acronyms 3](#_Toc62212635)

[2.2 Logical Scheme 3](#_Toc62212636)

[2.3 Objects 3](#_Toc62212637)

# 

# Business Description

## Business background

A household appliances store database is created to manage the store’s operation including customer management, order processing, product inventory, supplier tracking, and employee records.

## Problems. Current Situation

As the store grows it will require more information to be collected and stored in the database.  
Tracking and updating inventory could be also further automatized to save time.

## the Benefits of implementing a database. Project Vision

The database enables the following functionalities to the store:

* employees can process and monitor orders
* tracking inventory levels and supplier relationships
* using historical data for future analysis and decision making

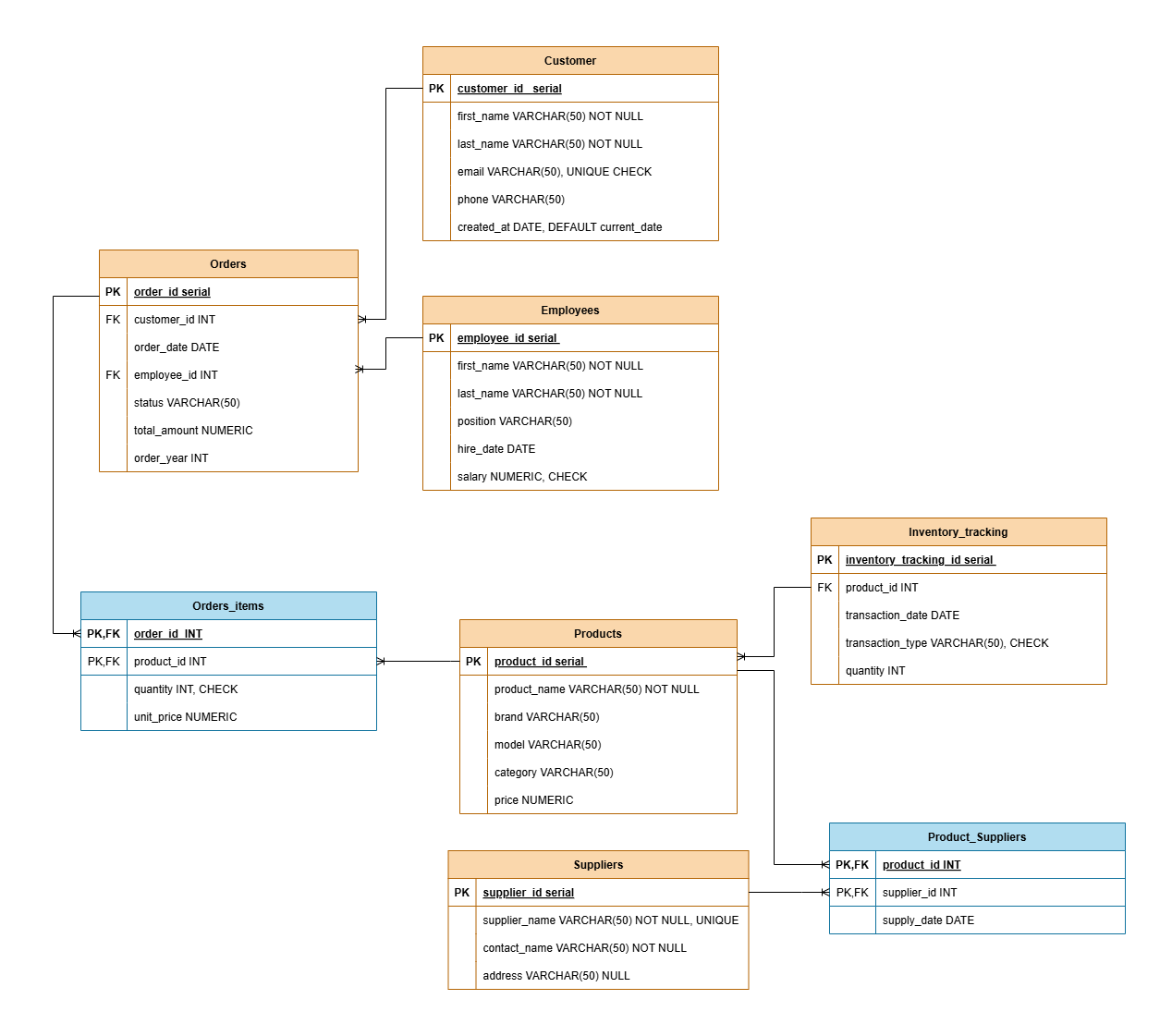
# Model description

## Definitions & Acronyms

The **UNIQUE** constraint ensures that specific columns, such as email addresses, supplier names, and product models, have distinct values.

The **CHECK** constraint is used to validate data ranges and formats. For example, numeric values like total amounts, salaries, quantities, and unit prices must be greater than zero, while dates for orders and supplies must occur after July 1, 2024. Email addresses are validated to follow the correct format: %@%.% and transaction types in inventory are limited to "sold" or "returned," and order statuses must be either "pending" or "delivered."  
The **NOT NULL** constraint is applied to essential fields, such as customer names, employee details, and order information

## Logical Scheme



## Objects

The database consists of the following core tables:

1. Customer: Tracks customer details, including contact information and registration date.
2. Orders: Stores details of customer orders, including the associated employee and order status.
3. Employees: Maintains employee records, such as positions, hire dates, and salaries.
4. Order\_Items: Links orders and products, detailing quantities and unit prices.
5. Products: Stores information about available products, including brand, model, category, and price.
6. Suppliers: Maintains supplier details, including contact and address information.
7. Product\_Suppliers: Represents the many-to-many relationship between products and suppliers.
8. Inventory\_Tracking: Tracks inventory transactions, including type (e.g., addition or removal) and quantity.
9. **Customer table**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Customer | customer\_id | PK - A unique identifier for each customer | serial |
| first\_name | The first name of the customer, NOT NULL | varchar(50) |
| last\_name | The last name of the customer, NOT NULL | varchar(50) |
| email | The email address of the customer, UNIQUE, CHECK format '%@%.%' | varchar(50) |
| phone | The phone number of the customer | varchar(50) |
| created\_at | he date when the customer was added, default current date and CHECK: the date should be after 2024-07-01 | date |

Customer (1:M): One customer can have multiple orders, but one specific order ID is connected to only one customer.

Example with data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| customer\_id | first\_name | last\_name | email | phone | created\_at |
| 1 | Alice | Smith | alice.smith@email.com | 823-100 | 2024-11-30 |

1. Orders table

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Orders | order\_id | PK - A unique identifier for each order | serial |
| customer\_id | FK - Links the order to a specific customer | integer |
| order\_date | The date the order was placed | date |
| employee\_id | FK - Links the order to the employee who handled it | integer |
| status | The current status of the order, CHECK "pending," "delivered" | varchar(50) |
| total\_amount | The total amount for the order,CHECK >0 | numeric |
| order\_year | The year the order was placed, GENERATED from order\_date | integer |

Orders (1:M): One order can include multiple products through the Order\_Items table, but each order ID is associated with only one employee and one customer.

Example with data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| order\_id | customer\_id | order\_date | employee\_id | status | total\_amount | order\_year |
| 1 | 1 | 2024-09-10 | 4 | deliered | 1500 | 2024 |

1. **Employees table**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Employees | employee\_id | PK - A unique identifier for each employee | serial |
| first\_name | The first name of the employee, NOT NULL | varchar(50) |
| last\_name | The last name of the employee, NOT NULL | varchar(50) |
| position | The role or title of the employee | varchar(50) |
| hire\_date | The date when the employee joined the company | date |
| salary | The salary of the employee, CHECK >0 | numeric |

Employees (1:M): One employee can process multiple orders, but each order ID is connected to only one employee.

Example with data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| employee\_id | first\_name | last\_name | position | hire\_date | salary |
| 1 | Oliver | Knight | Store Manager | 2024-09-01 | 60000 |

1. **Order\_items table**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Order\_items | order\_id | PK/FK - Links the row to a specific order | integer |
| product\_id | PK/FK Links the row to a specific product | integer |
| quantity | The number of units of the product in the order, CHECK >0 | integer |
| unit\_price | The price per unit of the product, CHECK >0 | numeric |

Order\_Items (M:M): A many-to-many relationship exists between orders and products, as one order can include multiple products, and one product can appear in multiple orders.

Example with data

|  |  |  |  |
| --- | --- | --- | --- |
| order\_id | product\_id | quantity | unit\_price |
| 1 | 1 | 2 | 1200 |

1. **Products table**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Products | product\_id | PK - A unique identifier for each product | serial |
| product\_name | The name of the product, NOT NULL | varchar(50) |
| brand | The brand of the product | varchar(50) |
| model | The model of the product, UNIQUE | varchar(50) |
| category | The category of the product | varchar(50) |
| price | The price of the product, CHECK >0 | numeric |

Products (M:M): Products are connected to suppliers via a many-to-many relationship, allowing multiple suppliers to provide the same product.

Example with data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| product\_id | product\_name | brand | model | category | price |
| 1 | Vacuum Cleaner | Dyson | V11 | Cleaning | 499.99 |

1. **Suppliers table**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Suppliers | supplier\_id | PK - A unique identifier for each supplier | serial |
| supplier\_name | The name of the supplier, NOT NULL, UNIQUE | varchar(50) |
| contact\_name | The name of the supplier's contact person, NOT NULL | varchar(50) |
| address | The supplier's address | varchar(50) |

Suppliers (M:M): Each supplier can supply multiple products through the Product\_Suppliers table.

Example with data

|  |  |  |  |
| --- | --- | --- | --- |
| supplier\_id | supplier\_name | contact\_name | address |
| 1 | Home Interiors and Gifts | James Clark | 123 Elm Street, Springfield |

1. **Inventory\_tracking table**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Inventory\_tracking | inventory\_tracking\_id | PK - A unique identifier for each inventory transaction | serial |
| product\_id | FK - Links the row to a specific product | integer |
| transaction\_date | The date of the inventory transaction, CHECK the date should be after 2024-07-01 | date |
| transaction\_type | The type of transaction, CHECK: sold or returned | varchar(50) |
| quantity | The quantity added or removed | integer |

Inventory\_Tracking (1:M): One product can have multiple inventory transactions, but each inventory transaction is tied to a single product ID.

Example with data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| inventory\_tracking\_id | product\_id | transaction\_date | transaction\_type | quantity |
| 1 | 1 | 2024-09-05 | sold | 20 |

1. **Product\_suppliers table**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Product\_suppliers | product\_id | PK/FK - Links the row to a specific product | integer |
| supplier\_id | PK/FK - Links the row to a specific supplier | integer |
| supply\_date | The date when the product was supplied by the supplier, CHECK date should be after 2024-07-01 | date |

Product\_suppliers (M:M): Each product can be linked to multiple suppliers, and each supplier can provide multiple products.

Example with data

|  |  |  |
| --- | --- | --- |
| product\_id | supplier\_id | supply\_date |
| 1 | 1 | 2024-09-01 |