CREATE DATABASE university\_sample;

-- Switch to the university\_sample database

USE university\_sample;

-- Create sample\_data table if it doesn't exist

Drop table sample\_data;

CREATE TABLE IF NOT EXISTS sample\_data (

data\_id INT PRIMARY KEY AUTO\_INCREMENT,

data\_value VARCHAR(100) NOT NULL

);

-- Create users (logins)

CREATE USER 'MarianoLogin'@'localhost' IDENTIFIED BY 'StrongPassword1!';

CREATE USER 'AmitLogin'@'localhost' IDENTIFIED BY 'StrongPassword2!';

CREATE USER 'SatoshiLogin'@'localhost' IDENTIFIED BY 'StrongPassword3!';

CREATE USER 'SuzanLogin'@'localhost' IDENTIFIED BY 'StrongPassword3!';

-- Grant permissions on sample\_data table

GRANT SELECT ON university\_sample.sample\_data TO 'SatoshiLogin'@'localhost';

GRANT INSERT ON university\_sample.sample\_data TO 'MarianoLogin'@'localhost';

GRANT UPDATE ON university\_sample.sample\_data TO 'AmitLogin'@'localhost';

GRANT DELETE ON university\_sample.sample\_data TO 'SuzanLogin'@'localhost';

GRANT SELECT ON university\_sample.sample\_data TO 'SuzanLogin'@'localhost', 'AmitLogin'@'localhost';

-- Create users table

CREATE TABLE users (

user\_id INT PRIMARY KEY,

name NVARCHAR(100) NOT NULL

);

-- Insert users into users table

INSERT INTO users (user\_id, name)

SELECT

user\_id,

user\_name

FROM

mysql.user

ORDER BY

user\_name;

-- Create authorizations table

CREATE TABLE authorizations (

auth\_id INT PRIMARY KEY AUTO\_INCREMENT,

user\_id INT,

table\_name NVARCHAR(100) NOT NULL,

can\_read BOOLEAN NOT NULL DEFAULT FALSE,

can\_insert BOOLEAN NOT NULL DEFAULT FALSE,

can\_update BOOLEAN NOT NULL DEFAULT FALSE,

can\_delete BOOLEAN NOT NULL DEFAULT FALSE,

FOREIGN KEY (user\_id) REFERENCES users (user\_id)

);

-- Insert authorizations into authorizations table

INSERT INTO authorizations (user\_id, table\_name, can\_read, can\_insert, can\_update, can\_delete)

SELECT

user\_id,

'sample\_data' AS table\_name,

MAX(CASE WHEN privilege\_type = 'SELECT' THEN 1 ELSE 0 END) AS can\_read,

MAX(CASE WHEN privilege\_type = 'INSERT' THEN 1 ELSE 0 END) AS can\_insert,

MAX(CASE WHEN privilege\_type = 'UPDATE' THEN 1 ELSE 0 END) AS can\_update,

MAX(CASE WHEN privilege\_type = 'DELETE' THEN 1 ELSE 0 END) AS can\_delete

FROM

information\_schema.user\_privileges

WHERE

table\_name = 'sample\_data'

GROUP BY

user\_id;

-- Create procedure to enforce read authorization

DELIMITER //

CREATE PROCEDURE enforce\_read\_auth()

BEGIN

IF NOT EXISTS (SELECT 1 FROM authorizations WHERE user\_id = CURRENT\_USER() AND table\_name = 'sample\_data' AND can\_read = 1) THEN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Read access denied';

ELSE

SELECT \* FROM sample\_data;

END IF;

END //

DELIMITER ;

-- Execute read authorization procedure

CALL enforce\_read\_auth();

-- Create trigger to enforce insert authorization

DELIMITER //

CREATE TRIGGER enforce\_insert\_auth

BEFORE INSERT ON sample\_data

FOR EACH ROW

BEGIN

IF NOT EXISTS (SELECT 1 FROM authorizations WHERE user\_id = CURRENT\_USER() AND table\_name = 'sample\_data' AND can\_insert = 1) THEN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Insert access denied';

END IF;

END //

DELIMITER ;

-- Create trigger to enforce update authorization

DELIMITER //

CREATE TRIGGER enforce\_update\_auth

BEFORE UPDATE ON sample\_data

FOR EACH ROW

BEGIN

IF NOT EXISTS (SELECT 1 FROM authorizations WHERE user\_id = CURRENT\_USER() AND table\_name = 'sample\_data' AND can\_update = 1) THEN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Update access denied';

END IF;

END //

DELIMITER ;

-- Create trigger to enforce delete authorization

DELIMITER //

CREATE TRIGGER enforce\_delete\_auth

BEFORE DELETE ON sample\_data

FOR EACH ROW

BEGIN

IF NOT EXISTS (SELECT 1 FROM authorizations WHERE user\_id = CURRENT\_USER() AND table\_name = 'sample\_data' AND can\_delete = 1) THEN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Delete access denied';

END IF;

END //

DELIMITER ;

-- Execute as different users to test authorizations

SET @user = 'AmitLogin';

CALL enforce\_read\_auth();

SET @user = 'MarianoLogin';

INSERT INTO sample\_data (data\_value) VALUES ('New Data');

SET @user = 'AmitLogin';

UPDATE sample\_data SET data\_value = 'Updated Data' WHERE data\_id = 1;

SET @user = 'SuzanLogin';

DELETE FROM sample\_data WHERE data\_id = 1;

-- Transfer and revoke privileges

GRANT SELECT ON university\_sample.sample\_data TO 'SatoshiLogin'@'localhost';

REVOKE SELECT ON university\_sample.sample\_data FROM 'AmitLogin'@'localhost', 'SatoshiLogin'@'localhost' CASCADE;