# **Pharmacy Drugs Inventory Management**

Milestone: Implementation in MySQL

Group 11

Sai Varun Kumar Namburi

FNU Meenal

857-265-1349 (Sai Varun)

848-667-4233 (Meenal)

namburi.sai@northeastern.edu

lnu.meenal@northeastern.edu

Percentage of Effort Contributed by Student1: 50%

Percentage of Effort Contributed by Student2: 50%

Signature of Student 1: Sai Varun

Signature of Student 2: Meenal

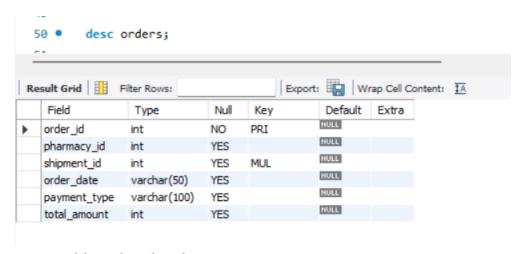
Submission Date: November 5, 2022

# **Implementation Model:**

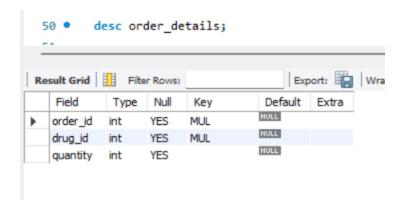
show databases; create database inventory; use inventory;

#### create table orders

(order\_id Int primary key, pharmacy\_id int, shipment\_id int, order\_date varchar(50), payment\_type varchar(100),total\_amount int);

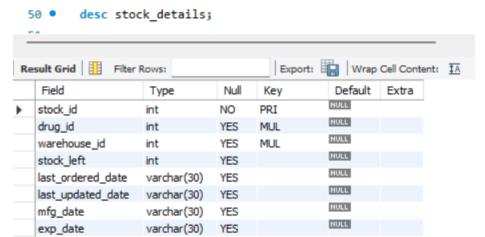


create table order\_details
(order\_id INT, drug\_id INT, quantity int);



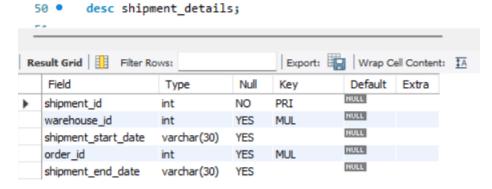
create table stock\_details

(stock\_id int primary key, drug\_id int, warehouse\_id int, stock\_left int, last\_ordered\_date varchar(30), last\_updated\_date varchar(30), mfg\_date varchar(30), exp\_date varchar(30));



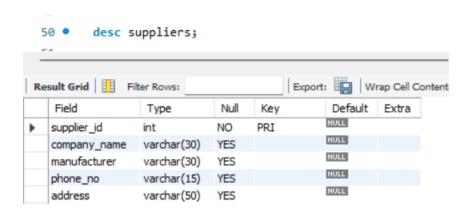
create table shipment\_details

(shipment\_id int primary key, warehouse\_id int, shipment\_start\_date varchar(30), order\_id int, shipment\_end\_date varchar(30));



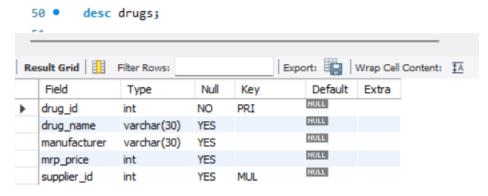
create table suppliers

(supplier\_id int primary key, company\_name varchar(30), manufacturer varchar(30),phone\_no int(10),address varchar(50));



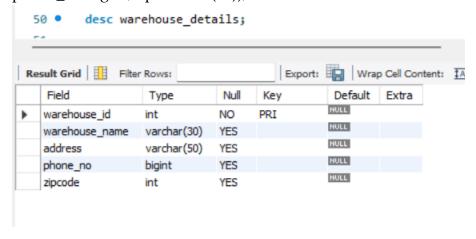
# create table drugs

(drug\_id int primary key, drug\_name varchar(30), manufacturer varchar(30), mrp\_price int, supplier\_id int);



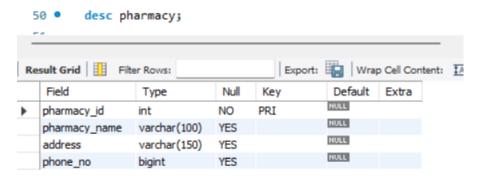
## create table warehouse details

(warehouse\_id int primary key, warehouse\_name varchar(30), address varchar(50), phone\_no bigint,zipcode int(10));

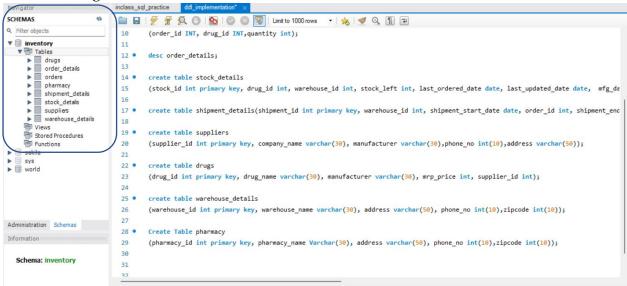


# Create Table pharmacy

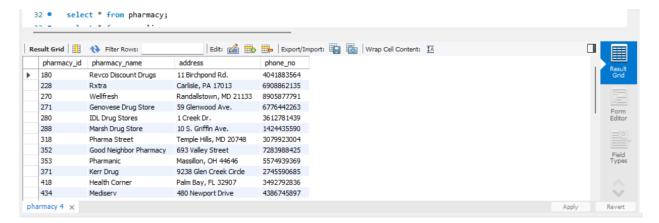
(pharmacy\_id int primary key, pharmacy\_name Varchar(100), address varchar(150), phone\_no bigint);



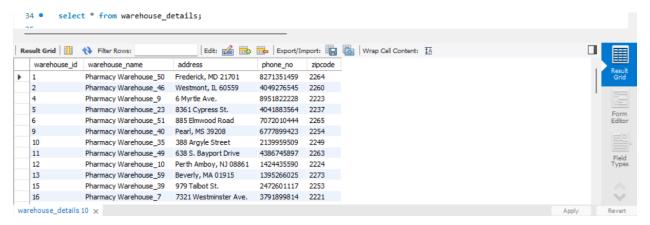
# After creating the table it is how it looks



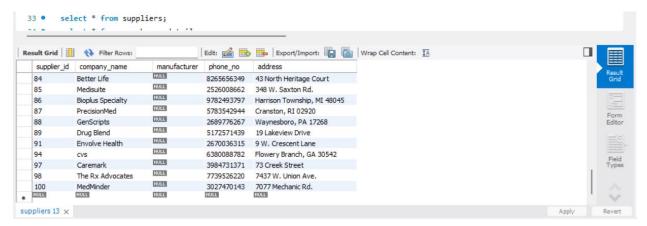
After inserting the data into the table, It looks like below, in the same way we have inserted the data into all the tables



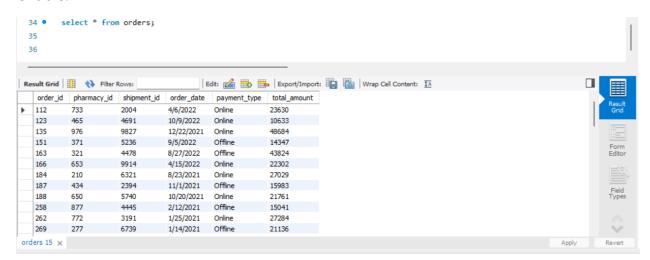
# Warehouse details



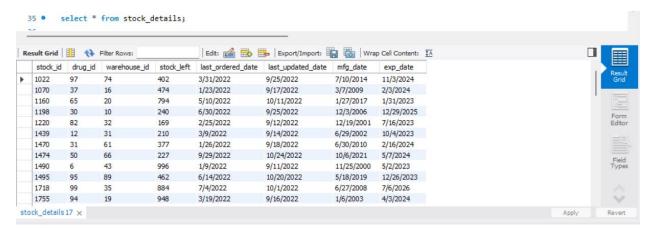
# Suppliers:



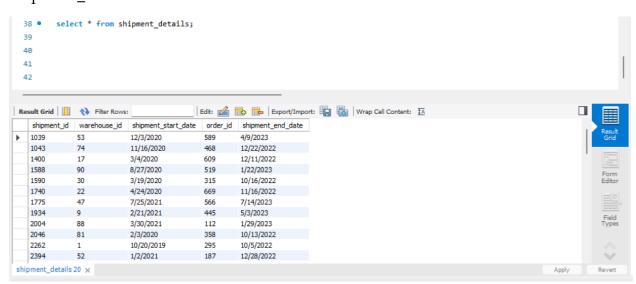
## Orders:



## Stock\_details:



# Shipment\_details:



# **Creating Foreign Key relation for the Tables:**

ALTER TABLE drugs ADD FOREIGN KEY (supplier\_id) REFERENCES suppliers(supplier\_id);

ALTER TABLE shipment\_details ADD FOREIGN KEY (warehouse\_id) REFERENCES warehouse\_id);

ALTER TABLE stock\_details ADD FOREIGN KEY (warehouse\_id) REFERENCES warehouse\_id);

ALTER TABLE order\_details ADD FOREIGN KEY (order\_id) REFERENCES orders(order\_id);

ALTER TABLE shipment\_details ADD FOREIGN KEY (order\_id) REFERENCES orders(order\_id);

ALTER TABLE order\_details ADD FOREIGN KEY (drug\_id) REFERENCES drugs(drug\_id);

ALTER TABLE stock\_details ADD FOREIGN KEY (drug\_id) REFERENCES drugs(drug\_id);

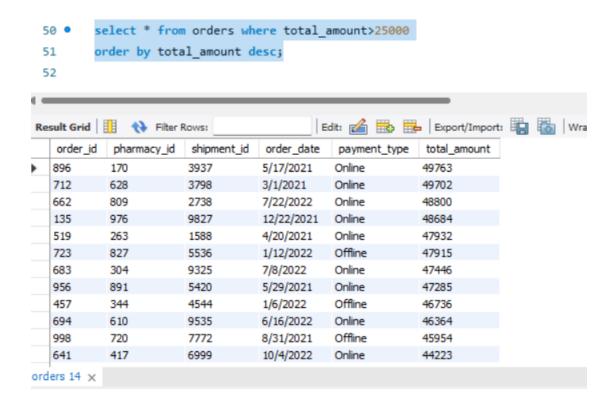
ALTER TABLE orders ADD FOREIGN KEY (shipment\_id) REFERENCES shipment\_details(shipment\_id);

# **Basic Analysis:**

1. select \* from orders where total\_amount>25000

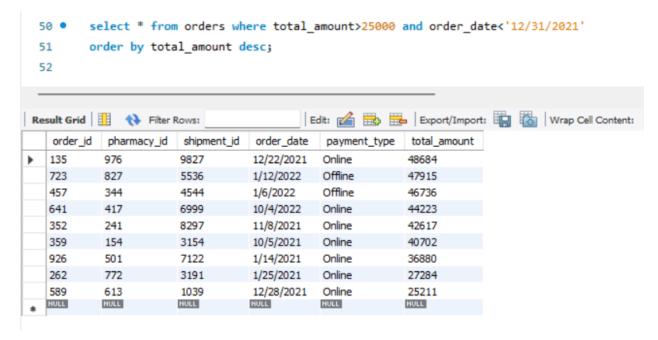
order by total\_amount desc;

Checking the all the orders which have total amount more than 25000 and ordered by total\_amount in descending order

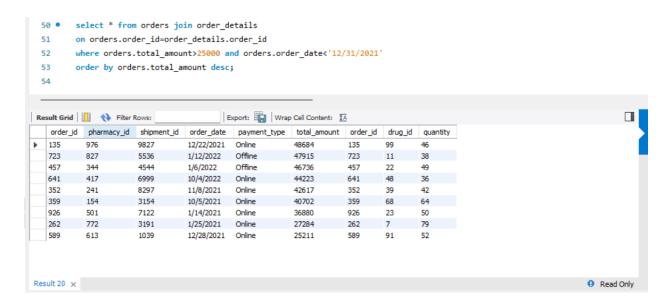


 select \* from orders where total\_amount>25000 and order\_date<'12/31/2021' order by total\_amount desc;

Now I have added one more filter condition to check orders of year 2021



3. select \* from orders join order\_details on orders.order\_id=order\_details.order\_id where orders.total\_amount>25000 and orders.order\_date<'12/31/2021' order by orders.total\_amount desc;



 select \* from stock\_details where stock\_left <200;</li>

Which means we have less stock left for those drugs

