Q1.

-- Create the database

CREATE DATABASE railway\_reservation;

-- Connect to the database

-- \c railway\_reservation

-- Create the TRAIN table

CREATE TABLE TRAIN (

TRAIN\_NO INT PRIMARY KEY,

TRAIN\_NAME VARCHAR(20),

DEPART\_TIME TIME,

ARRIVAL\_TIME TIME,

SOURCE\_STN VARCHAR(20),

DEST\_STN VARCHAR(20),

NO\_OF\_RES\_BOGIES INT,

BOGIE\_CAPACITY INT

);

-- Create the PASSENGER table

CREATE TABLE PASSENGER (

PASSENGER\_ID INT PRIMARY KEY,

PASSENGER\_NAME VARCHAR(20),

ADDRESS VARCHAR(30),

AGE INT,

GENDER CHAR(1) -- Changed to CHAR(1)

);

-- Create the TICKET table (M:M relationship)

CREATE TABLE TICKET (

TRAIN\_NO INT,

PASSENGER\_ID INT,

TICKET\_NO INT PRIMARY KEY,

BOGIE\_NO INT,

NO\_OF\_BERTHS INT,

DATE DATE,

TICKET\_AMT DECIMAL(7,2),

STATUS CHAR(1), -- Changed to CHAR(1)

FOREIGN KEY (TRAIN\_NO) REFERENCES TRAIN(TRAIN\_NO),

FOREIGN KEY (PASSENGER\_ID) REFERENCES PASSENGER(PASSENGER\_ID)

);

-- Insert sample data into TRAIN

INSERT INTO TRAIN (TRAIN\_NO, TRAIN\_NAME, DEPART\_TIME, ARRIVAL\_TIME, SOURCE\_STN, DEST\_STN, NO\_OF\_RES\_BOGIES, BOGIE\_CAPACITY) VALUES

(101, 'Rajdhani Exp', '17:00:00', '21:00:00', 'Delhi', 'Mumbai', 10, 72),

(102, 'Shatabdi Exp', '06:00:00', '12:00:00', 'Mumbai', 'Pune', 5, 72),

(103, 'Duronto Exp', '22:00:00', '08:00:00', 'Kolkata', 'Chennai', 12, 72);

-- Insert sample data into PASSENGER

INSERT INTO PASSENGER (PASSENGER\_ID, PASSENGER\_NAME, ADDRESS, AGE, GENDER) VALUES

(1, 'Amit Sharma', '123 MG Road, Delhi', 30, 'M'),

(2, 'Priya Patel', '456 Linking Rd, Mumbai', 25, 'F'),

(3, 'Rahul Singh', '789 Park St, Kolkata', 40, 'M');

-- Insert sample data into TICKET

INSERT INTO TICKET (TRAIN\_NO, PASSENGER\_ID, TICKET\_NO, BOGIE\_NO, NO\_OF\_BERTHS, DATE, TICKET\_AMT, STATUS) VALUES

(101, 1, 1001, 1, 2, '2024-01-20', 2500.00, 'C'),

(102, 2, 1002, 3, 1, '2024-01-20', 1200.00, 'C'),

(101, 3, 1003, 2, 3, '2024-01-21', 3750.00, 'W');

-- a) Trigger to validate train arrival time must be less than train departure time.

CREATE OR REPLACE FUNCTION validate\_train\_times()

RETURNS TRIGGER AS $$

BEGIN

IF NEW.ARRIVAL\_TIME >= NEW.DEPART\_TIME THEN

RAISE EXCEPTION 'Arrival time must be less than departure time.';

END IF;

RETURN NEW;

END;

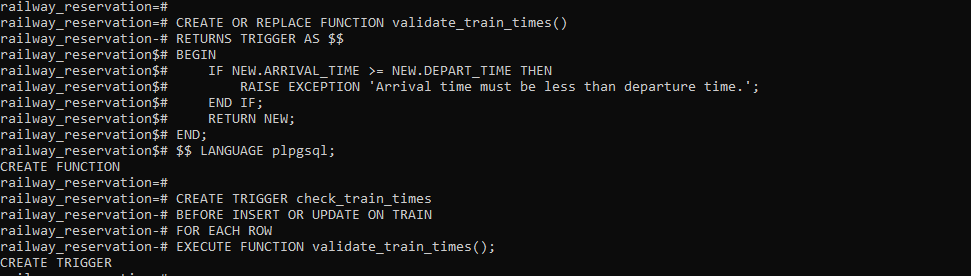
$$ LANGUAGE plpgsql;

CREATE TRIGGER check\_train\_times

BEFORE INSERT OR UPDATE ON TRAIN

FOR EACH ROW

EXECUTE FUNCTION validate\_train\_times();



-- b) Trigger to display message 'Ticket is updated' after changing ticket information.

CREATE OR REPLACE FUNCTION display\_ticket\_updated()

RETURNS TRIGGER AS $$

BEGIN

RAISE NOTICE 'Ticket is updated';

RETURN NEW;

END;

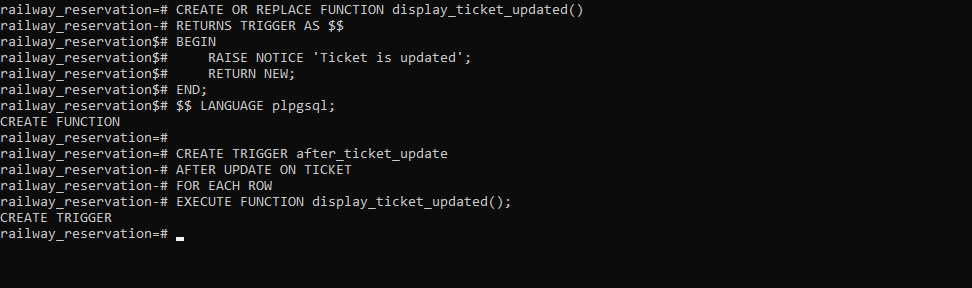
$$ LANGUAGE plpgsql;

CREATE TRIGGER after\_ticket\_update

AFTER UPDATE ON TICKET

FOR EACH ROW

EXECUTE FUNCTION display\_ticket\_updated();



Q.2

-- Create the database

CREATE DATABASE bus\_transport;

-- Connect to the database

-- \c bus\_transport;

-- Create the BUS table

CREATE TABLE BUS (

BUS\_NO INT PRIMARY KEY,

CAPACITY INT NOT NULL,

DEPOT\_NAME VARCHAR(20)

);

-- Create the ROUTE table

CREATE TABLE ROUTE (

ROUTE\_NO INT PRIMARY KEY,

SOURCE CHAR(20),

DESTINATION CHAR(20),

NO\_OF\_STATIONS INT

);

-- Create the DRIVER table

CREATE TABLE DRIVER (

DRIVER\_NO INT PRIMARY KEY,

DRIVER\_NAME CHAR(20),

LICENSE\_NO INT UNIQUE,

ADDRESS CHAR(20),

D\_AGE INT,

SALARY FLOAT

);

-- Create the BUS\_ROUTE table (M:1 relationship - corrected)

CREATE TABLE BUS\_ROUTE (

BUS\_NO INT,

ROUTE\_NO INT,

PRIMARY KEY (BUS\_NO, ROUTE\_NO), -- Added a primary key

FOREIGN KEY (BUS\_NO) REFERENCES BUS(BUS\_NO),

FOREIGN KEY (ROUTE\_NO) REFERENCES ROUTE(ROUTE\_NO)

);

-- Create the BUS\_DRIVER table (M:M relationship)

CREATE TABLE BUS\_DRIVER (

BUS\_NO INT,

DRIVER\_NO INT,

DATE\_OF\_DUTY DATE,

SHIFT VARCHAR(10), -- Changed SHIFT to VARCHAR

PRIMARY KEY (BUS\_NO, DRIVER\_NO, DATE\_OF\_DUTY), -- Added a primary key

FOREIGN KEY (BUS\_NO) REFERENCES BUS(BUS\_NO),

FOREIGN KEY (DRIVER\_NO) REFERENCES DRIVER(DRIVER\_NO)

);

-- Insert sample data into BUS

INSERT INTO BUS (BUS\_NO, CAPACITY, DEPOT\_NAME) VALUES

(101, 50, 'Central Depot'),

(102, 40, 'Eastern Depot'),

(103, 60, 'Central Depot'),

(104, 55, 'Western Depot');

-- Insert sample data into ROUTE

INSERT INTO ROUTE (ROUTE\_NO, SOURCE, DESTINATION, NO\_OF\_STATIONS) VALUES

(1, 'A', 'B', 10),

(2, 'C', 'D', 15),

(3, 'A', 'D', 20);

-- Insert sample data into DRIVER

INSERT INTO DRIVER (DRIVER\_NO, DRIVER\_NAME, LICENSE\_NO, ADDRESS, D\_AGE, SALARY) VALUES

(1001, 'John Smith', 12345, '1 Main St', 35, 50000.00),

(1002, 'Jane Doe', 67890, '2 Oak Ave', 28, 45000.00),

(1003, 'David Lee', 24680, '3 Pine Ln', 52, 60000.00),

(1004, 'Sarah Kim', 13579, '4 Elm Rd', 48, 55000.00);

-- Insert sample data into BUS\_ROUTE

INSERT INTO BUS\_ROUTE (BUS\_NO, ROUTE\_NO) VALUES

(101, 1),

(102, 2),

(103, 1),

(104, 3);

-- Insert sample data into BUS\_DRIVER

INSERT INTO BUS\_DRIVER (BUS\_NO, DRIVER\_NO, DATE\_OF\_DUTY, SHIFT) VALUES

(101, 1001, '2024-01-20', 'Morning'),

(101, 1002, '2024-01-20', 'Evening'),

(102, 1003, '2024-01-21', 'Morning'),

(103, 1001, '2024-01-21', 'Evening'),

(104, 1004, '2024-01-22', 'Morning');

-- a) Trigger to validate driver age before insert or update

CREATE OR REPLACE FUNCTION validate\_driver\_age()

RETURNS TRIGGER AS $$

BEGIN

IF NEW.D\_AGE >= 18 AND NEW.D\_AGE <= 50 THEN

RAISE NOTICE 'Valid entry';

RETURN NEW;

ELSE

RAISE EXCEPTION 'Invalid driver age. Age must be between 18 and 50.';

END IF;

END;

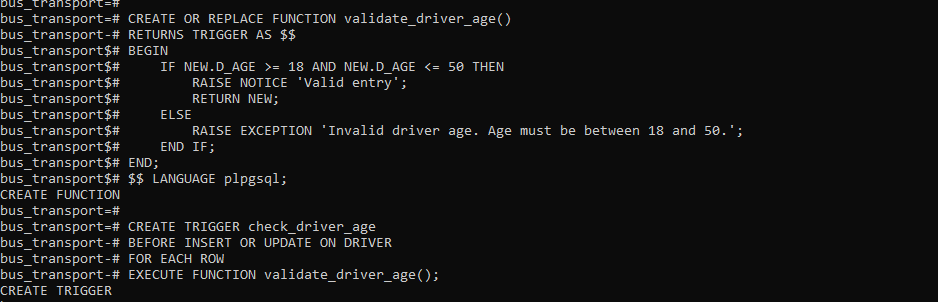
$$ LANGUAGE plpgsql;

CREATE TRIGGER check\_driver\_age

BEFORE INSERT OR UPDATE ON DRIVER

FOR EACH ROW

EXECUTE FUNCTION validate\_driver\_age();



-- b) Trigger to display message after deleting a bus with capacity < 10

CREATE OR REPLACE FUNCTION check\_bus\_capacity\_on\_delete()

RETURNS TRIGGER AS $$

BEGIN

IF OLD.CAPACITY < 10 THEN

RAISE NOTICE 'Bus with capacity % deleted.', OLD.CAPACITY;

END IF;

RETURN OLD;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER after\_delete\_bus

AFTER DELETE ON BUS

FOR EACH ROW

EXECUTE FUNCTION check\_bus\_capacity\_on\_delete();

