ITBAN2 – Advanced MySQL Queries with JSON Data

Database Setup:

Create a new MySQL database named `**e_commerce**`

Table Design:

Design the following tables:

- **products** table with columns:
  - product_id (INT, primary key)
  - name (VARCHAR)
  - description (TEXT)
  - attributes (JSON) – to store product attributes as JSON documents

- **orders** table with columns:
  - order_id (INT, primary key)
  - customer_id (INT, foreign key references **customers**)
  - order_date (DATE)

- **order_details** table with columns:
  - detail_id (INT, primary key)
  - order_id (INT, foreign key references **orders**)
  - product_id (INT, foreign key references **products)**
  - quantity (INT)
  - price (DECIMAL)

- **customers** table with columns:
  - customer_id (INT, primary key)
  - firstname (VARCHAR)
  - middlename (VARCHAR)
  - lastname (VARCHAR)
  - address (JSON)

Data Population

- Insert sample product data into the **products** table with attributes stored as JSON documents.

Ex.

```
{
  "color": "blue",
  "size": "medium",
  "price": 19.99,
  "brand": "BrandX"
}
```

- Populate the **orders** and **order_details** tables with sample data.

Note: You can use libraries or frameworks to auto-populate the tables such as Faker (Laravel) / Python. Records should be between 2000 – 5000 for **products** table, 1000 – 2000 for **orders** / **order_details** and 300 – 500 **customers**.

Document the following query questions. Provide screenshots of the query and the result set.

1. **Retrieve Product Information**:
   - Write a query to fetch the names and descriptions of all products.
   - Extend the previous query to include specific attributes such as color, size, and price.
2. **Query Orders and Order Details**:
   - Retrieve the details of all orders placed, including the order date, customer ID, product name, quantity, and price.
   - Calculate the total cost of each order.
3. **Filtering Products Based on Attributes**:
   - Write a query to find all products with a price greater than $50.
   - Filter products by color and brand, and display their names and prices.
4. **Calculating Aggregate Data**:
   - Calculate the total sales revenue generated by each product.
   - Determine the total quantity of each product ordered.
5. **Advanced Filtering and Aggregation**:
   - Find the top 5 best-selling products based on total quantity sold.
   - Identify the average price of products from a specific brand.
6. **Nested JSON Queries**:
   - Retrieve the color and size of a specific product.
   - Extract and display all available attributes of products in JSON format.
7. **Joining Multiple Tables**:
   - Write a query to find all orders placed by customers along with the products ordered and their quantities.
   - Calculate the total revenue generated by each customer.
8. **Data Manipulation with JSON Functions**:
   - Update the price of a specific product stored as JSON attribute.
   - Add a new attribute to all products with a default value.
9. **Advanced JSON Operations**:
   - Find products with specific attributes that match a given criteria using JSON path expressions.

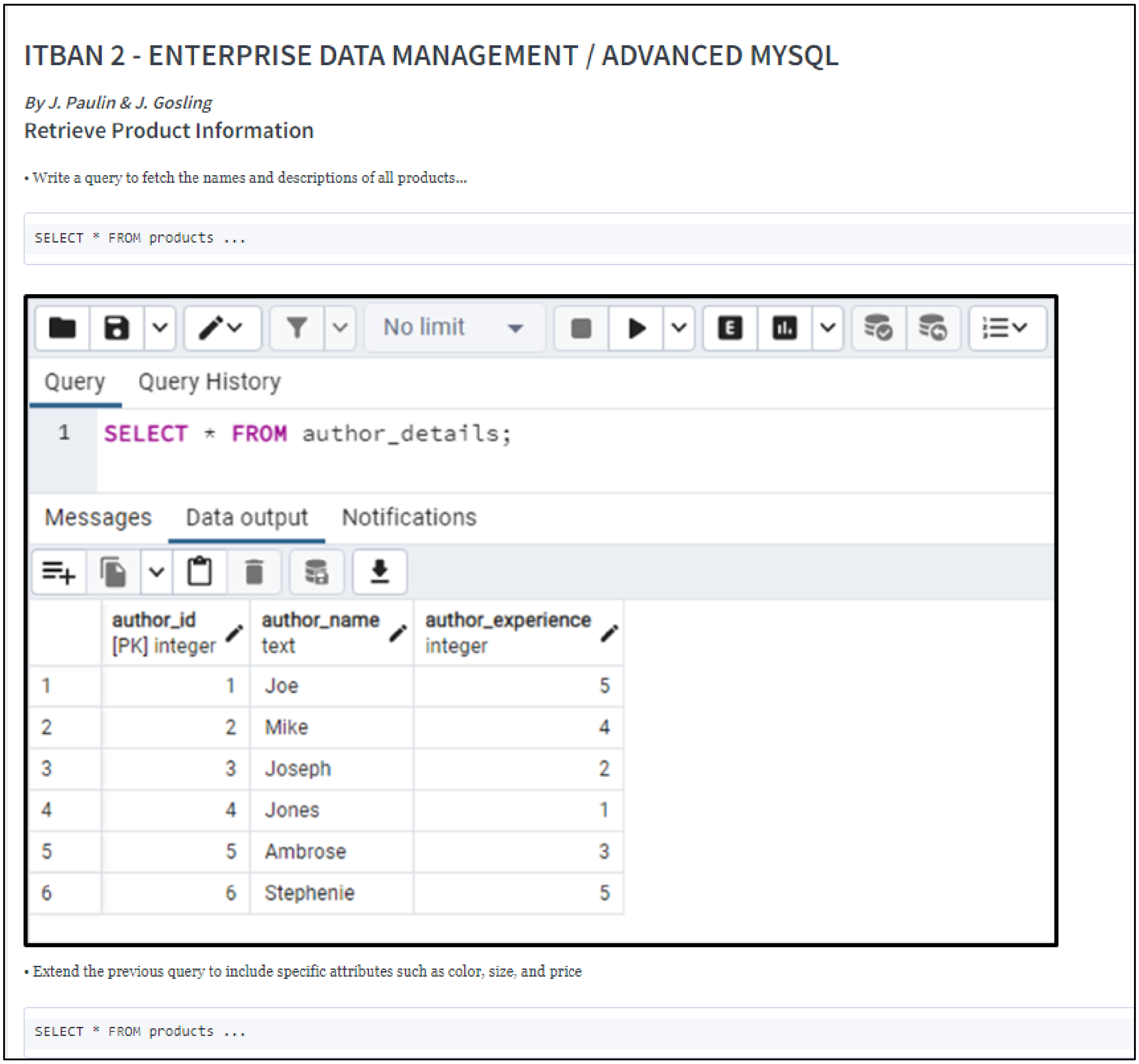
- Extract and display the first element of an array stored within a JSON attribute.

Deliverables:

- **Hard copy of documented query per item.**
- **GitHub link containing the sql dump and queries used and screenshots per item.**

Make use of the **GitHub Markdown** to properly format and style your readme.md file.

Example: (also provide the screenshots after the query)



ITBAN 2 - ENTERPRISE DATA MANAGEMENT / ADVANCED MYSQL

By J. Paulin & J. Gosling

Retrieve Product Information

- Write a query to fetch the names and descriptions of all products...

```
SELECT * FROM products ...
```

```
SELECT * FROM author_details;
```

	author_id [PK] integer	author_name text	author_experience integer
1	1	Joe	5
2	2	Mike	4
3	3	Joseph	2
4	4	Jones	1
5	5	Ambrose	3
6	6	Stephenie	5

- Extend the previous query to include specific attributes such as color, size, and price

```
SELECT * FROM products ...
```