EASTERN INTERNATIONAL UNIVERSITY
SCHOOL OF COMPUTING
AND
INFORMATION TECHNOLOGY

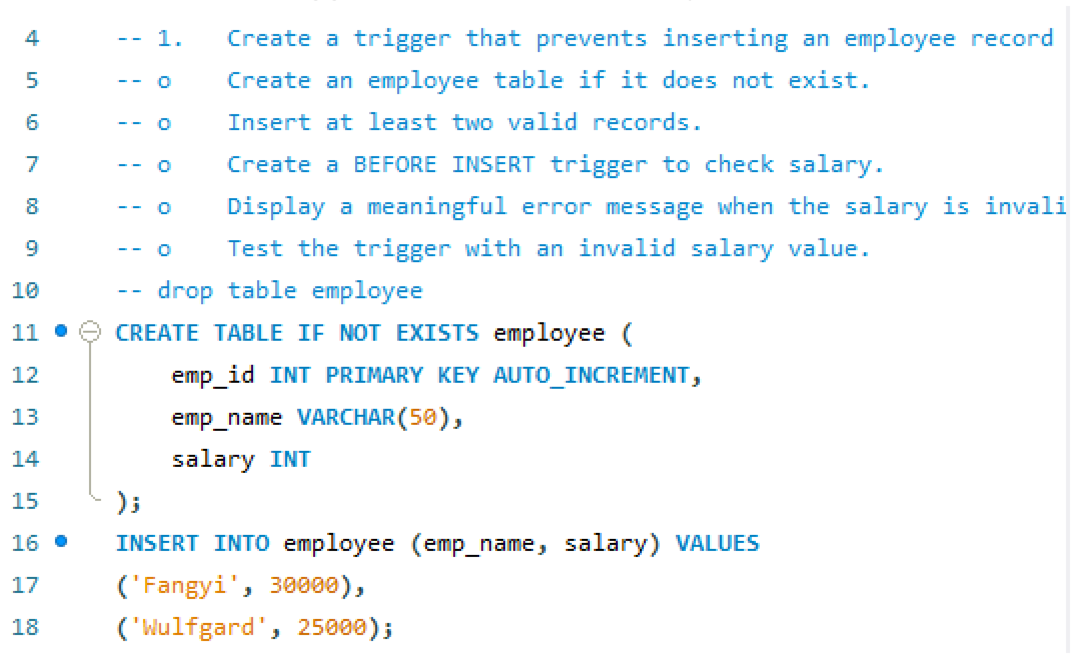
Week 5 Practice Assignment – Quarter 2, 2025-2026
Method: LAB 4 Practice 4
Course Name: Advanced Databases
Course Code: CSW 436
Student's Full Name:Phạm Trần Gia Hưng
Student ID: 2331200153

Important Instructions –

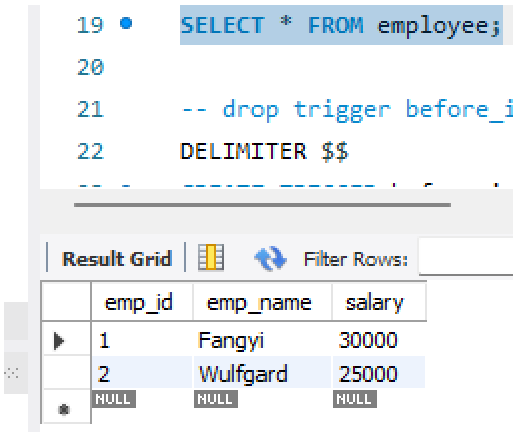
- Implement each question using MySQL Workbench.
- Document all question Output/results properly by capturing the screenshots of the output/results and SQL code for every question in a Word document and save it.
- After completing all questions, upload the document to Moodle.

Topic: Triggers and handle errors

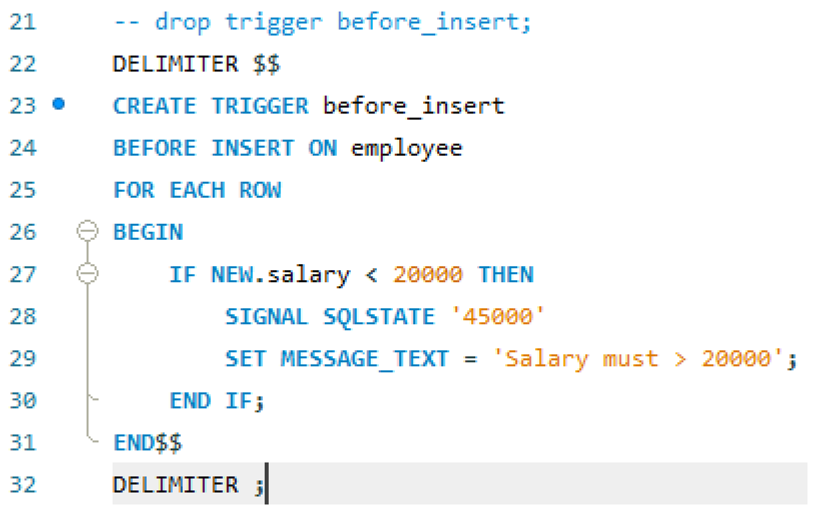
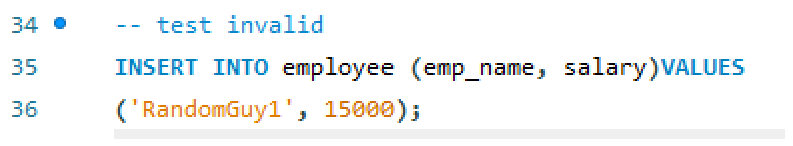
1. Create a trigger that prevents inserting an employee record if the salary is less than 20,000.
    - Create an `employee` table if it does not exist.
    - Insert at least two valid records.
    - Create a BEFORE INSERT trigger to check salary.
    - Display a meaningful error message when the salary is invalid.
    - Test the trigger with an invalid salary value.

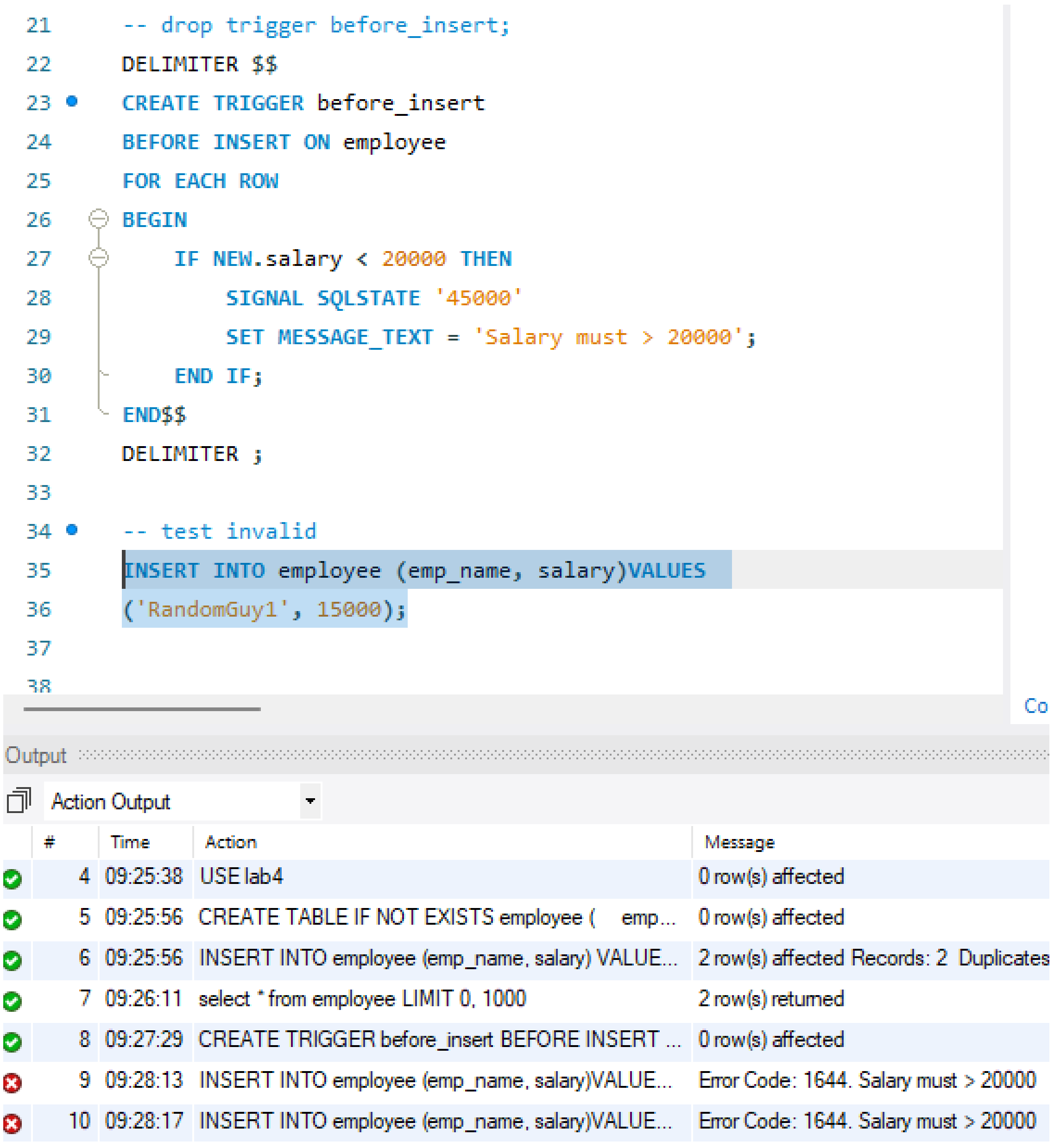


```
-- 1.    Create a trigger that prevents inserting an employee record
-- o     Create an employee table if it does not exist.
-- o     Insert at least two valid records.
-- o     Create a BEFORE INSERT trigger to check salary.
-- o     Display a meaningful error message when the salary is invali
-- o     Test the trigger with an invalid salary value.
-- drop table employee
CREATE TABLE IF NOT EXISTS employee (
    emp_id INT PRIMARY KEY AUTO_INCREMENT,
    emp_name VARCHAR(50),
    salary INT
);
INSERT INTO employee (emp_name, salary) VALUES
('Fangyi', 30000),
('Wulfgard', 25000);
```


```sql
SELECT * FROM employee;

-- drop trigger before_i
DELIMITER $$
```

emp_id	emp_name	salary
1	Fangyi	30000
2	Wulfgard	25000
NULL	NULL	NULL


```sql
-- drop trigger before_insert;
DELIMITER $$
CREATE TRIGGER before_insert
BEFORE INSERT ON employee
FOR EACH ROW
BEGIN
    IF NEW.salary < 20000 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Salary must > 20000';
    END IF;
END$$
DELIMITER ;
```

```sql
-- test invalid
INSERT INTO employee (emp_name, salary)VALUES
('RandomGuy1', 15000);
```



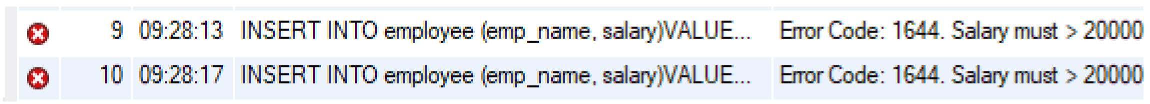
```sql
-- drop trigger before_insert;
DELIMITER $$
CREATE TRIGGER before_insert
BEFORE INSERT ON employee
FOR EACH ROW
BEGIN
    IF NEW.salary < 20000 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Salary must > 20000';
    END IF;
END$$
DELIMITER ;

-- test invalid
INSERT INTO employee (emp_name, salary)VALUES
('RandomGuy1', 15000);
```

Output

Action Output

	#	Time	Action	Message
✓	4	09:25:38	USE lab4	0 row(s) affected
✓	5	09:25:56	CREATE TABLE IF NOT EXISTS employee (emp...	0 row(s) affected
✓	6	09:25:56	INSERT INTO employee (emp_name, salary) VALUE...	2 row(s) affected Records: 2 Duplicates
✓	7	09:26:11	select * from employee LIMIT 0, 1000	2 row(s) returned
✓	8	09:27:29	CREATE TRIGGER before_insert BEFORE INSERT ...	0 row(s) affected
✗	9	09:28:13	INSERT INTO employee (emp_name, salary)VALUE...	Error Code: 1644. Salary must > 20000
✗	10	09:28:17	INSERT INTO employee (emp_name, salary)VALUE...	Error Code: 1644. Salary must > 20000

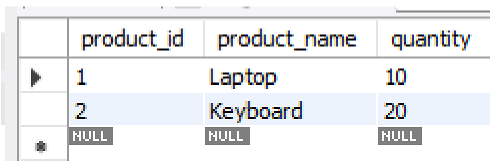
Result:



✗	9	09:28:13	INSERT INTO employee (emp_name, salary)VALUE...	Error Code: 1644. Salary must > 20000
✗	10	09:28:17	INSERT INTO employee (emp_name, salary)VALUE...	Error Code: 1644. Salary must > 20000

2. Create a trigger that prevents updating product quantity to zero or negative value. (Trigger to Maintain Stock Quantity)
    - Create a `product` table if it does not exist.
    - Insert sample records.
    - Create a BEFORE UPDATE trigger to validate quantity.
    - Display an error message if quantity is less than 1.
    - Test the trigger using an UPDATE statement.

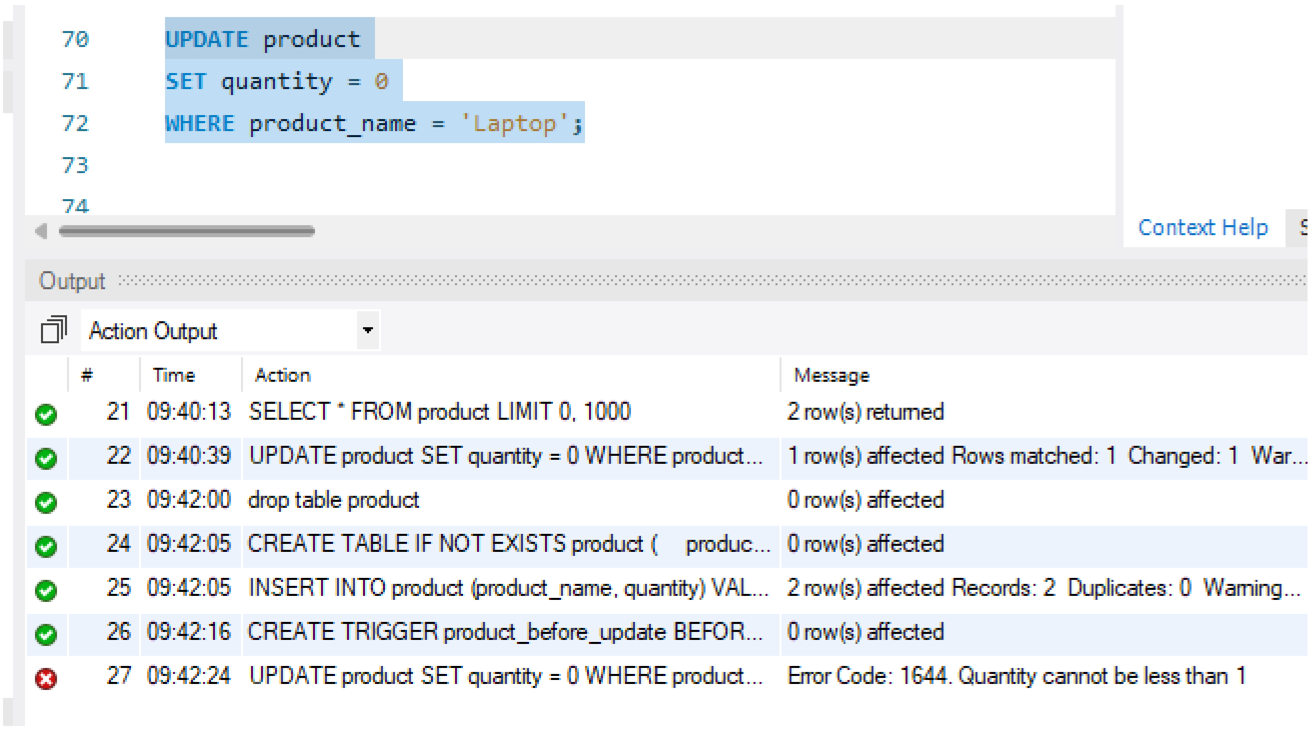
```sql
-- 2.   Create a trigger that prevents updating product quantity to
-- o    Create a product table if it does not exist.
-- o    Insert sample records.
-- o    Create a BEFORE UPDATE trigger to validate quantity.
-- o    Display an error message if quantity is less than 1.
-- o    Test the trigger using an UPDATE statement.
-- drop table product
CREATE TABLE IF NOT EXISTS product (
    product_id INT PRIMARY KEY AUTO_INCREMENT,
    product_name VARCHAR(50),
    quantity INT NOT NULL
);
INSERT INTO product (product_name, quantity) VALUES
('Laptop', 10),
('Keyboard', 20);
SELECT * FROM product;
```



product_id	product_name	quantity
1	Laptop	10
2	Keyboard	20
NULL	NULL	NULL

```sql
-- drop trigger product_before_update;
DELIMITER $$
CREATE TRIGGER product_before_update
BEFORE UPDATE ON product
FOR EACH ROW
BEGIN
    IF NEW.quantity < 1 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Quantity cannot be less than 1';
    END IF;
END$$
DELIMITER ;
```

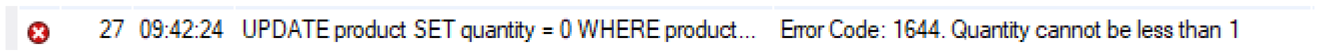
```sql
-- test
UPDATE product
SET quantity = 0
WHERE product_name = 'Laptop';
```



```
UPDATE product
SET quantity = 0
WHERE product_name = 'Laptop';
```

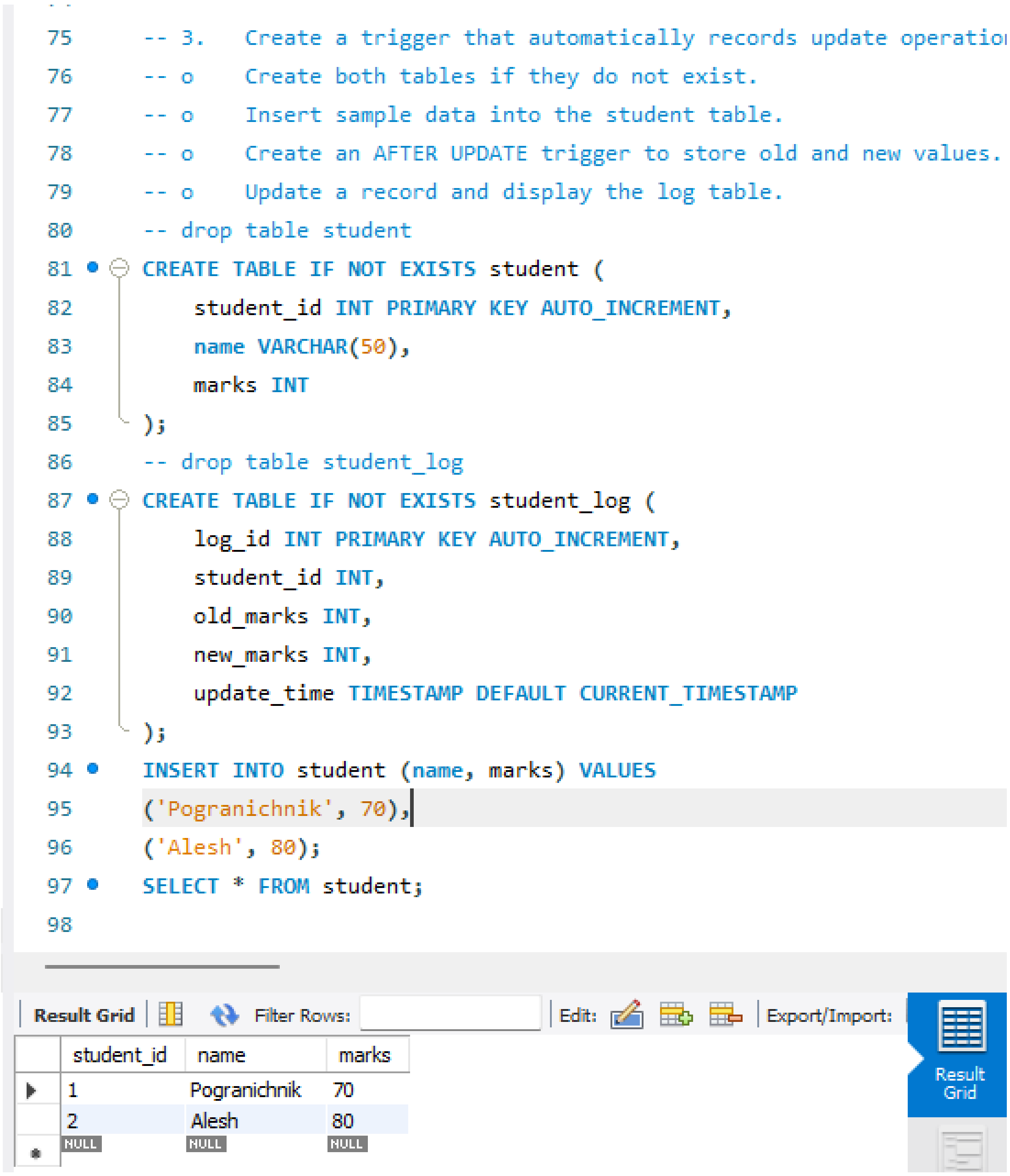
```
#   Time      Action                                              Message
21  09:40:13  SELECT * FROM product LIMIT 0, 1000                 2 row(s) returned
22  09:40:39  UPDATE product SET quantity = 0 WHERE product...    1 row(s) affected Rows matched: 1  Changed: 1  War...
23  09:42:00  drop table product                                  0 row(s) affected
24  09:42:05  CREATE TABLE IF NOT EXISTS product (    produc...   0 row(s) affected
25  09:42:05  INSERT INTO product (product_name, quantity) VAL... 2 row(s) affected Records: 2  Duplicates: 0  Warning...
26  09:42:16  CREATE TRIGGER product_before_update BEFOR...       0 row(s) affected
27  09:42:24  UPDATE product SET quantity = 0 WHERE product...    Error Code: 1644. Quantity cannot be less than 1
```

Result:



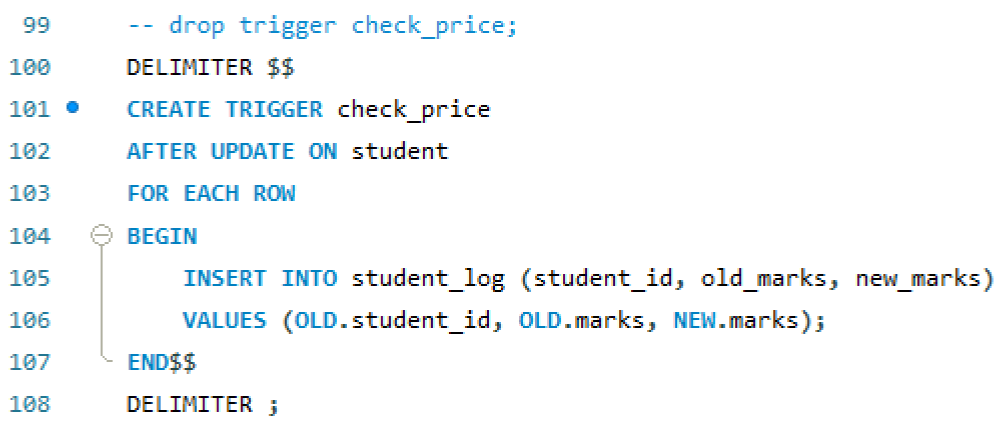
```
27  09:42:24  UPDATE product SET quantity = 0 WHERE product...    Error Code: 1644. Quantity cannot be less than 1
```

3. Create a trigger that automatically records update operations on a student table into a student_log table.
   - Create both tables if they do not exist.
   - Insert sample data into the `student` table.
   - Create an AFTER UPDATE trigger to store old and new values.
   - Update a record and display the log table.

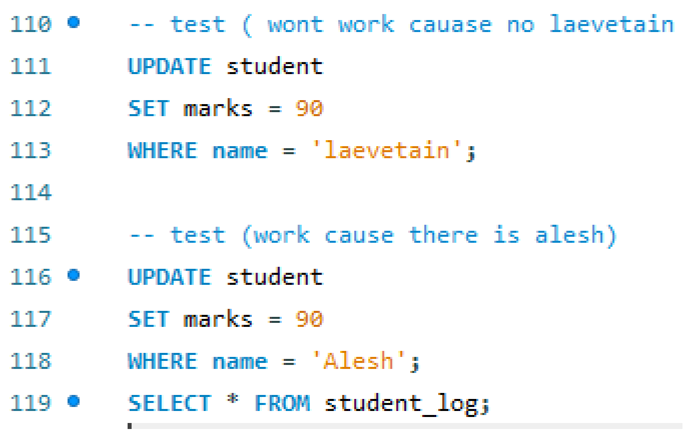


```sql
-- 3.   Create a trigger that automatically records update operatio
-- o    Create both tables if they do not exist.
-- o    Insert sample data into the student table.
-- o    Create an AFTER UPDATE trigger to store old and new values.
-- o    Update a record and display the log table.
-- drop table student
CREATE TABLE IF NOT EXISTS student (
    student_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(50),
    marks INT
);
-- drop table student_log
CREATE TABLE IF NOT EXISTS student_log (
    log_id INT PRIMARY KEY AUTO_INCREMENT,
    student_id INT,
    old_marks INT,
    new_marks INT,
    update_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
INSERT INTO student (name, marks) VALUES
('Pogranichnik', 70),
('Alesh', 80);
SELECT * FROM student;

```

student_id	name	marks
1	Pogranichnik	70
2	Alesh	80
NULL	NULL	NULL

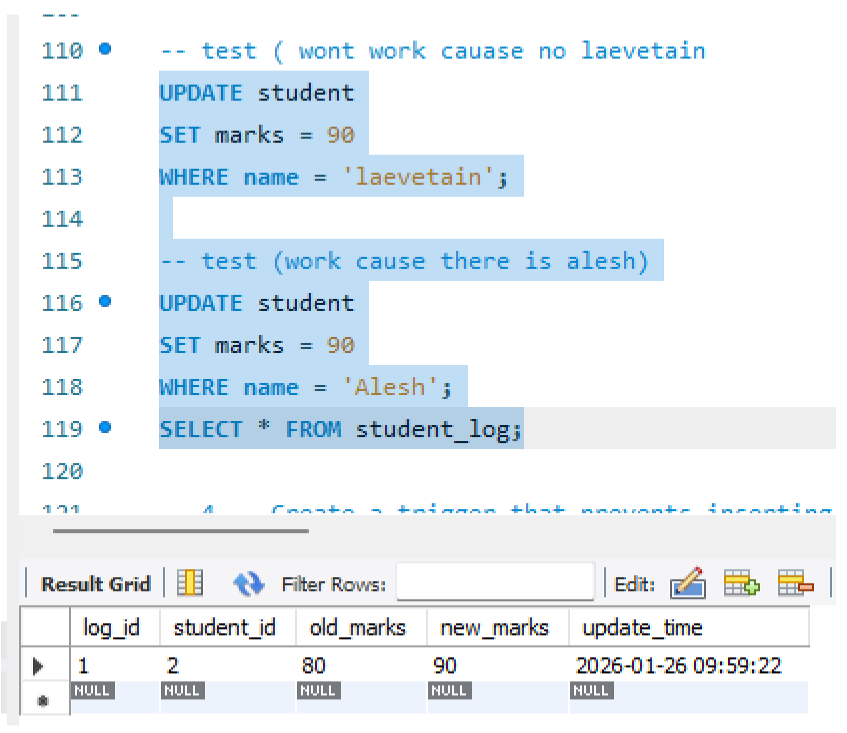


```sql
-- drop trigger check_price;
DELIMITER $$
CREATE TRIGGER check_price
AFTER UPDATE ON student
FOR EACH ROW
BEGIN
    INSERT INTO student_log (student_id, old_marks, new_marks)
    VALUES (OLD.student_id, OLD.marks, NEW.marks);
END$$
DELIMITER ;
```



```sql
-- test ( wont work cauase no laevetain
UPDATE student
SET marks = 90
WHERE name = 'laevetain';

-- test (work cause there is alesh)
UPDATE student
SET marks = 90
WHERE name = 'Alesh';
SELECT * FROM student_log;
```

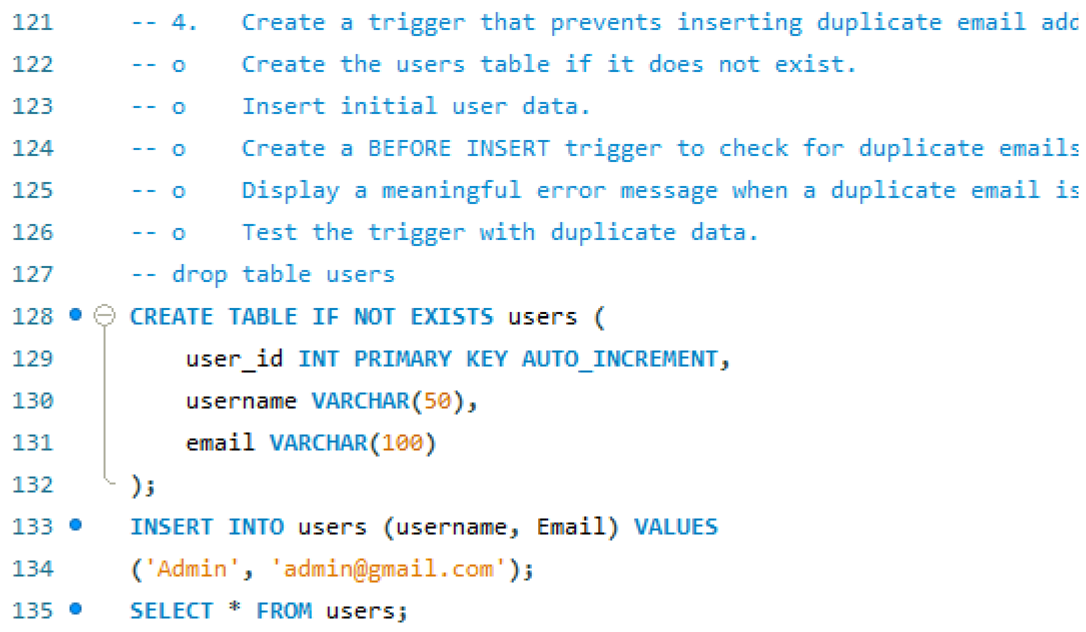
Result:



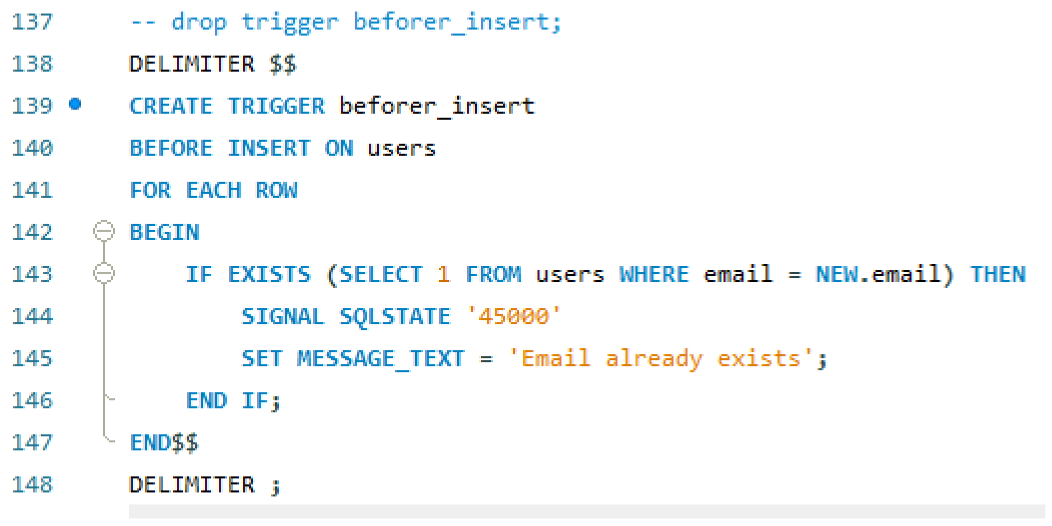
```sql
---
-- test ( wont work cauase no laevetain
UPDATE student
SET marks = 90
WHERE name = 'laevetain';

-- test (work cause there is alesh)
UPDATE student
SET marks = 90
WHERE name = 'Alesh';
SELECT * FROM student_log;

        4.   Create a trigger that prevents inserting
```

log_id	student_id	old_marks	new_marks	update_time
1	2	80	90	2026-01-26 09:59:22
NULL	NULL	NULL	NULL	NULL

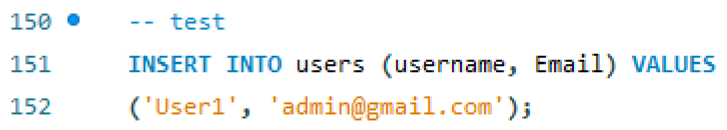
4. Create a trigger that prevents inserting duplicate email addresses into a user’s table.
   - Create the `users` table if it does not exist.
   - Insert initial user data.
   - Create a BEFORE INSERT trigger to check for duplicate emails.
   - Display a meaningful error message when a duplicate email is inserted.
   - Test the trigger with duplicate data.



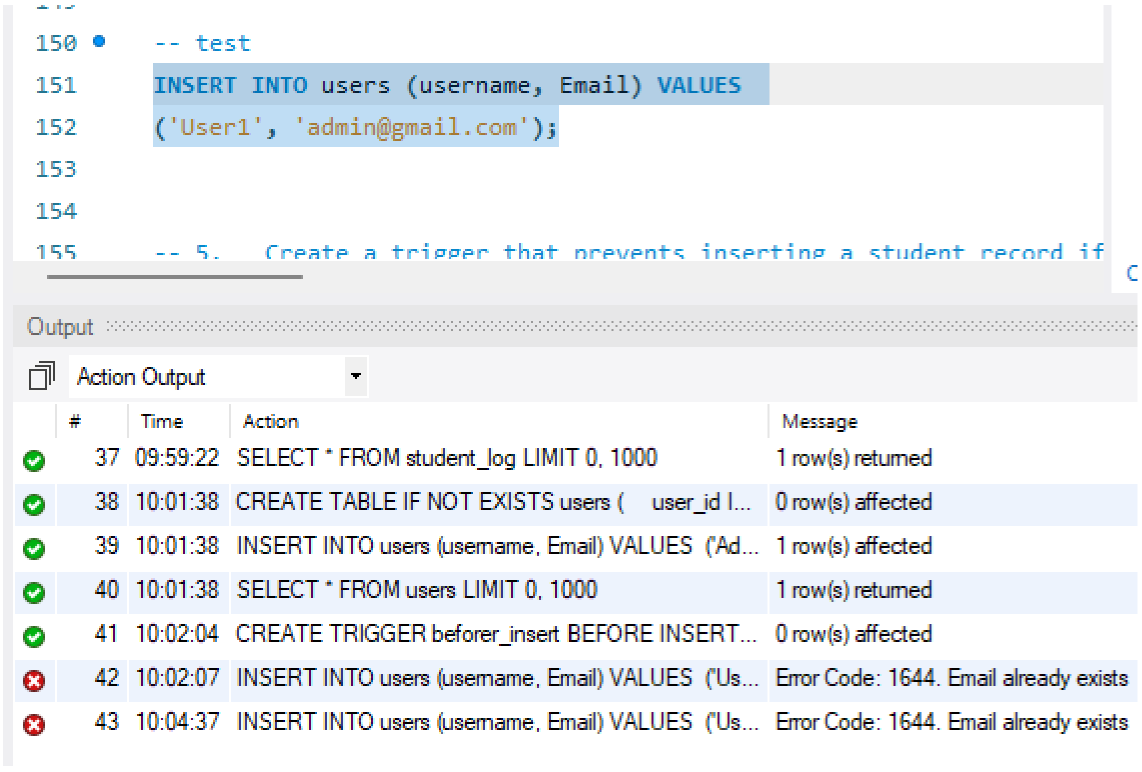
```sql
-- 4.   Create a trigger that prevents inserting duplicate email ad
-- o    Create the users table if it does not exist.
-- o    Insert initial user data.
-- o    Create a BEFORE INSERT trigger to check for duplicate emails
-- o    Display a meaningful error message when a duplicate email is
-- o    Test the trigger with duplicate data.
-- drop table users
CREATE TABLE IF NOT EXISTS users (
    user_id INT PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(50),
    email VARCHAR(100)
);
INSERT INTO users (username, Email) VALUES
('Admin', 'admin@gmail.com');
SELECT * FROM users;
```



```sql
-- drop trigger beforer_insert;
DELIMITER $$
CREATE TRIGGER beforer_insert
BEFORE INSERT ON users
FOR EACH ROW
BEGIN
    IF EXISTS (SELECT 1 FROM users WHERE email = NEW.email) THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Email already exists';
    END IF;
END$$
DELIMITER ;
```



```sql
-- test
INSERT INTO users (username, Email) VALUES
('User1', 'admin@gmail.com');
```



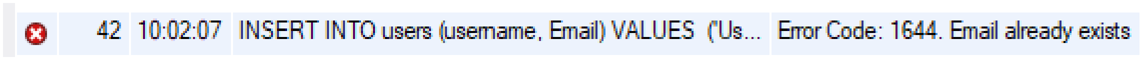
```
-- test
INSERT INTO users (username, Email) VALUES
('User1', 'admin@gmail.com');


-- 5.  Create a trigger that prevents inserting a student record if
```

Output

Action Output

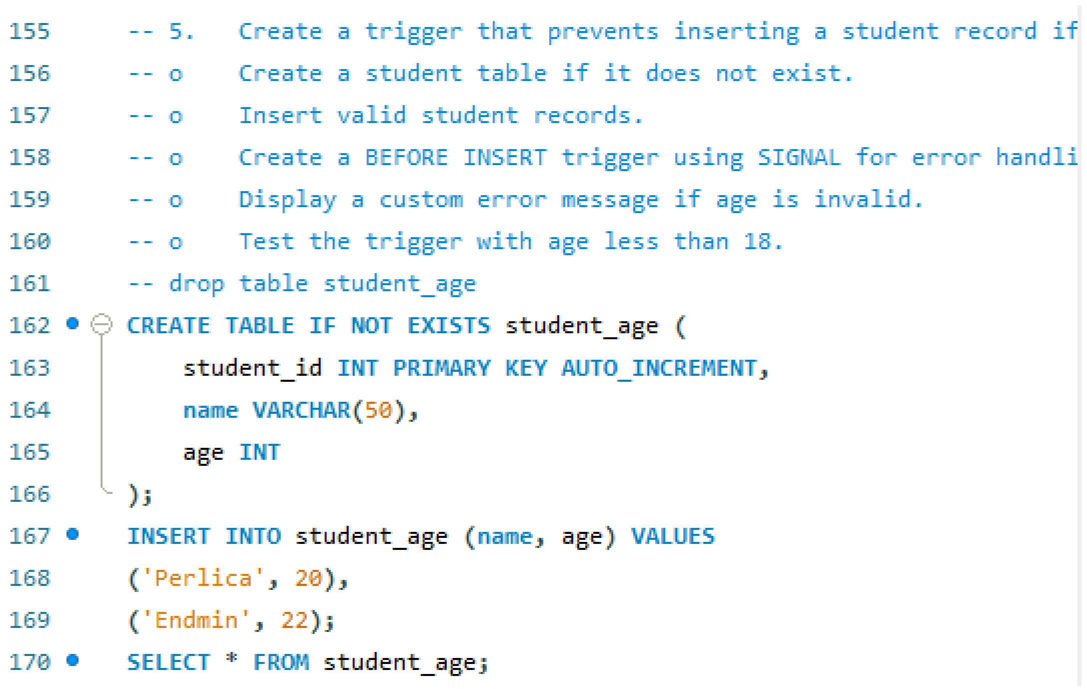
```
#   Time      Action                                            Message
37  09:59:22  SELECT * FROM student_log LIMIT 0, 1000           1 row(s) returned
38  10:01:38  CREATE TABLE IF NOT EXISTS users (    user_id I... 0 row(s) affected
39  10:01:38  INSERT INTO users (username, Email) VALUES  ('Ad... 1 row(s) affected
40  10:01:38  SELECT * FROM users LIMIT 0, 1000                 1 row(s) returned
41  10:02:04  CREATE TRIGGER beforer_insert BEFORE INSERT...    0 row(s) affected
42  10:02:07  INSERT INTO users (username, Email) VALUES  ('Us... Error Code: 1644. Email already exists
43  10:04:37  INSERT INTO users (username, Email) VALUES  ('Us... Error Code: 1644. Email already exists
```

Result:

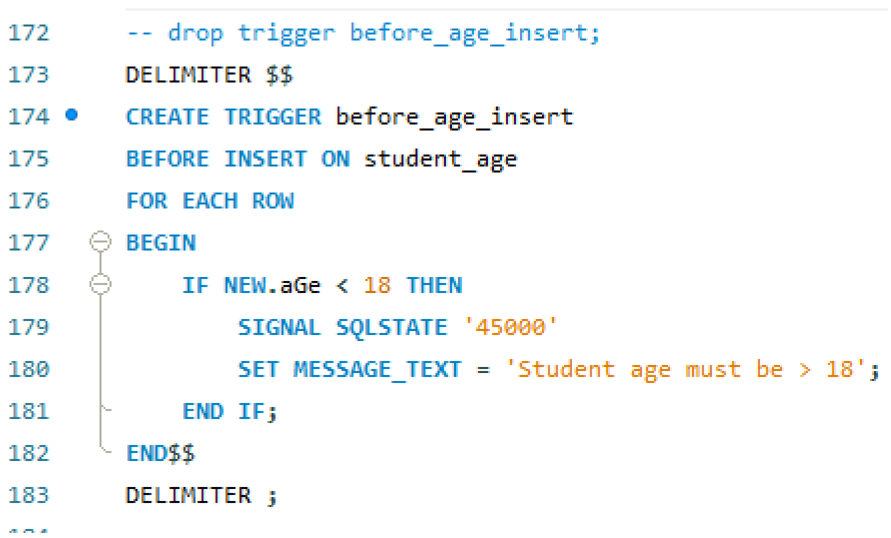


```
42  10:02:07  INSERT INTO users (username, Email) VALUES  ('Us... Error Code: 1644. Email already exists
```

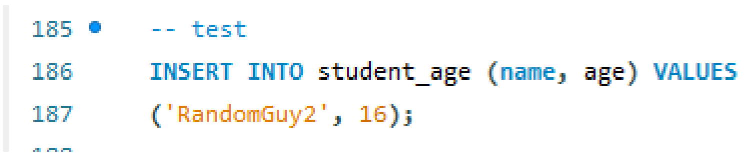
5. Create a trigger that prevents inserting a student record if age is less than 18.
   - Create a `student` table if it does not exist.
   - Insert valid student records.
   - Create a BEFORE INSERT trigger using `SIGNAL` for error handling.
   - Display a custom error message if age is invalid.
   - Test the trigger with age less than 18.



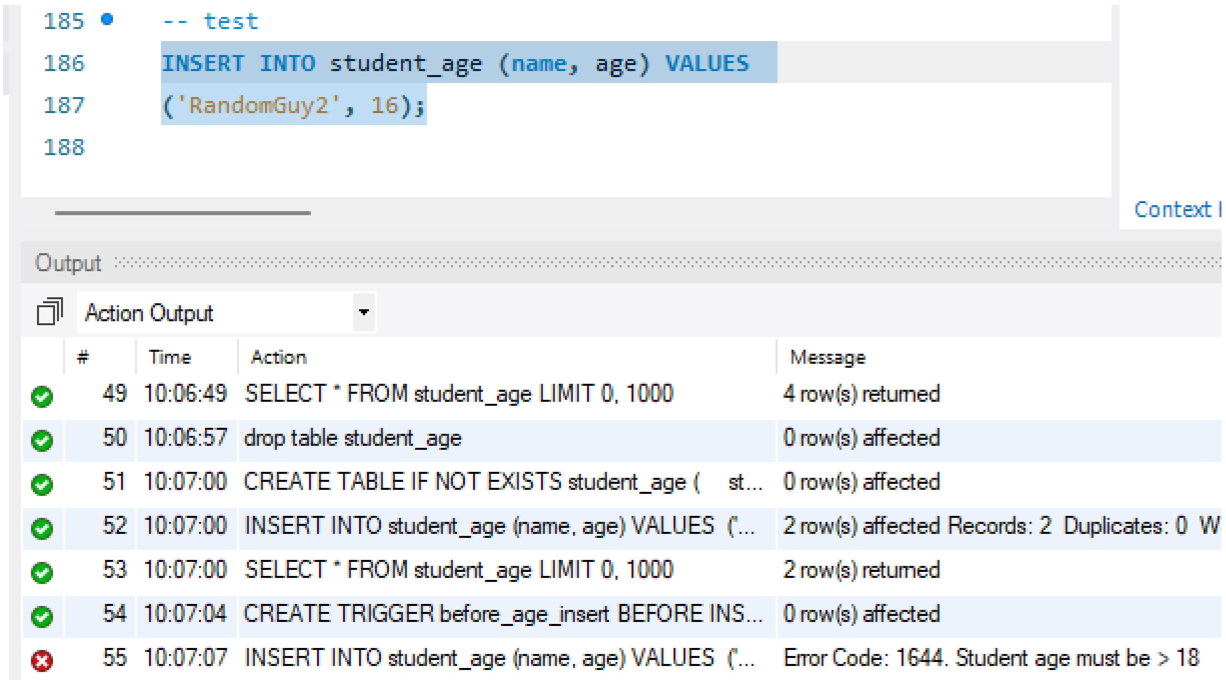
```sql
-- 5.   Create a trigger that prevents inserting a student record if
-- o    Create a student table if it does not exist.
-- o    Insert valid student records.
-- o    Create a BEFORE INSERT trigger using SIGNAL for error handli
-- o    Display a custom error message if age is invalid.
-- o    Test the trigger with age less than 18.
-- drop table student_age
CREATE TABLE IF NOT EXISTS student_age (
    student_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(50),
    age INT
);
INSERT INTO student_age (name, age) VALUES
('Perlica', 20),
('Endmin', 22);
SELECT * FROM student_age;
```



```sql
-- drop trigger before_age_insert;
DELIMITER $$
CREATE TRIGGER before_age_insert
BEFORE INSERT ON student_age
FOR EACH ROW
BEGIN
    IF NEW.aGe < 18 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Student age must be > 18';
    END IF;
END$$
DELIMITER ;
```

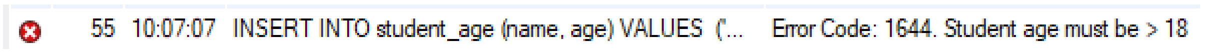


```sql
-- test
INSERT INTO student_age (name, age) VALUES
('RandomGuy2', 16);
```


```
-- test
INSERT INTO student_age (name, age) VALUES
('RandomGuy2', 16);
```

```
    #   Time      Action                                          Message
✓   49  10:06:49  SELECT * FROM student_age LIMIT 0, 1000         4 row(s) returned
✓   50  10:06:57  drop table student_age                          0 row(s) affected
✓   51  10:07:00  CREATE TABLE IF NOT EXISTS student_age (   st...  0 row(s) affected
✓   52  10:07:00  INSERT INTO student_age (name, age) VALUES  ('...  2 row(s) affected Records: 2  Duplicates: 0  W
✓   53  10:07:00  SELECT * FROM student_age LIMIT 0, 1000         2 row(s) returned
✓   54  10:07:04  CREATE TRIGGER before_age_insert BEFORE INS...  0 row(s) affected
✗   55  10:07:07  INSERT INTO student_age (name, age) VALUES  ('...  Error Code: 1644. Student age must be > 18
```

Result:


```
✗   55  10:07:07  INSERT INTO student_age (name, age) VALUES  ('...  Error Code: 1644. Student age must be > 18
```