Adisaputra Zidha

STUDY CASE
STORE MANAGEMENT SYSTEM

Mini Project
SQL Database Design and Implementation

BACKGROUND

A local grocery store **Jaya Abadi**, wants to improve its efficiency by implementing a computerized system. The store sells various items such as food, beverages, hygiene products, and other daily necessities. The main objective of this project is to automate the inventory, sales, and purchasing processes.

SYSTEM REQUIREMENTS

1. **Inventory Management**
    - Track stock items,
    - Receive notifications when stock reaches a minimum level.
2. **Sales Management**
    - Process sales transactions,
    - Issuing sales receipts.
3. **Purchasing Management**
    - Record purchases from suppliers,
    - Manage supplier information.

SYSTEM IMPLEMENTATION

1. **Database Creation**
   - Item Table for storing product information,
   - Supplier Table for supplier information,
   - Sales Table for sales transactions,
   - Purchase Table to record the purchase of goods.

2. **Report**
   - Stock report,
   - Daily sales report,
   - Purchase report.

TASK DESCRIPTION

You are required to design and implement a management system for **Jaya Abadi** Store. Create a database schema with the required tables and the relationships between them, use Oracle Database to create a database according to the schema that has been designed. Implement CRUD (Create, Read, Update, Delete) features for each table, and ensure that the system can process sales transactions and manage stock automatically.

OVERVIEW

To design and implement a management system for **Jaya Abadi** Store, I will build a database using **Oracle** and implement CRUD features and a sales transaction system that automatically manages stock. Here are the complete stages of this project.

DATABASE SCHEMA DESIGN

In this case, I created 4 main tables in the database schema, namely Products, Suppliers, Sales, and Purchases. Here is the description of each table:

Table: Products

- id: Primary Key, auto increment.
- product_name: Name of the product.
- category: Product category (food, drinks, hygiene, etc.).
- stock: Current stock quantity.
- purchase_price: Price of the product from the supplier.
- selling_price: Selling price of the product.
- minimum_stock: Minimum stock level before issuing a warning.

Table: Suppliers

- id: Primary Key, auto increment.
- supplier_name: Name of the supplier.
- address: Supplier’s address.
- phone_number: Supplier’s phone number.

DATABASE SCHEMA DESIGN

Table: Sales

- id: Primary Key, auto increment.
- sale_date: Date of the sales transaction.
- total_price: Total price of the sales transaction.

Table: Purchases

- id: Primary Key, auto increment.
- purchase_date: Date of the purchase transaction.
- id_supplier: Foreign Key from the Suppliers table.
- total_cost: Total cost of the purchase.

DATABASE SCHEMA DESIGN

Table: Sales Details

- id: Primary Key, auto increment.
- id_sale: FK from the Sales table.
- id_product: FK from the Products table.
- quantity: Quantity of products sold.
- subtotal: Total price for this product (quantity × selling_price).

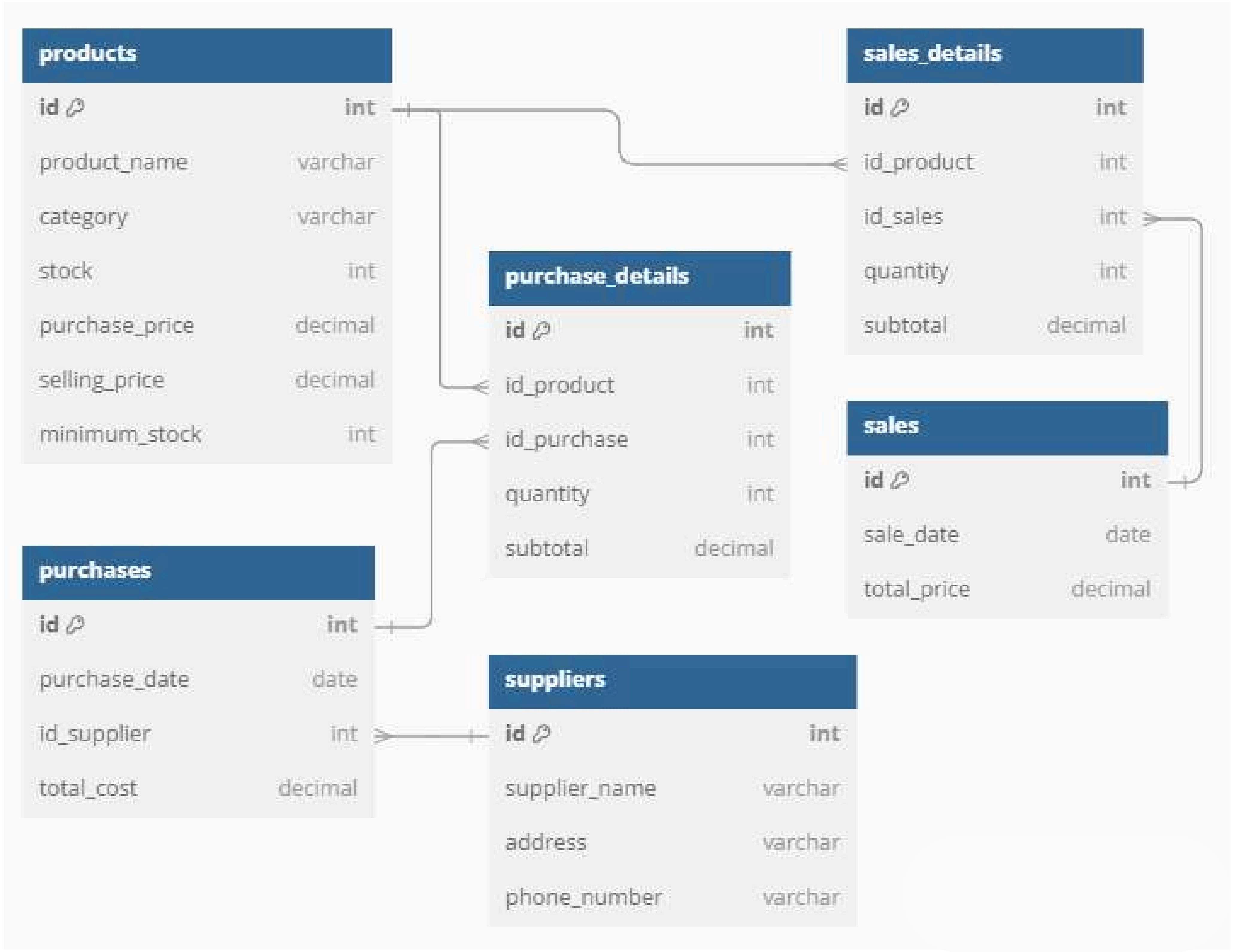
Table: Purchase Details

- id: Primary Key, auto increment.
- id_purchase: FK from the Purchases table.
- id_product: FK from the Products table.
- quantity: Quantity of products purchased.
- subtotal: Total price for this product (quantity × purchase_price).

DATABASE SCHEMA DESIGN

Relationships Between Tables

1. **Sales and Sales Details**
   - Each sale transaction recorded in the Sales table can have multiple details in the Sales_Details table.
2. **Products and Sales Details**
   - Each entry in Sales_Details refers to a specific product from the Products table.
3. **Suppliers and Purchases**
   - Each purchase recorded in the Purchases table is associated with a supplier listed in the Suppliers table.
4. **Purchases and Purchase Details**
   - Each purchase transaction recorded in the Purchases table can have multiple items listed in the Purchase_Details table.
5. **Products and Purchase Details**
   - Each entry in Purchase_Details refers to a specific product from the Products table.



DATABASE SCHEMA DESIGN

TABLE STRUCTURE IN ORACLE

```
-- Table: Products
CREATE TABLE products (
  id NUMBER GENERATED BY DEFAULT
     AS IDENTITY PRIMARY KEY,
  product_name VARCHAR2(255) NOT NULL,
  category VARCHAR2(100),
  stock NUMBER NOT NULL,
  purchase_price NUMBER(10, 2),
  selling_price NUMBER(10, 2),
  minimum_stock NUMBER
);
```

TABLE STRUCTURE IN ORACLE

```
-- Table: Suppliers
CREATE TABLE suppliers (
  id NUMBER GENERATED BY DEFAULT
     AS IDENTITY PRIMARY KEY,
  supplier_name VARCHAR2(255) NOT NULL,
  address VARCHAR2(255),
  phone_number VARCHAR2(20)
);
```

TABLE STRUCTURE IN ORACLE

```
-- Table: Sales
CREATE TABLE sales (
  id NUMBER GENERATED BY DEFAULT
     AS IDENTITY PRIMARY KEY,
  sale_date DATE DEFAULT SYSDATE
            NOT NULL,
  total_price NUMBER(10, 2)
);
```

TABLE STRUCTURE IN ORACLE

```
-- Table: Sales Detail
CREATE TABLE sales_details (
  id NUMBER GENERATED BY DEFAULT
     AS IDENTITY PRIMARY KEY,
  id_product NUMBER NOT NULL,
  id_sales NUMBER NOT NULL,
  quantity NUMBER NOT NULL,
  product_price_at_sale NUMBER(10, 2),
  subtotal NUMBER(10, 2),

  CONSTRAINT fk_product_sales FOREIGN KEY
     (id_product) REFERENCES products(id),
  CONSTRAINT fk_sales_details FOREIGN KEY
     (id_sales) REFERENCES sales(id)
);
```

TABLE STRUCTURE IN ORACLE

```
-- Table: Purchases
CREATE TABLE Purchases (
  id NUMBER GENERATED BY DEFAULT
     AS IDENTITY PRIMARY KEY,
  purchase_date DATE DEFAULT SYSDATE
     NOT NULL,
  id_supplier NUMBER NOT NULL,
  total_cost NUMBER(10, 2),

  CONSTRAINT fk_supplier FOREIGN KEY
     (id_supplier) REFERENCES
     suppliers(id)
);
```

TABLE STRUCTURE IN ORACLE

```
-- Table: Purchase Detail
CREATE TABLE purchase_details (
  id NUMBER GENERATED BY DEFAULT
     AS IDENTITY PRIMARY KEY,
  id_product NUMBER NOT NULL,
  id_purchase NUMBER NOT NULL,
  quantity NUMBER NOT NULL,
  subtotal NUMBER(10, 2),

  CONSTRAINT fk_product_purchase
     FOREIGN KEY (id_product)
     REFERENCES products(id),
  CONSTRAINT fk_purchase_details
     FOREIGN KEY (id_purchase)
     REFERENCES purchases(id)
);
```

CRUD FEATURE EXAMPLE

CREATE

```
INSERT INTO products (product_name, category_id, stock, purchase_price, selling_price, minimum_stock)
VALUES (:PRODUCT_NAME, :CATEGORY_ID, :STOCK, :PURCHASE_PRICE, :SELLING_PRICE, :MINIMUM_STOCK);

INSERT INTO suppliers (supplier_name, address, phone_number)
VALUES (:SUPPLIER_NAME, :ADDRESS, :PHONE_NUMBER);

INSERT INTO sales (sale_date, total_price, id_user)
VALUES (SYSDATE, :TOTAL_PRICE, :USER_ID);

INSERT INTO purchases (purchase_date, id_supplier, total_cost, id_user)
VALUES (SYSDATE, :SUPPLIER_ID, :TOTAL_COST, :USER_ID);
```

CRUD FEATURE EXAMPLE

READ

```
SELECT * FROM products;

SELECT * FROM sales;

SELECT * FROM purchases;

SELECT * FROM suppliers;
```

UPDATE

```
UPDATE products
  SET stock = :NEW_STOCK,
  selling_price = :NEW_SELL_PRICE
  WHERE id = :PRODUCT_ID;
```

DELETE

```
DELETE FROM products
  WHERE id = :PRODUCT_ID;
```

AUTOMATION IN STOCK MANAGEMENT

Reducing Stock After Sale

```
CREATE OR REPLACE TRIGGER
  trg_reduce_stock_after_sale
AFTER INSERT ON sales_details
FOR EACH ROW
BEGIN
  UPDATE products
  SET stock = stock -
     :NEW.quantity
  WHERE id = :NEW.id_product;
END;
```

Adding Stock After Purchase

```
CREATE OR REPLACE TRIGGER
  trg_add_stock_after_purchase
AFTER INSERT ON purchase_details
FOR EACH ROW
BEGIN
  UPDATE products
  SET stock = stock +
     :NEW.quantity
  WHERE id = :NEW.id_product;
END;
```

Stock Notification

```
SELECT product_name,
  stock, minimum_stock
FROM products
WHERE stock <=
  minimum_stock;
```

REPORTING

Stock Report

```
SELECT product_name, stock, minimum_stock
FROM products;
```

Daily Sales Report

```
SELECT s.id AS sale_id, s.sale_date, p.product_name, sd.quantity,
       p.selling_price AS product_price_at_sale, (sd.quantity * p.selling_price) AS subtotal
FROM sales s JOIN sales_details sd ON s.id = sd.id_sales
JOIN products p ON sd.id_product = p.id
WHERE TRUNC(s.sale_date) = TRUNC(SYSDATE);
```

Purchase Report

```
SELECT p.purchase_date, sup.supplier_name, pd.quantity, pd.subtotal
FROM purchases p JOIN purchase_details pd ON p.id = pd.id_purchase
JOIN suppliers sup ON p.id_supplier = sup.id
WHERE TRUNC(p.purchase_date) = TRUNC(SYSDATE);
```

THANKYOU &

LET'S CONNECT !

hizidha.site

hi.zidha@gmail.com

linkedin.com/in/adisaputrazidha