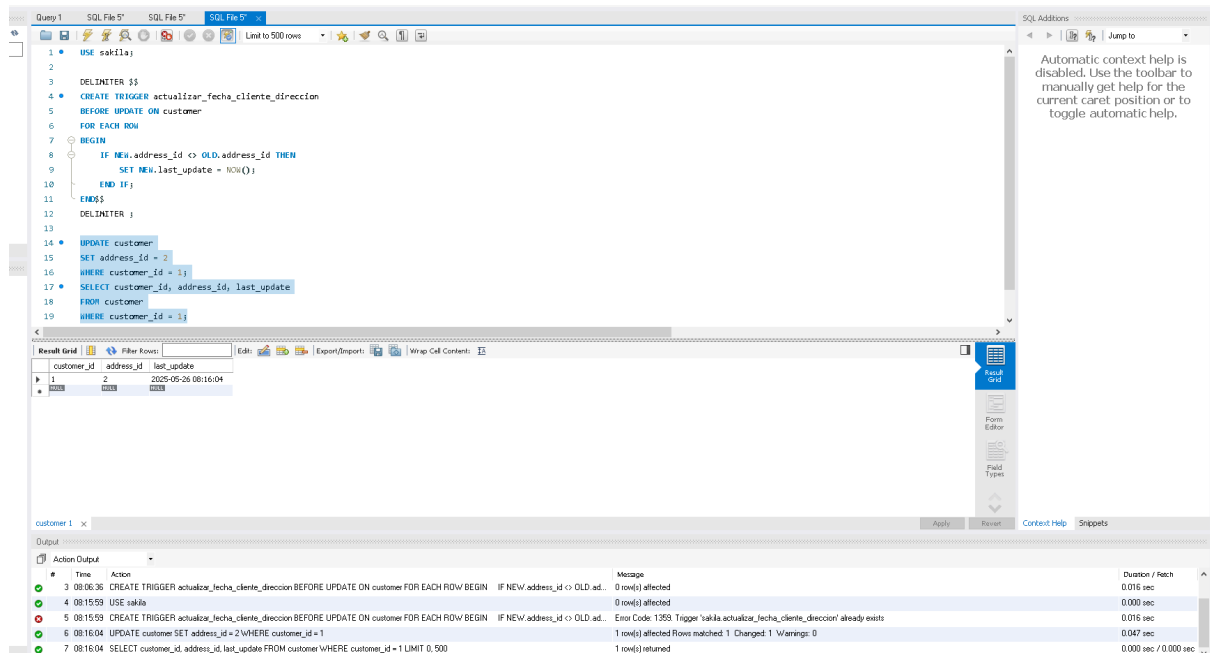


ACTIVIDAD 30

1. Actualizar last_update de customer cuando se actualice su dirección

Crea un trigger que actualice el campo last_update de la tabla customer cuando se modifique su address_id.



The screenshot shows the MySQL Workbench interface. The top pane displays a SQL query to create a trigger named 'actualizar_fecha_cliente_direccion' that updates the 'last_update' field of the 'customer' table whenever the 'address_id' is updated. The bottom pane shows the 'Results' grid with one row of data for customer_id 1, address_id 2, and last_update '2025-05-26 08:16:04'. The 'Output' pane at the bottom shows the execution log, including the creation of the trigger and the successful update of the customer record.

```
1 USE sakila;
2
3 DELIMITER $$
4 CREATE TRIGGER actualizar_fecha_cliente_direccion
5 BEFORE UPDATE ON customer
6 FOR EACH ROW
7 BEGIN
8     IF NEW.address_id <> OLD.address_id THEN
9         SET NEW.last_update = NOW();
10     END IF;
11 END $$
12 DELIMITER ;
13
14 UPDATE customer
15 SET address_id = 2
16 WHERE customer_id = 1;
17 SELECT customer_id, address_id, last_update
18 FROM customer
19 WHERE customer_id = 1;
```

customer_id	address_id	last_update
1	2	2025-05-26 08:16:04

Output:

#	Time	Action	Message	Duration / Fetch
3	08:06:36	CREATE TRIGGER actualizar_fecha_cliente_direccion BEFORE UPDATE ON customer FOR EACH ROW BEGIN IF NEW.address_id <> OLD.ad...	0 row(s) affected	0.016 sec
4	08:15:59	USE sakila	0 row(s) affected	0.000 sec
5	08:15:59	CREATE TRIGGER actualizar_fecha_cliente_direccion BEFORE UPDATE ON customer FOR EACH ROW BEGIN IF NEW.address_id <> OLD.ad...	Error Code: 1353. Trigger 'sakila.actualizar_fecha_cliente_direccion' already exists	0.016 sec
6	08:16:04	UPDATE customer SET address_id = 2 WHERE customer_id = 1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.047 sec
7	08:16:04	SELECT customer_id, address_id, last_update FROM customer WHERE customer_id = 1 LIMIT 0, 500	1 row(s) returned	0.000 sec / 0.000 sec

2. Actualizar last_update de rental al insertar un payment

Cuando se inserte un pago, actualiza el campo last_update del rental asociado.

The screenshot shows a SQL query window with the following code:

```

19 WHERE customer_id = 1;
20
21
22 DELIMITER $$
23 CREATE TRIGGER actualizar_fecha_renta_por_pago
24 AFTER INSERT ON payment
25 FOR EACH ROW
26 BEGIN
27     UPDATE rental
28     SET last_update = NOW()
29     WHERE rental_id = NEW.rental_id;
30 END$$
31 DELIMITER ;
32
33 INSERT INTO payment (customer_id, staff_id, rental_id, amount, payment_date)
34 VALUES (1, 1, 1, 4.99, NOW());
35 SELECT rental_id, last_update
36 FROM rental
37 WHERE rental_id = 1;

```

The results grid shows the following data:

rental_id	last_update
1	2025-05-26 08:18:36

The output window shows the following messages:

#	Time	Action	Message	Duration / Fetch
6	08:16:04	UPDATE customer SET address_id=2 WHERE customer_id=1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.047 sec
7	08:16:04	SELECT customer_id, address_id, last_update FROM customer WHERE customer_id = 1 LIMIT 0, 500	1 row(s) returned	0.000 sec / 0.000 sec
8	08:18:36	CREATE TRIGGER actualizar_fecha_renta_por_pago AFTER INSERT ON payment FOR EACH ROW BEGIN UPDATE rental SET last_update = NOW() WHERE rental_id = NEW.rental_id; END	0 row(s) affected	0.000 sec
9	08:18:36	INSERT INTO payment (customer_id, staff_id, rental_id, amount, payment_date) VALUES (1, 1, 1, 4.99, NOW())	1 row(s) affected	0.032 sec
10	08:18:36	SELECT rental_id, last_update FROM rental WHERE rental_id = 1 LIMIT 0, 500	1 row(s) returned	0.000 sec / 0.000 sec

3. Actualizar last_update de la tabla inventory al actualizar film_id

Quando se actualice el film_id de un inventario, actualiza su campo last_update.

The screenshot shows a SQL query window with the following code:

```

40 DELIMITER $$
41 CREATE TRIGGER actualizar_fecha_inventario_pelicula
42 BEFORE UPDATE ON inventory
43 FOR EACH ROW
44 BEGIN
45     IF NEW.film_id <> OLD.film_id THEN
46         SET NEW.last_update = NOW();
47     END IF;
48 END$$
49 DELIMITER ;
50
51 UPDATE inventory
52 SET film_id = 2
53 WHERE inventory_id = 1;
54 SELECT inventory_id, film_id, last_update
55 FROM inventory
56 WHERE inventory_id = 1;
57
58

```

The results grid shows the following data:

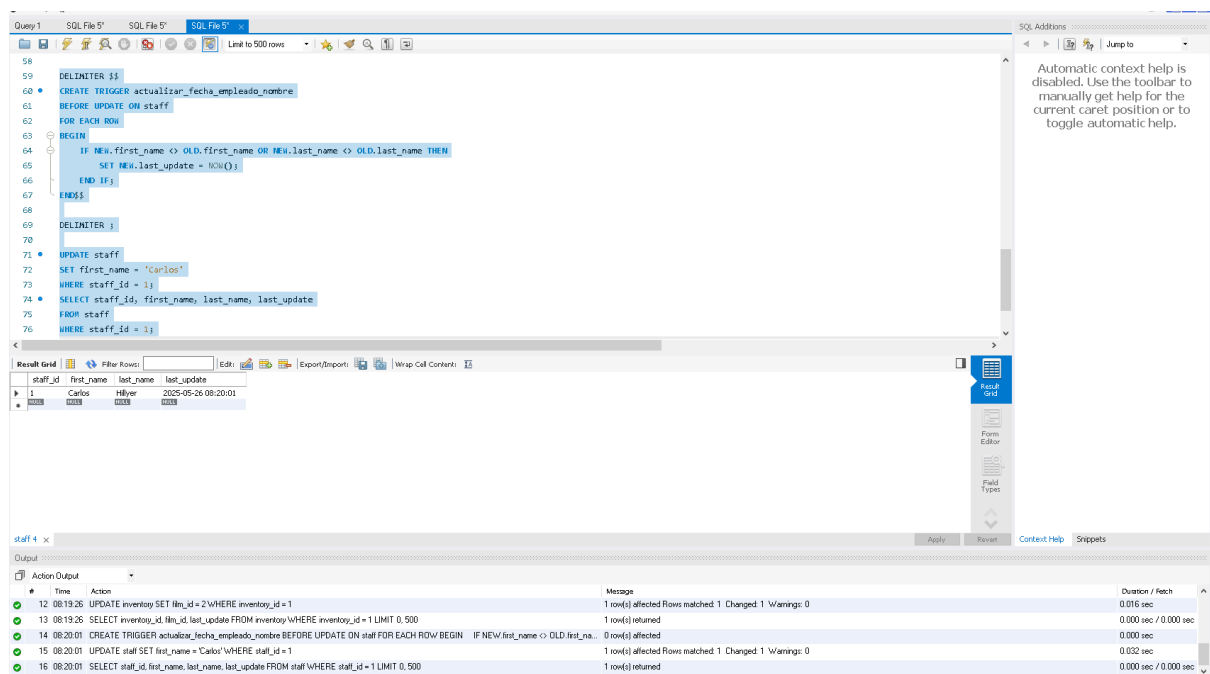
inventory_id	film_id	last_update
1	2	2025-05-26 08:19:26

The output window shows the following messages:

#	Time	Action	Message	Duration / Fetch
9	08:18:36	INSERT INTO payment (customer_id, staff_id, rental_id, amount, payment_date) VALUES (1, 1, 1, 4.99, NOW())	1 row(s) affected	0.032 sec
10	08:18:36	SELECT rental_id, last_update FROM rental WHERE rental_id = 1 LIMIT 0, 500	1 row(s) returned	0.000 sec / 0.000 sec
11	08:19:26	CREATE TRIGGER actualizar_fecha_inventario_pelicula BEFORE UPDATE ON inventory FOR EACH ROW BEGIN IF NEW.film_id <> OLD.film_id THEN SET NEW.last_update = NOW(); END IF; END	0 row(s) affected	0.015 sec
12	08:19:26	UPDATE inventory SET film_id=2 WHERE inventory_id=1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.016 sec
13	08:19:26	SELECT inventory_id, film_id, last_update FROM inventory WHERE inventory_id = 1 LIMIT 0, 500	1 row(s) returned	0.000 sec / 0.000 sec

4. Actualizar last_update de la tabla staff cuando se modifique su nombre

Quando se actualice el first_name o last_name de un empleado (staff), también debe actualizarse el campo last_update.



USE sakila;

DELIMITER \$\$

CREATE TRIGGER actualizar_fecha_cliente_direccion

BEFORE UPDATE ON customer

FOR EACH ROW

BEGIN

IF NEW.address_id <> OLD.address_id THEN

SET NEW.last_update = NOW();

END IF;

END\$\$

DELIMITER ;

UPDATE customer

SET address_id = 2

WHERE customer_id = 1;

SELECT customer_id, address_id, last_update

FROM customer

WHERE customer_id = 1;

```
DELIMITER $$
CREATE TRIGGER actualizar_fecha_renta_por_pago
AFTER INSERT ON payment
FOR EACH ROW
BEGIN
    UPDATE rental
    SET last_update = NOW()
    WHERE rental_id = NEW.rental_id;
END$$
DELIMITER ;
```

```
INSERT INTO payment (customer_id, staff_id, rental_id, amount, payment_date)
VALUES (1, 1, 1, 4.99, NOW());
SELECT rental_id, last_update
FROM rental
WHERE rental_id = 1;
```

```
DELIMITER $$
CREATE TRIGGER actualizar_fecha_inventario_pelicula
BEFORE UPDATE ON inventory
FOR EACH ROW
BEGIN
    IF NEW.film_id <> OLD.film_id THEN
        SET NEW.last_update = NOW();
    END IF;
END$$
DELIMITER ;
```

```
UPDATE inventory
SET film_id = 2
WHERE inventory_id = 1;
SELECT inventory_id, film_id, last_update
FROM inventory
```

```
WHERE inventory_id = 1;
```

```
DELIMITER $$
```

```
CREATE TRIGGER actualizar_fecha_empleado_nombre
```

```
BEFORE UPDATE ON staff
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    IF NEW.first_name <> OLD.first_name OR NEW.last_name <> OLD.last_name THEN
```

```
        SET NEW.last_update = NOW();
```

```
    END IF;
```

```
END$$
```

```
DELIMITER ;
```

```
UPDATE staff
```

```
SET first_name = 'Carlos'
```

```
WHERE staff_id = 1;
```

```
SELECT staff_id, first_name, last_name, last_update
```

```
FROM staff
```

```
WHERE staff_id = 1;
```

USANDO MIS BASE DE DATOS:

ESCUELA:

poner un nombre predeterminado.

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

```
USE escuela;

DELIMITER $$
CREATE TRIGGER antes_de_insertar_estudiante_nombre
BEFORE INSERT ON estudiante
FOR EACH ROW
BEGIN
    IF NEW.nombre = '' THEN
        SET NEW.nombre = 'Nombre Predeterminado';
    END IF;
END$$
DELIMITER ;

INSERT INTO estudiante (apellido, fecha_nacimiento, direccion, telefono, email, grado, fecha_inscripcion)
VALUES ('Gómez', '2010-01-01', 'Calle Falsa 123', '1234567890', 'ana.gomez@example.com', 'Primer', NOW());

SELECT id_estudiante, nombre, apellido
FROM estudiante
WHERE apellido = 'Gómez';
```

id_estudiante	nombre	apellido
0	Gómez	Gómez

estudiante 6 x

Action	Time	Message	Duration / Fech
INSERT INTO estudiante (apellido, fecha_nacimiento, direccion, telefono, email, grado, fecha_inscripcion) VALUES ('Gómez', '2010-01-01', 'Calle Fals...	08:29:27	1 row(s) affected, 2 warning(s): 1285 Data truncated for column 'fecha_inscripcion' at row 1 1364 Field 'id_estudiante' doesn't have a default value	0.015 sec
SELECT id_estudiante, nombre, apellido FROM estudiante WHERE apellido = 'Gómez' LIMIT 0.500	08:29:27	1 row(s) returned	0.000 sec / 0.000 sec
CREATE TRIGGER antes_de_insertar_estudiante_nombre BEFORE INSERT ON estudiante FOR EACH ROW BEGIN IF NEW.nombre = '' THEN ...	08:29:53	Error Code: 1363. Trigger 'antes_de_insertar_estudiante_nombre' already exists	0.016 sec
INSERT INTO estudiante (apellido, fecha_nacimiento, direccion, telefono, email, grado, fecha_inscripcion) VALUES ('Gómez', '2010-01-01', 'Calle Fals...	08:29:57	Error Code: 1062. Duplicate entry '0' for key 'PRIMARY'	0.000 sec
SELECT id_estudiante, nombre, apellido FROM estudiante WHERE apellido = 'Gómez' LIMIT 0.500	08:30:02	1 row(s) returned	0.000 sec / 0.000 sec

LIGA_FUTBOL:

Colocar equipo nuevo en caso de que no se insert el nombre

Query 1 SQL File 5* SQL File 5* SQL File 5* x

Limit to 500 rows

```

102
103 DELIMITER $$
104 • CREATE TRIGGER despies_colocar_equipo
105 BEFORE INSERT ON equipo
106 FOR EACH ROW
107 BEGIN
108     IF NEW.nombre = '' THEN
109         SET NEW.nombre = "Equipo Nuevo";
110     END IF;
111 END$$
112 DELIMITER ;
113
114 • INSERT INTO equipo (ciudad, fundacion) VALUES ( 'Madrid', '1903-04-26');
115
116 • SELECT * FROM equipo WHERE ciudad = "Madrid";
117
118

```

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

Result Grid

id	nombre	ciudad	fundacion
1	Real Madrid	Madrid	1902-03-06
3	Atlético de Madrid	Madrid	1903-04-26
20	Rayo Vallecano	Madrid	1924-05-29
21	Real Madrid Castilla	Madrid	1972-11-05
41	Equipo Nuevo	Madrid	1903-04-26
NULL	NULL	NULL	NULL

Form Editor

Field Types

equipo 5 x Apply Revert Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
17	07:25:23	CREATE TRIGGER despies_colocar_equipo BEFORE INSERT ON equipo FOR ...	Error Code: 1146. Table 'escuela.equipo' doesn't exist	0.000 sec
18	07:25:43	USE liga_futbol	0 row(s) affected	0.000 sec
19	07:25:43	CREATE TRIGGER despies_colocar_equipo BEFORE INSERT ON equipo FOR ...	0 row(s) affected	0.000 sec
20	07:25:47	INSERT INTO equipo (ciudad, fundacion) VALUES ('Madrid', '1903-04-26')	1 row(s) affected	0.000 sec
21	07:25:47	SELECT * FROM equipo WHERE ciudad = "Madrid" LIMIT 0, 500	5 row(s) returned	0.000 sec / 0.000 sec

TIENDA DEPARTAMENTAL:

Antes de cambiar de telefono, verificar que cumple con el formato XXX-XXX-XXXX

Query 1 SQL File 5* SQL File 5* SQL File 5* empleado

Limit to 500 rows

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

```
119
120
121 DELIMITER $$
122 • CREATE TRIGGER before_update_empleado
123 BEFORE UPDATE ON empleado
124 FOR EACH ROW
125 BEGIN
126 IF NEW.telefono NOT REGEXP '^[0-9]{3}-[0-9]{3}-[0-9]{4}$' THEN
127 SET NEW.telefono = OLD.telefono;
128 END IF;
129 END$$
130 DELIMITER ;
131
132 • UPDATE empleado SET telefono = '449-210-7183' WHERE nombre_empleado = "Juan Pérez";
133
134 • SELECT * FROM empleado WHERE nombre_empleado = 'Juan Pérez';
135
```

Result Grid

	id_empleado	nombre_empleado	cargo	telefono	email
▶	1	Juan Pérez	Vendedor	449-210-7183	juanperez@tienda.com
•	NULL	NULL	NULL	NULL	NULL

empleado 11 x Apply Revert Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 36	07:36:41	SELECT * FROM empleado WHERE nombre_empleado = 'Juan Perez' LIMIT 0, 500	1 row(s) returned	0.000 sec / 0.000 sec
✓ 37	07:37:14	UPDATE empleado SET telefono = 449-210-7183 WHERE nombre_empleado = '...	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0	0.000 sec
✓ 38	07:37:17	SELECT * FROM empleado WHERE nombre_empleado = 'Juan Pérez' LIMIT 0, 500	1 row(s) returned	0.000 sec / 0.000 sec
✓ 39	07:38:02	UPDATE empleado SET telefono = '449-210-7183' WHERE nombre_empleado = ...	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.016 sec
✓ 40	07:38:02	SELECT * FROM empleado WHERE nombre_empleado = 'Juan Pérez' LIMIT 0, 500	1 row(s) returned	0.000 sec / 0.000 sec

VIDEOJUEGOS:

Colocar como recompensa por default espada de diamante en caso de actualizar a null la recompensa

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

```
137 • USE videojuegos;
138
139 DELIMITER $$
140 • CREATE TRIGGER despues_actualizar_mision
141 BEFORE UPDATE ON mision
142 FOR EACH ROW
143 BEGIN
144     IF NEW.recompensa IS NULL THEN
145         SET NEW.recompensa = OLD.recompensa;
146     END IF;
147 END$$
148 DELIMITER ;
149
150
151 • UPDATE mision SET recompensa = NULL WHERE id_mision = 1;
152
153 • SELECT * FROM mision WHERE id_mision = 1;
```

id_mision	nombre_mision	descripcion	recompensa
1	Rescatar al Rey	Salvar al rey del castillo enemigo	Espada de diamante

mission 15 x Apply Revert Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
49	07:45:59	SELECT * FROM mision WHERE id_mision = 1 LIMIT 0, 500	1 row(s) returned	0.000 sec / 0.000 sec
50	07:46:11	UPDATE mision SET recompensa = 'Espada de diamante' WHERE id_mision = 1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
51	07:46:11	SELECT * FROM mision WHERE id_mision = 1 LIMIT 0, 500	1 row(s) returned	0.000 sec / 0.000 sec
52	07:46:22	UPDATE mision SET recompensa = NULL WHERE id_mision = 1	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0	0.000 sec
53	07:46:22	SELECT * FROM mision WHERE id_mision = 1 LIMIT 0, 500	1 row(s) returned	0.000 sec / 0.000 sec