Sales Database with SQL like Syntax for Fetching Records

Create an application that would allow the user to create a sales database and use SQL (Structured Query Language) like commands to fetch records from the database. Provide appropriate menu that will allow the user to create or use a database by specifying the database name (here the database name can be used to refer to a folder in which the files are stored). The user should be further allowed to add sales records for the following entities (these entities can be created using structure):

- **Customer** with at least the following attributes
    - customerId
    - firstName
    - middleName
    - gender
    - dob
    - contactNo
    - address
    - pincode
- **Order** with at least the following attributes
    - orderId
    - orderDate
    - customerId
- **Product** with at least the following attributes
    - productId
    - productName
    - price
    - UOM (Unit of Measurement)
- **OrderDetails** with at least the following attributes
    - orderId
    - productId
    - quantity

Each of the above mentioned entities can be stored in separate files (viz. Customer.dat, Order.dat, Product.dat, OrderDetails.dat) within a folder having a name that matches with the name of the database as provided by the user. Use this part of the application to add at least 10 customers, 20 products, and 30 orders each containing 1 to 10 products in order details.

Define your own language similar to Structured Query Language (SQL) in order to allow the user fetch records from the database. Provide an interface where the user can write a query to fetch records. The interface should generate appropriate error messages in case of any mistake in syntax within the query entered by the user. The query language should at least contain the operations mentioned in Appendix I.

Instructions:

1. Structure and File Management System should be appropriately used in the project.
2. Most of the features of 'C' programming language should be implemented in the project.
3. The project should be appropriately modularized and implemented using user-defined functions.
4. **Each group should equally distribute modules and tasks among the group members.**

Appendix I

Operation	Syntax	Example
Fetch all records and attributes of an entity	SELECT * FROM <Entity Name>;	SELECT * FROM Customer; *Will fetch all details of all customers.*
Fetch specific attributes of all records of an entity	SELECT <attribute1, attribute2, … attributeN> FROM <Entity Name>;	SELECT firstName, contactNo FROM Customer; *Will fetch first name and contact number of all customers.*
Fetch specific attributes of an entity with specific condition	SELECT <attribute1, attribute2, … attributeN> FROM <Entity Name> WHERE attributeR = value AND/OR arrtibuteM > value;	SELECT lastName FROM Customer WHERE gender = ‘M’ AND firstName = ‘Rajiv’ *Will fetch last name of all male customers whose first name is “Rajiv”*

Fetch specific attributes from multiple related entities	SELECT <A.attrib1, A.attrib2,…, A.attribP,… B.attrib1, B.attrib2, …, B.attribQ ,… N.attrib1, N.attrib2, … N.attribR>
	FROM <EntityA, EntityB, … EntityN>
	WHERE A.attribP = B.attribQ AND B.arrtibM = N.attribR;
		SELECT Customer.customerId, Customer.firstName, Order.orderId, Order.orderDate
		FROM Customer, Order WHERE Customer.customerId = Order.customerId
		Will fetch customer id, first name, order id, order date of all customers and their related orders
		SELECT Customer.customerId, Customer.firstName, Order.orderDate, Product.productName
		FROM Customer, Order, Product, OrderDetails
		WHERE Customer.customerId = Order.customerId AND Order.orderId = OrderDetails.orderId AND Product.productId = OrderDetails.productId
		Will fetch customer id, first name, order date, product name of all order details for all customers

Fetch all records and attributes of an entity in sorted order	SELECT * FROM <Entity Name> ORDER BY <attrib1, attrib2,...,attribN>;	SELECT * FROM Customer ORDER BY lastName, firstName;
		Will fetch all details of all customers sorted by last name and then by first name in ascending order.
		SELECT * FROM Product ORDER BY price, productName;
		Will fetch all details of all products sorted by price and then by product name in ascending order.

***Note: The operations and syntax mentioned here are not exhaustive and students can choose to implement other operations as well.**