CSE 311L(Database Management System)

LAB-Week 08

Triggers/Stored Procedure Examples & Demo

Trigger Example

- **Example 1:**

```
    CREATE TABLE salaries (
        id INT PRIMARY KEY,
        valid_from DATE NOT NULL,
        amount DECIMAL(12 , 2 ) NOT NULL DEFAULT 0
    );

    INSERT INTO salaries(id,valid_from,amount)
    VALUES
        (1002,'2000-01-01',50000),
        (1056,'2000-01-01',60000),
        (1076,'2000-01-01',70000);


    CREATE TABLE salary_archives (
        id INT PRIMARY KEY AUTO_INCREMENT,
        employee_id INT,
        valid_from DATE NOT NULL,
        amount DECIMAL(12 , 2 ) NOT NULL DEFAULT 0,
        deleted_at TIMESTAMP DEFAULT NOW()
  );


DELIMITER $$

    CREATE TRIGGER before_salaries_delete
    BEFORE DELETE
    ON salaries FOR EACH ROW
    BEGIN
        INSERT INTO salary_archives(employee_id,valid_from,amount)
        VALUES(OLD.id,OLD.valid_from,OLD.amount);
    END$$

DELIMITER ;


    DELETE FROM salaries
    WHERE ID = 1002;
```

- **Example 2:**

```
    CREATE TABLE salary_budgets(
        total DECIMAL(15,2) NOT NULL
    );
```

```
      INSERT INTO salary_budgets(total)
      SELECT SUM(amount)
      FROM salaries;


DELIMITER $$

      CREATE TRIGGER after_update
         AFTER UPDATE ON salaries
         FOR EACH ROW
      BEGIN
           IF OLD.amount <> NEW.amount THEN
               update salary_budgets
                  set total = (SELECT sum(amount) from salaries);

           END IF;
      END$$

DELIMITER ;



DELIMITER $$

      CREATE TRIGGER after_delete
         AFTER DELETE ON salaries
         FOR EACH ROW
      BEGIN
               update salary_budgets
                  set total = total - OLD.amount;
      END$$
      CREATE TRIGGER after_inserted

         AFTER INSERT ON salaries
         FOR EACH ROW
      BEGIN
               update salary_budgets
                  set total = total + NEW.amount;

      END$$

DELIMITER ;
```

- **Example 3:**

```
   CREATE TABLE Bank_account (
      Account INT PRIMARY KEY,
      Name VARCHAR2(35),
      Balance NUMBER(10,2),
      Pin INT
   );

   CREATE TABLE Account_Audit (
      Account INT,
      Prev_Balance DECIMAL(10,2),
      Curr_Balance DECIMAL(10,2),
      Audit_date Date,
      OP_Type VARCHAR2(10)
   );
```

```
DELIMITER $$

      Create or Replace trigger INS_OR_UPD_OR_DEL
            before insert or update or delete of Balance on Bank_Account
            for each row
      BEGIN
            if INSERTING then
                  insert into Account_Audit (Account, Curr_Balance, Audit_date,
      OP_Type)
                  values
                  (:new.Account, :new.Balance, Sysdate, 'Insert');
            end if;

            if UPDATING then
                  insert into Account_Audit (Account, Prev_Balance, Curr_Balance,
      Audit_date, OP_Type)
                  values
                  (:new.Account, :old.Balance, :new.Balance, Sysdate, 'Update');
            end if;

            if DELETING then
                  insert into Account_Audit (Account, Curr_Balance, Audit_date,
      OP_Type)
                  values
                  (:old.Account, :old.Balance, Sysdate, 'Delete');
            END IF;
      END$$

DELIMITER ;


Insert into Bank_account values (5000, 'M Faisal Nurnoby', 200, 7700);
Insert into Bank_account values (6000, 'Dr. Rashedur Rahman', 100, 7701);

Select * from Account_Audit;

Update Bank_account set Balance = 500 where Account = 5000;
Delete from Bank_account where Account = 5000;
```

Stored Procedure Example

```
CREATE TABLE BOOKSHELF (
      Title Varchar2(100) Primary Key,
      Publisher Varchar2(20),
      CategoryName Varchar2(20),
      Rating Varchar2(2)
);


insert into BOOK_ORDER values
('HP Part 1');

insert into BOOK_ORDER values
('HP Part 2');

insert into BOOK_ORDER values
('LOTR');

insert into BOOK_ORDER values
('Narnia');
```

```
create or replace procedure NEW_BOOK (aTitle in VARCHAR2, aPublisher in VARCHAR2,
aCategoryName in VARCHAR2)
as
BEGIN
      insert into BOOKSHELF (Title, Publisher, CategoryName, Rating)
            values (aTitle, aPublisher, aCategoryName, NULL);
      delete from BOOK_ORDER
            where Title = aTitle;
END;


execute NEW_BOOK('Narnia', 'Mcmillan','Fantasy');
```