A. -- Create Database (PostgreSQL doesn't usually require explicit CREATE DATABASE)

-- CREATE DATABASE ItemSupplierDB; -- You'd use this in psql or a similar tool

-- \c ItemSupplierDB; -- Connect to the database in psql

-- Create Tables

CREATE TABLE Item (

itemno INT PRIMARY KEY,

Itemname VARCHAR(20),

quantity INT

);

CREATE TABLE Supplier (

SupplierNo INT PRIMARY KEY,

SupplierName VARCHAR(20),

address VARCHAR(50),

city VARCHAR(20)

);

CREATE TABLE Item\_sup (

item\_no INT,

Supplier\_no INT,

Rate DECIMAL(10, 2),

PRIMARY KEY (item\_no, Supplier\_no),

FOREIGN KEY (item\_no) REFERENCES Item(itemno),

FOREIGN KEY (Supplier\_no) REFERENCES Supplier(SupplierNo)

);

-- Insert sample data

INSERT INTO Item (itemno, Itemname, quantity) VALUES

(101, 'Laptop', 100),

(102, 'Mouse', 500),

(103, 'Keyboard', 300);

INSERT INTO Supplier (SupplierNo, SupplierName, address, city) VALUES

(201, 'TechSupplies Inc', '123 Main St', 'New York'),

(202, 'GlobalTech Ltd', '456 Oak Ave', 'London'),

(203, 'BestDeals Co', '789 Pine Ln', 'Tokyo');

INSERT INTO Item\_sup (item\_no, Supplier\_no, Rate) VALUES

(101, 201, 1200.00),

(101, 202, 1300.00),

(102, 201, 25.00),

(102, 203, 30.00),

(103, 202, 20.00),

(103, 203, 22.00);

-- Trigger a: Before Update on Rate

CREATE OR REPLACE FUNCTION check\_rate\_update\_func()

RETURNS TRIGGER AS $$

BEGIN

IF (NEW.Rate - OLD.Rate) > 2000 THEN

RAISE EXCEPTION 'Error: Rate increase exceeds Rs 2000.';

END IF;

RETURN NEW;

END;

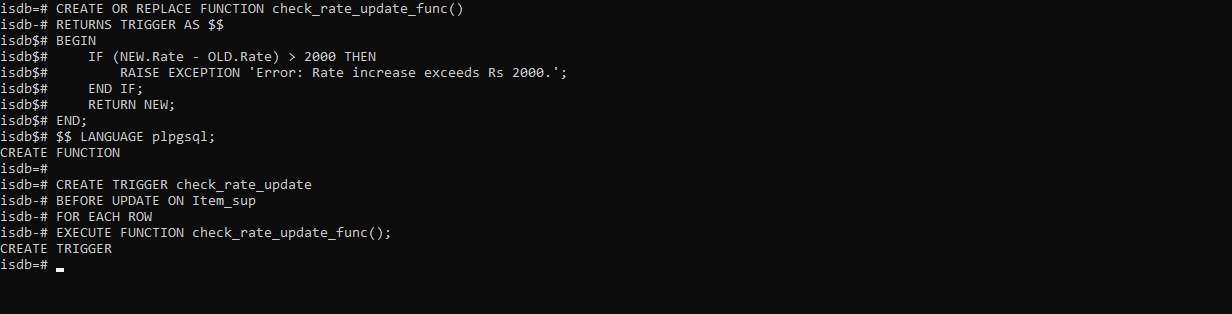
$$ LANGUAGE plpgsql;

CREATE TRIGGER check\_rate\_update

BEFORE UPDATE ON Item\_sup

FOR EACH ROW

EXECUTE FUNCTION check\_rate\_update\_func();



-- Trigger b: Before Insert on Rate

CREATE OR REPLACE FUNCTION check\_rate\_insert\_func()

RETURNS TRIGGER AS $$

BEGIN

IF NEW.Rate = 0 THEN

RAISE EXCEPTION 'Error: Zero rate not allowed.';

END IF;

RETURN NEW;

END;

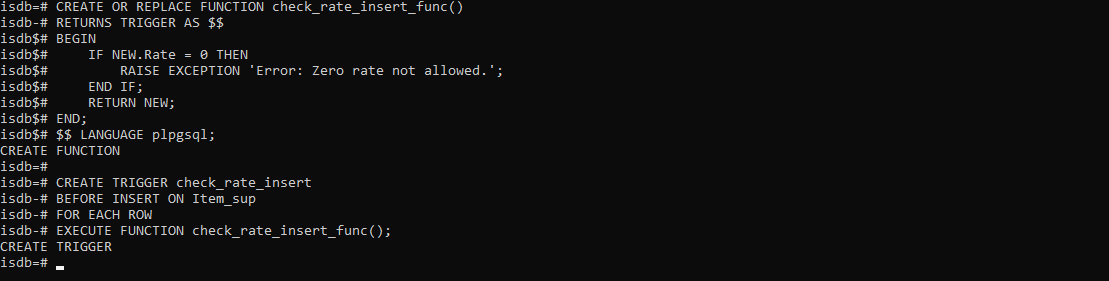
$$ LANGUAGE plpgsql;

CREATE TRIGGER check\_rate\_insert

BEFORE INSERT ON Item\_sup

FOR EACH ROW

EXECUTE FUNCTION check\_rate\_insert\_func();



b.

-- Create Database

-- CREATE DATABASE NewspaperDB;

-- \c NewspaperDB;

-- Create Tables

CREATE TABLE Newspaper (

name VARCHAR(20),

language VARCHAR(20),

Publisher VARCHAR(20),

cost DECIMAL(10, 2), -- Using DECIMAL for money

PRIMARY KEY (name)

);

CREATE TABLE Cities (

pincode VARCHAR(6),

city VARCHAR(20),

state VARCHAR(20),

PRIMARY KEY (pincode)

);

CREATE TABLE Newspaper\_Cities (

name VARCHAR(20),

pincode VARCHAR(6),

daily\_required INTEGER,

PRIMARY KEY (name, pincode),

FOREIGN KEY (name) REFERENCES Newspaper(name),

FOREIGN KEY (pincode) REFERENCES Cities(pincode)

);

-- Insert sample data

INSERT INTO Newspaper (name, language, Publisher, cost) VALUES

('Times of India', 'English', 'Bennett, Coleman & Co.', 5.00),

('Maharashtra Times', 'Marathi', 'Bennett, Coleman & Co.', 4.00),

('The Hindu', 'English', 'Kasturi & Sons Ltd.', 6.00),

('Lokmat', 'Marathi', 'Lokmat Media Pvt Ltd', 3.50);

INSERT INTO Cities (pincode, city, state) VALUES

('400001', 'Mumbai', 'Maharashtra'),

('110001', 'Delhi', 'Delhi'),

('600001', 'Chennai', 'Tamil Nadu'),

('411001', 'Pune', 'Maharashtra');

INSERT INTO Newspaper\_Cities (name, pincode, daily\_required) VALUES

('Times of India', '400001', 1000),

('Maharashtra Times', '400001', 500),

('The Hindu', '600001', 200),

('Lokmat', '411001', 300),

('Times of India', '110001', 800);

-- Trigger a: Check pincode length before insert into Cities

CREATE OR REPLACE FUNCTION check\_pincode\_length\_func()

RETURNS TRIGGER AS $$

BEGIN

IF LENGTH(NEW.pincode) <> 6 THEN

RAISE EXCEPTION 'Error: Pincode must be 6 digits. The provided pincode % has length %', NEW.pincode, LENGTH(NEW.pincode);

END IF;

RETURN NEW;

END;

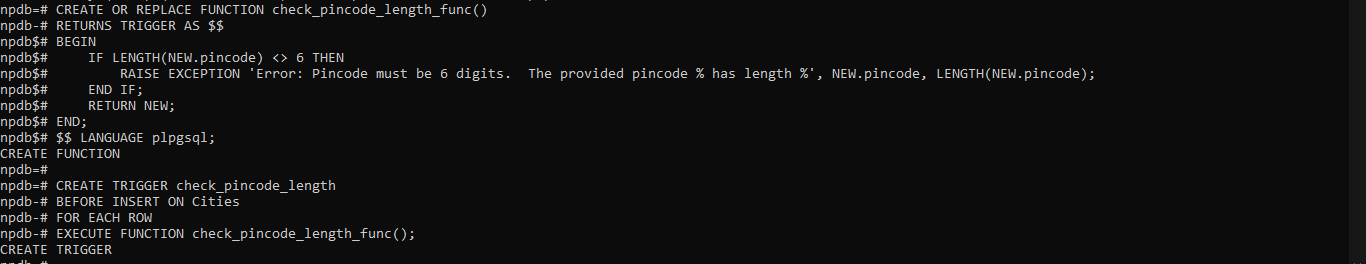
$$ LANGUAGE plpgsql;

CREATE TRIGGER check\_pincode\_length

BEFORE INSERT ON Cities

FOR EACH ROW

EXECUTE FUNCTION check\_pincode\_length\_func();



-- Trigger b: Prevent deleting cities from Maharashtra state

CREATE OR REPLACE FUNCTION prevent\_delete\_maharashtra\_func()

RETURNS TRIGGER AS $$

BEGIN

IF OLD.state = 'Maharashtra' THEN

RAISE EXCEPTION 'Error: Cannot delete cities from Maharashtra.';

END IF;

RETURN OLD;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER prevent\_delete\_maharashtra

BEFORE DELETE ON Cities

FOR EACH ROW

EXECUTE FUNCTION prevent\_delete\_maharashtra\_func();

