

DBMS ASSIGNMENT

Nikki Gautam

Question 2:

Part B:

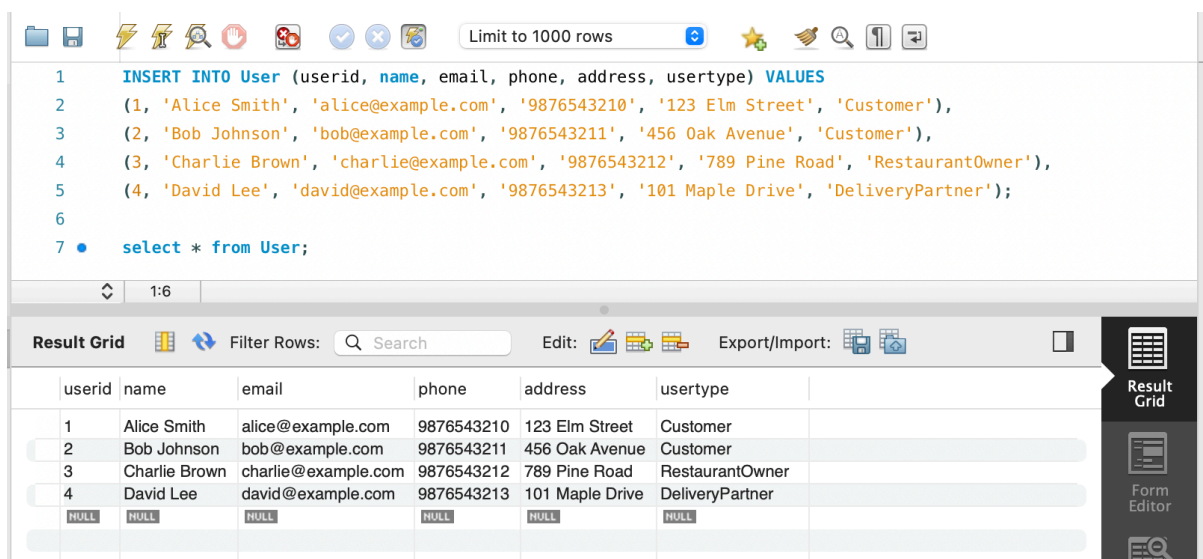
Once you have created the database and tables, we want you populate your tables with some mock data. You can use any programming language for this or produce a SQL script/statement to insert the data.

We want you think about the data types, lengths of fields, constraints, keys etc. while you are at it and make it as real-world ready as it can be.

Inserting Mock Data in the tables: (Using SQL Queries)

1. User table

```
INSERT INTO User (userid, name, email, phone, address, usertype) VALUES
(1, 'Alice Smith', 'alice@example.com', '9876543210', '123 Elm Street', 'Customer'),
(2, 'Bob Johnson', 'bob@example.com', '9876543211', '456 Oak Avenue', 'Customer'),
(3, 'Charlie Brown', 'charlie@example.com', '9876543212', '789 Pine Road', 'RestaurantOwner'),
(4, 'David Lee', 'david@example.com', '9876543213', '101 Maple Drive', 'DeliveryPartner');
```



The screenshot shows a database management tool interface. At the top, there's a toolbar with various icons and a 'Limit to 1000 rows' dropdown. Below the toolbar, the SQL query is entered in a text area. The query is as follows:

```
1 INSERT INTO User (userid, name, email, phone, address, usertype) VALUES
2 (1, 'Alice Smith', 'alice@example.com', '9876543210', '123 Elm Street', 'Customer'),
3 (2, 'Bob Johnson', 'bob@example.com', '9876543211', '456 Oak Avenue', 'Customer'),
4 (3, 'Charlie Brown', 'charlie@example.com', '9876543212', '789 Pine Road', 'RestaurantOwner'),
5 (4, 'David Lee', 'david@example.com', '9876543213', '101 Maple Drive', 'DeliveryPartner');
6
7 • select * from User;
```

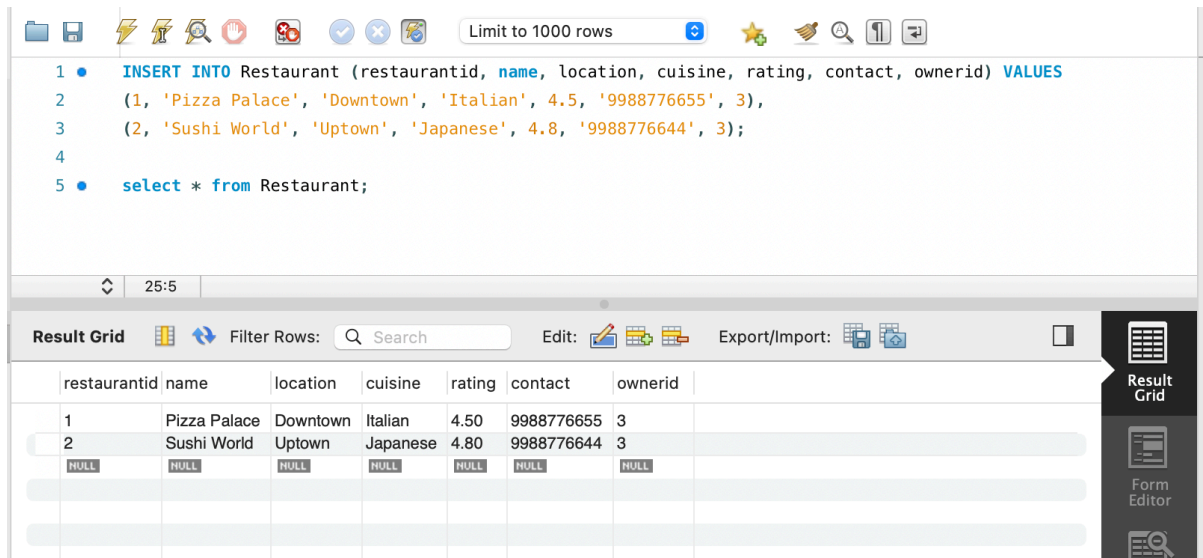
Below the query, the 'Result Grid' is displayed, showing the data inserted into the 'User' table. The grid has columns for 'userid', 'name', 'email', 'phone', 'address', and 'usertype'. The data is as follows:

userid	name	email	phone	address	usertype
1	Alice Smith	alice@example.com	9876543210	123 Elm Street	Customer
2	Bob Johnson	bob@example.com	9876543211	456 Oak Avenue	Customer
3	Charlie Brown	charlie@example.com	9876543212	789 Pine Road	RestaurantOwner
4	David Lee	david@example.com	9876543213	101 Maple Drive	DeliveryPartner
NULL	NULL	NULL	NULL	NULL	NULL

On the right side of the interface, there are buttons for 'Result Grid', 'Form Editor', and a search icon.

2. Restaurant table

```
INSERT INTO Restaurant (restaurantid, name, location, cuisine, rating, contact, ownerid) VALUES
(1, 'Pizza Palace', 'Downtown', 'Italian', 4.5,
'9988776655', 3),
(2, 'Sushi World', 'Uptown', 'Japanese', 4.8,
'9988776644', 3);
```



The screenshot shows a database management interface. At the top, there's a toolbar with various icons and a 'Limit to 1000 rows' dropdown. Below the toolbar, the SQL query is displayed in a text area:

```
1 • INSERT INTO Restaurant (restaurantid, name, location, cuisine, rating, contact, ownerid) VALUES
2 (1, 'Pizza Palace', 'Downtown', 'Italian', 4.5, '9988776655', 3),
3 (2, 'Sushi World', 'Uptown', 'Japanese', 4.8, '9988776644', 3);
4
5 • select * from Restaurant;
```

Below the query, the 'Result Grid' is shown. It has a search bar and 'Filter Rows' option. The grid displays the following data:

restaurantid	name	location	cuisine	rating	contact	ownerid
1	Pizza Palace	Downtown	Italian	4.50	9988776655	3
2	Sushi World	Uptown	Japanese	4.80	9988776644	3
NULL	NULL	NULL	NULL	NULL	NULL	NULL

On the right side of the interface, there are buttons for 'Result Grid', 'Form Editor', and a search icon.

3. MenuItem table

```
INSERT INTO MenuItem (itemid, restaurantid, name, price, preparationtime, availability, category)
VALUES
(1, 1, 'Pepperoni Pizza', 12.99, 20, TRUE, 'Main Course'),
(2, 1, 'Margherita Pizza', 10.99, 15, TRUE, 'Main Course'),
(3, 2, 'California Roll', 8.99, 10, TRUE, 'Appetizer'),
(4, 2, 'Spicy Tuna Roll', 9.99, 12, TRUE, 'Appetizer');
```

Limit to 1000 rows

```

1 • INSERT INTO MenuItem (itemid, restaurantid, name, price, preparationtime, availability, category) VALUES
2   (1, 1, 'Pepperoni Pizza', 12.99, 20, TRUE, 'Main Course'),
3   (2, 1, 'Margherita Pizza', 10.99, 15, TRUE, 'Main Course'),
4   (3, 2, 'California Roll', 8.99, 10, TRUE, 'Appetizer'),
5   (4, 2, 'Spicy Tuna Roll', 9.99, 12, TRUE, 'Appetizer');
6
7 • select * from MenuItem;

```

1:6

Result Grid Filter Rows: Search Edit: Export/Import:

itemid	restaurantid	name	price	preparationtime	availability	category
1	1	Pepperoni Pizza	12.99	20	1	Main Course
2	1	Margherita Pizza	10.99	15	1	Main Course
3	2	California Roll	8.99	10	1	Appetizer
4	2	Spicy Tuna Roll	9.99	12	1	Appetizer
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Result Grid Form Editor

4. Discount table

```

INSERT INTO Discount (discountid, description, percentage, code) VALUES
(1, 'New Year Discount', 10.00, 'NEWYEAR10'),
(2, 'Weekend Special', 15.00, 'WEEKEND15');

```

Limit to 1000 rows

```

1 • INSERT INTO Discount (discountid, description, percentage, code) VALUES
2   (1, 'New Year Discount', 10.00, 'NEWYEAR10'),
3   (2, 'Weekend Special', 15.00, 'WEEKEND15');
4
5 • select * from Discount;

```

1:4

Result Grid Filter Rows: Search Edit: Export/Import:

discountid	description	percentage	code
1	New Year Discount	10.00	NEWYEAR10
2	Weekend Special	15.00	WEEKEND15
NULL	NULL	NULL	NULL

Result Grid Form Editor

5. DeliveryPartner table

```
INSERT INTO DeliveryPartner (deliverypartnerid, name,
phone, vehicledetails) VALUES
(1, 'David Lee', '9876543213', 'Bike'),
(2, 'Eve Green', '9876543214', 'Car');
```

The screenshot shows a database management interface. At the top, there's a toolbar with various icons and a 'Limit to 1000 rows' dropdown. Below the toolbar, the SQL editor contains the following code:

```
1 • INSERT INTO DeliveryPartner (deliverypartnerid, name, phone, vehicledetails) VALUES
2   (1, 'David Lee', '9876543213', 'Bike'),
3   (2, 'Eve Green', '9876543214', 'Car');
4
5 • select * from DeliveryPartner;
```

Below the SQL editor, the 'Result Grid' tab is active, displaying the following data:

deliverypartne...	name	phone	vehicledetails
1	David Lee	9876543213	Bike
2	Eve Green	9876543214	Car
NULL	NULL	NULL	NULL

On the right side of the interface, there are buttons for 'Result Grid', 'Form Editor', and a search icon.

6. Order table

```
INSERT INTO `Order` (orderid, userid, restaurantid,
orderdate, totalamount, discountapplied,
paymentstatus, deliverypartnerid) VALUES
(1, 1, 1, '2025-01-15', 22.98, 1, 'Completed', 1),
(2, 2, 2, '2025-01-16', 17.98, 2, 'Pending', 2);
```

The screenshot shows a database management interface. At the top, there's a toolbar with various icons and a 'Limit to 1000 rows' dropdown. Below the toolbar, the SQL editor contains the following code:

```
1 • INSERT INTO `Order` (orderid, userid, restaurantid, orderdate, totalamount, discountapplied, paymentstatus, c
2   (1, 1, 1, '2025-01-15', 22.98, 1, 'Completed', 1),
3   (2, 2, 2, '2025-01-16', 17.98, 2, 'Pending', 2);
4
5 • select * from `Order`;
```

Below the SQL editor, the 'Result Grid' tab is active, displaying the following data:

orderid	userid	restaurantid	orderdate	totalamount	discountappli...	paymentstat...	deliverypartne...
1	1	1	2025-01-15	22.98	1	Completed	1
2	2	2	2025-01-16	17.98	2	Pending	2
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

On the right side of the interface, there are buttons for 'Result Grid', 'Form Editor', and a search icon.

7. Payment table

```
INSERT INTO Payment (paymentid, orderid,  
paymentmethod, paymentstatus, transactionid) VALUES  
(1, 1, 'Card', 'Completed', 'TXN123456'),  
(2, 2, 'UPI', 'Pending', 'TXN654321');
```

The screenshot shows a database management interface with a toolbar at the top containing icons for file operations, execution, and search. A text area contains the following SQL code:

```
1 • INSERT INTO Payment (paymentid, orderid, paymentmethod, paymentstatus, transactionid) VALUES  
2 (1, 1, 'Card', 'Completed', 'TXN123456'),  
3 (2, 2, 'UPI', 'Pending', 'TXN654321');  
4  
5 • select * from Payment;
```

Below the code, a "Result Grid" displays the data. The grid has a toolbar with "Filter Rows", "Search", "Edit", and "Export/Import" options. The data is as follows:

paymentid	orderid	paymentmeth...	paymentstat...	transactio...
1	1	Card	Completed	TXN123456
2	2	UPI	Pending	TXN654321
NULL	NULL	NULL	NULL	NULL

On the right side, there are buttons for "Result Grid", "Form Editor", and a search icon.

8. Feedback table

```
INSERT INTO Feedback (feedbackid, userid, orderid,  
rating, comments) VALUES  
(1, 1, 1, 4.5, 'Great pizza and delivery was  
quick!'),  
(2, 2, 2, NULL, NULL); -- Feedback pending
```

The screenshot shows a database management interface with a toolbar at the top containing icons for file operations, execution, and search. A text area contains the following SQL code:

```
1 • INSERT INTO Feedback (feedbackid, userid, orderid, rating, comments) VALUES  
2 (1, 1, 1, 4.5, 'Great pizza and delivery was quick!'),  
3 (2, 2, 2, NULL, NULL); -- Feedback pending  
4  
5 • select * from Feedback;
```

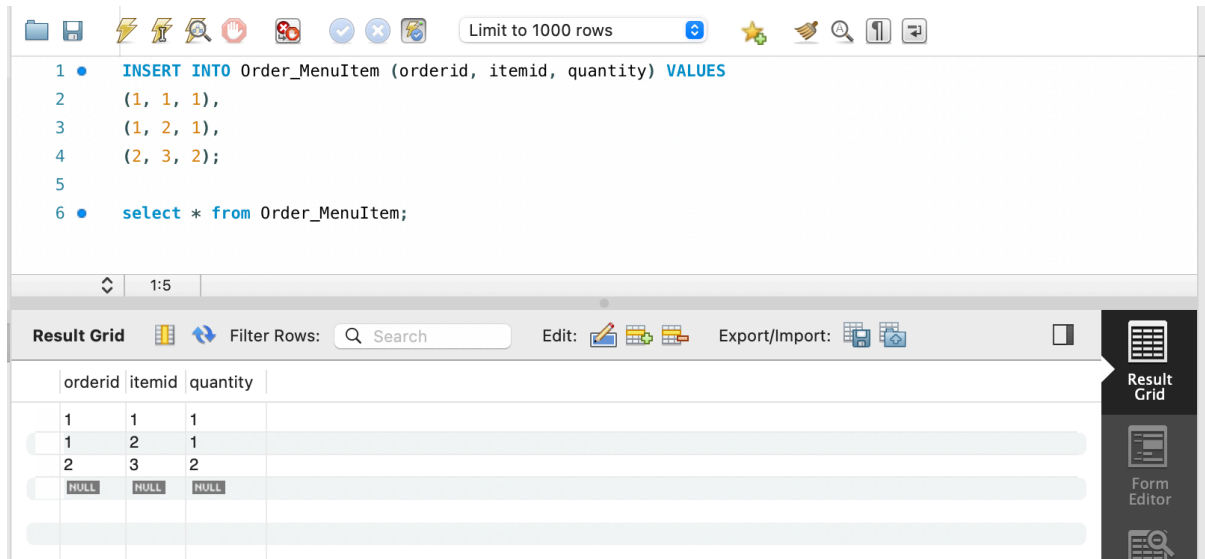
Below the code, a "Result Grid" displays the data. The grid has a toolbar with "Filter Rows", "Search", "Edit", and "Export/Import" options. The data is as follows:

feedbackid	userid	orderid	rating	comments
1	1	1	4.5	Great pizza and delivery was quick!
2	2	2	NULL	NULL
NULL	NULL	NULL	NULL	NULL

On the right side, there are buttons for "Result Grid", "Form Editor", and a search icon.

9. Order_MenuItem table

```
INSERT INTO Order_MenuItem (orderid, itemid,
quantity) VALUES
(1, 1, 1),
(1, 2, 1),
(2, 3, 2);
```



The screenshot shows a database management interface. At the top, there's a toolbar with various icons and a 'Limit to 1000 rows' dropdown. Below the toolbar, the SQL editor contains the following code:

```
1 • INSERT INTO Order_MenuItem (orderid, itemid, quantity) VALUES
2   (1, 1, 1),
3   (1, 2, 1),
4   (2, 3, 2);
5
6 • select * from Order_MenuItem;
```

Below the SQL editor, the 'Result Grid' tab is active, displaying the following data:

orderid	itemid	quantity
1	1	1
1	2	1
2	3	2
NULL	NULL	NULL

On the right side of the interface, there are buttons for 'Result Grid', 'Form Editor', and a search icon.

Data Distribution Justification:

1. User:

- Includes customers, a restaurant owner, and a delivery partner with distinct roles
- Data types align with constraints: “**email**” is unique, and “**usertype**” is constrained to specific values

2. Restaurant:

- Restaurants owned by the same user are included, reflecting real-world scenarios

3. MenuItem:

- Includes multiple items per restaurant with realistic prices, preparation times, and availability

4. Discount:

- Two discounts with descriptive codes and reasonable percentages

5. **DeliveryPartner:**

- Contains multiple delivery partners with unique contact details and vehicle information

6. **Order:**

- Reflects orders by customers, each tied to a restaurant, delivery partner, and discount

7. **Payment:**

- Shows different payment methods and statuses for completed and pending orders

8. **Feedback:**

- Includes feedback for a completed order and a placeholder for pending feedback

9. **Order_MenuItem:**

- Captures the association between orders and the items included in each order with quantities