**Institute of Computer Technology**

**Ganpat University**

**(2CSE301) DATABASE MANAGEMENT SYSTEM**

**Practical 11 MySQL Triggers**

1. Create a trigger that converts each newly inserted language name into uppercase

/\*1\*/

DELIMITER //

CREATE TRIGGER uppercase

BEFORE INSERT ON language

FOR EACH ROW

BEGIN

SET NEW.name = UPPER(NEW.name);

END;

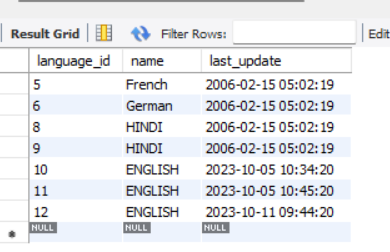
//

DELIMITER ;

INSERT INTO language (name) VALUES ('english');

select \*from language;

-- show triggers like 'uppercase';



1. Create a trigger that displays a message **“The record has been inserted successfully”** whenever a record is inserted in the language table.

/\*2\*/

CREATE TABLE insertion\_messages (

id INT AUTO\_INCREMENT PRIMARY KEY,

message VARCHAR(255),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

DELIMITER //

CREATE TRIGGER display

AFTER INSERT ON language

FOR EACH ROW

BEGIN

INSERT INTO insertion\_messages (message) VALUES ('The record has been inserted successfully');

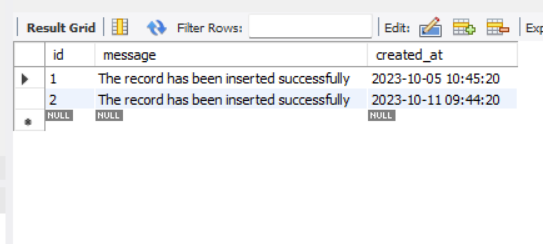
END;

//

DELIMITER ;

INSERT INTO language (name) VALUES ('english');

SELECT \* FROM insertion\_messages;



1. Create a new table **audit\_language\_update** with the following schema.

|  |  |
| --- | --- |
| **Column** | **Datatype and constraint** |
| language\_id | TINYINT(3) |
| name | CHAR(20) |
| last\_update | TIMESTAMP |
| Status | CHAR(20)    Should accept only “Before Update” and “After Update” |

For audit purpose, Create a trigger that inserts existing (old) record as well as updated record into above table whenever a record gets updated in ‘language’ table

CREATE TABLE audit\_language\_update (

id INT AUTO\_INCREMENT PRIMARY KEY,

language\_id TINYINT(3),

name CHAR(20),

last\_update TIMESTAMP,

status CHAR(20)

);

DELIMITER //

CREATE TRIGGER language\_audit\_trigger

AFTER UPDATE ON language

FOR EACH ROW

BEGIN

-- Insert old record (BEFORE UPDATE) into audit table

INSERT INTO audit\_language\_update (language\_id, name, last\_update, status)

VALUES (OLD.language\_id, OLD.name, OLD.last\_update, 'Before Update');

-- Insert updated record (AFTER UPDATE) into audit table

INSERT INTO audit\_language\_update (language\_id, name, last\_update, status)

VALUES (NEW.language\_id, NEW.name, NEW.last\_update, 'After Update');

END;

//

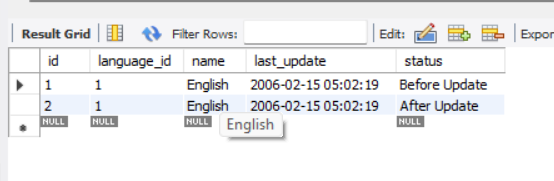
DELIMITER ;

UPDATE language

SET name = 'English'

WHERE language\_id = 1;

SELECT \* FROM audit\_language\_update;



1. Create a new table **language\_before\_delete** with the following schema.

|  |  |
| --- | --- |
| **Column** | **Datatype and constraint** |
| language\_id | TINYINT(3) |
| name | CHAR(20) |
| last\_update | TIMESTAMP |
| Status | CHAR(20)  **Default:** “Before Delete” |

Create a trigger that keeps backup of deleted records into above table whenever a record is deleted in ‘language’ table

CREATE TABLE language\_before\_delete (

language\_id TINYINT(3),

name CHAR(20),

last\_update TIMESTAMP,

status CHAR(20) DEFAULT 'Before Delete'

);

DELIMITER //

CREATE TRIGGER language\_delete\_trigger

AFTER DELETE ON language

FOR EACH ROW

BEGIN

-- Insert deleted record into backup table

INSERT INTO language\_before\_delete (language\_id, name, last\_update)

VALUES (OLD.language\_id, OLD.name, OLD.last\_update);

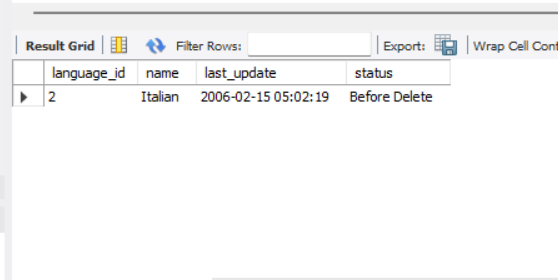
END;

//

DELIMITER ;

DELETE FROM language WHERE language\_id = 2;

SELECT \* FROM language\_before\_delete;



1. Create a new table **language\_after\_delete** with the following schema.

|  |  |
| --- | --- |
| **Column** | **Datatype and constraint** |
| language\_id | TINYINT(3) |
| name | CHAR(20) |
| Status | VARCHAR(200) |

Create a trigger that maintains the status of deleted records into above table whenever a record is deleted in ‘language’ table.

Status should be displayed as: “Language ***[language\_name]*** with ID ***[language\_id]*** was deleted on ***[timestamp when the record was deleted]***

***-- Create the language\_after\_delete table***

***CREATE TABLE language\_after\_delete (***

***language\_id TINYINT(3),***

***name CHAR(20),***

***Status VARCHAR(200)***

***);***

***-- Create the trigger to maintain deleted records in language\_after\_delete table***

***DELIMITER //***

***CREATE TRIGGER after\_language\_delete***

***AFTER DELETE ON language***

***FOR EACH ROW***

***BEGIN***

***INSERT INTO language\_after\_delete (language\_id, name, Status)***

***VALUES (OLD.language\_id, OLD.name,***

***CONCAT('Language ', OLD.name, ' with ID ', OLD.language\_id, ' was deleted on ', NOW()));***

***END;***

***//***

***DELIMITER ;***

***-- Insert some sample data into the language table***

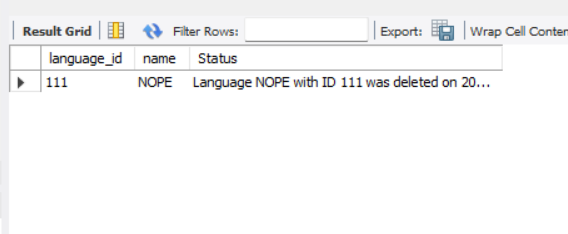
***INSERT INTO language (language\_id, name) VALUES (111, 'nope ');***

***INSERT INTO language (language\_id, name) VALUES (121, 'animal');***

***INSERT INTO language (language\_id, name) VALUES (131, 'code');***

***DELETE FROM language WHERE language\_id = 111;***

***SELECT \* FROM language\_after\_delete;***

******

1. Display all the triggers in your database.

show triggers;

