Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 2

з дисципліни

«Бази даних»

«Тролейбусне депо»

Варіант 2

Виконав(ла) *ІП-45 Янов. Б.Є.*

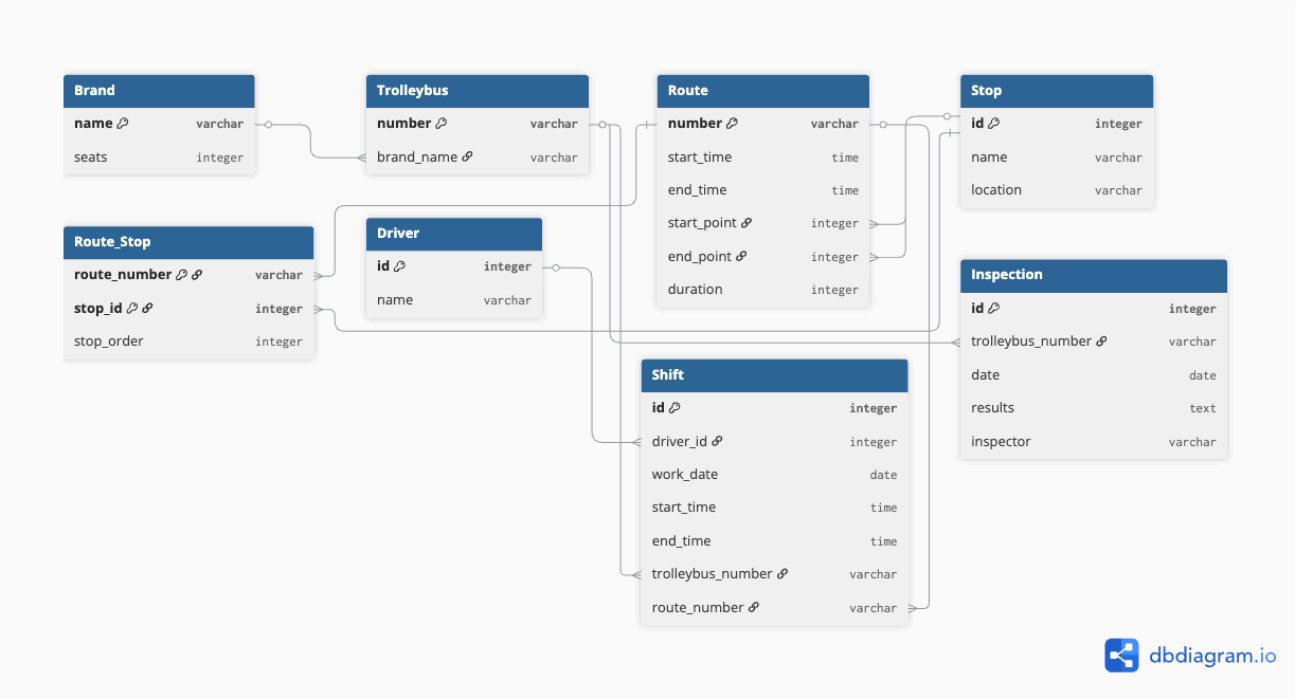
Перевірила Марченко О.І.

Київ 2025

Тролейбусне депо

Програмне забезпечення «Тролейбусне депо». База даних містить інформацію: відомості про водіїв (табельний номер; ПІБ; дата, час початку і закінчення роботи на маршруті), відомості про тролейбуси (номер; марка; кількість пасажирських місць; дата останнього технічного огляду), відомості про маршрути (номер; час початку і кінця роботи тролейбусів; початковий пункт; кінцевий пункт; список зупинок; тривалість маршруту). Кожен водій працює на одному тролейбусі та на одному маршруті. На одному маршруті працює кілька водіїв. Кількість пасажирських місць залежить тільки від марки тролейбуса.

ER-модель



https://dbdiagram.io/d/68d2169c7c85fb9961e53f0f

SQL-скрипти

Для виконання роботи використано PostgreSQL

1. Створення БД згідно з ER-моделлю

```
create database trolley_depot

    with

    encoding = 'UTF8'

    connection limit = -1;
```

2. Створення таблиць в БД засобами мови SQL

```
create table brand (
   name varchar(50) primary key,
   seats integer not null check (seats > 0)
);

create table stop (
   id integer generated by default as identity primary key,
   name varchar(255) not null,
   location varchar(255) not null
);

create table driver (
   id integer primary key,
   name varchar(255) not null
);

create table trolleybus (
   number varchar(20) primary key,
   brand_name varchar(50) not null
);

create table route (
   number varchar(20) primary key,
   start_time TIME not null,
   end_time TIME not null,
   start_point_id integer not null,
   end_point_id integer not null,
   duration_minutes integer not null check (duration_minutes > 0),
   constraint chk_route_time check (end_time > start_time)
);

create table route_stop (
   route_number varchar(20) not null,
   stop_id integer not null,
   stop_order integer not null check (stop_order > 0),
   primary key (route_number, stop_id)
```

```
);

create table shift (
   id integer generated by default as identity primary key,
   driver_id integer not null,
   work_date DATE not null,
   start_time TIME not null,
   end_time TIME not null,
   trolleybus_number varchar(20) not null,
   route_number varchar(20) not null,
   constraint chk_shift_time check (end_time > start_time)
);

create table inspection (
   id integer generated by default as identity primary key,
   trolleybus_number varchar(20) not null,
   inspection_date DATE not null default CURRENT_DATE,
   results TEXT, -- null in case of everything is ok
   inspector varchar(255) not null
);
```

3. Встановлення зв’язків між таблицями засобами мови SQL

```
ALTER table trolleybus
   ADD constraint fk_trolleybus_brand
       foreign key (brand_name) references brand(name)
       on update cascade on delete restrict;

ALTER table route
   ADD constraint fk_route_start_point
       foreign key (start_point_id) references stop(id)
       on delete restrict,
   ADD constraint fk_route_end_point
       foreign key (end_point_id) references stop(id)
       on delete restrict;

ALTER table route_stop
   ADD constraint fk_route_stop_route
```

```
        foreign key (route_number) references route(number)
        on delete cascade,
    ADD constraint fk_route_stop_stop
        foreign key (stop_id) references stop(id)
        on delete cascade;

ALTER table shift
    ADD constraint fk_shift_driver
        foreign key (driver_id) references driver(id)
        on update cascade on delete restrict,
    ADD constraint fk_shift_trolleybus
        foreign key (trolleybus_number) references trolleybus(number)
        on update cascade on delete restrict,
    ADD constraint fk_shift_route
        foreign key (route_number) references route(number)
        on update cascade on delete restrict;

ALTER table inspection
    ADD constraint fk_inspection_trolleybus
        foreign key (trolleybus_number) references trolleybus(number)
        on delete cascade;
```

4. Зміни в структурах таблиць, обмежень засобами мови SQL

```
ALTER table driver
    ADD COLUMN hire_date DATE,
    ALTER COLUMN hire_date
        SET default CURRENT_DATE,
    ADD constraint chk_hire_date
        check (hire_date <= CURRENT_DATE),
    ADD constraint uniq_driver_name
        UNIQUE (name);

ALTER table stop
    ALTER COLUMN location TYPE varchar(100);

ALTER table inspection
    RENAME COLUMN results TO notes;
```

```
ALTER table trolleybus
   ADD COLUMN status varchar(20) not null default 'in_service',
   ADD constraint chk_trolleybus_status
       check (status IN ('in_service', 'under_maintenance',
'decommissioned'));
```

5. видалення окремих елементів таблиць, обмежень засобами мови SQL

```
ALTER table trolleybus
   drop constraint chk_trolleybus_status,
   drop COLUMN status;


ALTER table inspection
   RENAME COLUMN notes TO results;


ALTER table stop
   ALTER COLUMN location TYPE varchar(255);


ALTER table driver
   drop constraint uniq_driver_name,
   drop constraint chk_hire_date,
   drop COLUMN hire_date;
```

6. Визначення основних типів користувачів
   - Аналітик - читає всі дані для звітів
   - Механік - читає дані про тролейбуси, бренди та огляди; може додавати та оновлювати огляди
   - Диспетчер - може читати все; може керувати змінами та маршрутами
7. Створення відповідних ролей

```
create role analyst;
create role mechanic;
create role dispatcher;


grant connect
   on database trolley_depot
   to analyst, mechanic, dispatcher;
grant usage
   on schema public
   to analyst, mechanic, dispatcher;
```

```
grant select
   on all tables in schema public
   to analyst;

alter default privileges
   in schema public
   grant select
       on tables
       to analyst;

grant select
   on trolleybus, brand, inspection
   to mechanic;

grant insert, update
   on inspection
   to mechanic;

grant usage
   on sequence inspection_id_seq
   to mechanic;

grant select
   on all tables in schema public
   to dispatcher;

grant insert, update, delete
   on route, route_stop, shift
   to dispatcher;

grant usage
   on sequence shift_id_seq
   to dispatcher;
```

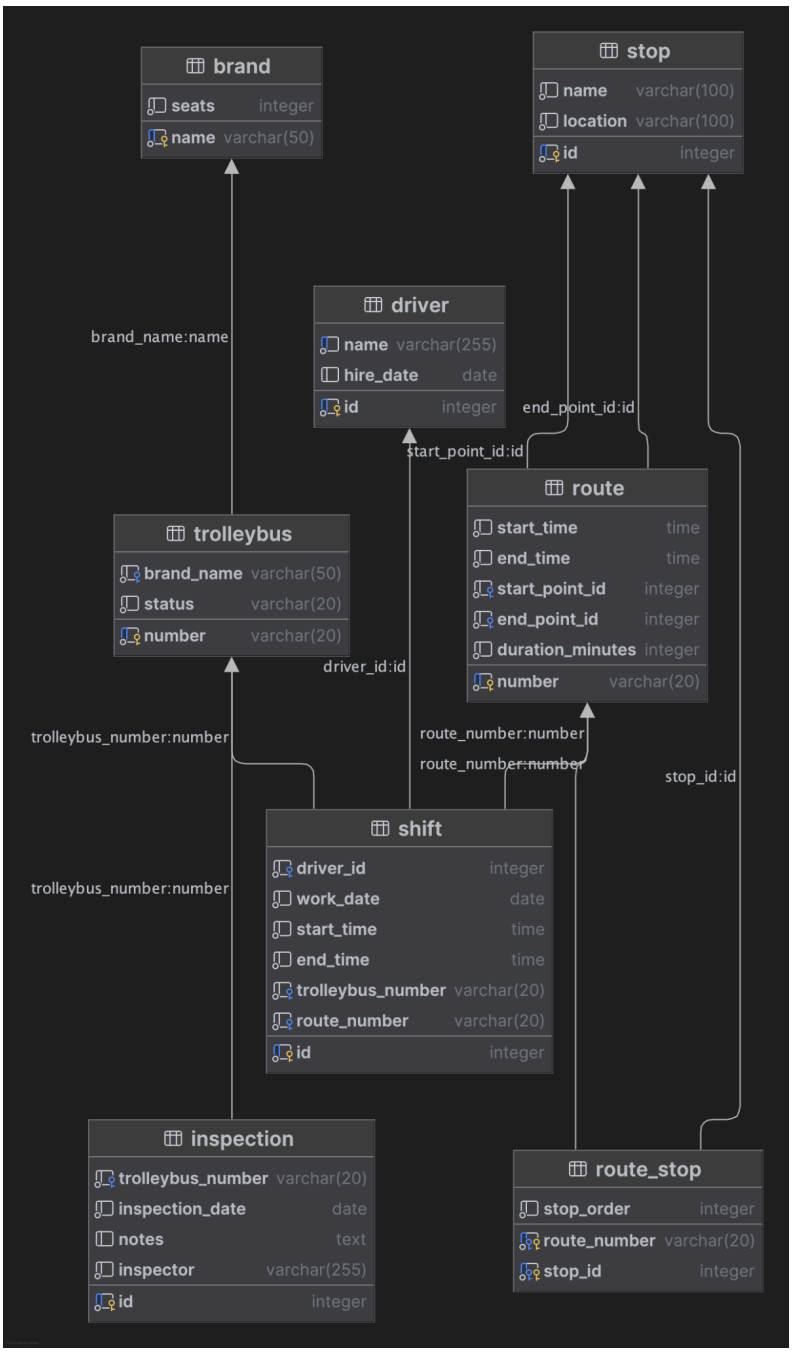
8. Створення користувачів для кожного типу

```
create user user_analyst with password '1234';
```

```
create user user_mechanic with password '1234';
create user user_dispatcher with password '1234';

grant analyst to user_analyst;
grant mechanic to user_mechanic;
grant dispatcher to user_dispatcher;
```

Схема даних



Для генерації використано DataGrip

Імпорт даних (без DML)

Для цього можна використати copy, оскільки bulk insert - особливість mssql.

```
copy brand(name, seats)
from '/path/to/brands.csv'
with (format csv, header true, delimiter ",");
```

Висновок

У ході виконання лабораторної роботи було створено фізичну структуру бази даних «Trolleybus Depot» засобами DDL мови SQL. Було реалізовано таблиці для раніше спроєктованих сутностей, встановлено обмеження цілісності (PRIMARY KEY, FOREIGN KEY, CHECK) та налаштовано зв'язки між ними. Отримана база даних дозволяє зберігати та обробляти інформацію про марки тролейбусів, індивідуальні тролейбуси, маршрути, зупинки, водіїв, графіки змін та інспекції. Також було налаштовано систему безпеки через створення користувачів та ролей (role_analyst, role_mechanic, role_dispatcher), що забезпечує розмежування прав доступу до даних.