MySkill

TechCorp SQL Practice:
Business Insights from Relational Data

Fadhila Dian Azhara

MySkill

Project Overview

This project simulates a relational database for TechCorp, covering product management, customer data, orders, and support operations. It was designed to strengthen SQL skills through real-world business scenarios such as analyzing customer orders, calculating revenue, and identifying key performance indicators.

MySkill

Analysis Performed

1. Identify the top 3 customers based on total order value.
2. Calculate average order value for each customer.
3. Find employees who have resolved more than 4 support tickets.
4. Detect products that have never been ordered.
5. Calculate total revenue from all product sales.
6. Determine categories with average product price above $500.
7. Find customers who placed at least one order over $1000.

Tools & Technologies

- Database: MySQL
- Techniques: SQL Joins, Aggregations, Subqueries, Common Table Expressions (CTE)
- Focus: Data analysis, database design, query performance

Database Scheme

1. **Tabel Produk (Products)**
   Tabel ini menyimpan informasi tentang produk-produk yang dijual.

2. **Tabel Pelanggan (Customers)**
   Tabel ini menyimpan informasi tentang pelanggan.

3. **Tabel Pesanan (Orders)**
   Tabel ini menyimpan informasi tentang pesanan yang dibuat oleh pelanggan.

4. **Tabel Detail Pesanan (OrderDetails)**
   Tabel ini menyimpan informasi detail dari setiap pesanan.

5. **Tabel Karyawan (Employees)**
   Tabel ini menyimpan informasi tentang karyawan yang bekerja di TechCorp.

6. **Tabel Tiket Dukungan (SupportTickets)**
   Tabel ini menyimpan informasi tentang tiket dukungan yang diajukan oleh pelanggan.

Products Table

Products	
product_id 🔑	**INT**
product_name	VARCHAR(100) NN
category	VARCHAR(50)
price	DECIMAL(10,2)
stock_quantity	INT

- **product_id**: Pengenal unik untuk setiap produk (Primary Key).
- **product_name**: Nama produk.
- **category**: Kategori produk (misalnya, Laptop, Smartphone, Aksesoris).
- **price**: Harga produk.
- **stock_quantity**: Jumlah unit produk yang tersedia dalam stok.
- **discount**: Diskon yang diterapkan pada harga produk, jika ada.

MySkill

Customers Table

Customers	
customer_id 🔑	INT
first_name	VARCHAR(50) NN
last_name	VARCHAR(50) NN
email	VARCHAR(100)
phone	VARCHAR(20)
address	VARCHAR(200)

- **customer_id:** Pengenal unik untuk setiap pelanggan (Primary Key).
- **first_name:** Nama depan pelanggan.
- **last_name:** Nama belakang pelanggan.
- **email:** Alamat email pelanggan.
- **phone:** Nomor telepon pelanggan.
- **address:** Alamat tempat tinggal pelanggan.

Orders Table

Orders	
order_id	INT
customer_id	INT
order_date	DATE
total_amount	DECIMAL(10,2)

- **order_id:** Pengenal unik untuk setiap pesanan (Primary Key).
- **customer_id:** Pengenal pelanggan yang membuat pesanan (Foreign Key yang merujuk ke `Customers.customer_id`).
- **order_date:** Tanggal pesanan dibuat.
- **total_amount:** Total nilai uang dari pesanan.

MySkill

OrderDetails Table

- **order_detail_id:** Pengenal unik untuk setiap entri detail pesanan (Primary Key).
- **order_id:** Pengenal pesanan yang dimiliki detail ini (Foreign Key yang merujuk ke Orders.order_id).
- **product_id:** Pengenal produk yang termasuk dalam pesanan (Foreign Key yang merujuk ke Products.product_id).
- **quantity:** Jumlah produk yang dipesan.
- **unit_price:** Harga per unit produk pada saat pesanan dibuat.

Employees Table

Employees	
employee_id	INT
first_name	VARCHAR(50) NN
last_name	VARCHAR(50) NN
email	VARCHAR(100)
phone	VARCHAR(20)
hire_date	DATE
department	VARCHAR(50)

- **employee_id:** Pengenal unik untuk setiap karyawan (Primary Key).
- **first_name:** Nama depan karyawan.
- **last_name:** Nama belakang karyawan.
- **email:** Alamat email karyawan.
- **phone:** Nomor telepon karyawan.
- **hire_date:** Tanggal karyawan dipekerjakan.
- **department:** Departemen tempat karyawan bekerja (misalnya, Support, Sales, Development).

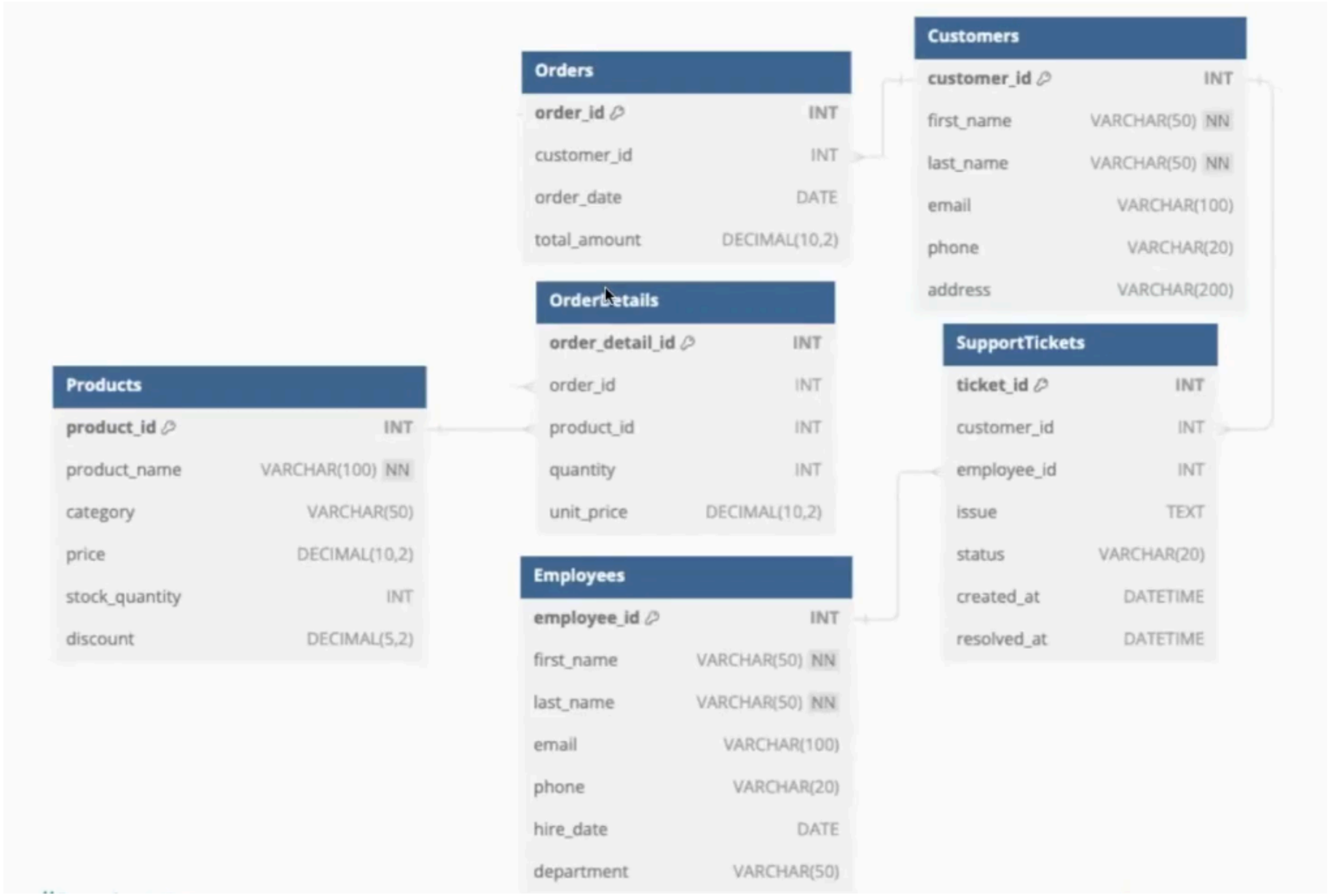
SupportTickets Table

SupportTickets	
ticket_id	INT
customer_id	INT
employee_id	INT
issue	TEXT
status	VARCHAR(20)
created_at	DATETIME
resolved_at	DATETIME

- **ticket_id:** Pengenal unik untuk setiap tiket support (Primary Key).
- **customer_id:** Pengenal pelanggan yang membuat tiket (Foreign Key yang merujuk ke Customers.customer_id).
- **employee_id:** Pengenal karyawan yang ditugaskan ke tiket (Foreign Key yang merujuk ke Employees.employee_id).
- **issue:** Deskripsi masalah yang dilaporkan oleh pelanggan.
- **status:** Status terkini dari tiket (misalnya, open, resolved).
- **created_at:** Tanggal dan waktu ketika tiket dibuat.
- **resolved_at:** Tanggal dan waktu ketika tiket diselesaikan (jika ada).

MySkill

Relational Data Structure



MySkill

Create Database

```
create database techcorp;
```

Create Table

```
create table Customers (
    customer_id int auto_increment primary key,
    first_name varchar(50) not null,
    last_name varchar(50) not null,
    email varchar(50) unique,
    phone varchar(20),
    address varchar(200)
);
```

```
create table Orders (
    order_id int auto_increment primary key,
    customer_id int,
    order_date date,
    total_amount decimal(10,2),
    foreign key (customer_id) references Customers(customer_id)
);
```

```
create table OrderDetails (
    order_detail_id int auto_increment primary key,
    order_id int,
    product_id int,
    quantity int,
    unit_price int,
    foreign key (order_id) references Orders(order_id),
    foreign key (product_id) references Products(product_id)
);
```

MySkill

Create Table

```sql
create table Employees (
    employee_id int auto_increment primary key,
    first_name varchar(50),
    last_name varchar(50),
    email varchar(100),
    phone varchar(20),
    hire_date date,
    department varchar(50)
);

create table SupportTickets (
    ticket_id int auto_increment primary key,
    customer_id int,
    employee_id int,
    issue text,
    status varchar(20),
    created_at datetime,
    resolved_at datetime,
    foreign key (customer_id) references Custoemrs(customer_id),
    foreign key (employee_id) references Employees(employee_id)
);
```

Insert values into Products table

```sql
INSERT INTO Products (product_name, category, price, stock_quantity, discount)
VALUES
('Laptop Pro 15', 'Laptop', 1500.00, 100, 0),
('Smartphone X', 'Smartphone', 800.00, 200, 0),
('Wireless Mouse', 'Accessories', 25.00, 500, 0),
('USB-C Charger', 'Accessories', 20.00, 300, 0),
('Gaming Laptop', 'Laptop', 2000.00, 50, 10),
('Budget Smartphone', 'Smartphone', 300.00, 150, 5),
('Noise Cancelling Headphones', 'Accessories', 150.00, 120, 15),
('Wireless Earphones', 'Accessories', 100.00, 100, 10);
```

Answer:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

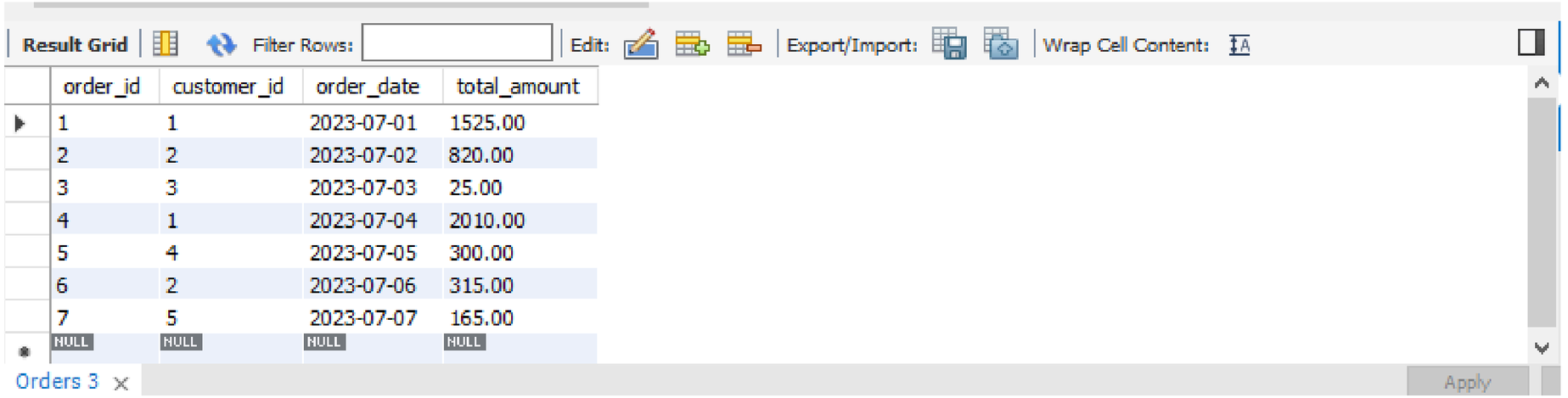
product_id	product_name	category	price	stock_quantity	discount
1	Laptop Pro 15	Laptop	1500.00	100	0.00
2	Smartphone X	Smartphone	800.00	200	0.00
3	Wireless Mouse	Accessories	25.00	500	0.00
4	USB-C Charger	Accessories	20.00	300	0.00
5	Gaming Laptop	Laptop	2000.00	50	10.00
6	Budget Smartphone	Smartphone	300.00	150	5.00
7	Noise Cancelling Headphones	Accessories	150.00	120	15.00
8	Wireless Earphones	Accessories	100.00	100	10.00

Products 1 | Apply

Insert values into Orders table

```
INSERT INTO Orders (customer_id, order_date, total_amount)
VALUES
(1, '2023-07-01', 1525.00),
(2, '2023-07-02', 820.00),
(3, '2023-07-03', 25.00),
(1, '2023-07-04', 2010.00),
(4, '2023-07-05', 300.00),
(2, '2023-07-06', 315.00),
(5, '2023-07-07', 165.00);
```

Answer:

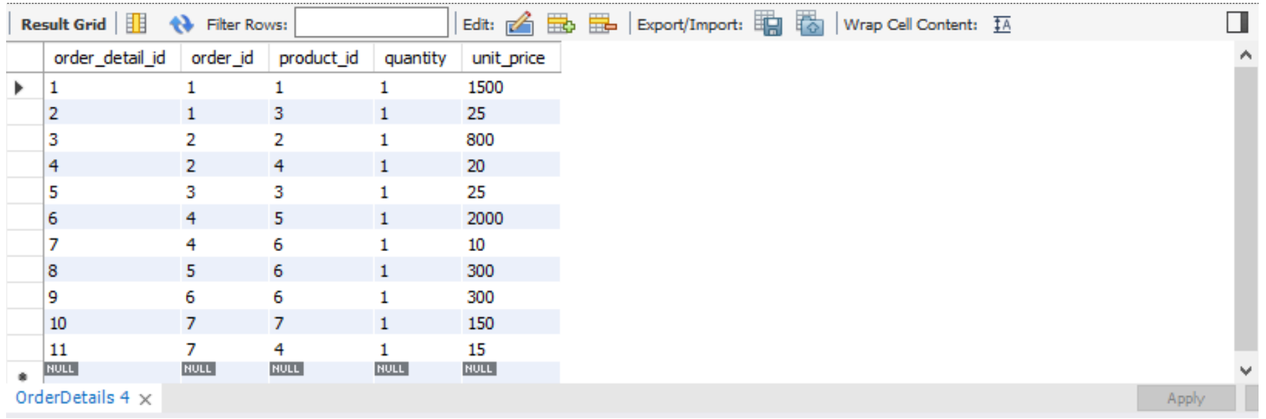


order_id	customer_id	order_date	total_amount
1	1	2023-07-01	1525.00
2	2	2023-07-02	820.00
3	3	2023-07-03	25.00
4	1	2023-07-04	2010.00
5	4	2023-07-05	300.00
6	2	2023-07-06	315.00
7	5	2023-07-07	165.00
NULL	NULL	NULL	NULL

Insert values into OrderDetails table

```
INSERT INTO OrderDetails (order_id, product_id, quantity, unit_price)
VALUES
(1, 1, 1, 1500.00),
(1, 3, 1, 25.00),
(2, 2, 1, 800.00),
(2, 4, 1, 20.00),
(3, 3, 1, 25.00),
(4, 5, 1, 2000.00),
(4, 6, 1, 10.00),
(5, 6, 1, 300.00),
(6, 6, 1, 300.00),
(7, 7, 1, 150.00),
(7, 4, 1, 15.00);
```

Answer:

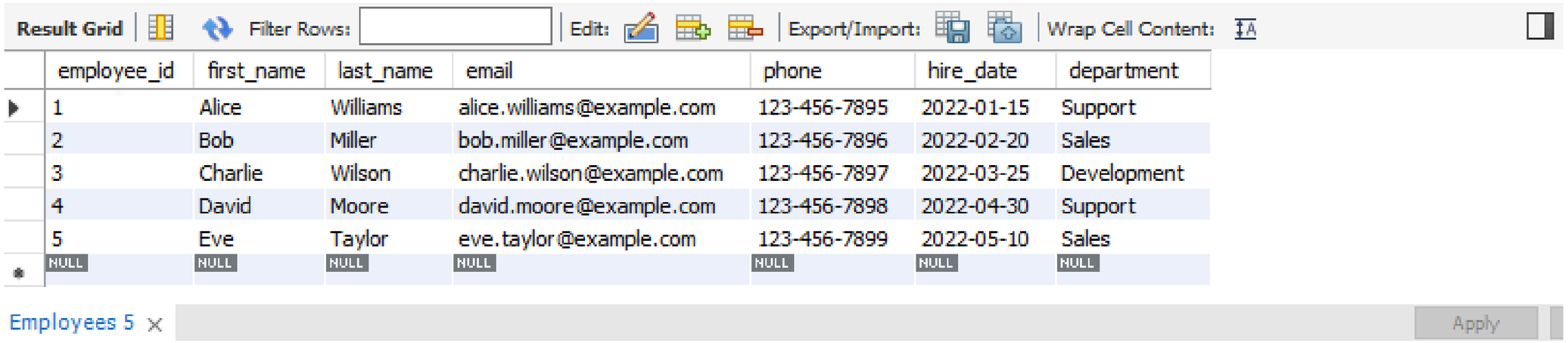


order_detail_id	order_id	product_id	quantity	unit_price
1	1	1	1	1500
2	1	3	1	25
3	2	2	1	800
4	2	4	1	20
5	3	3	1	25
6	4	5	1	2000
7	4	6	1	10
8	5	6	1	300
9	6	6	1	300
10	7	7	1	150
11	7	4	1	15
NULL	NULL	NULL	NULL	NULL

Insert values into Employees table

```
INSERT INTO Employees (first_name, last_name, email, phone, hire_date, department)
VALUES
('Alice', 'Williams', 'alice.williams@example.com', '123-456-7895', '2022-01-15', 'Support'),
('Bob', 'Miller', 'bob.miller@example.com', '123-456-7896', '2022-02-20', 'Sales'),
('Charlie', 'Wilson', 'charlie.wilson@example.com', '123-456-7897', '2022-03-25', 'Development'),
('David', 'Moore', 'david.moore@example.com', '123-456-7898', '2022-04-30', 'Support'),
('Eve', 'Taylor', 'eve.taylor@example.com', '123-456-7899', '2022-05-10', 'Sales');
```

Answer:

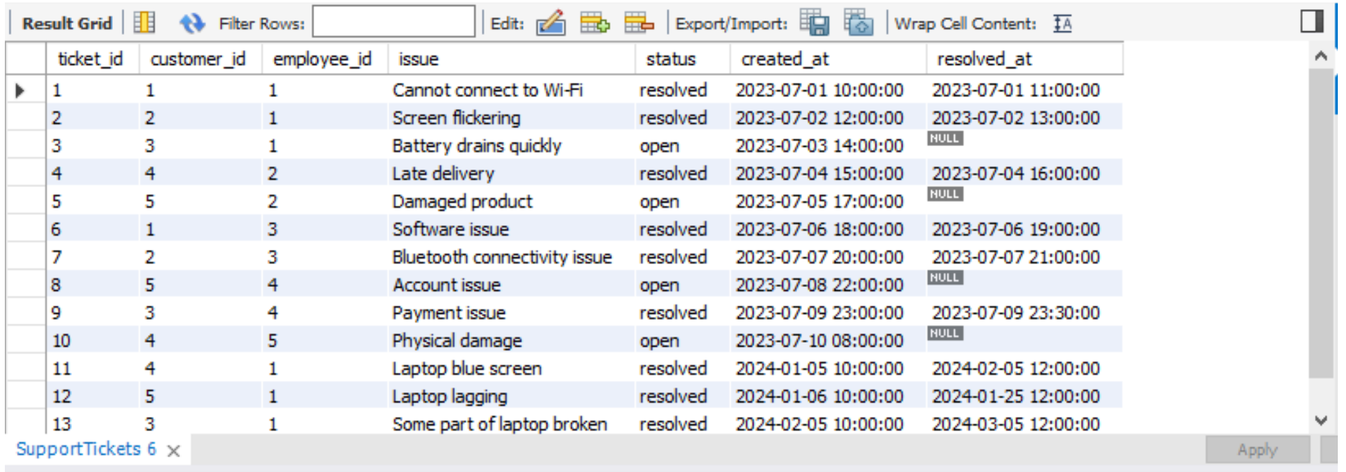


employee_id	first_name	last_name	email	phone	hire_date	department
1	Alice	Williams	alice.williams@example.com	123-456-7895	2022-01-15	Support
2	Bob	Miller	bob.miller@example.com	123-456-7896	2022-02-20	Sales
3	Charlie	Wilson	charlie.wilson@example.com	123-456-7897	2022-03-25	Development
4	David	Moore	david.moore@example.com	123-456-7898	2022-04-30	Support
5	Eve	Taylor	eve.taylor@example.com	123-456-7899	2022-05-10	Sales
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Insert values into SupportTickets table

```
INSERT INTO SupportTickets (customer_id, employee_id, issue, status, created_at, resolved_at)
VALUES
(1, 1, 'Cannot connect to Wi-Fi', 'resolved', '2023-07-01 10:00:00', '2023-07-01 11:00:00'),
(2, 1, 'Screen flickering', 'resolved', '2023-07-02 12:00:00', '2023-07-02 13:00:00'),
(3, 1, 'Battery drains quickly', 'open', '2023-07-03 14:00:00', NULL),
(4, 2, 'Late delivery', 'resolved', '2023-07-04 15:00:00', '2023-07-04 16:00:00'),
(5, 2, 'Damaged product', 'open', '2023-07-05 17:00:00', NULL),
(1, 3, 'Software issue', 'resolved', '2023-07-06 18:00:00', '2023-07-06 19:00:00'),
(2, 3, 'Bluetooth connectivity issue', 'resolved', '2023-07-07 20:00:00', '2023-07-07 21:00:00'),
(5, 4, 'Account issue', 'open', '2023-07-08 22:00:00', NULL),
(3, 4, 'Payment issue', 'resolved', '2023-07-09 23:00:00', '2023-07-09 23:30:00'),
(4, 5, 'Physical damage', 'open', '2023-07-10 08:00:00', NULL),
(4, 1, 'Laptop blue screen', 'resolved', '2024-01-05 10:00:00', '2024-02-05 12:00:00'),
(5, 1, 'Laptop lagging', 'resolved', '2024-01-06 10:00:00', '2024-01-25 12:00:00'),
(3, 1, 'Some part of laptop broken', 'resolved', '2024-02-05 10:00:00', '2024-03-05 12:00:00');
```

Answer:



ticket_id	customer_id	employee_id	issue	status	created_at	resolved_at
1	1	1	Cannot connect to Wi-Fi	resolved	2023-07-01 10:00:00	2023-07-01 11:00:00
2	2	1	Screen flickering	resolved	2023-07-02 12:00:00	2023-07-02 13:00:00
3	3	1	Battery drains quickly	open	2023-07-03 14:00:00	NULL
4	4	2	Late delivery	resolved	2023-07-04 15:00:00	2023-07-04 16:00:00
5	5	2	Damaged product	open	2023-07-05 17:00:00	NULL
6	1	3	Software issue	resolved	2023-07-06 18:00:00	2023-07-06 19:00:00
7	2	3	Bluetooth connectivity issue	resolved	2023-07-07 20:00:00	2023-07-07 21:00:00
8	5	4	Account issue	open	2023-07-08 22:00:00	NULL
9	3	4	Payment issue	resolved	2023-07-09 23:00:00	2023-07-09 23:30:00
10	4	5	Physical damage	open	2023-07-10 08:00:00	NULL
11	4	1	Laptop blue screen	resolved	2024-01-05 10:00:00	2024-02-05 12:00:00
12	5	1	Laptop lagging	resolved	2024-01-06 10:00:00	2024-01-25 12:00:00
13	3	1	Some part of laptop broken	resolved	2024-02-05 10:00:00	2024-03-05 12:00:00

Top 3 customers based on orders

```sql
-- 1.top 3 customers based on orders
select
c.first_name,
c.last_name,
sum(total_amount) as total_order_amount
from customers as c
join orders o on o.customer_id = c.customer_id
group by c.customer_id
order by total_order_amount desc
limit 3;
```

Answer:

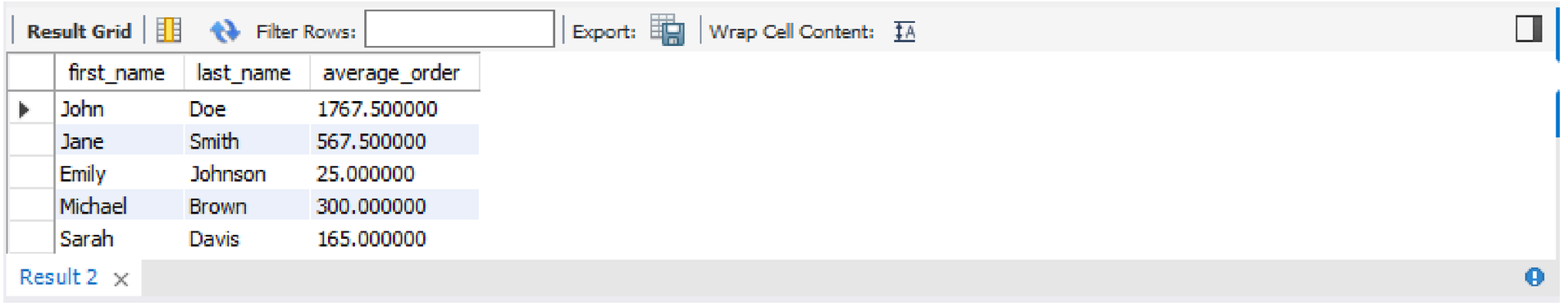
Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

first_name	last_name	total_order_amount
John	Doe	3535.00
Jane	Smith	1135.00
Michael	Brown	300.00

Average order value for each customer

```sql
-- 2. average order value for each customer
select
c.first_name,
c.last_name,
avg(total_amount) as average_order
from customers c
join orders o on o.customer_id = c.customer_id
group by c.customer_id;
```

Answer:



Employees with >4 resolved ticket support

```sql
-- 3. employees with >4 resolved ticket support
select
e.first_name,
e.last_name,
count(s.ticket_id) as ticket_support
from employees e
join supporttickets s on e.employee_id = s.employee_id
where s.status = 'resolved'
group by e.employee_id
having count(s.ticket_id) >4;
```

Answer:

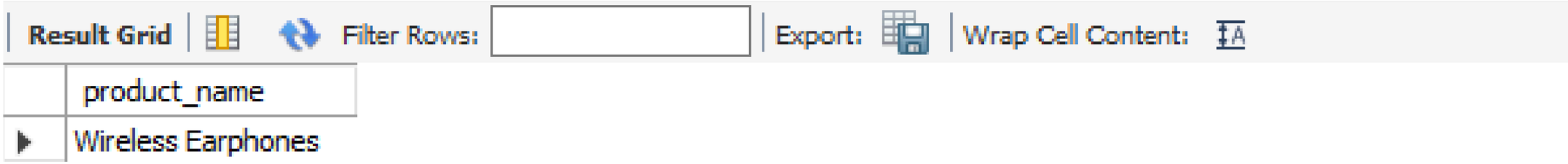
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	first_name	last_name	ticket_support
▶	Alice	Williams	5

Products that have never been ordered

```sql
-- 4. products that have never been ordered
select p.product_name
from products p
left join orderdetails od on od.product_id = p.product_id
where od.order_id is null;
```

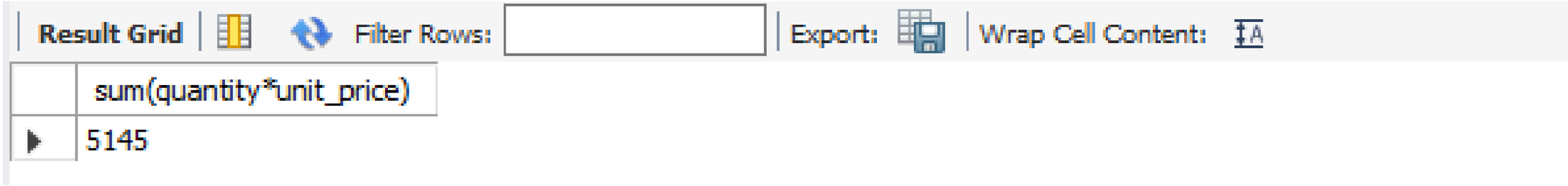
Answer:



Calculate total revenue generated from product sales

```
-- 5. calculate total revenue generated from product sales
select
sum(quantity*unit_price)
from orderdetails;
```

Answer:



Find the average price for each category and list categories with an average price higher than $500

```sql
-- 6. find the average price for each category and list categories with an average price higher than $500
with cte_avg_price as (
select
category, avg(price) as rerata
from products
group by category
)
select *
from cte_avg_price
where rerata > 500;
```

Answer:

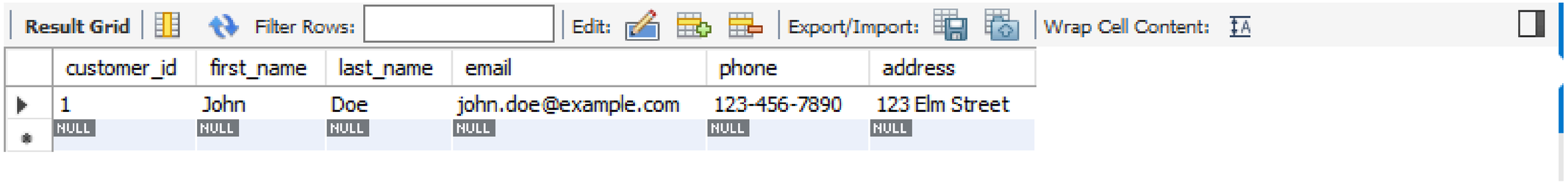
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	category	rerata
▶	Laptop	1750.000000
	Smartphone	550.000000

Find customers who have placed at least one order with a total amount greater than $1000

```sql
-- 7. find customers who have placed at least one order with a total amount greater than $1000
select *
from customers
where customer_id in
(select
customer_id
from orders
where total_amount >1000
);
```

Answer:



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

customer_id	first_name	last_name	email	phone	address
1	John	Doe	john.doe@example.com	123-456-7890	123 Elm Street
NULL	NULL	NULL	NULL	NULL	NULL

MySkill

Insight

- Built and normalized five relational tables: Products, Customers, Orders, OrderDetails, Employees, and SupportTickets.
- Applied JOINs, subqueries, CTEs, aggregation, and filtering for business reporting.
- Gained experience simulating real e-commerce data structures.
- Strengthened understanding of SQL logic and query optimization.

Let's connect!

+6287771811236
fadhiladianazhara@gmail.com
linkedin.com/in/fadhiladianaz