Міністерство освіти та науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря
Сікорського» Факультет інформатики та обчислювальної
техніки
Кафедра інформатики і програмної інженерії

Звіт
з дисципліни «Бази даних»
Лабораторна робота №2
"Створення бази даних. Користувачі, ролі, права."

Виконав:
Студент II курсу
гр. ІП-33
Соколов О. В.

Перевірила:
Марченко О. І.

2024

Лабораторна робота № 2.
Створення бази даних. Користувачі, ролі, права.

Мета:

- Створення бази даних шляхом визначення схеми БД
- Навчитися проектувати бази даних, вводити і редагувати структуру таблиць та дані в таблицях
- Вивчити DDL-команди SQL для роботи з таблицями (створення, модифікації та видалення таблиць)
- Вивчити використовувані в SQL засоби для підтримки цілісності даних та їх практичне застосування
- Вивчити основні принципи керування обліковими записами та ролями

Теоретичні основи

Викладені в лекційному матеріалі

Постановка задачі лабораторної роботи No 2

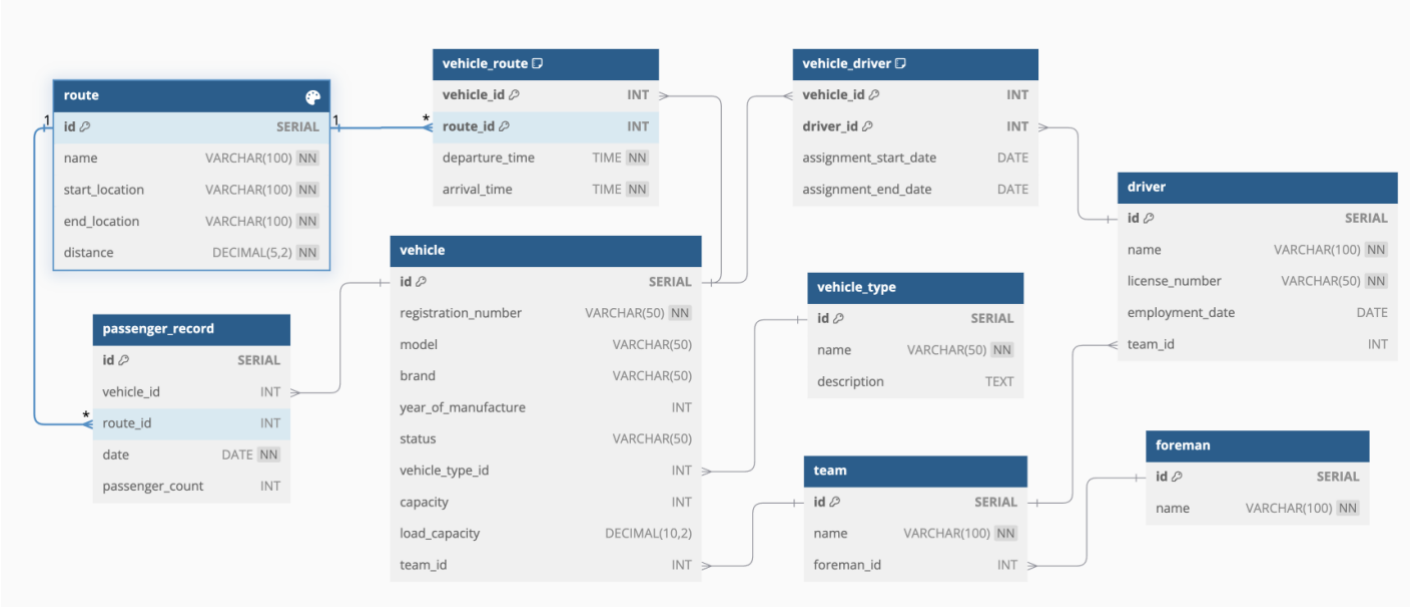
При виконанні лабораторної роботи необхідно виконати наступні дії:

1) Розробити SQL-скрипти для:
    a. створення БД згідно з розробленою в роботі No1 ER-моделлю;
    b. створення таблиць в БД засобами мови SQL. Передбачити наявність обмежень для підтримки цілісності та коректності даних, котрі зберігаються та вводяться;
    c. встановлення зв'язків між таблицями засобами мови SQL;
    d. зміни в структурах таблиць, обмежень засобами мови SQL (до 10 різних за суттю запитів для декількох таблиць (використати DDL- команди SQL));
    e. видалення окремих елементів таблиць/обмежень або самих таблиць засобами мови SQL (до 10 різних за суттю запитів (використати DDL-команди SQL));
    f. визначити декілька (2-3) типів користувачів, котрі будуть працювати з розробленою базою даних. Для кожного користувача визначити набір привілеїв, котрі він буде мати;
    g. для визначених типів користувачів створити відповідні ролі та наділити їх необхідними привілеями;
    h. створити по одному користувачу в базі
2) Згенерувати схему даних засобами СУБД
3) Імпортувати дані в створену БД з використанням засобів СУБД, а не DML SQL.

Програмне забезпечення автопідприємства.

Автопідприємство міста займається організацією пасажирських і вантажних перевезень всередині міста. У віданні підприємства знаходиться автотранспорт різного призначення: автобуси, таксі, маршрутні таксі, інший легковий транспорт, вантажний транспорт, транспорт допоміжного характеру, представлений різними марками. Кожна з перерахованих категорій транспорту має характеристики, властиві тільки цій категорії: наприклад, до характеристик вантажного транспорту відноситься вантажопідйомність, пасажирський транспорт характеризується місткістю і т.д. З плином часу, з одного боку, транспорт старіє і списується (можливо, продається), а з іншого, підприємство поповнюється новим автотранспортом. Підприємство має штат водіїв, закріплених за автомобілями (за одним автомобілем може бути закріплено більше одного водія). Водії об'єднується в бригади, якими керують бригадири. Пасажирський автотранспорт (автобуси, маршрутні таксі) перевозить пасажирів за визначеними маршрутами, за кожним з них закріплені окремі одиниці автотранспорту. Ведеться облік числа перевезених пасажирів, на підставі чого проводиться перерозподіл транспорту з одного маршруту на інший.

ER-модель



Основні множини сутностей

Vehicle (Транспортний засіб):

- **id**: Унікальний ідентифікатор транспортного засобу.
- **registration_number**: Номер реєстрації транспортного засобу.
- **model**: Модель транспортного засобу.
- **brand**: Бренд транспортного засобу.
- **year_of_manufacture**: Рік виготовлення транспортного засобу.
- **status**: Статус транспортного засобу (наприклад, активний, списаний, проданий).
- **vehicle_type_id**: Зовнішній ключ, що посилається на тип транспортного засобу (VehicleType).
- **capacity**: (Опціонально, залежно від типу транспортного засобу) Місткість для пасажирських транспортних засобів.
- **load_capacity**: (Опціонально, залежно від типу транспортного засобу) Вантажопідйомність для вантажних транспортних засобів.
- **team_id**: Зовнішній ключ, що посилається на команду (Team).

VehicleType (Тип транспортного засобу):

- **id**: Унікальний ідентифікатор типу транспортного засобу.
- **name**: Назва типу транспортного засобу (наприклад, автобус, таксі, вантажівка тощо).
- **description**: Короткий опис типу транспортного засобу.

Driver (Водій):

- **id**: Унікальний ідентифікатор водія.
- **name**: Повне ім'я водія.
- **license_number**: Номер водійських прав водія.
- **employment_date**: Дата прийняття водія на роботу.
- **team_id**: Зовнішній ключ, що посилається на команду (Team).

Team (Команда):

- **id**: Унікальний ідентифікатор команди.
- **name**: Назва команди.
- **foreman_id**: Зовнішній ключ, що посилається на бригадира (Foreman).

Foreman (Бригадир):

- **id**: Унікальний ідентифікатор бригадира.
- **name**: Повне ім'я бригадира.

Route (Маршрут):

- **id**: Унікальний ідентифікатор маршруту.
- **name**: Назва або номер маршруту.
- **start_location**: Початкове місце маршруту.
- **end_location**: Кінцеве місце маршруту.
- **distance**: Відстань маршруту в кілометрах.

PassengerRecord (Запис про пасажирів):

- **id**: Унікальний ідентифікатор запису про пасажирів.
- **vehicle_id**: Зовнішній ключ, що посилається на транспортний засіб (Vehicle).
- **route_id**: Зовнішній ключ, що посилається на маршрут (Route).
- **date**: Дата запису.
- **passenger_count**: Кількість перевезених пасажирів.

SQL Скрипти

lab2.sql

```
-- a. створення БД згідно з розробленою в роботі No1 ER-моделлю;

-- CREATE DATABASE transport_company;

-- \c lab2;

-- b. створення таблиць в БД засобами мови SQL. Передбачити наявність
-- обмежень для підтримки цілісності та коректності даних, котрі
-- зберігаються та вводяться;

-- c. встановлення зв’язків між таблицями засобами мови SQL;
CREATE TABLE foreman (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);

CREATE TABLE team (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    foreman_id INT,
    FOREIGN KEY (foreman_id) REFERENCES foreman(id) ON DELETE SET NULL
);

CREATE TABLE driver (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    license_number VARCHAR(50) UNIQUE NOT NULL,
    employment_date DATE,
    team_id INT,
    FOREIGN KEY (team_id) REFERENCES team(id) ON DELETE SET NULL
);

CREATE TABLE vehicle_type (
    id SERIAL PRIMARY KEY,
    name VARCHAR(50) UNIQUE NOT NULL,
    description TEXT
);

CREATE TABLE vehicle (
    id SERIAL PRIMARY KEY,
    registration_number VARCHAR(50) UNIQUE NOT NULL,
    model VARCHAR(50),
    brand VARCHAR(50),
    year_of_manufacture INT CHECK (year_of_manufacture >= 1886 AND
year_of_manufacture <= EXTRACT(YEAR FROM CURRENT_DATE)),
    status VARCHAR(50),
    vehicle_type_id INT,
```

```
    capacity INT CHECK (capacity > 0),
    load_capacity DECIMAL(10, 2) CHECK (load_capacity >= 0),
    team_id INT,
    FOREIGN KEY (vehicle_type_id) REFERENCES vehicle_type(id) ON DELETE SET
NULL,
    FOREIGN KEY (team_id) REFERENCES team(id) ON DELETE SET NULL
);

CREATE TABLE route (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    start_location VARCHAR(100) NOT NULL,
    end_location VARCHAR(100) NOT NULL,
    distance DECIMAL(5, 2) CHECK (distance >= 0) NOT NULL
);

CREATE TABLE passenger_record (
    id SERIAL PRIMARY KEY,
    vehicle_id INT,
    route_id INT,
    date DATE NOT NULL,
    passenger_count INT CHECK (passenger_count >= 0),
    FOREIGN KEY (vehicle_id) REFERENCES vehicle(id) ON DELETE CASCADE,
    FOREIGN KEY (route_id) REFERENCES route(id) ON DELETE CASCADE
);

CREATE TABLE vehicle_driver (
    vehicle_id INT,
    driver_id INT,
    assignment_start_date DATE,
    assignment_end_date DATE,
    PRIMARY KEY (vehicle_id, driver_id),
    FOREIGN KEY (vehicle_id) REFERENCES vehicle(id) ON DELETE CASCADE,
    FOREIGN KEY (driver_id) REFERENCES driver(id) ON DELETE CASCADE
);

CREATE TABLE vehicle_route (
    vehicle_id INT,
    route_id INT,
    departure_time TIME NOT NULL,
    arrival_time TIME NOT NULL,
    PRIMARY KEY (vehicle_id, route_id),
    FOREIGN KEY (vehicle_id) REFERENCES vehicle(id) ON DELETE CASCADE,
    FOREIGN KEY (route_id) REFERENCES route(id) ON DELETE CASCADE
);

-- d. зміни в структурах таблиць, обмежень засобами мови SQL (до 10
-- різних за суттю запитів для декількох таблиць (використати DDL-
-- команди SQL));
```

```
ALTER TABLE driver
ADD COLUMN phone_number VARCHAR(15);

ALTER TABLE vehicle
ALTER COLUMN status TYPE VARCHAR(20);

ALTER TABLE route
ADD CONSTRAINT unique_route_name UNIQUE (name);

ALTER TABLE vehicle_type
DROP COLUMN description;

ALTER TABLE vehicle
ADD CONSTRAINT vehicle_status_check CHECK (status IN ('active', 'maintenance',
'retired'));

ALTER TABLE team
ADD COLUMN team_lead BOOLEAN DEFAULT FALSE;

ALTER TABLE vehicle
DROP CONSTRAINT vehicle_year_of_manufacture_check;

ALTER TABLE vehicle
ADD CONSTRAINT vehicle_year_of_manufacture_check CHECK (year_of_manufacture
>= 1886 AND year_of_manufacture <= EXTRACT(YEAR FROM CURRENT_DATE) + 1);

ALTER TABLE vehicle_route
ADD CONSTRAINT valid_time_check CHECK (departure_time < arrival_time);

-- e. видалення окремих елементів таблиць/обмежень або самих таблиць
-- засобами мови SQL (до 10 різних за суттю запитів (використати
-- DDL-команди SQL));

ALTER TABLE driver
DROP COLUMN phone_number;

ALTER TABLE route
DROP CONSTRAINT unique_route_name;

ALTER TABLE vehicle
DROP CONSTRAINT vehicle_status_check;

-- DROP TABLE vehicle_route;

ALTER TABLE driver
DROP CONSTRAINT driver_team_id_fkey;

-- f. визначити декілька (2-3) типів користувачів, котрі будуть
-- працювати з розробленою базою даних. Для кожного користувача
-- визначити набір привілеїв, котрі він буде мати;
```

```
-- g. для визначених типів користувачів створити відповідні ролі та
-- наділити їх необхідними привілеями;

-- h. створити по одному користувачу в базі даних для кожного типу та
-- присвоїти їм відповідні ролі.

CREATE ROLE admin;
CREATE ROLE manager;
CREATE ROLE driver;

GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON team, driver, vehicle TO manager;
GRANT SELECT ON passenger_record, route TO driver;

CREATE USER admin_user WITH PASSWORD 'admin_password';
CREATE USER manager_user WITH PASSWORD 'manager_password';
CREATE USER driver_user WITH PASSWORD 'driver_password';

GRANT admin TO admin_user;
GRANT manager TO manager_user;
GRANT driver TO driver_user;
```

fill.sql

```
COPY foreman (id, name) FROM '/docker-entrypoint-initdb.d/csvs/foreman.csv'
DELIMITER ',' CSV HEADER;

COPY team (id, name, foreman_id) FROM '/docker-entrypoint-initdb.d/csvs/team.csv'
DELIMITER ',' CSV HEADER;

COPY driver (id, name, license_number, employment_date, team_id) FROM '/docker-
entrypoint-initdb.d/csvs/driver.csv' DELIMITER ',' CSV HEADER;

COPY vehicle_type (id, name) FROM '/docker-entrypoint-initdb.d/csvs/vehicle_type.csv'
DELIMITER ',' CSV HEADER;

COPY vehicle (id, registration_number, model, brand, year_of_manufacture, status,
vehicle_type_id, capacity, load_capacity, team_id) FROM '/docker-entrypoint-
initdb.d/csvs/vehicle.csv' DELIMITER ',' CSV HEADER;

COPY route (id, name, start_location, end_location, distance) FROM '/docker-entrypoint-
initdb.d/csvs/route.csv' DELIMITER ',' CSV HEADER;

COPY passenger_record (id, vehicle_id, route_id, date, passenger_count) FROM '/docker-
entrypoint-initdb.d/csvs/passenger_record.csv' DELIMITER ',' CSV HEADER;

COPY vehicle_driver (vehicle_id, driver_id, assignment_start_date, assignment_end_date)
FROM '/docker-entrypoint-initdb.d/csvs/vehicle_driver.csv' DELIMITER ',' CSV HEADER;
```

```
COPY vehicle_route (vehicle_id, route_id, departure_time, arrival_time) FROM '/docker-entrypoint-initdb.d/csvs/vehicle_route.csv' DELIMITER ',' CSV HEADER;
```

Git-репозиторій додається: https://github.com/sokolovgit/database_course

Висновок: Під час виконання даної лабораторної роботи було реалізовано розробку бази даних для системи автопідприємства. Використовуючи знання з теорії проектування баз даних та DDL-команди SQL, вдалося створити структуру бази даних, що включає таблиці з необхідними обмеженнями для забезпечення цілісності та коректності даних. Крім того, було встановлено зв'язки між таблицями та виконано зміни в їхніх структурах відповідно до поставлених вимог.

Також було визначено ролі та привілеї для різних типів користувачів, які працюватимуть із базою даних, та наділено їх відповідними правами доступу. Це дозволило отримати практичні навички в керуванні обліковими записами та ролями в системах керування базами даних (СУБД).

Лабораторна робота допомогла закріпити теоретичні знання з проектування та управління базами даних, а також ознайомитися з практичними аспектами роботи з SQL для створення і модифікації структур даних, керування користувачами та підтриманням цілісності даних.