

## HW Murach Ch 16 Triggers Project

**GROUP 21: Ara Chung, Dalton, Pooja , and Yogesh**

### Question :

Add two triggers of your choice (one has to be the audit table example but you can choose if you use insert, update or delete) to the Campus Eats database system. Test your triggers.

### Ans:

**Added two triggers in Campus eats database**

- 1. Trigger on order table. Whenever an update occurs on the order table, trigger creates an entry in order\_audit table.**
- 2. Trigger on ratings table. Whenever rating is deleted from the ratings table, row in driver\_ratings, restaurant\_rating is deleted with the same rating\_id.**

```
use campus_eats_fall2020;
DROP TABLE IF EXISTS order_audit;
CREATE TABLE order_audit
(
  order_id      INT NOT NULL,
  delivery_id   INT NOT NULL,
  total_price   float NOT NULL,
  action_type   VARCHAR(50) NOT NULL,
  action_date   DATETIME NOT NULL
);

Drop Trigger IF EXISTS trigger_Update_after_Order;
Drop Trigger IF EXISTS trigger_delete_before_ratings;

DELIMITER $$
-- Trigger 1 on update on order details.
-- Insert in order_audit table whenever an update occurs on order table
CREATE TRIGGER trigger_update_after_Order
  AFTER Update ON `order`
  FOR EACH ROW
BEGIN
  INSERT INTO order_audit VALUES
    (NEW.order_id, NEW.delivery_id, NEW.total_price, 'UPDATE', NOW());
END $$
```

```

-- Trigger 2 on ratings table.
-- On Delete of row from ratings table deletes row with same rating_id from driver_ratings,
restaurant_ratings
CREATE TRIGGER trigger_delete_before_ratings
  BEFORE delete ON `ratings`
  FOR EACH ROW
BEGIN
  Delete from driver_ratings where rating_id = OLD.rating_id;
  Delete from restaurant_ratings where rating_id = OLD.rating_id;
END $$
DELIMITER ;

-- Test the trigger on order_audit table
Update `order` set total_price=350 where order_id = 4;
select * from order_audit;

-- Test the trigger on ratings table
select d.rating_id, ontime_rating, courteous_rating, food_rating, price_rating
from driver_ratings d join restaurant_ratings r on d.rating_id = r.rating_id order by d.rating_id;
delete from ratings where rating_id = 2;
select d.rating_id, ontime_rating, courteous_rating, food_rating, price_rating
from driver_ratings d join restaurant_ratings r on d.rating_id = r.rating_id order by d.rating_id;

```

### **Screenshots showing successful completion of triggers:**

1. Screenshot with trigger1

Trigger SQL File 6\*

```

1 use campus_eats_fall2020;
2 DROP TABLE IF EXISTS order_audit;
3 CREATE TABLE order_audit
4 (
5     order_id          INT NOT NULL,
6     delivery_id       INT NOT NULL,
7     total_price       float NOT NULL,
8     action_type       VARCHAR(50) NOT NULL,
9     action_date       DATETIME NOT NULL,
10 );
11 DROP TRIGGER IF EXISTS trigger_update_after_order;
12 DROP TRIGGER IF EXISTS trigger_delete_before_ratings;
13 DELIMITER $$
14 -- Trigger 1 on update on order details.
15 -- Insert in order_audit table whenever an update occurs on order table
16 CREATE TRIGGER trigger_update_after_order
17 AFTER Update ON `order`
18 FOR EACH ROW
19 BEGIN
20     INSERT INTO order_audit VALUES
21     (NEW.order_id, NEW.delivery_id, NEW.total_price, 'UPDATE', NOW());
22 END $$
23

```

Result Grid

order_id	delivery_id	total_price	action_type	action_date
4	4	350	UPDATE	2021-04-11 14:36:26

order\_audit 4 x

Output

#	Time	Action	Message	Duration / Fetch
14	14:36:26	use campus_eats_fall2020	0 row(s) affected	0.000 sec
15	14:36:26	DROP TABLE IF EXISTS order_audit	0 row(s) affected	0.031 sec
16	14:36:26	CREATE TABLE order_audit ( order_id INT NOT NULL, delivery_id INT NOT NULL, total_price float NOT NULL, action_type ...	0 row(s) affected	0.031 sec
17	14:36:26	Drop Trigger IF EXISTS trigger_update_after_order	0 row(s) affected	0.031 sec
18	14:36:26	Drop Trigger IF EXISTS trigger_delete_before_ratings	0 row(s) affected	0.016 sec
19	14:36:26	-- Trigger 1 on update on order details -- Insert in order_audit table whenever an update occurs on order table CREATE TRIGGER trigger_update...	0 row(s) affected	0.015 sec
20	14:36:26	CREATE TRIGGER trigger_delete_before_ratings BEFORE delete ON `ratings` FOR EACH ROW BEGIN Delete from driver_ratings where...	0 row(s) affected	0.016 sec

## 2. With trigger 2 and Output of order\_audit table

Trigger SQL File 6\*

```

22 END $$
23
24 -- Trigger 2 on ratings table.
25 -- On Delete of row from ratings table deletes row with same rating_id from driver_ratings, restaurant_ratings
26 CREATE TRIGGER trigger_delete_before_ratings
27 BEFORE delete ON `ratings`
28 FOR EACH ROW
29 BEGIN
30     Delete from driver_ratings where rating_id = OLD.rating_id;
31     Delete from restaurant_ratings where rating_id = OLD.rating_id;
32 END $$
33 DELIMITER ;
34
35 -- Test the trigger on order_audit table
36 Update `order` set total_price=350 where order_id = 4;
37 select * from order_audit;
38
39 -- Test the trigger on ratings table
40 select d.rating_id, online_rating, courteous_rating, food_rating, price_rating
41 from driver_ratings d join restaurant_ratings r on d.rating_id = r.rating_id order by d.rating_id;
42 delete from ratings where rating_id = 2;
43 select d.rating_id, online_rating, courteous_rating, food_rating, price_rating
44 from driver_ratings d join restaurant_ratings r on d.rating_id = r.rating_id order by d.rating_id;

```

Result Grid

order_id	delivery_id	total_price	action_type	action_date
4	4	350	UPDATE	2021-04-11 14:36:26

order\_audit 4 x

Output

#	Time	Action	Message	Duration / Fetch
14	14:36:26	use campus_eats_fall2020	0 row(s) affected	0.000 sec
15	14:36:26	DROP TABLE IF EXISTS order_audit	0 row(s) affected	0.031 sec
16	14:36:26	CREATE TABLE order_audit ( order_id INT NOT NULL, delivery_id INT NOT NULL, total_price float NOT NULL, action_type ...	0 row(s) affected	0.031 sec
17	14:36:26	Drop Trigger IF EXISTS trigger_update_after_order	0 row(s) affected	0.031 sec
18	14:36:26	Drop Trigger IF EXISTS trigger_delete_before_ratings	0 row(s) affected	0.016 sec
19	14:36:26	-- Trigger 1 on update on order details -- Insert in order_audit table whenever an update occurs on order table CREATE TRIGGER trigger_update...	0 row(s) affected	0.015 sec
20	14:36:26	CREATE TRIGGER trigger_delete_before_ratings BEFORE delete ON `ratings` FOR EACH ROW BEGIN Delete from driver_ratings where...	0 row(s) affected	0.016 sec

## 3. Output before delete on ratings table

Triggers

```

31 Delete from restaurant_ratings where rating_id = OLD.rating_id;
32 END $$
33 DELIMITER ;
34
35 -- Test the trigger on order_audit table
36 Update 'order' set total_price=350 where order_id = 4;
37 select * from order_audit;
38
39 -- Test the trigger on ratings table
40 select d.rating_id, ontime_rating, courteous_rating, food_rating, price_rating
41 from driver_ratings d join restaurant_ratings r on d.rating_id = r.rating_id order by d.rating_id;
42 delete from ratings where rating_id = 2;
43 select d.rating_id, ontime_rating, courteous_rating, food_rating, price_rating
44 from driver_ratings d join restaurant_ratings r on d.rating_id = r.rating_id order by d.rating_id;

```

Result Grid

rating_id	ontime_rating	courteous_rating	food_rating	price_rating
2	9	1	2	10
3	4	6	10	7
4	5	7	1	10
5	3	10	10	10
6	9	10	3	10
7	4	9	7	9
8	5	8	1	9
9	2	4	2	5
10	9	7	5	4
11	7	8	3	3
12	2	8	1	3
13	10	6	6	1
14	6	5	8	3
15	8	6	7	5
16	5	4	5	10
17	1	3	5	1

order\_audit 1 Result 2 x Result 3

Output

#	Time	Action	Message	Duration / Fetch
8	14:02:57	-- Test the trigger on order_audit table Update 'order' set total_price=350 where order_id = 4	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.015 sec
9	14:02:57	select * from order_audit LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
10	14:02:57	select d.rating_id, ontime_rating, courteous_rating, food_rating, price_rating from driver_ratings d join restaurant_ratings r on d.rating_id = r.rating_id	39 row(s) returned	0.000 sec / 0.000 sec
11	14:02:57	delete from ratings where rating_id = 2	1 row(s) affected	0.015 sec
12	14:02:57	select d.rating_id, ontime_rating, courteous_rating, food_rating, price_rating from driver_ratings d join restaurant_ratings r on d.rating_id = r.rating_id	38 row(s) returned	0.000 sec / 0.000 sec
13	14:05:16	SELECT * FROM campus_eats_fal2020.order_audit LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

#### 4. Output after delete on ratings table

Triggers

```

31 Delete from restaurant_ratings where rating_id = OLD.rating_id;
32 END $$
33 DELIMITER ;
34
35 -- Test the trigger on order_audit table
36 Update 'order' set total_price=350 where order_id = 4;
37 select * from order_audit;
38
39 -- Test the trigger on ratings table
40 select d.rating_id, ontime_rating, courteous_rating, food_rating, price_rating
41 from driver_ratings d join restaurant_ratings r on d.rating_id = r.rating_id order by d.rating_id;
42 delete from ratings where rating_id = 2;
43 select d.rating_id, ontime_rating, courteous_rating, food_rating, price_rating
44 from driver_ratings d join restaurant_ratings r on d.rating_id = r.rating_id order by d.rating_id;

```

Result Grid

rating_id	ontime_rating	courteous_rating	food_rating	price_rating
3	4	6	10	7
4	5	7	1	10
5	3	10	10	10
6	9	10	3	10
7	4	9	7	9
8	5	8	1	9
9	2	4	2	5
10	9	7	5	4
11	7	8	3	3
12	2	8	1	3
13	10	6	6	1
14	6	5	8	3
15	8	6	7	5
16	5	4	5	10
17	1	3	5	1
18	10	3	2	8

order\_audit 1 Result 2 Result 3 x

Output

#	Time	Action	Message	Duration / Fetch
8	14:02:57	-- Test the trigger on order_audit table Update 'order' set total_price=350 where order_id = 4	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.015 sec
9	14:02:57	select * from order_audit LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
10	14:02:57	select d.rating_id, ontime_rating, courteous_rating, food_rating, price_rating from driver_ratings d join restaurant_ratings r on d.rating_id = r.rating_id	39 row(s) returned	0.000 sec / 0.000 sec
11	14:02:57	delete from ratings where rating_id = 2	1 row(s) affected	0.015 sec
12	14:02:57	select d.rating_id, ontime_rating, courteous_rating, food_rating, price_rating from driver_ratings d join restaurant_ratings r on d.rating_id = r.rating_id	38 row(s) returned	0.000 sec / 0.000 sec
13	14:05:16	SELECT * FROM campus_eats_fal2020.order_audit LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec