CSE 311L(Database Management System)

Instructor: Nazmul Alam Diptu

LAB-Week 09 (Part A)

Controlling User Access

Topics:

- Creating Users
- Granting System Privileges
- What is a role?
- Creating and Granting Privileges to a Role
- Changing Password
- Granting Object Privileges
- Using WITH GRANT OPTION and PUBLIC key

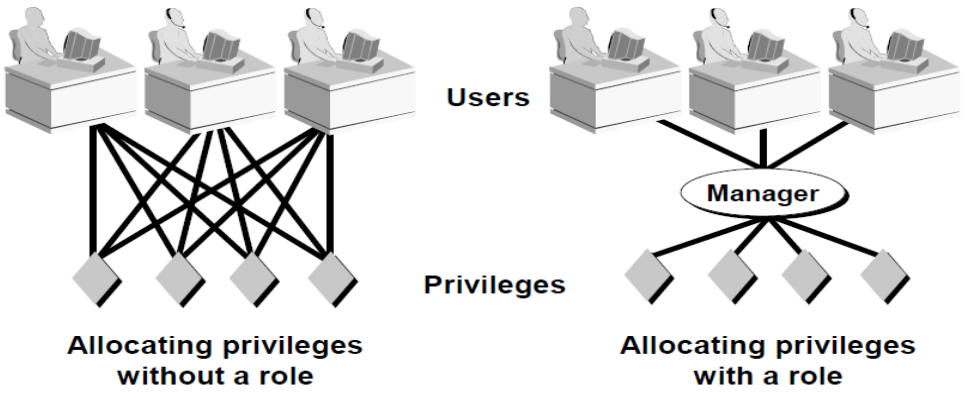
Creating Users

```
CREATE USER IF NOT EXISTS diptu
IDENTIFIED BY 'random_password';
```

Show Default Privileges Granted to the user

```
SHOW GRANTS
FOR diptu;
```

What is a role?



Creating and Granting Privileges to a Role

```
CREATE ROLE manager;

GRANT ALL
ON lab2.*
TO manager;

GRANT manager
TO diptu;
```

Changing Password

MySQL 5.7.6 and later

```
ALTER USER diptu
IDENTIFIED BY 'NEW_USER_PASSWORD';
```

MySQL 5.7.5 and earlier

```
SET PASSWORD FOR diptu = PASSWORD('NEW_USER_PASSWORD');
```

Granting Object Privileges

```
GRANT INSERT, UPDATE, DELETE
ON employees
TO tom, jerry;
```

Grant privileges to update specific columns

```
GRANT
        SELECT (fname,lname, phone),
        UPDATE(lname)
ON students
TO tom;
```

Give a user authority to pass along privileges

```
GRANT
        SELECT,UPDATE,INSERT
        ON employees
        TO diptu
WITH GRANT OPTION
```

Allow all user from the system to query from a specific table

```
GRANT
        SELECT
ON lab2.departments
TO '%';
```

CSE 311L(Database Management System)

LAB-Week 09 (Part B)

Instructor: Nazmul Alam Diptu

Triggers/Stored Procedure Examples & Demo

Trigger Example

```
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;

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 ;
```