**May 2,2025**

**CC-215 DATABASE SYSTEM**

FAIQA LATIF (110790) TOOBA FATIMA (110789)

**Instructor: Miss Sehrish khan**

Database Design and Implementation in MySQL

* Entity Relationship Diagram
* Relational schema\table schema
* Normalization
* Implementation in MySQL

**Problem Statement**

**Grocery Delivery App with Vendor Reviews and Weekly Subscriptions**

A grocery delivery system enables users to either place instant grocery orders or subscribe to curated weekly bundles (like "Healthy Heart Kit"). Products come from multiple verified vendors. The system tracks vendor reliability and quality through customer ratings. Users can customize subscriptions, postpone deliveries, or swap out items. The inventory system predicts restocking needs and manages perishable item expiration alerts.

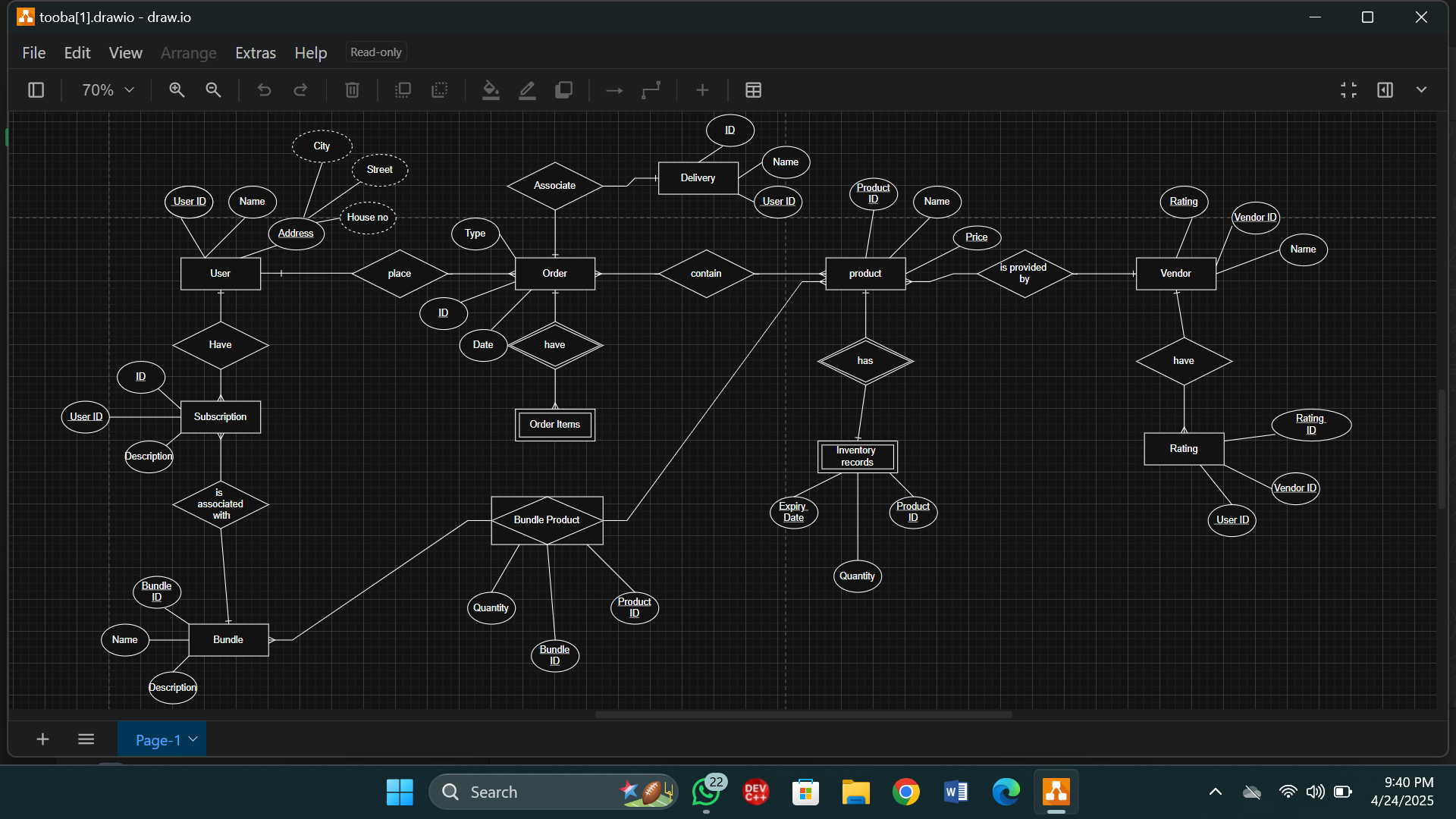
**ENTITY RELATIONSHIP DIAGRAM**

ERD stands for Entity-Relationship Diagram. It's a visual representation of the structure of a database, showing the relationships between entities (tables) and their attributes.

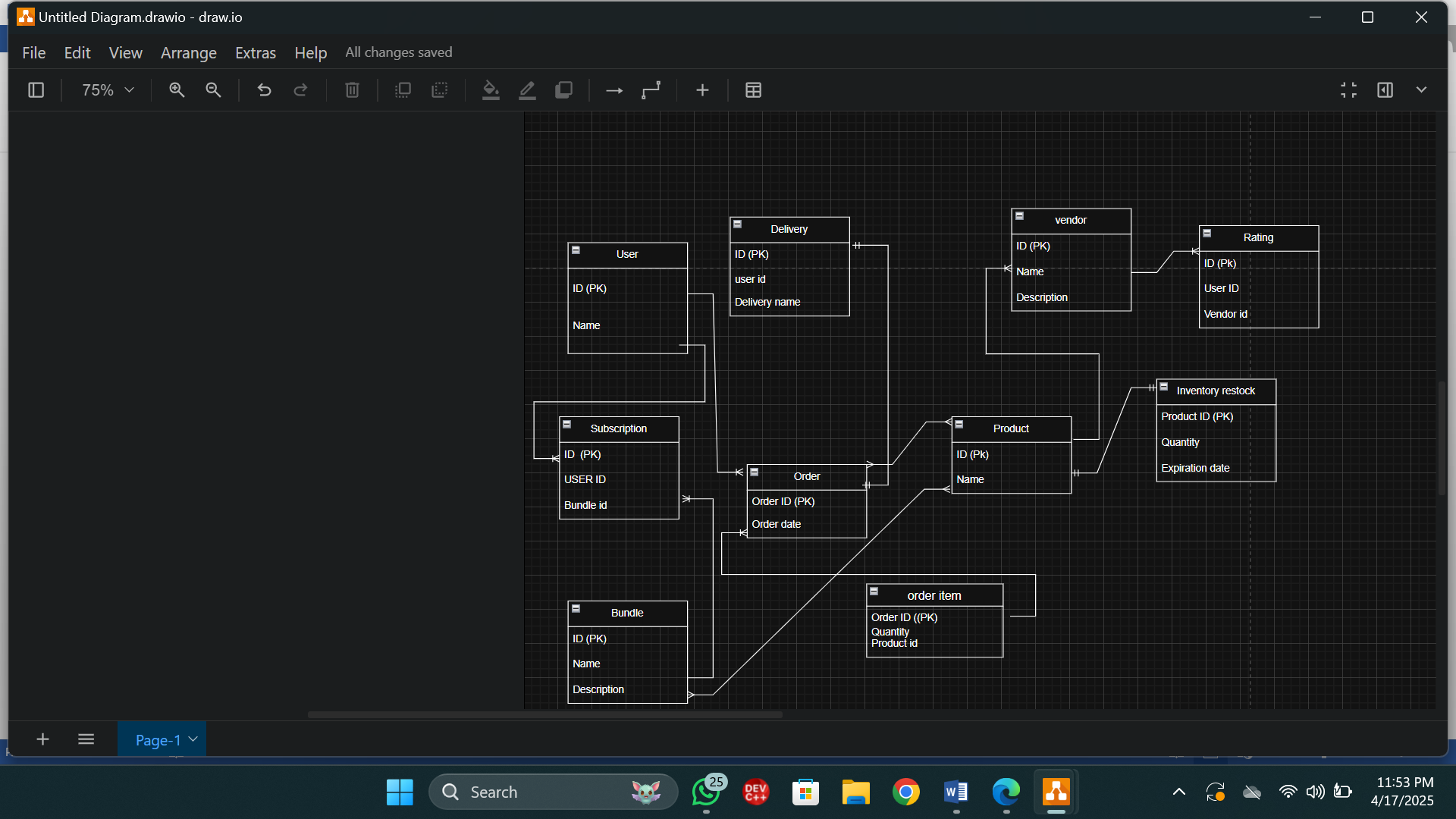
Entity relationship diagram is of 2 type:

* Crow Feet Model
* Chen Model

**ENTITY RELATIONSHIP DIAGRAM USING CHEN MODEL**



**CROW FOOT MODEL:**



**Relational schema\table schema**

**User** ( User\_ID (PK), Name, Email, Address)

**Vendor(** Vendor\_ID (PK), Name, Rating)

**Product (** Product\_ID (PK), Name, Price, Vendor\_ID (FK), Category)

**Order** (Order\_ID (PK), User\_ID (FK), Order\_Date, Total\_Cost, Status)

**Order\_item** ( Order\_ID (FK), Product\_ID (FK), Quantity)

**Bundle**  (Bundle\_ID (PK), Name, Description)

**Subscription** (Subscription\_ID (PK), User\_ID (FK), Bundle\_ID (FK), Frequency, Status)

**Product\_Bundle** (Bundle\_ID (FK), Product\_ID (FK), Quantity)

**Inventory** (Product\_ID (FK), Quantity, Expiration\_Date, Restock\_Threshold)

**Delivery** (Delivery\_ID (PK), Order\_ID (FK), Delivery\_Date, Status)

**Rating** (Rating\_ID (PK), User\_ID (FK), Vendor\_ID (FK), Rating, Review)

**NORMALIZATION**

Normalization is a process of organizing data in a database to minimize redundancy and improve data integrity.

The **main objective** of database normalization is to eliminate redundant data, minimize data modification errors, and simplify the query process.

* **Every table has primary key.**
* **Other keys depend upon primary key.**
* **Field must contain atomic values.**

**FIRST NORMAL FORM (1NF):**

Each table cell contains a single value. No repeating groups or arrays in a single column. 1NF eliminates data redundancy and improves data integrity by organizing data into well-structured tables. In 1NF there must be a primary key.

Each attribute contains only atomic (indivisible) values, and there are no repeating groups of columns.

**SECOND NORMAL FORM (2NF):**

Second Normal Form (2NF) is a level of database normalization that builds on First Normal Form (1NF). A table is in 2NF if:

There is no partial dependency of any column on a composite primary key.

All non-key attributes depend on the entire primary key.

All non-key attributes are fully functionally dependent on the entire primary key in above schema.

**THIRD NORMAL FORM (3NF):**

Third Normal Form (3NF) in database normalization eliminates transitive dependencies, ensuring that each non-key attribute in a table depends only on the primary key, not on other non-key attributes. It builds upon First Normal Form (1NF) and Second Normal Form (2NF).

3NF eliminates indirect dependencies, ensuring that each non-key attribute depends directly on the primary key.

**COMMON COMMANDS:**

**1. To see existing databases:**

**show databases**;

**2. To Create a database:**

**CREATE DATABASE database\_name;**

**3. To use the database:**

**use database\_name;**

**4. To see existing TABLES:**

**show tables;**

**5. To Create a tables:**

**create table table\_name(attribute\_id\_1 type(domain) primary key, attribute\_2 type(domain),…,n);**

**6. To describe table:**

**describe table\_name;**

**7. To see values from table:**

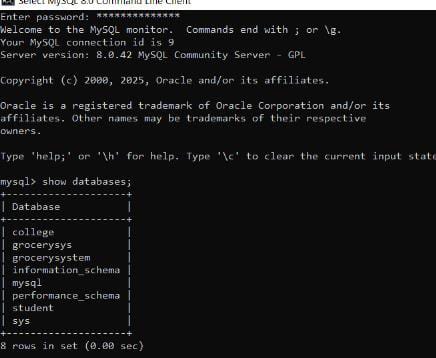
**select \* from table\_name;**

**8. To add foreign key in existing table:**

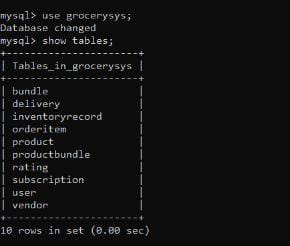
**alter table table\_name ADD constraint fk\_table\_name foreign key (attribute) references ref\_table(ref\_attribute);**

**IMPLEMENTATION IN MYSQL**

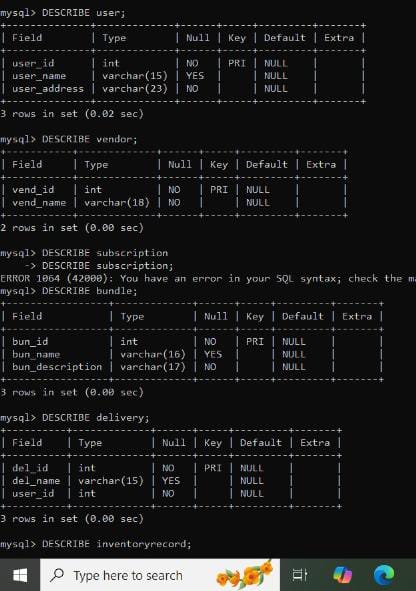
* **show databases;**
* **create database grocerysys;**



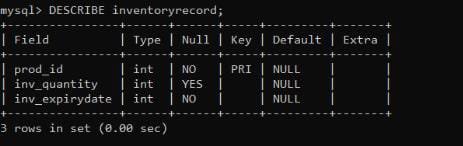
* use grocerysys;
* show tables;



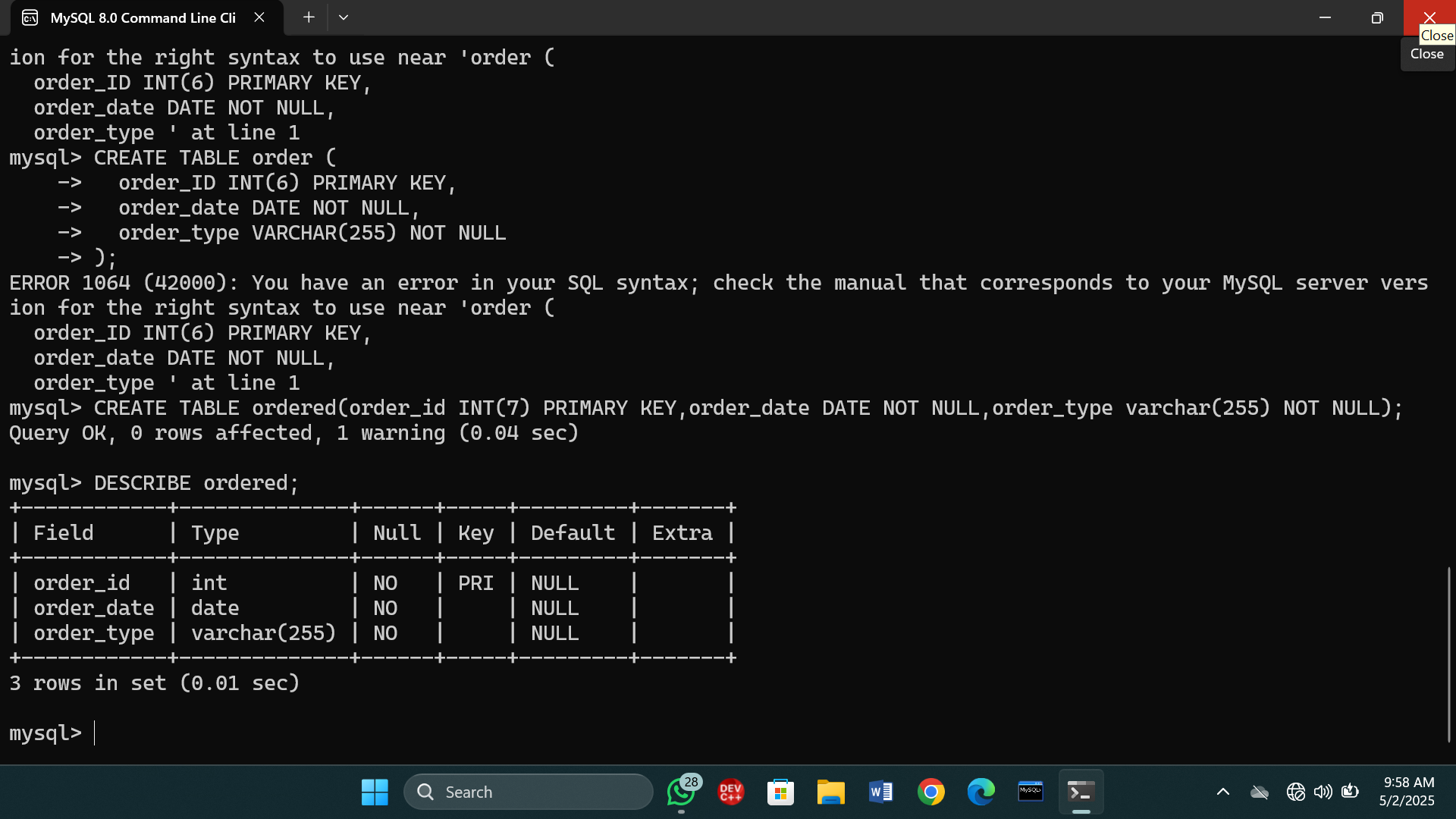
* create table user(user\_id INT(3) PRIMARY KEY,user\_name varchar (15),user\_address varchar(23) NOT NULL);
* DESCRIBE user;
* create table vendor(vend\_id INT(4) PRIMARY KEY,vend\_name varchar (18) NOT NULL);
* DESCRIBE vendor;
* Create table bundle(bun\_id INT (5) PRIMARY KEY,bun\_name varchar(16),bun\_description varchar(17) NOT NULL);
* DESCRIBE bundle;
* Create table delivery(del\_id INT(5) PRIMARY KEY,del\_namevarchar(16),user\_id INT(7) NOT NULL);
* DESCRIBE delivery;



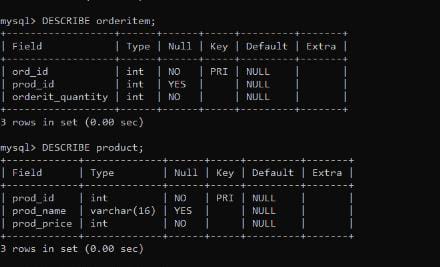
* Create table inventoryrecord(prod\_id INT(5) PRIMARY KEY,inv\_quantity INT(19),inv\_expirydate INT(6) NOT NULL);
* DESCRIBE inventoryrecord;



* Create table orderd(order\_id INT(7) PRIMARY KEY,order\_date DATE NOT NULL,order\_type varchar(255) NOT NULL);
* DESCRIBE ordered;



* create table orderitem(ord\_id INT(5) PRIMARY KEY,prod\_id INT(7),orderit\_quantity INT(7) NOT NULL);
* DESCRIBE orderitem;
* Create table product(prod\_id INT(6) PRIMARY KEY,prod\_name varchar(15),prod\_price INT(4) NOT NULL);
* DESCRIBE product;



* Create table productbundle(prod\_id INT(6) PRIMARY KEY,bun\_id INT(7),productbun\_quantity(7) NOT NULL);
* DESCRIBE productbundle;
* Create table rating(rat\_id INT(8) PRIMARY KEY,vend\_id INT(7),user\_id INT(5) NOT NULL);
* DESCRIBE rating;
* Create table subscription(sub\_id INT(7) PRIMARY KEY,user\_id INT(5),sub\_description varchar(25)NOT NULL);
* DESCRIBE subscription;

