SQL TASK 10

1. Create a table named teachers with fields id, name, subject, experience and salary and insert 8 rows.

2. Create a before insert trigger named before\_insert\_teacher that will raise an error “salary cannot be negative” if the salary inserted to the table is less than zero.

--This trigger checks the salary value before a new row is added.If salary is less than zero the trigger raises an error message.

**Advantage**—This trigger enforces data integrity by preventing invalid data from entering the database. To achieve this advantage I run this query.

CREATE TRIGGER before\_insert\_teacher

ON teachers

INSTEAD OF INSERT

AS

BEGIN

IF EXISTS (SELECT 1 FROM inserted WHERE salary<0)

BEGIN

RAISERROR ('Salary cannot be negative',16,1);

ROLLBACK TRANSACTION;

END

ELSE

BEGIN

INSERT INTO teachers (Id, Name, subject, experience, salary) SELECT id,name,subject,experience,salary from inserted;

END

END;

INSERT INTO teachers (Id, Name, subject, Experience, Salary) VALUES (9, 'Pramod kumar','Science', 6, -55000.07)

A screenshot of a computer

Description automatically generated

3. Create an after insert trigger named after\_insert\_teacher that inserts a row with teacher\_id,action, timestamp to a table called teacher\_log when a new entry gets inserted to the teacher table. tecaher\_id -> column of teacher table, action -> the trigger action, timestamp -> time at which the new row has got inserted.

--This AFTER INSERT trigger runs immediately after a new row is added to the teachers table. It inserts a log entry into the teacher\_log table, storing the teacher\_id, action as "INSERT", and the current timestamp (NOW()).

**Advantage**: By logging every insertion, this trigger enables tracking of changes and provides a way to audit the data added to the teachers table. This can be helpful for compliance and troubleshooting purposes.

We can achieve this by

CREATE table teacher\_log (

teacher\_id INT PRIMARY KEY,

action NVARCHAR(50),

Timestamp DATETIME);

CREATE TRIGGER after\_insert\_teacher

on teachers

AFTER INSERT

AS

BEGIN

INSERT INTO teacher\_log (teacher\_id, action, timestamp)

SELECT id, 'insert', GETDATE()

FROM inserted;

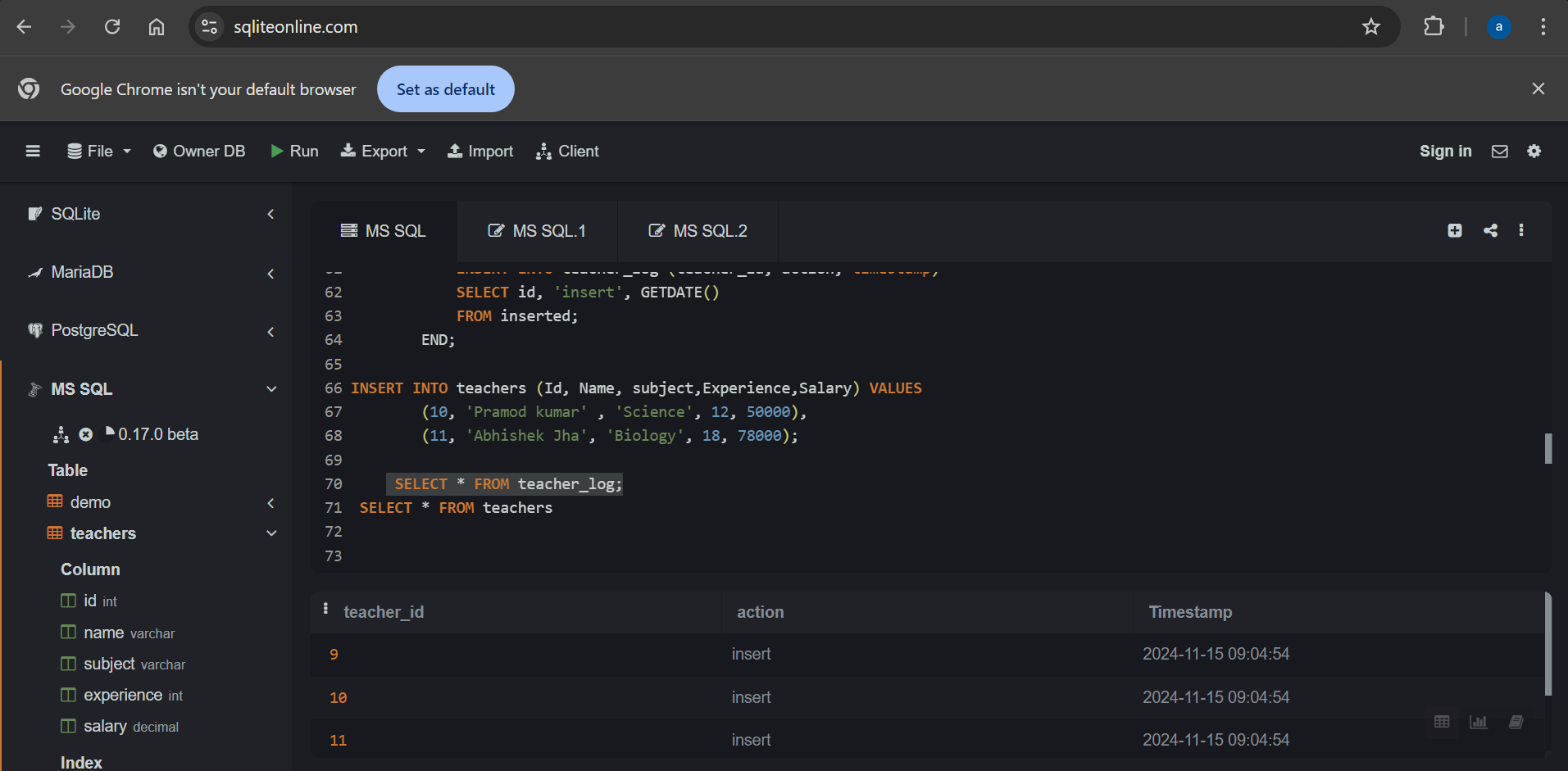
END;

INSERT INTO teachers (Id, Name, subject,Experience,Salary) VALUES

(10, 'Pramod kumar' , 'Science', 12, 50000),

(11, 'Abhishek Jha', 'Biology', 18, 78000);

SELECT \* FROM teacher\_log;



4. Create a before delete trigger that will raise an error when you try to delete a row that has experience greater than 10 years.

--This Before delete trigger checks if the experience of the teacher being deleted is greater than 10,if yes it raises an error preventing the deletion.

**Advantage:**The trigger safeguards data by ensuring that some data which is not supposed to be deleted gets deleted unintentionally.

To achieve this

CREATE TRIGGER before\_delete\_teacher

ON teachers

INSTEAD OF DELETE

AS

BEGIN

IF EXISTS (SELECT 1 from deleted where experience>10)

BEGIN

RAISERROR ('Cannot delete a teacher with more than 10 year experience',16,1);

ROLLBACK TRANSACTION;

END

ELSE

BEGIN

DELETE FROM teachers

WHERE id in (SELECT id FROM deleted);

END

END;

DELETE from teachers

WHERE id=6;

A screenshot of a computer

Description automatically generated

5. Create an after delete trigger that will insert a row to teacher\_log table when that row is deleted from teacher table.

--This trigger records an entry in teacher\_log after a row is deleted from teachers table storing the teacher\_id,action and the deletion timestamp.

**Advantage**:It helps in keeping clear record of deletion.

CREATE TRIGGER after\_delete\_trigger

ON teachers

AFTER DELETE

AS

BEGIN

INSERT INTO teacher\_log (teacher\_id, action, timestamp)

SELECT id, 'DELETE', GETDATE() FROM deleted;

END;

DELETE from teachers

WHERE id=4;

SELECT \* from teacher\_log;

