

# **Indian Institute of Technology Gandhinagar**



## **Databases CS 432**

### **Report: Assignment 2**

#### **Authors**

22110050	Birudugadda Srivibhav	20110101	Pramod Limbore
22110260	Srivaths Vamsi Chaturvedula	20110106	Shriyash Mandavekar
22110124	Sriman reddy	20110071	Haikoo Khandor
22110162	Nikhilesh Myanapuri	20110104	Madhav Kanda

**Under the Supervision of  
Prof. Mayank Singh**

**February 16, 2024**

## Responsibility of G1

**Github Repository:** <https://github.com/Sparky1743/Food-Delivery-System-Website>

### **3.1.1: ACID and Constraints:**

- 1. Atomicity:** In our food delivery system, we ensure that when a delivery person accepts an order, the whole process is like a single package. If something goes wrong (like a payment failure), we can easily undo the entire process and go back to where we started. We make use of the database system's features to handle this, ensuring transactions can be rolled back if there's an error.
- 2. Consistency:** Before processing any order, our system checks to make sure everything is in order. For example, we confirm that each order is assigned to only one restaurant by using foreign keys. We also validate that each order has a proper delivery address and accurate payment info, and this is tracked through data definition.
- 3. Isolation:** To make sure different tasks happening at the same time don't mess with each other, we use a locking system in our database. When a delivery person takes on an order, the relevant parts of the database get locked until the entire process is done. This is made possible using foreign keys in the respective table.
- 4. Durability:** We've made our food delivery system robust by setting up a system that regularly backs up the database to a safe place. Though we haven't put it into action yet, once the real database is in place, we can easily recover it after a system failure. This ensures our food delivery system stands up to the test of time.

Our food delivery management system ensures the dependability and coherence of its database transactions by using the ACID properties. This method makes sure that the data in the database is precise, even when faced with errors or failures.

We have applied these constraints in our database management system to make sure that the data in the database tables follows certain rules and limits. These constraints are present in the SQL dump that we have submitted.

1. ***delivery\_rating***: This constraint ensures that the customer's rating for the delivery is from 0 to 5 (inclusive).
2. ***item\_rating***: This constraint ensures that the customer's rating for the restaurant is from 0 to 5 (inclusive).
3. ***delivery\_charges***: This constraint ensures that an order's delivery charges are not negative (i.e., at least 0).
4. ***amount***: This constraint makes sure that the amount paid for an order is more than 0.
5. ***tip***: This constraint ensures that the tip for a delivery is not negative (i.e., at least 0).
6. ***delivery\_time***: This constraint ensures that an order's delivery time is later than the pickup time.
7. ***payment\_method***: This constraint ensures that the payment method used for an order is one of the permitted options (UPI, net banking, Debit Card, or Credit Card).
8. ***order\_status***: This constraint ensures that the order status is one of the permitted options (pending, processing or delivered).
9. ***email, phone and license\_id*** are UNIQUE.

By applying these constraints, we ensure that the data in our database is correct, consistent, and valid, which helps enhance our system's overall quality and reliability.

### 3.1.2: Indexing, user-defined data type, table extensions:

#### Indexing

- Indexing in a database is employed for search operations to enhance query performance by swiftly locating specific records. By organising data in a structured format, indexing reduces the time and resources required for searching, thereby optimising database retrieval efficiency. This leads to faster data access and improved overall system responsiveness, particularly in large datasets.
- We utilised the "fd beta" database, specifically focusing on the "Food\_Item" entity containing approximately 10,000 instances or rows. Our objective was to observe the time difference between conducting searches with and without indexing, particularly targeting the "food\_item" attribute by searching for a specific URL. The process is described below through screenshots.
- We used the following commands
  - EXPLAIN ANALYZE  
SELECT \* FROM Food\_Item WHERE photo\_url LIKE "%http://weber.info%";
  - CREATE INDEX idx\_food\_item\_photo\_url ON Food\_Item(photo\_url);
  - EXPLAIN ANALYZE  
SELECT \* FROM Food\_Item WHERE photo\_url LIKE "%http://weber.info%";
- Output of Explain analyze command

```
Database changed
mysql> EXPLAIN ANALYZE SELECT * FROM Food_Item WHERE photo_url LIKE "%http://weber.info%";
+-----+
| EXPLAIN
+-----+
| -> Filter: (Food_Item.photo_url like '%http://weber.info%') (cost=999 rows=1083) (actual time=9.01..9.02 rows=1 loops=1)
    |   -> Table scan on Food_Item (cost=999 rows=9751) (actual time=0.191..6.42 rows=10000 loops=1)
    |
+-----+
1 row in set (0.01 sec)

mysql> CREATE INDEX idx_food_item_photo_url ON Food_Item(photo_url);
Query OK, 0 rows affected (0.07 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> EXPLAIN ANALYZE SELECT * FROM Food_Item WHERE photo_url LIKE "%http://weber.info%";
+-----+
| EXPLAIN
+-----+
| -> Filter: (Food_Item.photo_url like '%http://weber.info%') (cost=999 rows=1083) (actual time=8.6..8.61 rows=1 loops=1)
    |   -> Table scan on Food_Item (cost=999 rows=9751) (actual time=0.131..6.07 rows=10000 loops=1)
    |
+-----+
1 row in set (0.01 sec)
```

Performing Search	Without Indexing	With Indexing
Execution Duration	0.191 sec	0.131 sec

## User-defined data types:

A user-defined data type is one that the user creates instead of the system. It lets users make their own data type. The following code shows how to do this:

```
-- CREATE TYPE details AS (
--   email varchar(50) NOT NULL,
--   phone varchar(50) NOT NULL
-- );
-- As CREATE TYPE is not in mySQL we use JSON datatype

DROP TABLE IF EXISTS Customers;
CREATE TABLE Customers (
  customer_id int(8) NOT NULL,
  first_name varchar(50) NOT NULL,
  middle_name varchar(50) DEFAULT NULL,
  last_name varchar(50) NOT NULL,
  dob date DEFAULT NULL,
  age int(2) DEFAULT NULL,
  contact_details JSON, -- JSON data type for contact details
  password varchar(50) NOT NULL,
  PRIMARY KEY (customer_id)
) ;

INSERT INTO Customers (customer_id, first_name, middle_name, last_name, dob, age, contact_details, password)
VALUES(1,'Rajesh','Kumar','Sharma','1980-05-10',41,'{"email":"rajesh.sharma@example.com","phone":"9876543210"}','password1');
```

Here, we have set the datatype of “contact\_details” to JSON as CREATE TYPE is not in MySQL. So, we can now insert any data that is of type JSON. The content of JSON objects can be of any type like numeric, char, varchar, int, etc

## Table Extension

In MySQL, extending a table involves augmenting it with additional columns without altering the existing columns or data. Essentially, it's about expanding upon an existing table by incorporating new columns while retaining the original structure intact. This approach proves beneficial when you need to integrate fresh information into an existing table without creating an entirely new one.

There are some methods to achieve the result; they are described below

- To achieve this, you can utilise the `ALTER TABLE table_name ADD COLUMN column_name datatype` command to seamlessly add more columns to the extended table. We have added an additional column tip comment for the Delivery table, which is shown below.

Table before using the command

order_id	agent_id	delivery_review	delivery_rating	delivery_charges	pickup_time	delivery_time	delivery_status	tip
3557	2457	Porro voluptatum possimus quis. Debitis cupiditate	0.00	183.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	993.00
6403	11855		5.00	26.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	91.00
17874	20525		0.00	191.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	956.00
66537	21888		0.00	136.00	2004-11-29 00:00:00	0000-00-00 00:00:00	delivered	332.00
74834	22577		5.00	113.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	768.00
89909	29866		1.00	125.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	30.00
112462	32046		0.00	74.00	2004-11-00 00:00:00	0000-00-00 00:00:00	delivered	661.00
116681	36778	Atque rerum ut omnis consectetur aspernatur ea. Ex	0.00	183.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	978.00
124666	43622		0.00	76.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	507.00

Command used

```
mysql> ALTER TABLE Delivery ADD COLUMN tip_comment varchar(255);
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Table after using the command

order_id	agent_id	delivery_review	delivery_rating	delivery_charges	pickup_time	delivery_time	delivery_status	tip	tip_comment
3557	2457	Porro voluptatum possimus quis. Debitis cupiditate	0.00	183.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	993.00	NULL
6403	11855		5.00	26.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	91.00	NULL
17874	20525		0.00	191.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	956.00	NULL
66537	21888		0.00	136.00	2004-11-29 00:00:00	0000-00-00 00:00:00	delivered	332.00	NULL
74834	22577		5.00	113.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	768.00	NULL
89909	29866		1.00	125.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	30.00	NULL
112462	32046		0.00	74.00	2004-11-00 00:00:00	0000-00-00 00:00:00	delivered	661.00	NULL
116681	36778	Atque rerum ut omnis consectetur aspernatur ea. Ex	0.00	183.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	978.00	NULL
124666	43622		0.00	70.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	507.00	NULL
124661	54456	Non amet ex sit sed. Quo facere fuga nam consequuntur	0.00	78.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	451.00	NULL
126247	56987	Quaerat voluptatem quia ut nihil vero. Rerum inven	0.00	7.00	0000-00-00 00:00:00	0000-00-00 00:00:00	delivered	817.00	NULL

- We can use `CREATE TABLE Top_Orders as (SELECT * FROM Ordered_items WHERE (Ordered_items.item_rating=5))`; This will create a table Top\_Orders, which will have all columns from Orders and contain all the orders whose item rating is 5.

Table before using the command

order_id	item_id	item_quantity	item_rating	item_review	notes
3557	59	18	3.00	Vero molestiae ex nulla dicta et maxime. Libero re	Esse dolores dolores voluptate et est in et. Illum ratione iste nihil maiores in tempora non.
6403	79	10	1.00	Neniam dolores et corporis iste nisi facere. Offic	
17874	81	16	0.00	Atque at vel rerum accusantium magnam. Earum quo	
66537	240	19	0.00	Tempora dolorum est repellat voluptatum. Libero ac	Laborum aut et et aut et.
74834	288	20	2.00	Odit omnis delectus qui alias. Dolores aut perfere	Adipisci quis vitae magnam quas.
89909	333	16	0.00		Rem voluptates tempore tempora molestiae porro molestiae ipsa voluptatem. Tempore in soluta fugit er
112462	386	4	0.00	Eos perferendis occaecati et nulla velit. Earum es	
116681	383	13	0.00	Porro aut maxime asperiores ratione cumque dolore.	Aut deserunt enim accusantium doloremque suscipit dignissimos praesentium maiores. Aut ut dolor in d
124666	581	11	1.00	Perspicillatis repudiandae dolorem sed dolor. A eius	Nunquam qui eum tempore enim nobis quo animi aperiam.
124661	543	6	0.00		

Command used

```
mysql> CREATE TABLE Top_Orders as (SELECT * FROM Ordered_items WHERE (Ordered_items.item_rating=5));
Query OK, 859 rows affected (0.05 sec)
Records: 859  Duplicates: 0  Warnings: 0
```

Table after using the command

order_id	item_id	item_quantity	item_rating	item_review	notes
409295	3448	18	5.00	Quis paratur est et est qui deserunt. Modi quae r	Recusandae facere eum non non aliquid. Culpa expedita id ad.
763467	6280	18	5.00		
782950	6353	10	5.00	Est sunt deserunt et eorum. Excepturi excepturi al	
930497	6789	13	5.00	A commodi fugiat et rerum enim recusandae. Iure se	Ut cunque quam cum praesentium a. Et voluptates natus laboriosam voluptatem iusto.
873678	7591	2	5.00		Eos soluta amet velit deleniti eum.
902805	7991	19	5.00		
1036421	9039	14	5.00	Et autem explicabo ea odit iusto error. Autem sed	
1386554	12178	2	5.00	Suscipit autem voluptatem odit. Magni in dicta ull	
1399563	12529	2	5.00	Iusto amet ex sed suscipit expedita voluptatum. Pe	Laborum sed similique consectetur nisi explicabo tenetur quis. Minus perspiciatis sit dignissimos po
1404427	12630	16	5.00	Distinctio deleniti voluptatem molestias id except	

3. We can create a new table with an additional column we want to add and then use a foreign key constraint to link the new table to the original.  
The detailed code and results are provided in the image below

```
mysql> CREATE TABLE Restaurant_extended(restaurant_id INT PRIMARY KEY, famous_item varchar(255), FOREIGN KEY (restaurant_id) REFERENCES Restaurant(restaurant_id));
Query OK, 0 rows affected (0.02 sec)

mysql> SELECT * FROM Restaurant_extended LIMIT 10;
Empty set (0.00 sec)

mysql> INSERT INTO Restaurant_extended(restaurant_id) SELECT restaurant_id FROM Restaurant;
Query OK, 1000 rows affected (0.02 sec)
Records: 1000  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM Restaurant_extended LIMIT 10;
+-----+-----+
| restaurant_id | famous_item |
+-----+-----+
|        149 |      NULL |
|        171 |      NULL |
|        346 |      NULL |
|        435 |      NULL |
|        564 |      NULL |
|        650 |      NULL |
|        675 |      NULL |
|        920 |      NULL |
|        929 |      NULL |
|        958 |      NULL |
+-----+-----+
10 rows in set (0.00 sec)
```

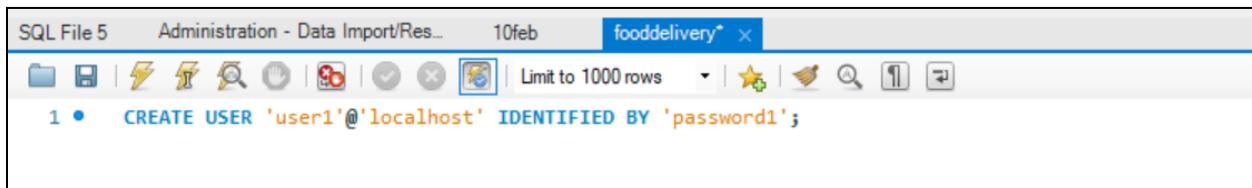
### Additional Note:

- In type 1, where we used the ALTER command, we will get NULL values to the added columns initially as no data has been inserted into the same and in type 3, the CREATE TABLE statement creates a new table with the same structure as the original table without any data. In other words, the new table is empty, and all the columns have NULL values until you insert data into them. For type 3, the created table Restaurant\_extended has the restaurant\_id empty initially, but I have inserted the data into that column(Please refer to the figure in type 3 for the same).
- In type 2, you use the CREATE TABLE ... AS SELECT statement to create a new table based on a SELECT query. The SELECT query retrieves data from the Ordered\_item table and inserts it into the new table so it contains actual data.

## Responsibility of G2

We imported data into the MySQL server, then:

### 3.2.1 Created a user named "user1" with the password "password1"

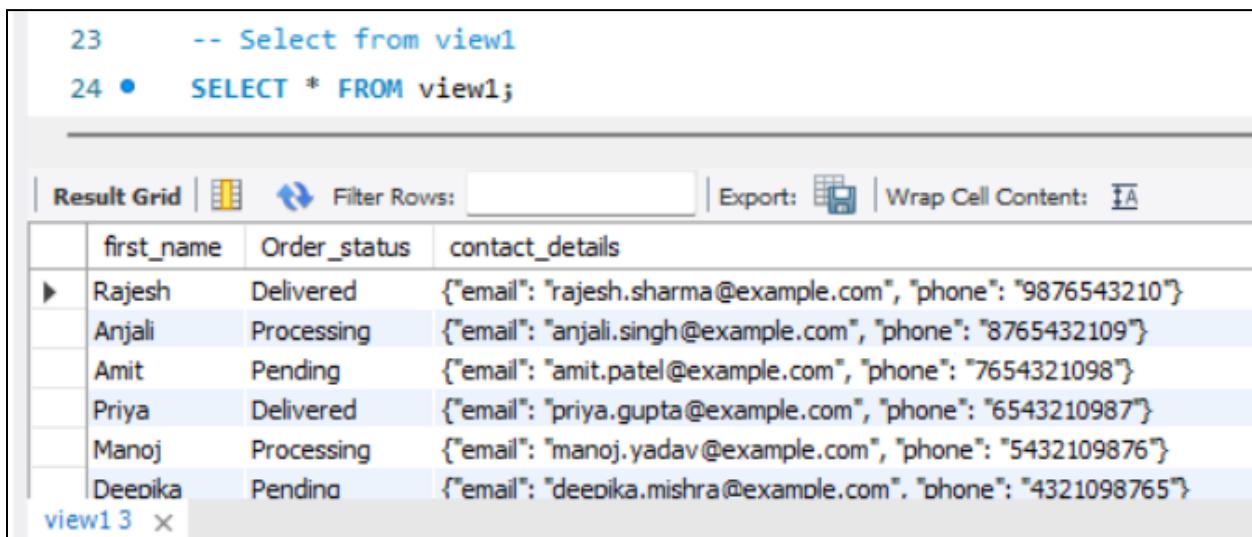


The screenshot shows the MySQL Workbench interface. The title bar says "SQL File 5 Administration - Data Import/Res... 10feb fooddelivery\*". The main pane contains the SQL command: "CREATE USER 'user1'@'localhost' IDENTIFIED BY 'password1';". Below the command are various icons for database management.

### 3.2.2 We then Created Views on the two tables formed by G1 as view1 and view2.

```
11    -- Create view1
12 •  CREATE VIEW view1 AS
13      SELECT customers.first_name, Orders.Order_status, customers.contact_details
14      FROM Orders
15      JOIN customers ON Orders.customer_id = customers.customer_id;
16
17    -- Create view2
18 •  CREATE VIEW view2 AS
19      SELECT Restaurant.restaurant_name, Food_Item.item_name, Restaurant.contact_details
20      FROM Food_Item
21      JOIN Restaurant ON Food_Item.restaurant_id = Restaurant.restaurant_id;
```

view1 contains columns as first\_name, contact\_details(column which has user-defined data type) from customers table and Order\_status from order table



The screenshot shows the MySQL Workbench interface with a result grid. The grid has columns: first\_name, Order\_status, and contact\_details. The data is as follows:

	first_name	Order_status	contact_details
▶	Rajesh	Delivered	{"email": "rajesh.sharma@example.com", "phone": "9876543210"}
	Anjali	Processing	{"email": "anjali.singh@example.com", "phone": "8765432109"}
	Amit	Pending	{"email": "amit.patel@example.com", "phone": "7654321098"}
	Priya	Delivered	{"email": "priya.gupta@example.com", "phone": "6543210987"}
	Manoj	Processing	{"email": "manoj.yadav@example.com", "phone": "5432109876"}
	Deepika	Pendina	{"email": "deepika.mishra@example.com", "phone": "4321098765"}

view2 contains columns as restaurant\_name, contact\_details(column which has user-defined data type) from Restaurant table and item\_name from Item table

```
26    -- Select from view2
27 •  SELECT * FROM view2;
```

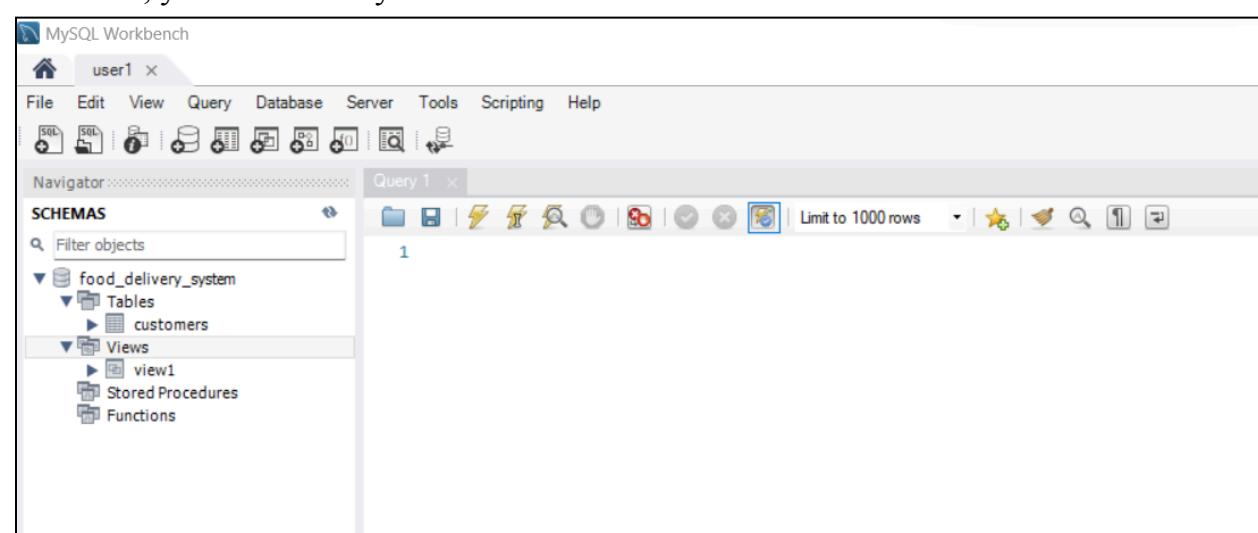
Result Grid	Filter Rows:	Export:	Wrap Cell Content:
restaurant_name	item_name	contact_details	
Taj Mahal Restaurant	Paneer Tikka	{"email": "tajmahal@example.com", "phone": "+91 9876543210"}	
Spice Garden	Chicken Biryani	{"email": "spicegarden@example.com", "phone": "+91 8765432109"}	
Punjabi Dhaba	Butter Chicken	{"email": "punjabidhaba@example.com", "phone": "+91 7654321098"}	
Southern Spice	Masala Dosa	{"email": "southern spice@example.com", "phone": "+91 6543210987"}	
The Mughal Feast	Mutton Rogan Josh	{"email": "mughalfeast@example.com", "phone": "+91 5432109876"}	
Coastal Curvv House	Fish Currv	{"email": "coastalcurrv@example.com", "phone": "+91 4321098765"}	

3.2.3 & 3.2.4 Granted "user1" the following permissions on the customers table:  
SELECT, UPDATE, DELETE

Granted "user1" the following permissions on view1: SELECT

```
29    -- Grant permissions to 'user1'
30 •  GRANT SELECT, UPDATE, DELETE ON customers TO 'user1'@'localhost';
31 •  GRANT SELECT ON view1 TO 'user1'@'localhost';
--
```

For user1, you can see only the customers' table and view1 as follows:



### 3.2.5 Operations on customers table as user1:

- SELECT:

```
1 • use food_delivery_system;
2 • SELECT * FROM customers;
```

Result Grid								
	customer_id	first_name	middle_name	last_name	dob	age	contact_details	password
▶	1	Rajesh	Kumar	Sharma	1980-05-10	41	{"email": "rajesh.sharma@example.com", "phon...}	password1
	2	Anjali	HULL	Singh	1995-08-20	26	{"email": "anjali.singh@example.com", "phone":...}	password2
	3	Amit	Kumar	Patel	1988-03-15	33	{"email": "amit.patel@example.com", "phone": "...}	password3
	4	Priya	HULL	Gupta	1992-11-25	29	{"email": "priya.gupta@example.com", "phone": "...}	password4
	5	Manoj	Kumar	Yadav	1982-07-08	39	{"email": "manoj.yadav@example.com", "phone...}	password5
	6	Deepika	HULL	Mishra	1987-09-18	34	{"email": "deepika.mishra@example.com", "pho...}	password6
=								

- UPDATE:

```
1 • use food_delivery_system;
2 • SELECT * FROM customers;
3
4 • UPDATE customers SET password = 'raju1234' WHERE customer_id=1;
```

Output		
#	Time	Action
1	16:19:50	UPDATE customers SET password = 'raju1234' WHERE customer_id=1

Message  
1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0 Duration / Fetch  
0.016 sec

- DELETE:

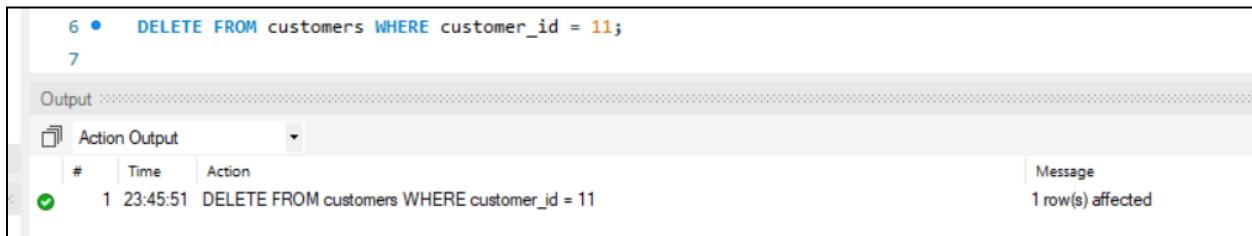
Here, when attempting to delete the customer with customer\_id = 15, an error occurs due to the usage of customer\_id as a foreign key in the 'orders' table without specifying 'ON DELETE CASCADE' for that foreign key constraint.

```
6 • DELETE FROM customers WHERE customer_id = 15;
7
```

Output		
#	Time	Action
1	23:03:02	DELETE FROM customers WHERE customer_id = 15

Message  
Error Code: 1217. Cannot delete or update a parent row: a foreign key constraint fails

After specifying the 'ON DELETE CASCADE' as that foreign key constraint, we are able to perform the delete operation



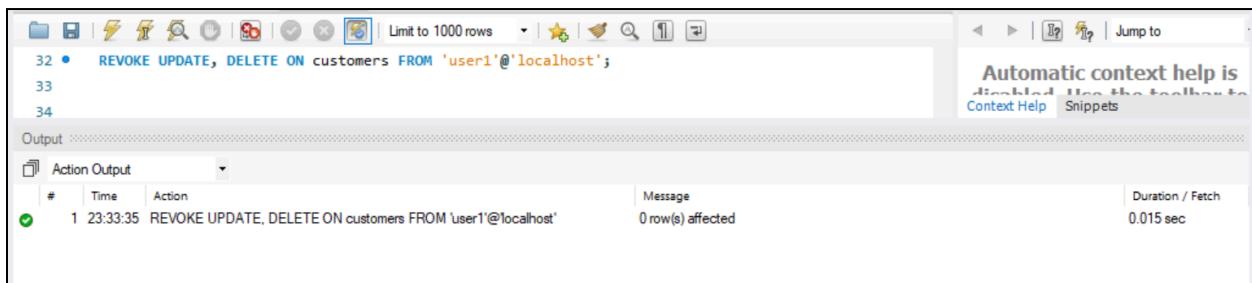
The screenshot shows the MySQL Workbench interface. In the SQL tab, the following command is run:

```
6 •  DELETE FROM customers WHERE customer_id = 11;
```

The output pane shows the result of the command:

#	Time	Action	Message	Duration / Fetch
1	23:45:51	DELETE FROM customers WHERE customer_id = 11	1 row(s) affected	

**3.2.6** For revoking the permission for user1 on the customers table, we can use



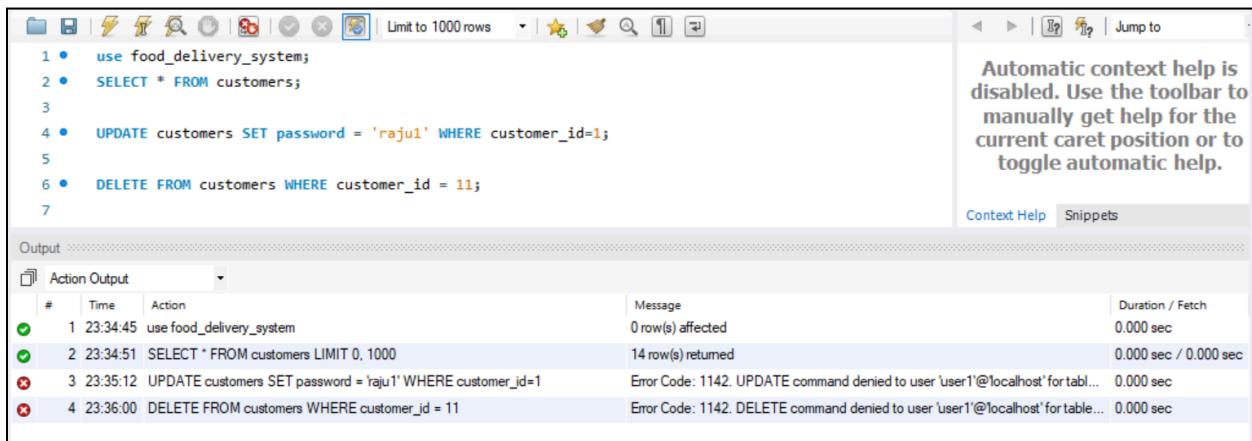
The screenshot shows the MySQL Workbench interface. In the SQL tab, the following command is run:

```
32 •  REVOKE UPDATE, DELETE ON customers FROM 'user1'@'localhost';
```

The output pane shows the result of the command:

#	Time	Action	Message	Duration / Fetch
1	23:33:35	REVOKE UPDATE, DELETE ON customers FROM 'user1'@'localhost'	0 row(s) affected	0.015 sec

After revoking the permission to update and delete, the user can not update and delete the customers table.



The screenshot shows the MySQL Workbench interface. In the SQL tab, the following commands are run:

```
1 •  use food_delivery_system;
2 •  SELECT * FROM customers;
3
4 •  UPDATE customers SET password = 'raju1' WHERE customer_id=1;
5
6 •  DELETE FROM customers WHERE customer_id = 11;
```

The output pane shows the results of these commands. The UPDATE command fails with an error, and the DELETE command fails with an error. The context help panel indicates that automatic context help is disabled.

#	Time	Action	Message	Duration / Fetch
1	23:34:45	use food_delivery_system	0 row(s) affected	0.000 sec
2	23:34:51	SELECT * FROM customers LIMIT 0, 1000	14 row(s) returned	0.000 sec / 0.000 sec
3	23:35:12	UPDATE customers SET password = 'raju1' WHERE customer_id=1	Error Code: 1142. UPDATE command denied to user 'user1'@'localhost' for table...	0.000 sec
4	23:36:00	DELETE FROM customers WHERE customer_id = 11	Error Code: 1142. DELETE command denied to user 'user1'@'localhost' for table...	0.000 sec

### 3.2.7 SELECT command on view1

The screenshot shows the MySQL Workbench interface. In the SQL editor, the following commands are run:

```
1 • use food_delivery_system;
2 • SELECT * FROM view1;
3
```

The results are displayed in a Result Grid:

	first_name	Order_status	contact_details
▶	Rajesh	Delivered	{"email": "rajesh.sharma@example.com", "phon...
▶	Anjali	Processing	{"email": "anjali.singh@example.com", "phone":...
▶	Amit	Pending	{"email": "amit.patel@example.com", "phone": "...
▶	Priya	Delivered	{"email": "priya.gupta@example.com", "phone":..."
▶	Manoi	Processing	{"email": "manoi.vadav@example.com", "phone":...

### UPDATE command on view1

The user cannot update entries in view1 as they do not have permission to update the underlying tables in the database.

The screenshot shows the MySQL Workbench interface. In the SQL editor, the following commands are run:

```
1 • use food_delivery_system;
2 • SELECT * FROM view1;
3
4 • UPDATE view1 SET order_status = 'Pending' WHERE first_name='Rajesh';
5
```

A tooltip message is displayed in the top right corner:

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to Context Help Snippets

The Output window shows the following log:

#	Time	Action	Message	Duration / Fetch
1	23:39:39	SELECT * FROM view1 LIMIT 0, 1000	14 row(s) returned	0.000 sec / 0.000 sec
2	23:41:33	UPDATE view1 SET order_status = 'Pending' WHERE first_name='Rajesh'	Error Code: 1142. UPDATE command denied to user 'user1'@localhost' for table 'view1'	0.000 sec

### DELETE command on view1

The user cannot delete entries from view1 as they do not have permission to delete from the underlying tables in the database.

The screenshot shows the MySQL Workbench interface. In the SQL editor, the following commands are run:

```
1 • use food_delivery_system;
2 • SELECT * FROM view1;
3
4 • UPDATE view1 SET order_status = 'Pending' WHERE first_name='Rajesh';
5
6 • DELETE FROM customers WHERE first_name = 'Rajesh';
7
```

A tooltip message is displayed in the top right corner:

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

The Output window shows the following log:

#	Time	Action	Message	Duration / Fetch
1	23:39:39	SELECT * FROM view1 LIMIT 0, 1000	14 row(s) returned	0.000 sec / 0.000 sec
2	23:41:33	UPDATE view1 SET order_status = 'Pending' WHERE first_name='Rajesh'	Error Code: 1142. UPDATE command denied to user 'user1'@localhost' for table 'view1'	0.000 sec
3	23:43:07	SELECT * FROM view1 LIMIT 0, 1000	14 row(s) returned	0.000 sec / 0.000 sec
4	23:43:59	DELETE FROM customers WHERE first_name = 'Rajesh'	Error Code: 1142. DELETE command denied to user 'user1'@localhost' for table 'customers'	0.000 sec

## Referential integrity:

1. If we try to delete the delivery agent from the database, it would leave that specific delivery without assigning it to another delivery agent.

The screenshot shows the MySQL Workbench interface. In the SQL tab, the following command is run:

```
48
49 •  delete from delivery_agent where agent_id = 2;
50
51
52
53
```

In the Output tab, the results of the operation are shown in a table:

#	Time	Action	Message	Duration / Fetch
1	22:33:15	delete from delivery_agent where agent_id = 2	1 row(s) affected	0.000 sec
2	22:33:31	SELECT * FROM food_delivery_system.delivery_agent LIMIT 0, 1000	14 row(s) returned	0.000 sec / 0.000 sec
3	22:33:34	SELECT * FROM food_delivery_system.delivery_agent LIMIT 0, 1000	14 row(s) returned	0.000 sec / 0.000 sec

A tooltip message is visible on the right side of the interface: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help." It also includes links to "Context Help" and "Snippets".

Solution: We can tackle this problem by not including '**ON DELETE CASCADE**' in the delivery table where delivery\_agent is a foreign key.

The screenshot shows the MySQL Workbench interface. In the SQL tab, the following CREATE TABLE statement is run:

```
212 •  CREATE TABLE Delivery (
213     order_id int(8) DEFAULT NULL,
214     agent_id int(8) DEFAULT NULL,
215     delivery_review varchar(50) DEFAULT NULL,
216     delivery_rating decimal(3,2) DEFAULT NULL,
217     delivery_charges decimal(10,2) DEFAULT NULL,
218     pickup_time datetime DEFAULT NULL,
219     delivery_time datetime DEFAULT NULL,
220     delivery_status varchar(50) DEFAULT NULL,
221     tip decimal(10,2) DEFAULT NULL,
222     KEY order_id (order_id),
223     KEY agent_id (agent_id),
224     CONSTRAINT Delivery_ibfk_1 FOREIGN KEY (order_id) REFERENCES Orders (order_id) ON DELETE CA
225     CONSTRAINT Delivery_ibfk_2 FOREIGN KEY (agent_id) REFERENCES Delivery_Agent (agent_id)
226 );
227
```

In the Output tab, the results of the operation are shown in a table:

#	Time	Action	Message	Duration / Fetch
			Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.	

A tooltip message is visible on the right side of the interface: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help." It also includes links to "Context Help" and "Snippets".

After that, if we try to delete the delivery\_agent, it will result in an error saying that the foreign constraint fails.

The screenshot shows the MySQL Workbench interface. In the SQL tab, the following command is run:

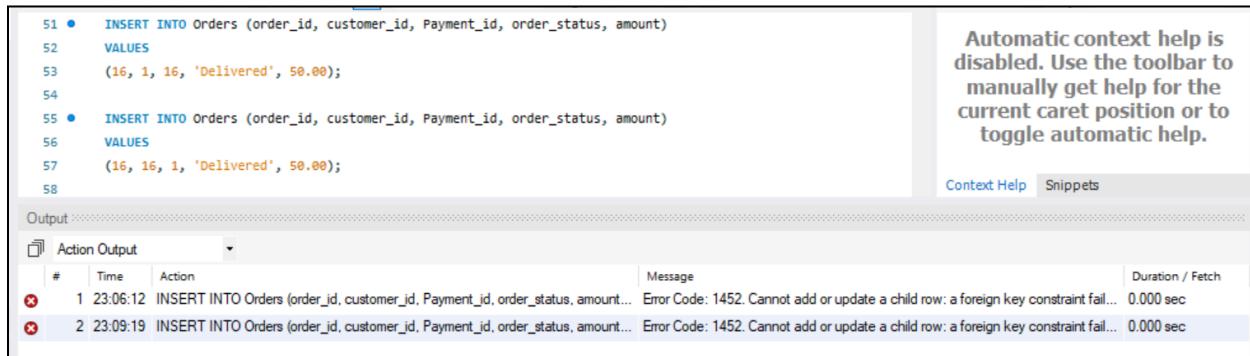
```
49 •  delete from delivery_agent where agent_id = 2;
50
51
52
```

In the Output tab, the results of the operation are shown in a table:

#	Time	Action	Message	Duration / Fetch
1	22:58:55	delete from delivery_agent where agent_id = 2	Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails.	0.000 sec

A tooltip message is visible on the right side of the interface: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help." It also includes links to "Context Help" and "Snippets".

2. If you try to add an order without a valid payment\_id or customer\_id, this will violate the referential integrity between the order and customer tables.



```

51 •  INSERT INTO Orders (order_id, customer_id, Payment_id, order_status, amount)
52   VALUES
53     (16, 1, 16, 'Delivered', 50.00);
54
55 •  INSERT INTO Orders (order_id, customer_id, Payment_id, order_status, amount)
56   VALUES
57     (16, 16, 1, 'Delivered', 50.00);
58

```

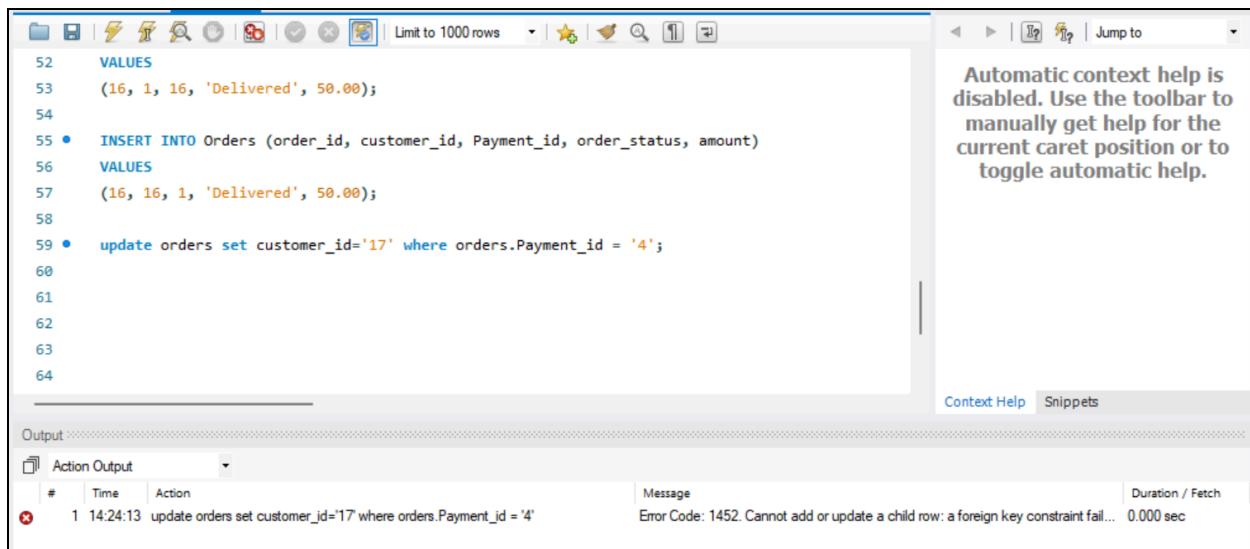
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

Output:

Action Output	#	Time	Action	Message	Duration / Fetch
	1	23:06:12	INSERT INTO Orders (order_id, customer_id, Payment_id, order_status, amount)	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fail...	0.000 sec
	2	23:09:19	INSERT INTO Orders (order_id, customer_id, Payment_id, order_status, amount)	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fail...	0.000 sec

3. If you try to update the order with an invalid customer\_id, it will violate the referential integrity between the customer and the orders table.



```

52   VALUES
53     (16, 1, 16, 'Delivered', 50.00);
54
55 •  INSERT INTO Orders (order_id, customer_id, Payment_id, order_status, amount)
56   VALUES
57     (16, 16, 1, 'Delivered', 50.00);
58
59 •  update orders set customer_id='17' where orders.Payment_id = '4';
60
61
62
63
64

```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

Output:

Action Output	#	Time	Action	Message	Duration / Fetch
	1	14:24:13	update orders set customer_id='17' where orders.Payment_id = '4'	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fail...	0.000 sec

4. Delivery\_rating should be between 0 and 5. If any user tries to update the rating to greater than five, the database will deny the request.

Solution: To Include the above constraint for delivery\_rating, we need to include.

**CONSTRAINT CHK\_delivery\_rating\_range CHECK (delivery\_rating >= 0 AND delivery\_rating <= 5)**

The screenshot shows a MySQL Workbench interface. In the SQL pane, lines 58 through 63 are visible:

```

58
59 • update orders set customer_id='17' where orders.Payment_id = '4';
60
61 • update delivery set delivery_rating = 6 where order_id = 3;
62
63
64

```

In the Output pane, there is one entry:

#	Time	Action	Message	Duration / Fetch
1	14:41:38	update delivery set delivery_rating = 6 where order_id = 3	Error Code: 3819. Check constraint 'CHK_delivery_rating_range' is violated.	0.000 sec

A tooltip on the right side of the interface states: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

5. Another constraint is that the delivery\_time should always be after pickup.

The screenshot shows a MySQL Workbench interface. In the SQL pane, lines 58 through 68 are visible:

```

58
59 • update orders set customer_id='17' where orders.Payment_id = '4';
60
61 • update delivery set delivery_rating = 6 where order_id = 3;
62
63 • INSERT INTO Delivery (order_id, agent_id, delivery_review, delivery_rating, delivery_charges,
64   VALUES
65   (16, 1, 'Good service', 4, 5.00, '2024-02-14 12:30:00', '2024-02-14 12:00:00', 'Delivered', 2
66
67
68
69

```

In the Output pane, two entries are shown:

#	Time	Action	Message	Duration / Fetch
1	14:41:38	update delivery set delivery_rating = 6 where order_id = 3	Error Code: 3819. Check constraint 'CHK_delivery_rating_range' is violated.	0.000 sec
2	14:45:56	INSERT INTO Delivery (order_id, agent_id, delivery_review, delivery_rating, deli...	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fail...	0.000 sec

A tooltip on the right side of the interface states: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

6. Similarly, restaurant\_rating should be less than 5 and greater than 0.

7. Order\_status can only take values from 'Delivered', 'Processing', and 'Pending'.

Solution: `order_status ENUM('Delivered', 'Processing', 'Pending') NOT NULL`

The screenshot shows a MySQL Workbench interface. In the SQL pane, lines 62 through 74 are visible:

```

62
63 • INSERT INTO Delivery (order_id, agent_id, delivery_review, delivery_rating, delivery_charges,
64   VALUES
65   (16, 1, 'Good service', 4, 5.00, '2024-02-14 12:30:00', '2024-02-14 12:00:00', 'Delivered', 2
66
67 • update orders set order_status = 'Remaining' where order_id = 2;
68
69
70
71
72
73
74

```

In the Output pane, one entry is shown:

#	Time	Action	Message	Duration / Fetch
1	14:57:30	update orders set order_status = 'Remaining' where order_id = 2	Error Code: 1265. Data truncated for column 'order_status' at row 1	0.000 sec

A tooltip on the right side of the interface states: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

8. Payment\_status can only take values from ‘Successful’, ‘Failed’ or ‘Pending’.

Updating a payment with an invalid payment status will result in an error.

Solution: **payment\_status ENUM('Successful', 'Failed', 'Pending') NOT NULL**

The screenshot shows a MySQL Workbench interface. In the SQL editor pane, lines 64 through 73 are visible, containing DML statements. Lines 67 and 69 are marked with a red dot, indicating they are the source of the error. The output pane shows a single row in the 'Action Output' table:

#	Time	Action	Message	Duration / Fetch
1	15:02:10	update payment set payment_status = 'Unsuccessful' where payment_id = 2	Error Code: 1265. Data truncated for column 'payment_status' at row 1	0.000 sec

A tooltip on the right side of the interface states: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

## Responsibility of G1 and G2

### 3.3.1 Listing out the restaurant with the highest rating

*Relational algebra:*

$$\pi_{\text{restaurant\_name}} - \pi_{\text{Restaurant.restaurant\_name}}(\sigma_{\text{Restaurant.rating} < d.\text{rating}}(\text{Restaurant} \times \rho_d \text{ Restaurant}))$$

To calculate the restaurant with the highest rating, we first have to compute a temporary relation consisting of restaurants that do not have the largest rating. Then, we take the set difference between the relation restaurant name and the temporary relation.

The screenshot shows a database management interface with a query editor and a results pane. The query editor contains the following SQL code:

```
71 • SELECT Restaurant.restaurant_name
72   FROM Restaurant
73   LEFT JOIN (
74     SELECT Restaurant.rating
75       FROM Restaurant
76   ) AS d ON Restaurant.rating < d.rating
77   WHERE d.rating IS NULL;
78
79
80
```

The results pane shows a single row in the 'Result Grid' table:

restaurant_name
Bengali Bhavan

Below the results, the 'Output' section displays the execution log:

#	Time	Action	Message	Duration / Fetch
1	18:17:05	SELECT Restaurant.restaurant_name FROM Restaurant LEFT JOIN ( SELE...	1 row(s) returned	0.000 sec / 0.000 sec

### 3.3.2 Average, max and min age of customers:

The screenshot shows a database management interface with a query editor and a results pane. The query editor contains the following SQL code:

```
10  SELECT AVG(age) AS average_age, MAX(age) as max_age, MIN(age) as min_age
11  FROM Customers;
12
```

The results pane shows the calculated statistics in a table:

average_age	max_age	min_age
32.2000	41	23

### **Relational algebra:**

$$\pi_{\text{average\_age}, \text{max\_age}, \text{min\_age}}(\gamma_{\text{AVG}(\text{age}), \text{MAX}(\text{age}), \text{MIN}(\text{age})}(\text{Customers}))$$

**3.3.3** Insert an entry for Delivery with negative delivery\_rating on which our database throws an error:

The screenshot shows a SQL interface with the following command:

```
1 • INSERT INTO Delivery (order_id, agent_id, delivery_review, delivery_rating, delivery_charges, pickup_time, delivery_time, delivery_status, tip)
2 VALUES
3     (1, 2, 'Good service', -1, 5.00, '2024-02-14 12:00:00', '2024-02-14 12:30:00', 'Delivered', 2.50);
```

In the Output pane, there is an error message:

Action Output	#	Time	Action	Message
	1	17:27:55	INSERT INTO Delivery (order_id, agent_id, delivery_review, delivery_rating, delivery_charges, pickup_time, delivery_time, delivery_status, tip)	VALUES (1, 2, 'Good service', -1, 5.00, '2024-02-14 12:00:00', '2024-02-14 12:30:00', 'Delivered', 2.50)

Error Code: 3819. Check constraint 'CHK\_delivery\_rating\_range' is violated.

### **Relational algebra:**

$$\text{Delivery} \leftarrow \text{Delivery} \cup \{(1, 2, 'Good service', -1, 5.00, '2024-02-14 12:00:00', '2024-02-14 12:30:00', 'Delivered', 2.50)\}$$

**3.3.4** Attempting to insert an existing customer with customer\_id = 1

The screenshot shows a SQL interface with the following command:

```
5      -- Attempting to insert an existing customer with customer_id = 1
6 • INSERT INTO Customers (customer_id, first_name, middle_name, last_name, dob, age, contact_details, password)
7 VALUES
8     (1, 'Vasu', 'das', 'rao', '1980-04-10', 43, '{"email": "Vasu.rao@example.com", "phone": "9676443210"}', 'password3');
```

In the Output pane, there is an error message:

Action Output	#	Time	Action	Message
	1	19:32:05	INSERT INTO Customers (customer_id, first_name, middle_name, last_name, dob, age, contact_details, pass...	Error Code: 1062. Duplicate entry '1' for key 'customers.PRIMARY'

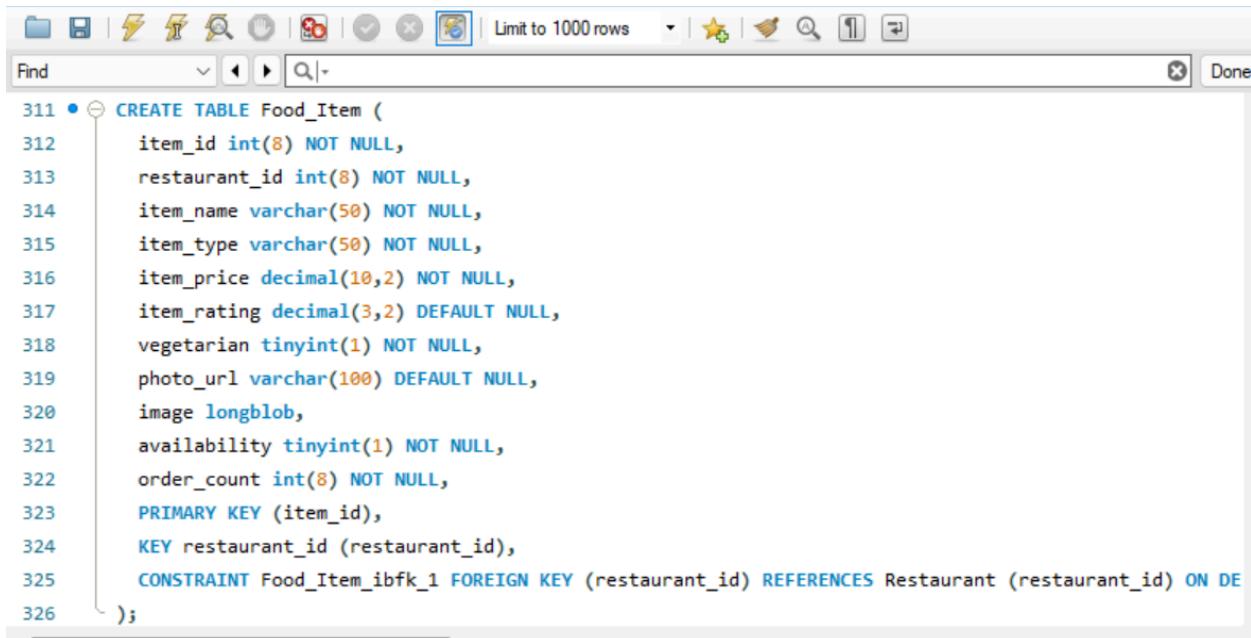
### **Relational algebra:**

$$\text{Customers} \leftarrow \text{Customers} \cup \{1, 'Vasu', 'das', 'rao', '1980 - 04 - 10', 43, \{"email": "Vasu.rao@example.com", "phone": "9676443210"\}, 'password3'\}$$

This expression represents the union of the existing Customers relation with a new tuple containing the specified attribute values, effectively inserting a new record into the Customers table.

### 3.3.5 Insert an image into the MySQL database.

The image's data type must be BLOB for storing images in MySQL. BLOB stands for “Binary Large Object” and represents a database type to store binary data. It stores images, videos, audio, or any other data type that cannot be represented using text. Depending on the size of the image, the data type can be blob or longblob.

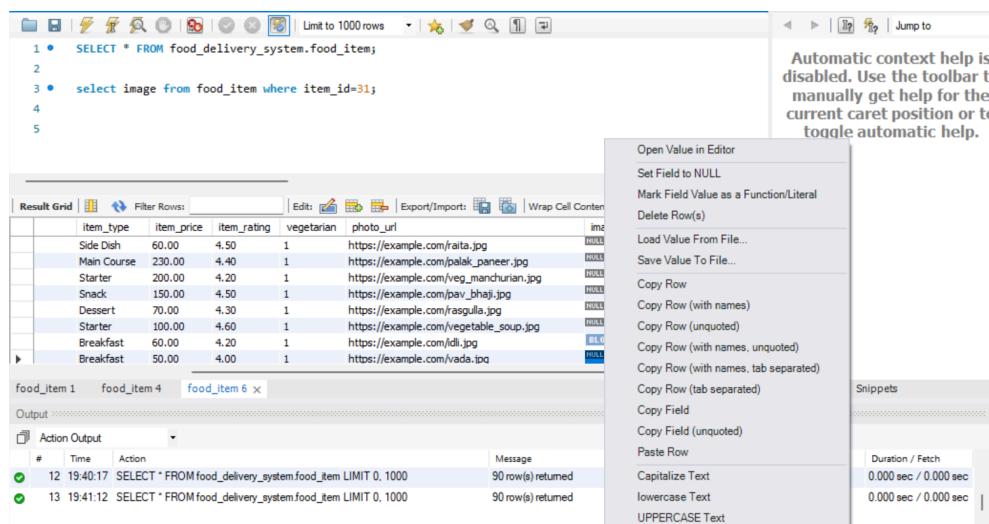


A screenshot of the MySQL Workbench interface. The main window shows the SQL editor with the following code:

```
311 • CREATE TABLE Food_Item (
312     item_id int(8) NOT NULL,
313     restaurant_id int(8) NOT NULL,
314     item_name varchar(50) NOT NULL,
315     item_type varchar(50) NOT NULL,
316     item_price decimal(10,2) NOT NULL,
317     item_rating decimal(3,2) DEFAULT NULL,
318     vegetarian tinyint(1) NOT NULL,
319     photo_url varchar(100) DEFAULT NULL,
320     image longblob,
321     availability tinyint(1) NOT NULL,
322     order_count int(8) NOT NULL,
323     PRIMARY KEY (item_id),
324     KEY restaurant_id (restaurant_id),
325     CONSTRAINT Food_Item_ibfk_1 FOREIGN KEY (restaurant_id) REFERENCES Restaurant (restaurant_id) ON DE
326 );
```

The interface includes a toolbar at the top with various icons for file operations, search, and help. The status bar at the bottom indicates "Limit to 1000 rows".

We have stored the image referring to the [article](#) provided in the assignment. You can also add captions by editing the fields.



A screenshot of the MySQL Workbench interface. The SQL editor contains the following query:

```
1 • SELECT * FROM food_delivery_system.food_item;
2
3 • select image from food_item where item_id=31;
```

The result grid shows the following data:

	item_type	item_price	item_rating	vegetarian	photo_url	image
1	Side Dish	60.00	4.50	1	https://example.com/rata.jpg	BLOB
2	Main Course	230.00	4.40	1	https://example.com/palak_paneer.jpg	BLOB
3	Starter	200.00	4.20	1	https://example.com/veg_manchurian.jpg	BLOB
4	Snack	150.00	4.50	1	https://example.com/pav_bhaji.jpg	BLOB
5	Dessert	70.00	4.30	1	https://example.com/rasgulla.jpg	BLOB
6	Starter	100.00	4.60	1	https://example.com/vegetable_soup.jpg	BLOB
7	Breakfast	60.00	4.20	1	https://example.com/dli.jpg	BLOB
8	Breakfast	50.00	4.00	1	https://example.com/vada.ipq	BLOB

The context menu for the "image" column is open, displaying options such as "Open Value in Editor", "Set Field to NULL", and "Mark Field Value as a Function/Literal". The status bar at the bottom shows "Duration / Fetch: 0.000 sec / 0.000 sec".

Another way to add the image is by using the update command with the image parameter as a link to the image.

The screenshot shows the MySQL Workbench interface. In the top-left pane, there is a SQL editor window containing the following code:

```

1 •  SELECT * FROM food_delivery_system.food_item;
2
3 •  select image from food_item where item_id=31;
4
5 •  update food_item set image = 'https://www.shutterstock.com/image-photo/south-indian-breakfast-food-idli...'.

```

In the top-right pane, a message box displays: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

The bottom half of the interface shows a "Result Grid" displaying a table of food items. The columns are: item\_type, item\_price, item\_rating, vegetarian, photo\_url, image, availability, and order\_c. The data includes various items like Side Dish, Main Course, Starter, Snack, Dessert, etc., with their respective details.

item_type	item_price	item_rating	vegetarian	photo_url	image	availability	order_c
Side Dish	60.00	4.50	1	https://example.com/raita.jpg	NULL	1	35
Main Course	230.00	4.40	1	https://example.com/palak_paneer.jpg	NULL	1	60
Starter	200.00	4.20	1	https://example.com/veg_manchurian.jpg	NULL	1	45
Snack	150.00	4.50	1	https://example.com/pav_bhaji.jpg	NULL	1	70
Dessert	70.00	4.30	1	https://example.com/rasgulla.jpg	NULL	1	40
Starter	100.00	4.60	1	https://example.com/vegetable_soup.jpg	NULL	1	55
Breakfast	60.00	4.20	1	https://example.com/idli.jpg	blob	1	30
Breakfast	50.00	4.00	1	https://example.com/vada.jpg	blob	1	25

Below the grid, there are tabs for "food\_item 1", "food\_item 4", and "food\_item 10 x". On the right, there are buttons for "Apply" and "Revert".

The bottom section is labeled "Output" and shows the execution log:

Action	Time	Action	Message	Duration / Fetch
update food_item set image = 'https://www.shutterstock.com/image-photo/south-indian-breakfast-food-idli...'	18 19:48:18		0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0	0.000 sec
SELECT * FROM food_delivery_system.food_item LIMIT 0, 1000	19 19:48:31		90 row(s) returned	0.000 sec / 0.000 sec

### 3.3.6 Finding statistical information about food items:

It includes average, minimum, and maximum prices, along with the corresponding restaurants for each extreme price.

#### *Relational algebra:*

$$\begin{aligned}
 Item\_statistics\_with\_joints &\leftarrow \rho_{item\_name, avg\_price, min\_price, max\_price, min\_price\_item\_id, min\_price\_restaurant, \\
 &\quad 'max\_price\_item\_id, max\_price\_restaurant} (\sigma_{(Item\_statistics.item\_name = Food\_Item.item\_name \text{ AND} \\
 &\quad 'Item\_statistics.min\_price = Food\_Item.item\_price)} (Item\_statistics \bowtie_{Food\_Item} \bowtie_{Restaurant} \\
 &\quad \bowtie_{\rho_{(min\_price\_item\_id, restaurant\_id, item\_name, min\_price\_restaurant}} (Food\_Item) \\
 &\quad \bowtie_{\rho_{max\_price\_item\_id, restaurant\_id, item\_name, max\_price\_restaurant}} (Food\_Item))
 \end{aligned}$$

```

15 ● CREATE TABLE Item_statistics AS
16     SELECT distinct
17         item_name,
18             AVG(item_price) OVER(PARTITION BY item_name) AS avg_price,
19             MIN(item_price) OVER(PARTITION BY item_name) AS min_price,
20             MAX(item_price) OVER(PARTITION BY item_name) AS max_price
21     FROM
22         Food_Item;
23
24 ● SELECT distinct
25     i.item_name,
26     i.avg_price,
27     i.min_price,
28     i.max_price,
29     r_min.restaurant_name AS min_price_restaurant,
30     r_max.restaurant_name AS max_price_restaurant
31     FROM
32         Item_statistics i
33     JOIN
34         Food_Item f_min ON i.item_name = f_min.item_name AND i.min_price = f_min.item_price
35     JOIN
36         Restaurant r_min ON f_min.restaurant_id = r_min.restaurant_id
37     JOIN
38         Food_Item f_max ON i.item_name = f_max.item_name AND i.max_price = f_max.item_price
39     JOIN
40         Restaurant r_max ON f_max.restaurant_id = r_max.restaurant_id;

```

Result Grid | Filter Rows:  Export: Wrap Cell Content:

	item_name	avg_price	min_price	max_price	min_price_restaurant	max_price_restaurant
▶	Vegetable Biryani	226.000000	210.00	240.00	Southern Spice	Spice Garden
	Aloo Paratha	118.000000	100.00	130.00	The Mughal Feast	Spice Garden
	Dal Makhani	201.000000	180.00	220.00	Spice Garden	Coastal Curry House
	Chicken Tikka	272.000000	250.00	300.00	Spice Garden	Rajasthani Delight
	Vegetable Biryani	226.000000	210.00	240.00	Southern Spice	Rajasthani Delight

3.3.7 Based on the average price of the item we classified the items into High and medium categories.

```

1 ● START TRANSACTION;
2 ● CREATE TABLE Item_Statistics AS
3     SELECT
4         item_name,
5             MAX(item_price) AS max_price,
6             MIN(item_price) AS min_price,
7             AVG(item_price) AS avg_price,
8             CASE
9                 WHEN AVG(item_price) > 50 THEN 'High'
10                WHEN AVG(item_price) > 25 THEN 'Medium'
11                ELSE 'Low'
12            END AS price_category
13     FROM
14         Food_Item
15     GROUP BY
16         item_name;
17 ●     SELECT * FROM Item_Statistics;
18
19 -- Assuming everything is successful Commit command is used
20 ● COMMIT;
21
22

```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

## Work Distribution

Team Member	Contribution
1. Birudugadda Srivibhav(G1)	Generation of random data, Putting up the necessary constraints. Making up the final report. Implementing indexing, user-defined data types, and table extensions, Generation of random data
2. Srivathsa Vamsi(G1)	Generation of random data, Putting up the necessary constraints. Making up the final report. Implementing indexing, user-defined data types, and table extensions, Generation of random data
3. Sriman Reddy(G1)	Generation of random data, Putting up the necessary constraints. Making up the final report. Implementing indexing, user-defined data types, and table extensions, Generation of random data
4. Nikhilesh Myanapuri(G1)	Generation of random data, Putting up the necessary constraints. Making up the final report. Implementing indexing, user-defined data types, and table extensions, Generation of random data
5. Haikoo Khandor(G2)	Implementing indexing, user-defined data types, and table extensions. Creating a user, granting different permission on tables and views
6. Madhav Kanda(G2)	Situation where referential integrity is violated, updates in the table and how we can solve such problems
7. Pramod Limbore(G2)	Write up five operations and their corresponding tuple relational calculus/ relational algebra and SQL queries, Adding up the photo into the database.
8. Shriyash Mandavekar(G2)	Write up five operations and their corresponding tuple relational calculus/ relational algebra and SQL queries, Adding up the photo into the database.

## Acknowledgements

We want to express our thanks to Prof Mayank Singh and TA Ritesh Patidar for their support in the assignment.

## References

- <https://forums.mysql.com/read.php?20,17671,27914>
- [https://www.geeksforgeeks.org/cardinality-in-dbms/?ref=ml\\_lbp](https://www.geeksforgeeks.org/cardinality-in-dbms/?ref=ml_lbp)
- <https://www.geeksforgeeks.org/design-restaurant-management-system-system-design/#database-design>
- <https://www.gleek.io/blog/relational-schema>
- Classroom Slides