

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра МО ЭВМ**

**ОТЧЕТ**  
**по лабораторной работе №2**  
**по дисциплине «Базы данных»**  
**Тема: Реализация базы данных в СУБД PostgreSQL**

Студент гр. 3341

Мальцев К.Л.

Преподаватель

Заславский М.М.

Санкт-Петербург

2025

## Задание

Целью данной лабораторной работы является развёртывание базы данных в СУБД PostgreSQL

## Задание

Необходимо развернуть PostgreSQL локально:

- Написать запросы для создания таблиц из предыдущей лабораторной работы
- Заполнить тестовыми данными: 5-10 строк на каждую таблицу, обязательно наличие связи между ними, данные приближены к реальности.
- Написать запросы к БД, отвечающие на вопросы из предыдущей лабораторной работы
- Исходный код выложить на [www.db-fiddle.com](http://www.db-fiddle.com) для проверки работоспособности
- Исходный код в виде .sql файла загрузить в виде PR в репо

В отчете описать:

- Цель
- Текст задания в соответствии с вариантом
- Скриншоты работы с СУБД PostgreSQL (psql / DBeaver / Datagrip, ...)
- Скриншоты на каждый запрос (или группу запросов) на изменение/таблицы с выводом результатов (ответ)
- Исходный код в приложении
- Ссылку на исходный код [www.db-fiddle.com](http://www.db-fiddle.com) в приложении
- Ссылка на PR в приложении
- Вывод

## Вариант 9

- Пусть требуется создать программную систему, предназначенную для диспетчера автобусного парка. Такая система должна обеспечивать хранение сведений о водителях, о маршрутах и характеристиках авто-

бусов. Каждый водитель характеризуется паспортными данными, классом, стажем работы и окладом, причем оклад зависит от класса и стажа работы. Маршрут автобуса характеризуется номером маршрута, названием начального и конечного пункта движения, временем начала и конца движения, интервалом движения и протяженностью в минутах (время движения от кольца до кольца). Характеристиками автобуса являются: номер государственной регистрации автобуса, его тип и вместимость, причем вместимость автобуса зависит от его типа. Каждый водитель закреплен за отдельным автобусом и работает на определенном маршруте, но в случае поломки своего автобуса или болезни другого водителя может пересест на другую машину. В базе должен храниться график работы водителей. Необходимо предусмотреть возможность корректировки БД в случаях поступления на работу нового водителя, списания старого автобуса, введения нового маршрута или изменения старого и т.п. Диспетчеру автопарка могут потребоваться следующие сведения:

- Список водителей, работающих на определенном маршруте с указанием графика их работы?
- Какие автобусы обслуживают данный маршрут?
- Какие маршруты начинаются или заканчиваются в пункте с заданным названием?
- Когда начинается и заканчивается движение автобусов на каждом маршруте?
- Какова протяженность определенного маршрута?
- Какова общая протяженность маршрутов, обслуживаемых автопарком?
- Какие автобусы не вышли на линию, и по какой причине (неисправность, отсутствие водителя)?

В лабораторной работе № 1 была построена реляционная модель:

```
erDiagram
    Driver ||--o{ WorkSchedule : "has"
    Driver ||--}| SalaryRule : "has"
    DriverBusRoute ||--o{ Driver : "has"
    DriverBusRoute ||--}| SalaryRule : "has"
    DriverBusRoute ||--}| Route : "has"
    Bus ||--}| BusType : "has"
    Bus ||--}| Issues : "has"
```

The diagram illustrates the following entities and their attributes:

- Driver**: driver\_id, passport\_data, class, work\_experience, salary\_rule\_id
- WorkSchedule**: driver\_id, working\_days
- SalaryRule**: salary\_rule\_id, class, work\_experience, salary
- DriverBusRoute**: driver\_id, bus\_id, route\_id
- Bus**: bus\_id, state\_registration\_number, bus\_type\_id
- BusType**: bus\_type\_id, type\_name, capacity
- Route**: route\_id, number, start\_point, end\_point, start\_time, end\_time, movement\_interval, travel\_time
- Issues**: bus\_id, reason

The relationships are defined as follows:

- Driver** to **WorkSchedule**: One-to-many relationship with a mandatory one end and an optional many end.
- Driver** to **SalaryRule**: One-to-many relationship with a mandatory one end and a mandatory many end.
- DriverBusRoute** to **Driver**: One-to-many relationship with a mandatory one end and an optional many end.
- DriverBusRoute** to **SalaryRule**: One-to-many relationship with a mandatory one end and a mandatory many end.
- DriverBusRoute** to **Route**: One-to-many relationship with a mandatory one end and a mandatory many end.
- Bus** to **BusType**: One-to-many relationship with a mandatory one end and a mandatory many end.
- Bus** to **Issues**: One-to-many relationship with a mandatory one end and an optional many end.

В качестве инструмента разработки воспользуемся графическим интерфейсом pgAdmin4. Создадим новую базу данных `bus_fleet` для нашего задания с помощью запроса *CREATE DATABASE music\_band*

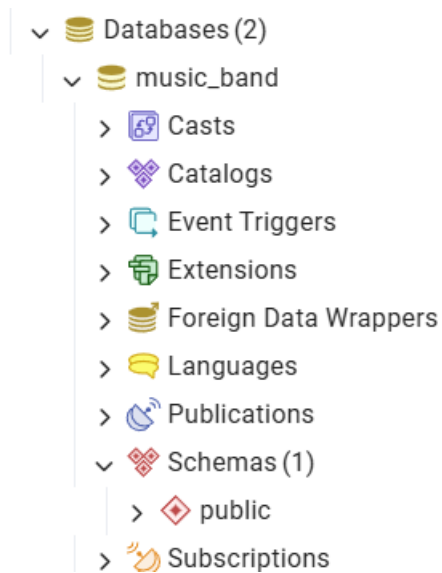


Рисунок № 2 – Результат *CREATE DATABASE*

С помощью *CREATE TABLE* созданы таблицы *BusType*, *SalaryRule*, *Driver*, *Bus*, *Issue*, *Route*, *WorkSchedule*.

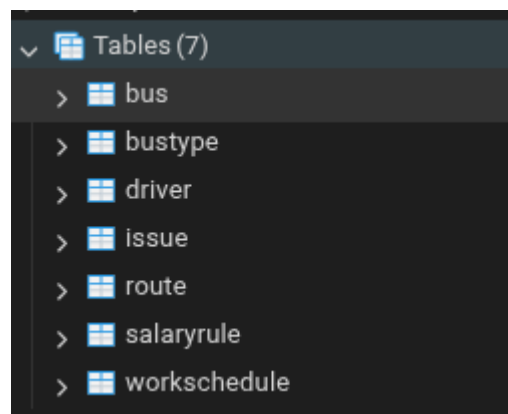


Рисунок № 3 – Результат создания таблиц

Заполним таблицы произвольными данными с помощью *INSERT INTO VALUES*

Data Output Messages Notifications				
<div> <div>≡+</div> <div>📄</div> <div>▼</div> <div>📋</div> <div>▼</div> <div>🗑️</div> <div>🗄️</div> <div>⬇️</div> <div>📈</div> <div>SQL</div> </div>				
	salary_rule_id [PK] integer	worker_class integer	experience_from integer	salary integer
1	1	1	0	25000
2	2	2	0	40000
3	3	3	0	55000
4	4	1	3	30000
5	5	2	3	45000
6	6	3	3	60000
7	7	1	6	35000
8	8	2	6	50000
9	9	3	6	65000

Data Output Messages Notifications				
<div> <div>≡+</div> <div>📄</div> <div>▼</div> <div>📋</div> <div>▼</div> <div>🗑️</div> <div>🗄️</div> <div>⬇️</div> <div>📈</div> <div>SQL</div> </div>				
	salary_rule_id [PK] integer	worker_class integer	experience_from integer	salary integer
1	1	1	0	25000
2	2	2	0	40000
3	3	3	0	55000
4	4	1	3	30000
5	5	2	3	45000
6	6	3	3	60000
7	7	1	6	35000
8	8	2	6	50000
9	9	3	6	65000

Data Output Messages Notifications				
	driver_id [PK] integer	passport_data character varying (256)	worker_class integer	experience integer
1	1	AB1234561	1	1
2	2	AB1234562	1	2
3	3	AB1234563	1	3
4	4	AB1234564	1	4
5	5	AB1234565	1	5
6	6	AB1234566	2	1
7	7	AB1234567	2	2
8	8	AB1234568	2	3
9	9	AB1234569	2	4
10	10	AB1234570	2	5
11	11	AB1234571	2	6
12	12	AB1234572	1	1
13	13	AB1234573	1	2
14	14	AB1234574	1	3
15	15	AB1234575	1	4
16	16	AB1234576	2	1
17	17	AB1234577	2	2
18	18	AB1234578	2	3
19	19	AB1234579	2	4
20	20	AB1234580	2	5

Data Output Messages Notifications			
	SQL		
	bus_id [PK] integer	bus_type_id integer	registration_number character varying (16)
1	1	1	AB1001
2	2	1	AB1002
3	3	1	AB1003
4	4	1	AB1004
5	5	1	AB1005
6	6	1	AB1006
7	7	1	AB1007
8	8	1	AB1008
9	9	1	AB1009
10	10	1	AB1010
11	11	2	CD2001
12	12	2	CD2002
13	13	2	CD2003
14	14	2	CD2004
15	15	2	CD2005
16	16	2	CD2006
17	17	2	CD2007
18	18	2	CD2008
19	19	2	CD2009
20	20	2	CD2010
21	21	3	EF3001
22	22	3	EF3002
23	23	3	EF3003
24	24	3	EF3004
25	25	3	EF3005
26	26	3	EF3006
27	27	3	EF3007
28	28	3	EF3008
29	29	3	EF3009
30	30	3	EF3010



Data Output

Messages

Notifications

≡+

▼

▼

SQL

	<div>issue_id</div> <div>[PK] integer </div>	<div>bus_id</div> <div>integer </div>	<div>reason</div> <div>character varying (1024) </div>	
1	1	1	Engine failure	
2	2	5	Flat tire	
3	3	12	Broken door	
4	4	22	Battery issues	
5	5	30	Regular maintenance	

Data Output

Messages

Notifications

≡

SQL

Showing rows: 1 to 10

Page

	route_id [PK] integer	number integer	start_point character varying (256)	end_point character varying (256)	start_time time without time zone	end_time time without time zone	movement_interval time without time zone	travel_time time without time zone
1	1	1	Central Station	Airport	06:00:00	22:00:00	00:15:00	01:00:00
2	2	2	Downtown	Stadium	07:00:00	20:00:00	00:10:00	00:45:00
3	3	3	Suburb	Mall	05:30:00	23:00:00	00:20:00	01:15:00
4	4	4	East Park	West End	06:30:00	21:30:00	00:12:00	00:50:00
5	5	5	North Square	South Market	06:00:00	22:30:00	00:18:00	01:05:00
6	6	6	Museum	Library	08:00:00	20:00:00	00:20:00	00:40:00
7	7	7	Hospital	City Hall	06:15:00	22:00:00	00:15:00	00:55:00
8	8	8	Stadium	Central Park	07:00:00	21:00:00	00:10:00	00:35:00
9	9	9	Old Town	New Town	06:00:00	22:00:00	00:15:00	01:10:00
10	10	10	West Station	East Station	05:45:00	23:00:00	00:25:00	01:20:00

Data Output Messages Notifications					
	work_schedule_id [PK] integer	driver_id integer	bus_id integer	route_id integer	work_date date
1	1	1	1	1	2025-10-20
2	2	2	2	1	2025-10-20
3	3	3	3	2	2025-10-20
4	4	4	4	3	2025-10-20
5	5	5	5	4	2025-10-20
6	6	6	6	5	2025-10-20
7	7	7	7	6	2025-10-20
8	8	8	8	7	2025-10-20
9	9	9	9	8	2025-10-20
10	10	10	10	9	2025-10-20
11	11	11	11	10	2025-10-20
12	12	12	12	1	2025-10-20
13	13	13	13	2	2025-10-20
14	14	14	14	3	2025-10-20
15	15	15	15	4	2025-10-20
16	16	16	16	5	2025-10-20
17	17	17	17	6	2025-10-20
18	18	18	18	7	2025-10-20
19	19	19	19	8	2025-10-20
20	20	20	20	9	2025-10-20
21	21	1	21	10	2025-10-21
22	22	2	22	1	2025-10-21
23	23	3	23	2	2025-10-21
24	24	4	24	3	2025-10-21
25	25	5	25	4	2025-10-21
26	26	6	26	5	2025-10-21
27	27	7	27	6	2025-10-21
28	28	8	28	7	2025-10-21
29	29	9	29	8	2025-10-21
30	30	10	30	9	2025-10-21

Реализация запросов для ответов на вопросы из предыдущей лабораторной работы.

```
-- List of drivers working on a specific route along with their work schedule?
-- 8
SELECT D.*, W.*
FROM Driver AS D
JOIN WorkSchedule AS W
ON D.driver_id = W.driver_id
WHERE W.route_id = 10;

-- Which buses serve the given route?
-- 9
SELECT B.*, W.*
FROM Bus as B
```

```

JOIN WorkSchedule AS W
ON B.bus_id = W.bus_id
WHERE W.route_id = 10;

-- Which routes start or end at a location with the specified name?
-- 10
SELECT *
FROM Route as R
WHERE (R.start_point = 'Stadium' or R.end_point = 'Stadium');

-- When does bus movement start and end on each route?
-- 11
SELECT number, start_time, end_time
FROM Route;

-- What is the travel time for a particular route?
-- 12
SELECT number, travel_time
FROM Route;

-- What is the total travel time of all routes served by the fleet?
-- 13
SELECT SUM(travel_time)
FROM Route;

-- Which buses did not go on line and for what reason (malfunction, absence of driver)?
-- 14
SELECT W.*, I.*
FROM WorkSchedule as W
JOIN Issue as I
ON W.bus_id = I.bus_id;

```

Query #8

drive_r_id	passport_data	worker_class	experience	work_schedule_id	drive_r_id	bus_id	route_id	work_date
1	AB1234561	1	1	21	1	21	10	2025-10-21
11	AB1234571	2	6	11	11	11	10	2025-10-20

Query #9

bus_id	bus_type_id	registration_number	work_schedule_id	driver_id	bus_id	route_id	work_date
11	2	CD2001	11	11	11	10	2025-10-20
21	3	EF3001	21	1	21	10	2025-10-21

Query #10

route_id	number	start_point	end_point	start_time	end_time	movement_interval	travel_time
2	2	Downtown	Stadium	07:00:00	20:00:00	00:10:00	00:45:00
8	8	Stadium	Central Park	07:00:00	21:00:00	00:10:00	00:35:00

Query #11

number	start_time	end_time
1	06:00:00	22:00:00
2	07:00:00	20:00:00
3	05:30:00	23:00:00
4	06:30:00	21:30:00
5	06:00:00	22:30:00
6	08:00:00	20:00:00
7	06:15:00	22:00:00
8	07:00:00	21:00:00
9	06:00:00	22:00:00

10	05:45:00	23:00:00
----	----------	----------

Query #12

number	travel_time
1	01:00:00
2	00:45:00
3	01:15:00
4	00:50:00
5	01:05:00
6	00:40:00
7	00:55:00
8	00:35:00
9	01:10:00
10	01:20:00

Query #13

sum
09:35:00

Query #14

work_schedule_id	driver_id	bus_id	route_id	work_date	issue_id	bus_id	reason
------------------	-----------	--------	----------	-----------	----------	--------	--------

1	1	1	1	2025-10-20	1	1	Engine failure
5	5	5	4	2025-10-20	2	5	Flat tire
12	12	12	1	2025-10-20	3	12	Broken door
22	2	22	1	2025-10-21	4	22	Battery issues
30	10	30	9	2025-10-21	5	30	Regular maintenance

## **Вывод**

В результате выполнения лабораторной работы была развёрнута база данных в СУБД PostgreSQL с помощью pgAdmin4, соответствующая структуре базе данных, спроектированной в предыдущей лабораторной работе. Реализованы необходимые запросы для реализованных таблиц.

## ПРИЛОЖЕНИЕ А

### ИСХОДНЫЙ КОД ПРОГРАММЫ

Pull Request: <https://github.com/moevm/sql-2025-3341/pull/21>

DB Fiddle: <https://www.db-fiddle.com/f/uTrxZpPqMJ77KAvXDfwXk/42>

Файл lab2.sql

```
-- Schema SQL

CREATE TABLE BusType (
    bus_type_id SERIAL PRIMARY KEY,
    type_name VARCHAR(256) NOT NULL,
    capacity INT NOT NULL
);

CREATE TABLE SalaryRule (
    salary_rule_id SERIAL PRIMARY KEY,
    worker_class INT NOT NULL,
    experience_from INT NOT NULL,
    salary INT NOT NULL
);

CREATE TABLE Driver (
    driver_id SERIAL PRIMARY KEY,
    passport_data VARCHAR(256) NOT NULL,
    worker_class INT NOT NULL,
    experience INT NOT NULL
);

CREATE TABLE Bus (
    bus_id SERIAL PRIMARY KEY,
    bus_type_id INT NOT NULL,
```



```

registration_number VARCHAR(16) NOT NULL,
FOREIGN KEY (bus_type_id) REFERENCES BusType (bus_type_id)
);

CREATE TABLE Issue (
issue_id SERIAL PRIMARY KEY,
bus_id INT NOT NULL,
reason VARCHAR(1024) NOT NULL,
FOREIGN KEY (bus_id) REFERENCES Bus (bus_id) ON DELETE CASCADE
);

CREATE TABLE Route (
route_id SERIAL PRIMARY KEY,
number INT NOT NULL,
start_point VARCHAR(256) NOT NULL,
end_point VARCHAR(256) NOT NULL,
start_time TIME NOT NULL,
end_time TIME NOT NULL,
movement_interval TIME NOT NULL,
travel_time TIME NOT NULL
);

CREATE TABLE WorkSchedule (
work_schedule_id SERIAL PRIMARY KEY,
driver_id INT,
bus_id INT,
route_id INT NOT NULL,
work_date DATE NOT NULL,
FOREIGN KEY (driver_id) REFERENCES Driver (driver_id) ON DELETE SET
NULL,

```

```
FOREIGN KEY (bus_id) REFERENCES Bus (bus_id) ON DELETE SET NULL,  
FOREIGN KEY (route_id) REFERENCES Route (route_id) ON DELETE CASCADE  
);
```

```
-- BusType data
```

```
INSERT INTO BusType (type_name, capacity) VALUES  
( 'Standard Bus', 50),  
( 'Double Decker', 80),  
( 'Mini Bus', 20);
```

```
-- SalaryRule data
```

```
INSERT INTO SalaryRule (worker_class, experience_from, salary) VALUES  
(1, 0, 25000), (2, 0, 40000), (3, 0, 55000),  
(1, 3, 30000), (2, 3, 45000), (3, 3, 60000),  
(1, 6, 35000), (2, 6, 50000), (3, 6, 65000);
```

```
-- Driver data
```

```
INSERT INTO Driver (passport_data, worker_class, experience) VALUES  
( 'AB1234561', 1, 1),  
( 'AB1234562', 1, 2),  
( 'AB1234563', 1, 3),  
( 'AB1234564', 1, 4),  
( 'AB1234565', 1, 5),  
( 'AB1234566', 2, 1),  
( 'AB1234567', 2, 2),  
( 'AB1234568', 2, 3),  
( 'AB1234569', 2, 4),  
( 'AB1234570', 2, 5),  
( 'AB1234571', 2, 6),  
( 'AB1234572', 1, 1),
```

```

('AB1234573', 1, 2),
('AB1234574', 1, 3),
('AB1234575', 1, 4),
('AB1234576', 2, 1),
('AB1234577', 2, 2),
('AB1234578', 2, 3),
('AB1234579', 2, 4),
('AB1234580', 2, 5);

```

-- Bus data

```

INSERT INTO Bus (bus_type_id, registration_number) VALUES

(1, 'AB1001'), (1, 'AB1002'), (1, 'AB1003'), (1, 'AB1004'), (1,
'AB1005'),

(1, 'AB1006'), (1, 'AB1007'), (1, 'AB1008'), (1, 'AB1009'), (1,
'AB1010'),

(2, 'CD2001'), (2, 'CD2002'), (2, 'CD2003'), (2, 'CD2004'), (2,
'CD2005'),

(2, 'CD2006'), (2, 'CD2007'), (2, 'CD2008'), (2, 'CD2009'), (2,
'CD2010'),

(3, 'EF3001'), (3, 'EF3002'), (3, 'EF3003'), (3, 'EF3004'), (3,
'EF3005'),

(3, 'EF3006'), (3, 'EF3007'), (3, 'EF3008'), (3, 'EF3009'), (3,
'EF3010');

```

-- Issue data

```

INSERT INTO Issue (bus_id, reason) VALUES

(1, 'Engine failure'),

(5, 'Flat tire'),

(12, 'Broken door'),

(22, 'Battery issues'),

(30, 'Regular maintenance');

```

-- Route data

```

INSERT INTO Route (number, start_point, end_point, start_time,
end_time, movement_interval, travel_time) VALUES

(1, 'Central Station', 'Airport', '06:00:00', '22:00:00', '00:15:00',
'01:00:00'),

(2, 'Downtown', 'Stadium', '07:00:00', '20:00:00', '00:10:00',
'00:45:00'),

(3, 'Suburb', 'Mall', '05:30:00', '23:00:00', '00:20:00', '01:15:00'),

(4, 'East Park', 'West End', '06:30:00', '21:30:00', '00:12:00',
'00:50:00'),

(5, 'North Square', 'South Market', '06:00:00', '22:30:00', '00:18:00',
'01:05:00'),

(6, 'Museum', 'Library', '08:00:00', '20:00:00', '00:20:00',
'00:40:00'),

(7, 'Hospital', 'City Hall', '06:15:00', '22:00:00', '00:15:00',
'00:55:00'),

(8, 'Stadium', 'Central Park', '07:00:00', '21:00:00', '00:10:00',
'00:35:00'),

(9, 'Old Town', 'New Town', '06:00:00', '22:00:00', '00:15:00',
'01:10:00'),

(10, 'West Station', 'East Station', '05:45:00', '23:00:00',
'00:25:00', '01:20:00');

```

-- WorkSchedule data

```

INSERT INTO WorkSchedule (driver_id, bus_id, route_id, work_date)
VALUES

(1, 1, 1, '2025-10-20'),

(2, 2, 1, '2025-10-20'),

(3, 3, 2, '2025-10-20'),

(4, 4, 3, '2025-10-20'),

(5, 5, 4, '2025-10-20'),

(6, 6, 5, '2025-10-20'),

(7, 7, 6, '2025-10-20'),

(8, 8, 7, '2025-10-20'),

(9, 9, 8, '2025-10-20'),

(10, 10, 9, '2025-10-20'),

```

```

(11, 11, 10, '2025-10-20'),
(12, 12, 1, '2025-10-20'),
(13, 13, 2, '2025-10-20'),
(14, 14, 3, '2025-10-20'),
(15, 15, 4, '2025-10-20'),
(16, 16, 5, '2025-10-20'),
(17, 17, 6, '2025-10-20'),
(18, 18, 7, '2025-10-20'),
(19, 19, 8, '2025-10-20'),
(20, 20, 9, '2025-10-20'),
(1, 21, 10, '2025-10-21'),
(2, 22, 1, '2025-10-21'),
(3, 23, 2, '2025-10-21'),
(4, 24, 3, '2025-10-21'),
(5, 25, 4, '2025-10-21'),
(6, 26, 5, '2025-10-21'),
(7, 27, 6, '2025-10-21'),
(8, 28, 7, '2025-10-21'),
(9, 29, 8, '2025-10-21'),
(10, 30, 9, '2025-10-21');

```

```
-- Query SQL
```

```
-- Display relations
```

```
SELECT * FROM BusType; -- 1
```

```
SELECT * FROM SalaryRule; -- 2
```

```
SELECT * FROM Driver; -- 3
```

```
SELECT * FROM Bus; -- 4
```

```
SELECT * FROM Issue; -- 5
```

```
SELECT * FROM Route; -- 6
```

```
SELECT * FROM WorkSchedule; -- 7
```

```
-- List of drivers working on a specific route along with their work  
schedule?
```

```
-- 8
```

```
SELECT D.*, W.*  
  
FROM Driver AS D  
  
JOIN WorkSchedule AS W  
  
ON D.driver_id = W.driver_id  
  
WHERE W.route_id = 10;
```

```
-- Which buses serve the given route?
```

```
-- 9
```

```
SELECT B.*, W.*  
  
FROM Bus as B  
  
JOIN WorkSchedule AS W  
  
ON B.bus_id = W.bus_id  
  
WHERE W.route_id = 10;
```

```
-- Which routes start or end at a location with the specified name?
```

```
-- 10
```

```
SELECT *  
  
FROM Route as R  
  
WHERE (R.start_point = 'Stadium' or R.end_point = 'Stadium');
```

```
-- When does bus movement start and end on each route?
```

```
-- 11
```

```
SELECT number, start_time, end_time  
  
FROM Route;
```

```

-- What is the travel time for a particular route?

-- 12

SELECT number, travel_time

FROM Route;


-- What is the total travel time of all routes served by the fleet?

-- 13

SELECT SUM(travel_time)

FROM Route;


-- Which buses did not go on line and for what reason (malfunction,
absence of driver)?

-- 14

SELECT W.*, I.*

FROM WorkSchedule as W

JOIN Issue as I

ON W.bus_id = I.bus_id;


-- Add a new driver to the database

-- 15

INSERT INTO Driver (passport_data, worker_class, experience)

VALUES ('AB1234581', 1, 0);


-- Remove a bus from the database (does NOT remove schedule entries
automatically)

-- 16

DELETE FROM Bus

WHERE bus_id = 15;


-- Add a new route to the database

-- 17

```

```

INSERT INTO Route (number, start_point, end_point, start_time,
end_time, movement_interval, travel_time)

VALUES (11, 'New Start', 'New End', '06:00:00', '22:00:00', '00:15:00',
'01:00:00');

-- Update an existing route's times
-- 18

UPDATE Route

SET start_time = '07:00:00', end_time = '21:00:00'

WHERE route_id = 3;

-- Update driver's experience
-- 19

UPDATE Driver

SET experience = 3

WHERE driver_id = 1;

-- Display relations

SELECT * FROM BusType; -- 20

SELECT * FROM SalaryRule; -- 21

SELECT * FROM Driver; -- 22

SELECT * FROM Bus; -- 23

SELECT * FROM Issue; -- 24

SELECT * FROM Route; -- 25

SELECT * FROM WorkSchedule; -- 26

```