THE YOUNGS PIZZA MANAGEMENT SYSTEM

Yaw Dapaa & Genevieve Donkor Armah

Problem Statement & Purpose

Youngs Pizza is a growing family-owned pizzeria

Still relies on handwritten orders

System has become messy, inefficient and prone to mistakes in orders

Build a centralized, automated and reliable relational database system to streamline operations and optimize all operations using a relational database.

Technical Requirements

Database System

Oracle Database

Selected for robustness and reliability in business environments

Development Environment

Oracle SQL Developer

Used for designing, developing and maintaining the database

Database Normalization

Third Normal Form (3NF)

Eliminates redundancy and prevents update anomalies

Core Database Entities

Customer
Stores customer information

Employee
Stores staff details

MenuItem
Menu items and pricing

Orders
Order information

OrderItem
Junction table for orders and menu items

Payment
Payment details

Database Design & Entity Relationships

Key Relationships

Customer to Orders
One customer places many orders. CustomerID is the foreign key in Order.

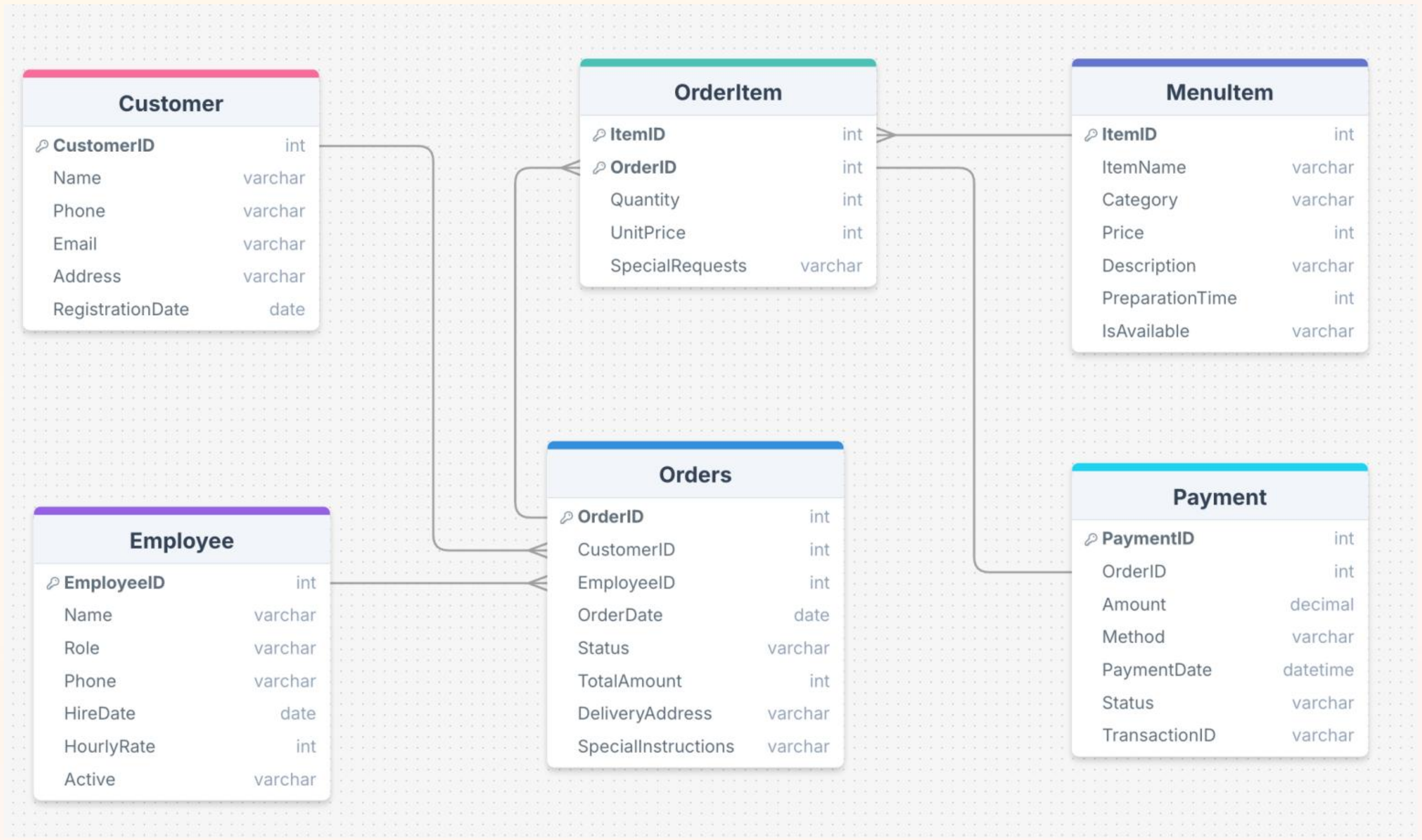
Employee to Orders
One employee handles many orders. EmployeeID is the foreign key in Order.

Orders to OrderItem
One order has many order items. OrderID is the foreign key in OrderItem.

MenuItem to OrderItem
One menu item can appear in many order items. ItemID is the foreign key in OrderItem.

Order to Payment
One order can have many payments. OrderID is the foreign key in Payment.

Entity Relationship Diagram



Customer	
CustomerID	int
Name	varchar
Phone	varchar
Email	varchar
Address	varchar
RegistrationDate	date

OrderItem	
ItemID	int
OrderID	int
Quantity	int
UnitPrice	int
SpecialRequests	varchar

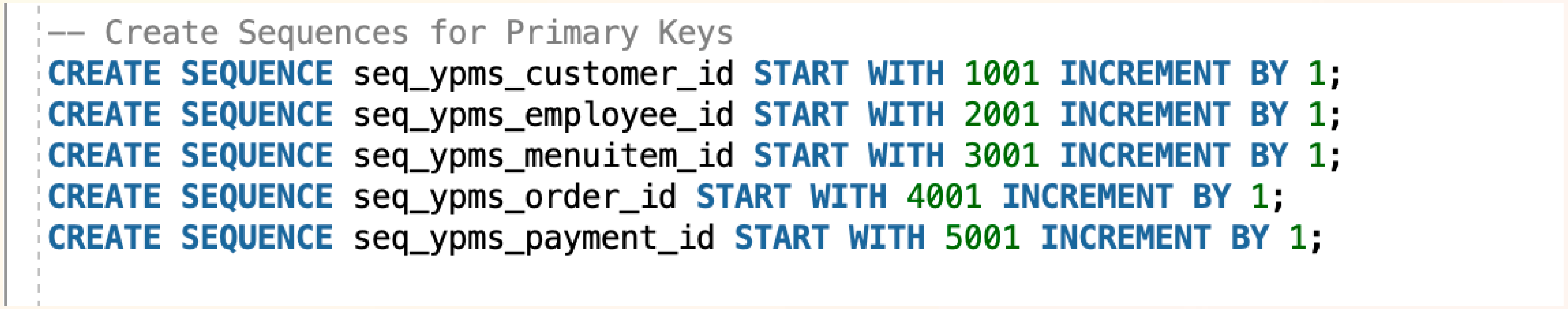
MenuItem	
ItemID	int
ItemName	varchar
Category	varchar
Price	int
Description	varchar
PreparationTime	int
IsAvailable	varchar

Employee	
EmployeeID	int
Name	varchar
Role	varchar
Phone	varchar
HireDate	date
HourlyRate	int
Active	varchar

Orders	
OrderID	int
CustomerID	int
EmployeeID	int
OrderDate	date
Status	varchar
TotalAmount	int
DeliveryAddress	varchar
SpecialInstructions	varchar

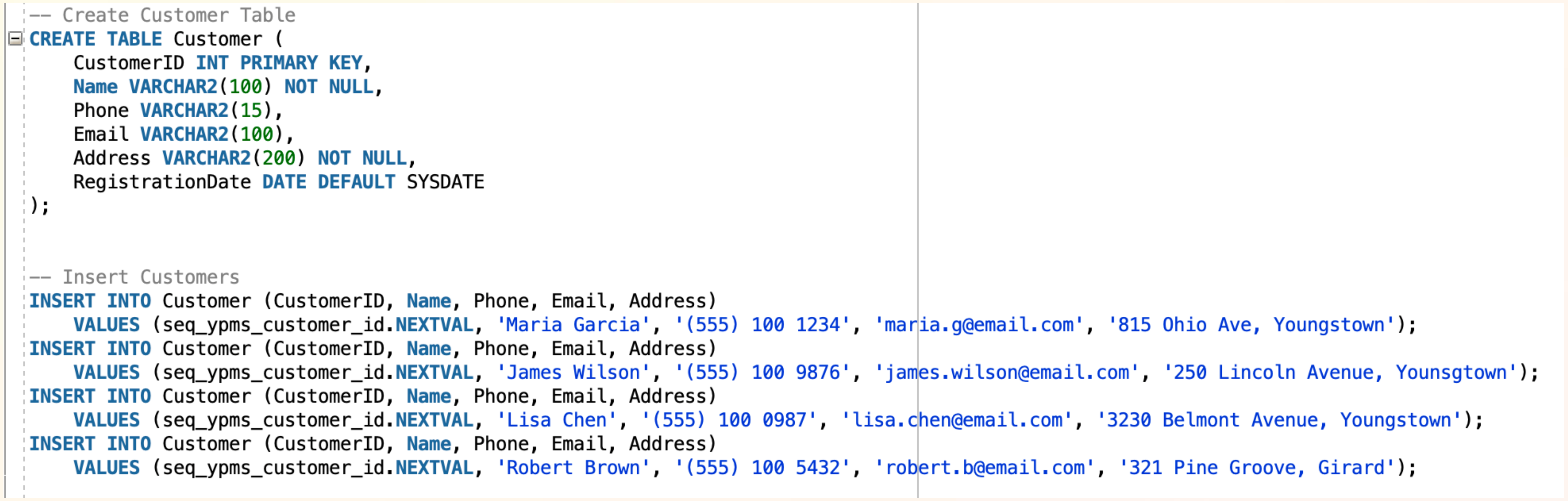
Payment	
PaymentID	int
OrderID	int
Amount	decimal
Method	varchar
PaymentDate	datetime
Status	varchar
TransactionID	varchar

Create Sequence For the Primary Keys



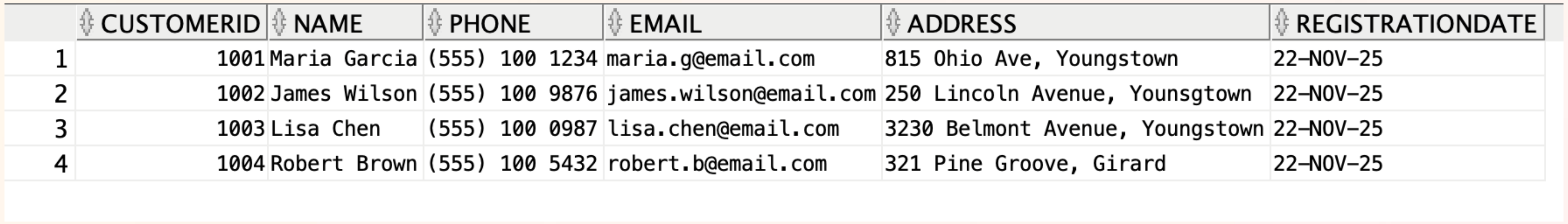
```
-- Create Sequences for Primary Keys
CREATE SEQUENCE seq_ypms_customer_id START WITH 1001 INCREMENT BY 1;
CREATE SEQUENCE seq_ypms_employee_id START WITH 2001 INCREMENT BY 1;
CREATE SEQUENCE seq_ypms_menuitem_id START WITH 3001 INCREMENT BY 1;
CREATE SEQUENCE seq_ypms_order_id START WITH 4001 INCREMENT BY 1;
CREATE SEQUENCE seq_ypms_payment_id START WITH 5001 INCREMENT BY 1;
```

Create Customer Table and Sample Inserts



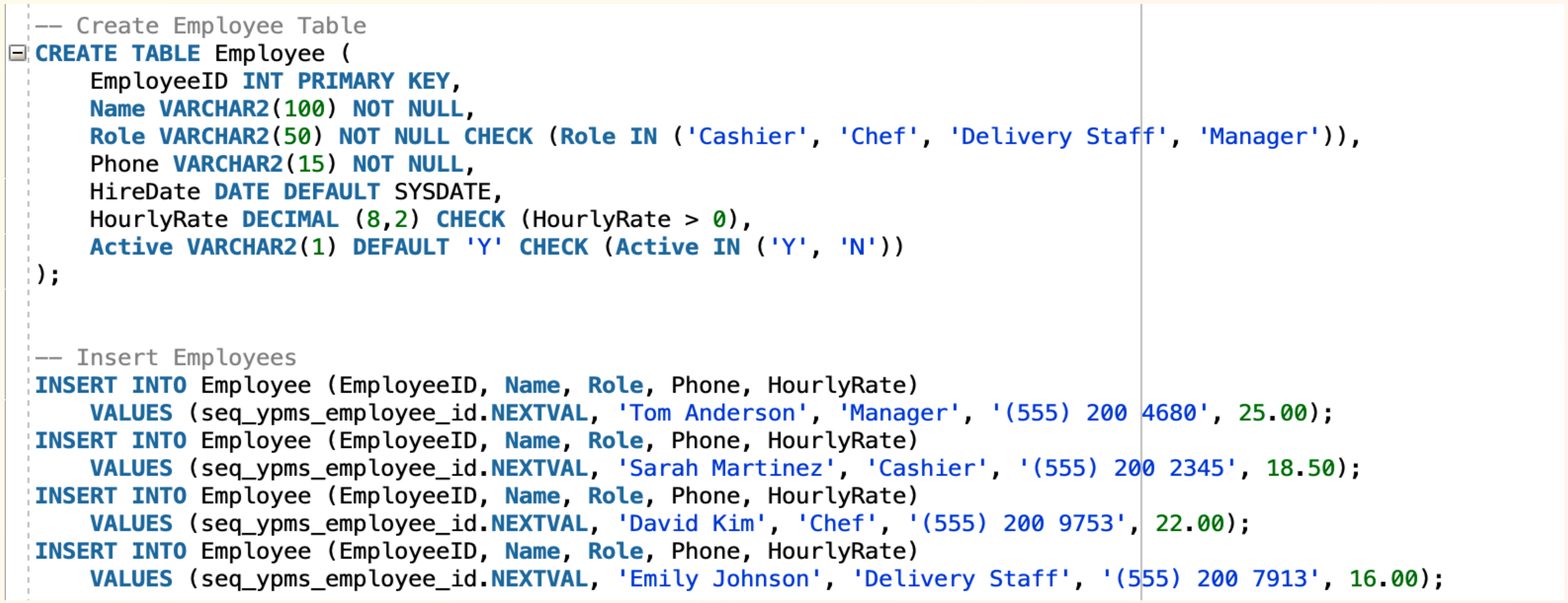
```sql
-- Create Customer Table
CREATE TABLE Customer (
    CustomerID INT PRIMARY KEY,
    Name VARCHAR2(100) NOT NULL,
    Phone VARCHAR2(15),
    Email VARCHAR2(100),
    Address VARCHAR2(200) NOT NULL,
    RegistrationDate DATE DEFAULT SYSDATE
);


-- Insert Customers
INSERT INTO Customer (CustomerID, Name, Phone, Email, Address)
    VALUES (seq_ypms_customer_id.NEXTVAL, 'Maria Garcia', '(555) 100 1234', 'maria.g@email.com', '815 Ohio Ave, Youngstown');
INSERT INTO Customer (CustomerID, Name, Phone, Email, Address)
    VALUES (seq_ypms_customer_id.NEXTVAL, 'James Wilson', '(555) 100 9876', 'james.wilson@email.com', '250 Lincoln Avenue, Younsgtown');
INSERT INTO Customer (CustomerID, Name, Phone, Email, Address)
    VALUES (seq_ypms_customer_id.NEXTVAL, 'Lisa Chen', '(555) 100 0987', 'lisa.chen@email.com', '3230 Belmont Avenue, Youngstown');
INSERT INTO Customer (CustomerID, Name, Phone, Email, Address)
    VALUES (seq_ypms_customer_id.NEXTVAL, 'Robert Brown', '(555) 100 5432', 'robert.b@email.com', '321 Pine Groove, Girard');
```

Output:



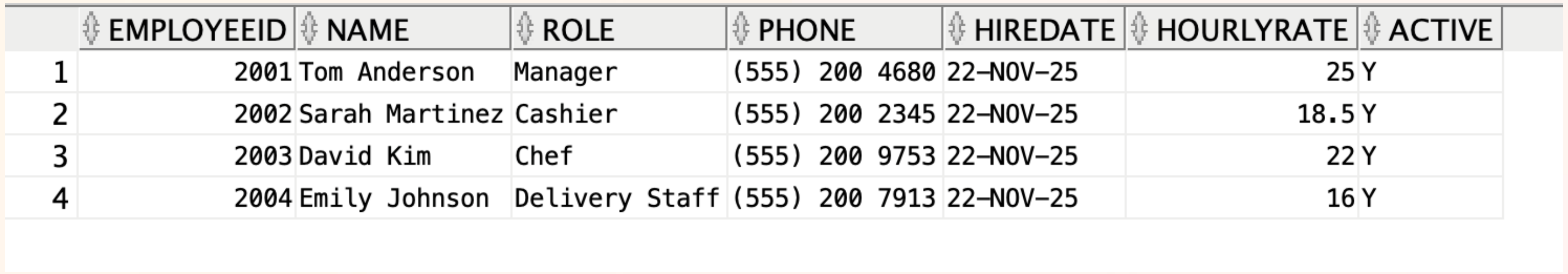
	CUSTOMERID	NAME	PHONE	EMAIL	ADDRESS	REGISTRATIONDATE
1	1001	Maria Garcia	(555) 100 1234	maria.g@email.com	815 Ohio Ave, Youngstown	22-NOV-25
2	1002	James Wilson	(555) 100 9876	james.wilson@email.com	250 Lincoln Avenue, Younsgtown	22-NOV-25
3	1003	Lisa Chen	(555) 100 0987	lisa.chen@email.com	3230 Belmont Avenue, Youngstown	22-NOV-25
4	1004	Robert Brown	(555) 100 5432	robert.b@email.com	321 Pine Groove, Girard	22-NOV-25

Create Employee Table and Sample Inserts



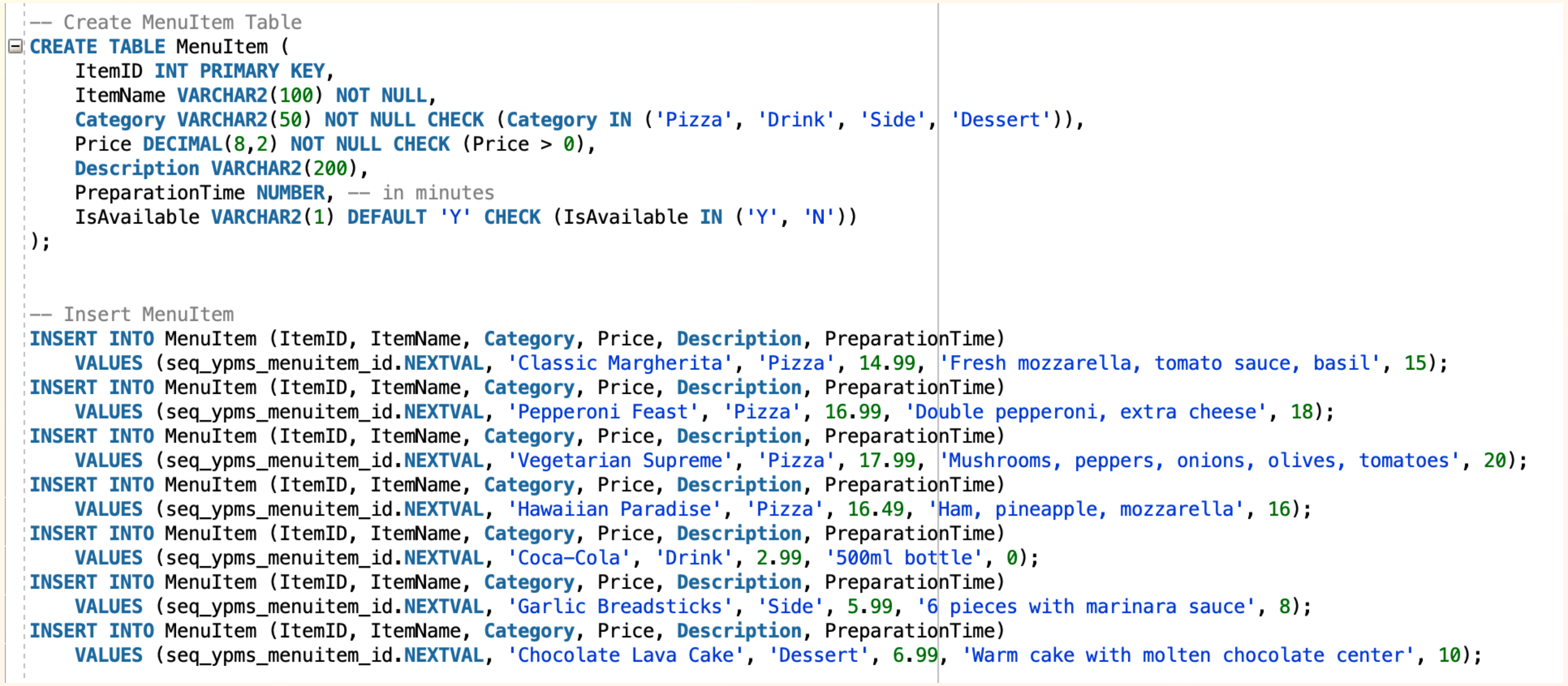
```sql
-- Create Employee Table
CREATE TABLE Employee (
    EmployeeID INT PRIMARY KEY,
    Name VARCHAR2(100) NOT NULL,
    Role VARCHAR2(50) NOT NULL CHECK (Role IN ('Cashier', 'Chef', 'Delivery Staff', 'Manager')),
    Phone VARCHAR2(15) NOT NULL,
    HireDate DATE DEFAULT SYSDATE,
    HourlyRate DECIMAL (8,2) CHECK (HourlyRate > 0),
    Active VARCHAR2(1) DEFAULT 'Y' CHECK (Active IN ('Y', 'N'))
);


-- Insert Employees
INSERT INTO Employee (EmployeeID, Name, Role, Phone, HourlyRate)
    VALUES (seq_ypms_employee_id.NEXTVAL, 'Tom Anderson', 'Manager', '(555) 200 4680', 25.00);
INSERT INTO Employee (EmployeeID, Name, Role, Phone, HourlyRate)
    VALUES (seq_ypms_employee_id.NEXTVAL, 'Sarah Martinez', 'Cashier', '(555) 200 2345', 18.50);
INSERT INTO Employee (EmployeeID, Name, Role, Phone, HourlyRate)
    VALUES (seq_ypms_employee_id.NEXTVAL, 'David Kim', 'Chef', '(555) 200 9753', 22.00);
INSERT INTO Employee (EmployeeID, Name, Role, Phone, HourlyRate)
    VALUES (seq_ypms_employee_id.NEXTVAL, 'Emily Johnson', 'Delivery Staff', '(555) 200 7913', 16.00);
```

Output:



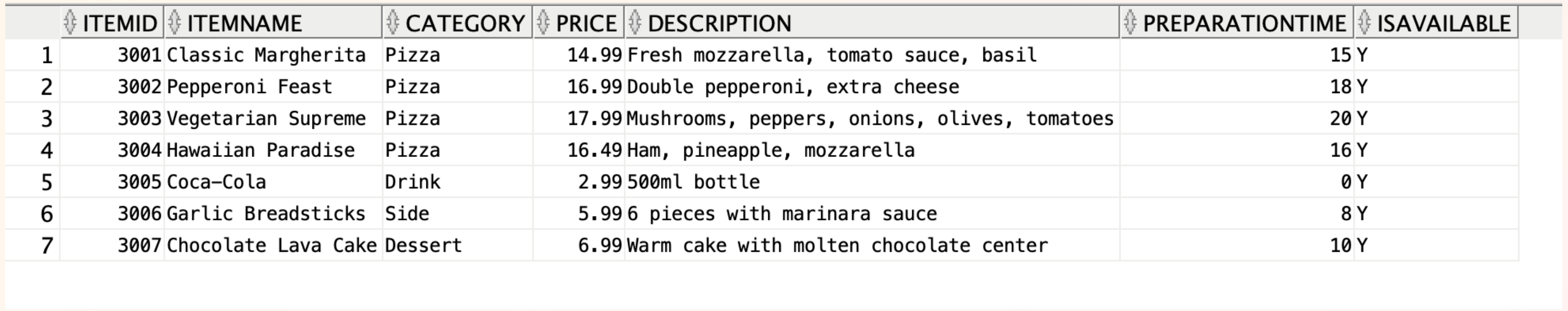
	EMPLOYEEID	NAME	ROLE	PHONE	HIREDATE	HOURLYRATE	ACTIVE
1	2001	Tom Anderson	Manager	(555) 200 4680	22-NOV-25	25	Y
2	2002	Sarah Martinez	Cashier	(555) 200 2345	22-NOV-25	18.5	Y
3	2003	David Kim	Chef	(555) 200 9753	22-NOV-25	22	Y
4	2004	Emily Johnson	Delivery Staff	(555) 200 7913	22-NOV-25	16	Y

Create Menu Item Table and Sample Inserts



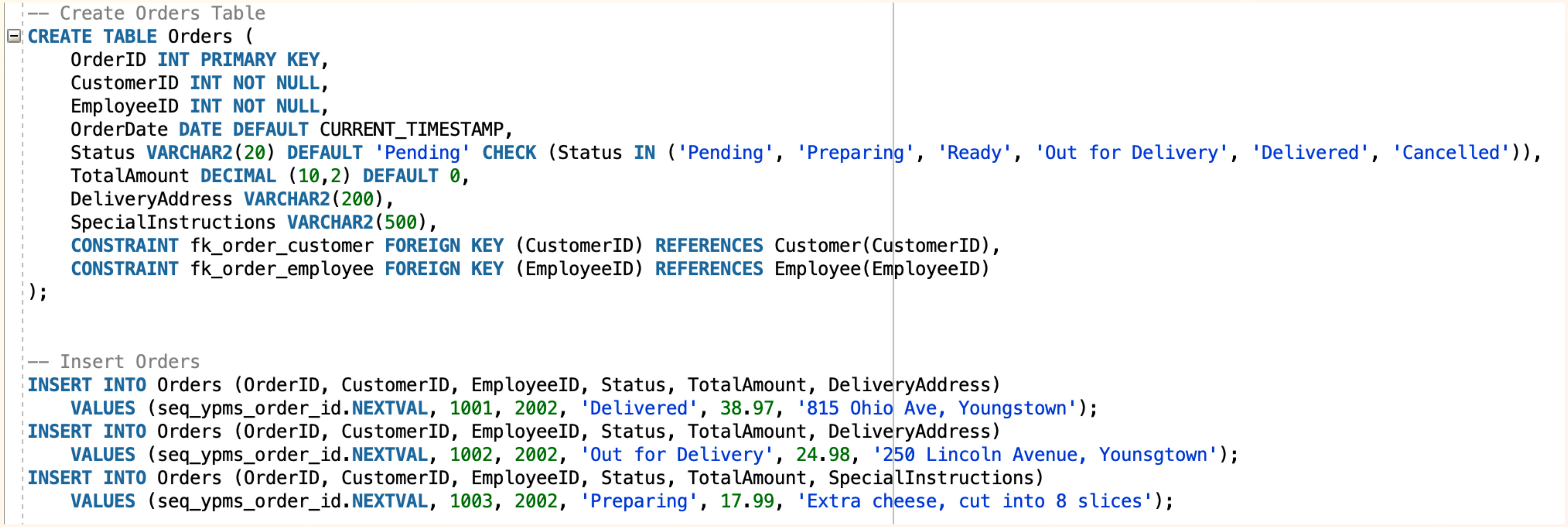
```
-- Create MenuItem Table
CREATE TABLE MenuItem (
    ItemID INT PRIMARY KEY,
    ItemName VARCHAR2(100) NOT NULL,
    Category VARCHAR2(50) NOT NULL CHECK (Category IN ('Pizza', 'Drink', 'Side', 'Dessert')),
    Price DECIMAL(8,2) NOT NULL CHECK (Price > 0),
    Description VARCHAR2(200),
    PreparationTime NUMBER, -- in minutes
    IsAvailable VARCHAR2(1) DEFAULT 'Y' CHECK (IsAvailable IN ('Y', 'N'))
);


-- Insert MenuItem
INSERT INTO MenuItem (ItemID, ItemName, Category, Price, Description, PreparationTime)
    VALUES (seq_ypms_menuitem_id.NEXTVAL, 'Classic Margherita', 'Pizza', 14.99, 'Fresh mozzarella, tomato sauce, basil', 15);
INSERT INTO MenuItem (ItemID, ItemName, Category, Price, Description, PreparationTime)
    VALUES (seq_ypms_menuitem_id.NEXTVAL, 'Pepperoni Feast', 'Pizza', 16.99, 'Double pepperoni, extra cheese', 18);
INSERT INTO MenuItem (ItemID, ItemName, Category, Price, Description, PreparationTime)
    VALUES (seq_ypms_menuitem_id.NEXTVAL, 'Vegetarian Supreme', 'Pizza', 17.99, 'Mushrooms, peppers, onions, olives, tomatoes', 20);
INSERT INTO MenuItem (ItemID, ItemName, Category, Price, Description, PreparationTime)
    VALUES (seq_ypms_menuitem_id.NEXTVAL, 'Hawaiian Paradise', 'Pizza', 16.49, 'Ham, pineapple, mozzarella', 16);
INSERT INTO MenuItem (ItemID, ItemName, Category, Price, Description, PreparationTime)
    VALUES (seq_ypms_menuitem_id.NEXTVAL, 'Coca-Cola', 'Drink', 2.99, '500ml bottle', 0);
INSERT INTO MenuItem (ItemID, ItemName, Category, Price, Description, PreparationTime)
    VALUES (seq_ypms_menuitem_id.NEXTVAL, 'Garlic Breadsticks', 'Side', 5.99, '6 pieces with marinara sauce', 8);
INSERT INTO MenuItem (ItemID, ItemName, Category, Price, Description, PreparationTime)
    VALUES (seq_ypms_menuitem_id.NEXTVAL, 'Chocolate Lava Cake', 'Dessert', 6.99, 'Warm cake with molten chocolate center', 10);
```

Output:



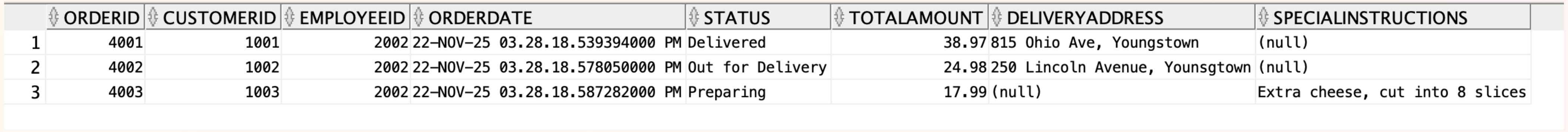
	ITEMID	ITEMNAME	CATEGORY	PRICE	DESCRIPTION	PREPARATIONTIME	ISAVAILABLE
1	3001	Classic Margherita	Pizza	14.99	Fresh mozzarella, tomato sauce, basil	15	Y
2	3002	Pepperoni Feast	Pizza	16.99	Double pepperoni, extra cheese	18	Y
3	3003	Vegetarian Supreme	Pizza	17.99	Mushrooms, peppers, onions, olives, tomatoes	20	Y
4	3004	Hawaiian Paradise	Pizza	16.49	Ham, pineapple, mozzarella	16	Y
5	3005	Coca-Cola	Drink	2.99	500ml bottle	0	Y
6	3006	Garlic Breadsticks	Side	5.99	6 pieces with marinara sauce	8	Y
7	3007	Chocolate Lava Cake	Dessert	6.99	Warm cake with molten chocolate center	10	Y

Create Orders Table and Sample Inserts



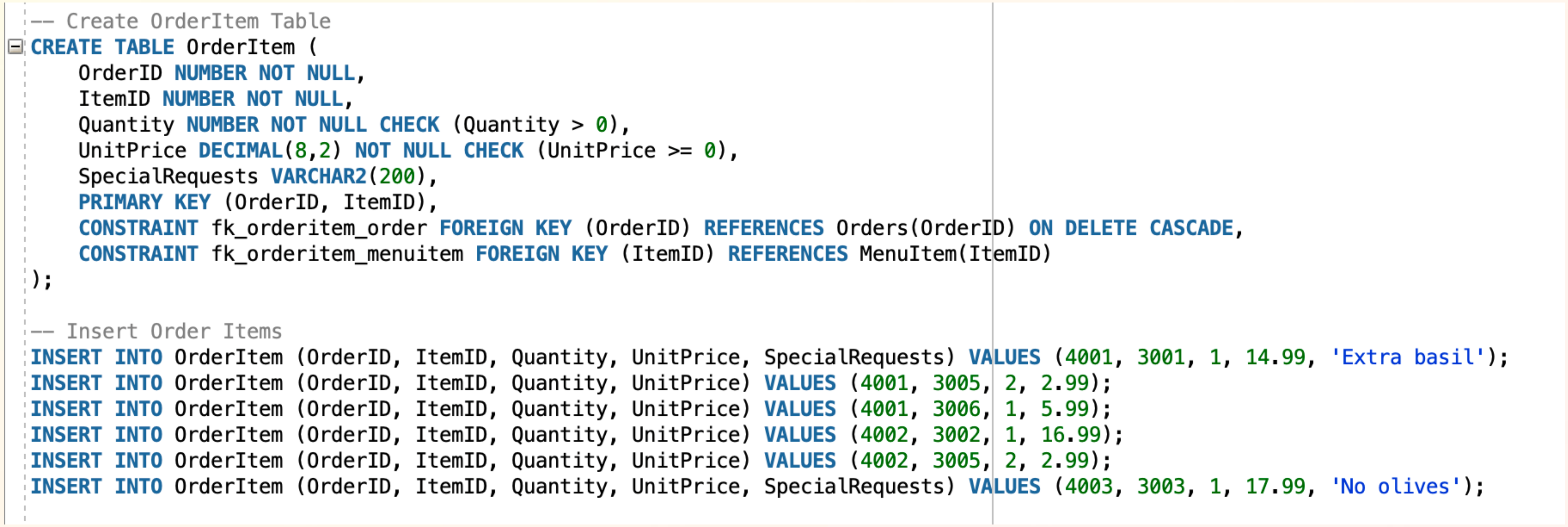
```sql
-- Create Orders Table
CREATE TABLE Orders (
    OrderID INT PRIMARY KEY,
    CustomerID INT NOT NULL,
    EmployeeID INT NOT NULL,
    OrderDate DATE DEFAULT CURRENT_TIMESTAMP,
    Status VARCHAR2(20) DEFAULT 'Pending' CHECK (Status IN ('Pending', 'Preparing', 'Ready', 'Out for Delivery', 'Delivered', 'Cancelled')),
    TotalAmount DECIMAL (10,2) DEFAULT 0,
    DeliveryAddress VARCHAR2(200),
    SpecialInstructions VARCHAR2(500),
    CONSTRAINT fk_order_customer FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID),
    CONSTRAINT fk_order_employee FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)
);


-- Insert Orders
INSERT INTO Orders (OrderID, CustomerID, EmployeeID, Status, TotalAmount, DeliveryAddress)
    VALUES (seq_ypms_order_id.NEXTVAL, 1001, 2002, 'Delivered', 38.97, '815 Ohio Ave, Youngstown');
INSERT INTO Orders (OrderID, CustomerID, EmployeeID, Status, TotalAmount, DeliveryAddress)
    VALUES (seq_ypms_order_id.NEXTVAL, 1002, 2002, 'Out for Delivery', 24.98, '250 Lincoln Avenue, Younsgtown');
INSERT INTO Orders (OrderID, CustomerID, EmployeeID, Status, TotalAmount, SpecialInstructions)
    VALUES (seq_ypms_order_id.NEXTVAL, 1003, 2002, 'Preparing', 17.99, 'Extra cheese, cut into 8 slices');
```

Output:



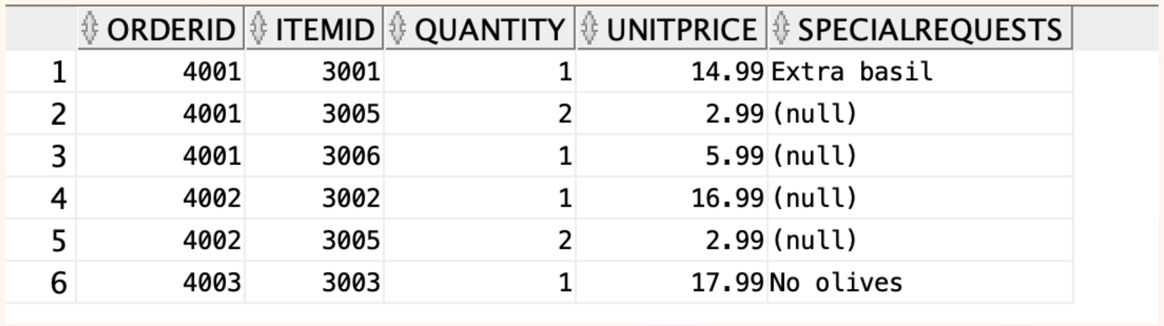
	ORDERID	CUSTOMERID	EMPLOYEEID	ORDERDATE	STATUS	TOTALAMOUNT	DELIVERYADDRESS	SPECIALINSTRUCTIONS
1	4001	1001	2002	22-NOV-25 03.28.18.539394000 PM	Delivered	38.97	815 Ohio Ave, Youngstown	(null)
2	4002	1002	2002	22-NOV-25 03.28.18.578050000 PM	Out for Delivery	24.98	250 Lincoln Avenue, Younsgtown	(null)
3	4003	1003	2002	22-NOV-25 03.28.18.587282000 PM	Preparing	17.99	(null)	Extra cheese, cut into 8 slices

Create Order Item Table and Sample Inserts



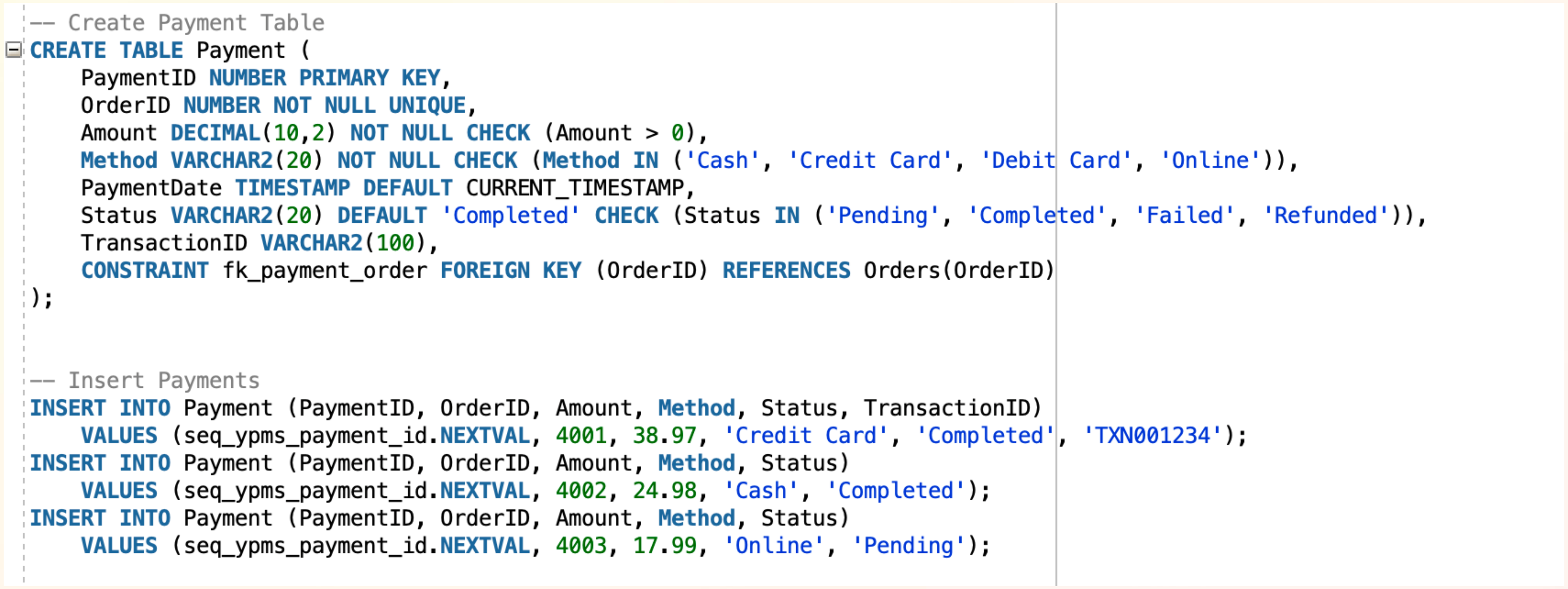
```
-- Create OrderItem Table
CREATE TABLE OrderItem (
    OrderID NUMBER NOT NULL,
    ItemID NUMBER NOT NULL,
    Quantity NUMBER NOT NULL CHECK (Quantity > 0),
    UnitPrice DECIMAL(8,2) NOT NULL CHECK (UnitPrice >= 0),
    SpecialRequests VARCHAR2(200),
    PRIMARY KEY (OrderID, ItemID),
    CONSTRAINT fk_orderitem_order FOREIGN KEY (OrderID) REFERENCES Orders(OrderID) ON DELETE CASCADE,
    CONSTRAINT fk_orderitem_menuitem FOREIGN KEY (ItemID) REFERENCES MenuItem(ItemID)
);

-- Insert Order Items
INSERT INTO OrderItem (OrderID, ItemID, Quantity, UnitPrice, SpecialRequests) VALUES (4001, 3001, 1, 14.99, 'Extra basil');
INSERT INTO OrderItem (OrderID, ItemID, Quantity, UnitPrice) VALUES (4001, 3005, 2, 2.99);
INSERT INTO OrderItem (OrderID, ItemID, Quantity, UnitPrice) VALUES (4001, 3006, 1, 5.99);
INSERT INTO OrderItem (OrderID, ItemID, Quantity, UnitPrice) VALUES (4002, 3002, 1, 16.99);
INSERT INTO OrderItem (OrderID, ItemID, Quantity, UnitPrice) VALUES (4002, 3005, 2, 2.99);
INSERT INTO OrderItem (OrderID, ItemID, Quantity, UnitPrice, SpecialRequests) VALUES (4003, 3003, 1, 17.99, 'No olives');
```

Output:



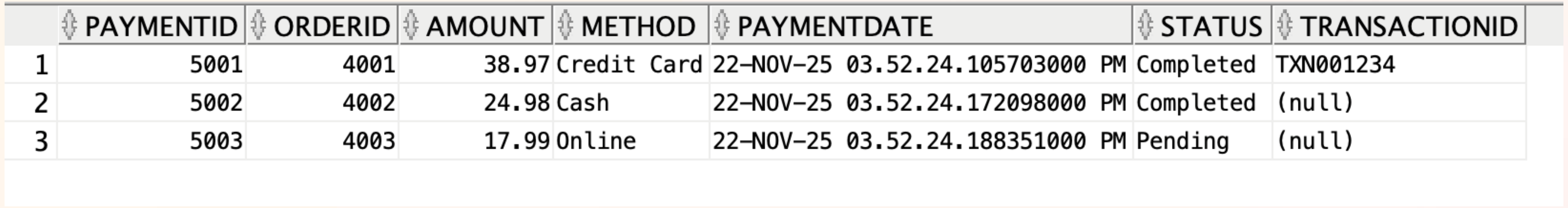
	ORDERID	ITEMID	QUANTITY	UNITPRICE	SPECIALREQUESTS
1	4001	3001	1	14.99	Extra basil
2	4001	3005	2	2.99	(null)
3	4001	3006	1	5.99	(null)
4	4002	3002	1	16.99	(null)
5	4002	3005	2	2.99	(null)
6	4003	3003	1	17.99	No olives

Create Payment Table and Sample Inserts



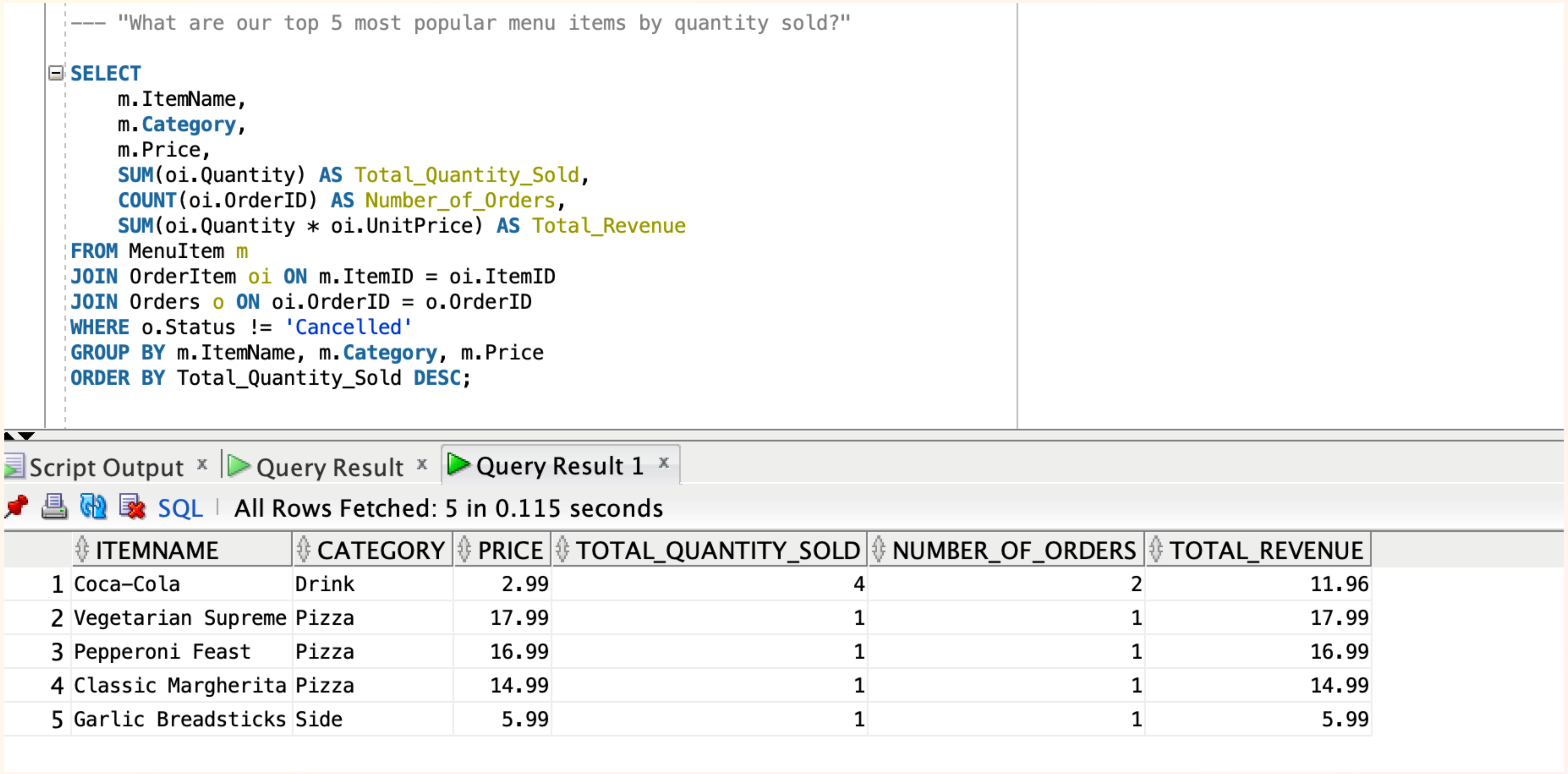
```
-- Create Payment Table
CREATE TABLE Payment (
    PaymentID NUMBER PRIMARY KEY,
    OrderID NUMBER NOT NULL UNIQUE,
    Amount DECIMAL(10,2) NOT NULL CHECK (Amount > 0),
    Method VARCHAR2(20) NOT NULL CHECK (Method IN ('Cash', 'Credit Card', 'Debit Card', 'Online')),
    PaymentDate TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    Status VARCHAR2(20) DEFAULT 'Completed' CHECK (Status IN ('Pending', 'Completed', 'Failed', 'Refunded')),
    TransactionID VARCHAR2(100),
    CONSTRAINT fk_payment_order FOREIGN KEY (OrderID) REFERENCES Orders(OrderID)
);


-- Insert Payments
INSERT INTO Payment (PaymentID, OrderID, Amount, Method, Status, TransactionID)
    VALUES (seq_ypms_payment_id.NEXTVAL, 4001, 38.97, 'Credit Card', 'Completed', 'TXN001234');
INSERT INTO Payment (PaymentID, OrderID, Amount, Method, Status)
    VALUES (seq_ypms_payment_id.NEXTVAL, 4002, 24.98, 'Cash', 'Completed');
INSERT INTO Payment (PaymentID, OrderID, Amount, Method, Status)
    VALUES (seq_ypms_payment_id.NEXTVAL, 4003, 17.99, 'Online', 'Pending');
```

Output:



	PAYMENTID	ORDERID	AMOUNT	METHOD	PAYMENTDATE	STATUS	TRANSACTIONID
1	5001	4001	38.97	Credit Card	22-NOV-25 03.52.24.105703000 PM	Completed	TXN001234
2	5002	4002	24.98	Cash	22-NOV-25 03.52.24.172098000 PM	Completed	(null)
3	5003	4003	17.99	Online	22-NOV-25 03.52.24.188351000 PM	Pending	(null)

Scenario 1: Finding the top 5 most popular menu items by quantity sold

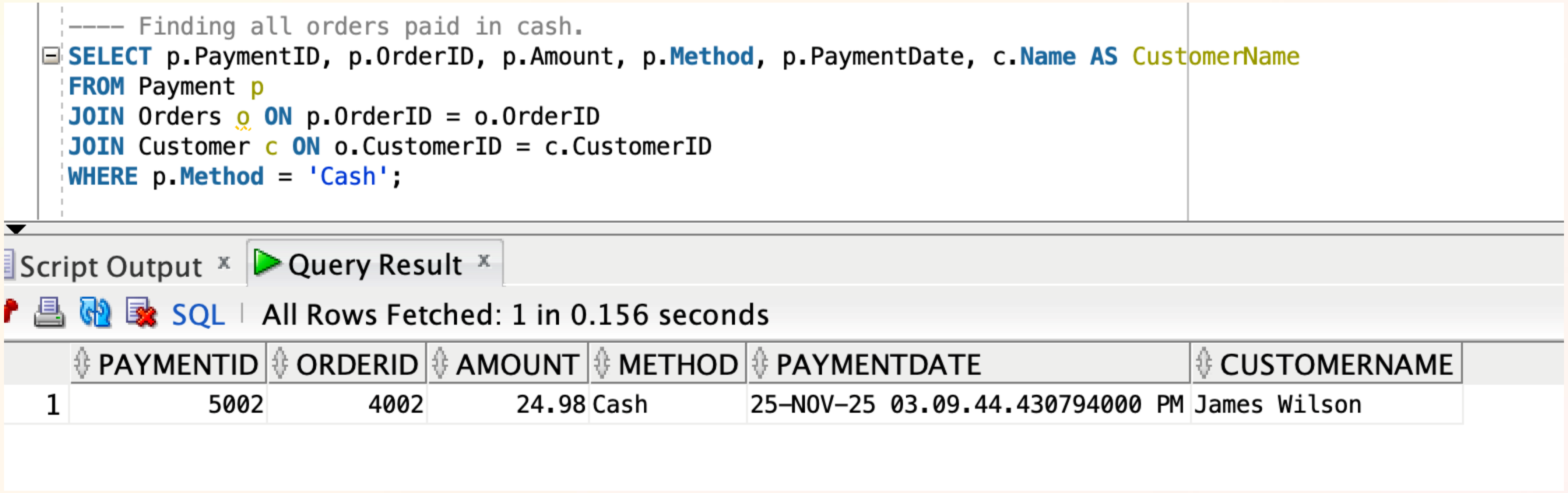


```
--- "What are our top 5 most popular menu items by quantity sold?"

SELECT
    m.ItemName,
    m.Category,
    m.Price,
    SUM(oi.Quantity) AS Total_Quantity_Sold,
    COUNT(oi.OrderID) AS Number_of_Orders,
    SUM(oi.Quantity * oi.UnitPrice) AS Total_Revenue
FROM MenuItem m
JOIN OrderItem oi ON m.ItemID = oi.ItemID
JOIN Orders o ON oi.OrderID = o.OrderID
WHERE o.Status != 'Cancelled'
GROUP BY m.ItemName, m.Category, m.Price
ORDER BY Total_Quantity_Sold DESC;
```

All Rows Fetched: 5 in 0.115 seconds

	ITEMNAME	CATEGORY	PRICE	TOTAL_QUANTITY_SOLD	NUMBER_OF_ORDERS	TOTAL_REVENUE
1	Coca-Cola	Drink	2.99	4	2	11.96
2	Vegetarian Supreme	Pizza	17.99	1	1	17.99
3	Pepperoni Feast	Pizza	16.99	1	1	16.99
4	Classic Margherita	Pizza	14.99	1	1	14.99
5	Garlic Breadsticks	Side	5.99	1	1	5.99

Scenario 2: Orders paid in Cash

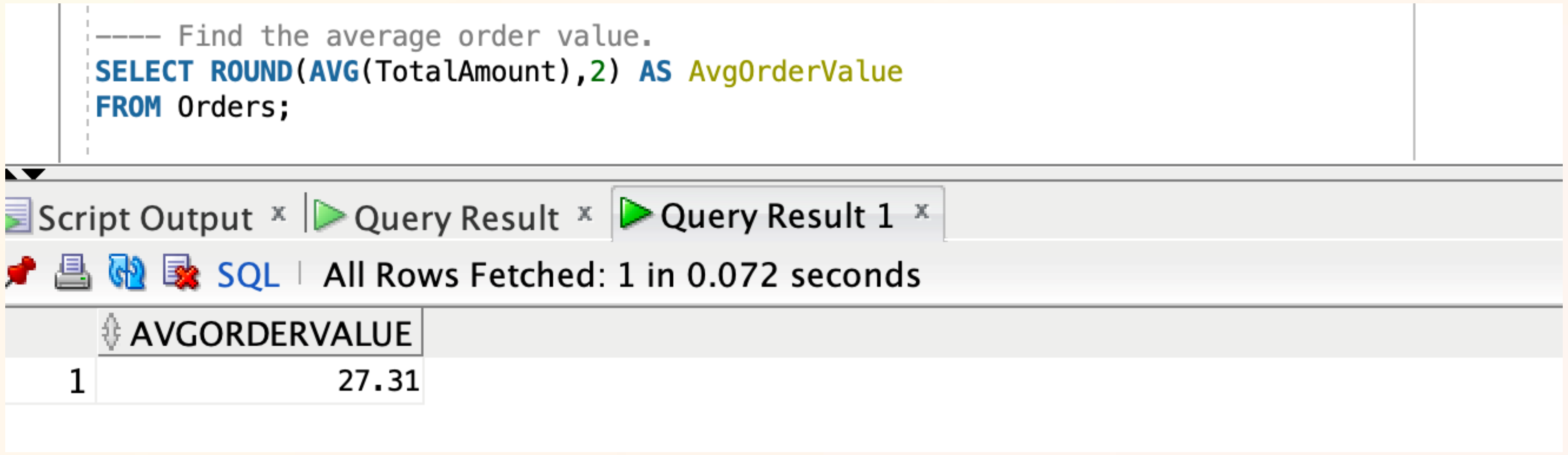


```
---- Finding all orders paid in cash.
SELECT p.PaymentID, p.OrderID, p.Amount, p.Method, p.PaymentDate, c.Name AS CustomerName
FROM Payment p
JOIN Orders o ON p.OrderID = o.OrderID
JOIN Customer c ON o.CustomerID = c.CustomerID
WHERE p.Method = 'Cash';
```

All Rows Fetched: 1 in 0.156 seconds

	PAYMENTID	ORDERID	AMOUNT	METHOD	PAYMENTDATE	CUSTOMERNAME
1	5002	4002	24.98	Cash	25-NOV-25 03.09.44.430794000 PM	James Wilson

Scenario 3: Finding the average order value



```
---- Find the average order value.
SELECT ROUND(AVG(TotalAmount),2) AS AvgOrderValue
FROM Orders;
```

```
All Rows Fetched: 1 in 0.072 seconds

  AVGORDERVALUE
1         27.31
```

Thank You!