

Міністерство освіти і науки України

НТУУ «Київський політехнічний інститут»

Фізико-технічний інститут

Розрахунково-графічна робота

Бази даних та інформаційні системи

Варіант - 9

**Виконав:**

Студент 4 курсу ФТІ

групи ФІ-41

Льовкін М.С.

Київ 2017

Информационная система диспетчера автопарка.

БД должна обеспечивать хранение сведений о водителях, о маршрутах и характеристиках автобусов.

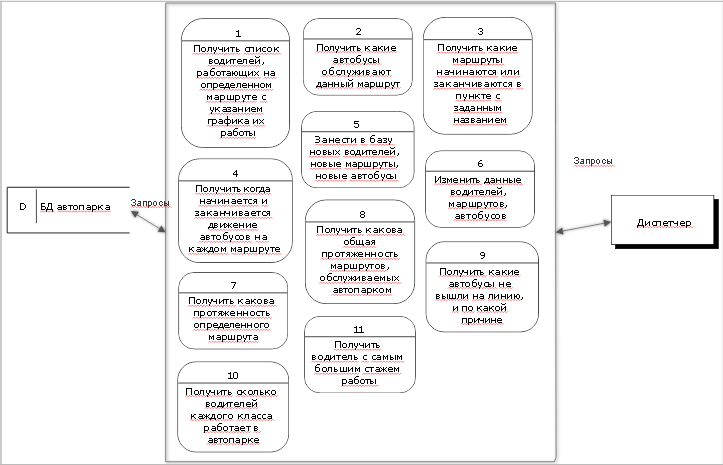
Каждый водитель характеризуется паспортными данными, классом, стажем работы и окладом, причем оклад зависит от класса и стажа работы. Маршрут автобуса характеризуется номером маршрута, названием начального и конечного пункта движения, временем начала и конца движения, интервалом движения и протяженностью 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 пересесть 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:

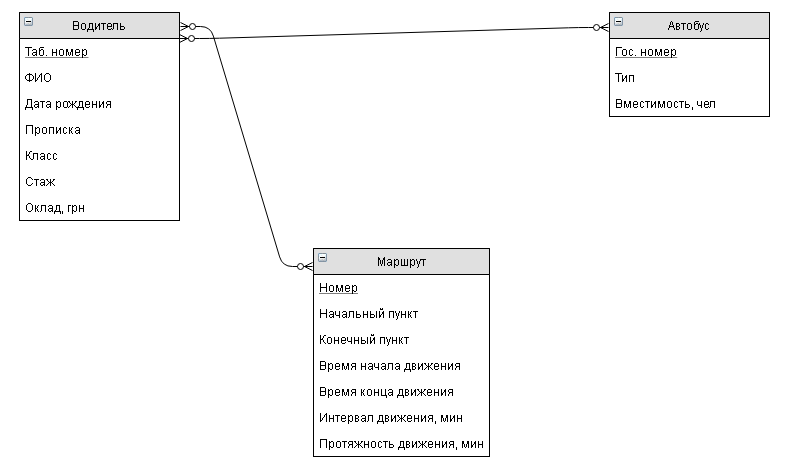
* Список водителей, работающих на определенном маршруте с указанием графика их работы?
* Какие автобусы обслуживают данный маршрут?
* Какие маршруты начинаются или заканчиваются в пункте с заданным названием?
* Когда начинается и заканчивается движение автобусов на каждом маршруте?
* Какова протяженность определенного маршрута?
* Какова общая протяженность маршрутов, обслуживаемых автопарком?
* Какие автобусы не вышли на линию, и по какой причине (неисправность, отсутствие водителя)?
* Сколько водителей каждого класса работает в автопарке?
* Водитель с самым большим стажем работы?
* Необходимо предусмотреть возможность выдачи справки о маршруте (протяженность, время и интервал движения, конечные пункты, кто обслуживает) и отчета по автопарку, сгруппированного по типам автобусов, с указанием маршрутов, обслуживаемых автобусами каждого типа. Для маршрутов должны быть указаны все характеристики, включая списки автобусов и водителей, обслуживающих каждый маршрут. Отчет должен содержать сведения о суммарной протяженности обслуживаемых маршрутов, о количестве имеющихся в автопарке автобусов каждого типа, о количестве водителей, их среднем возрасте и стаже.

2. ЗАВДАННЯ НА розрахунково-графічних робіт

2.1. Промоделювати функціональні вимоги до системи, що проектується за допомогою DFD-діаграми:

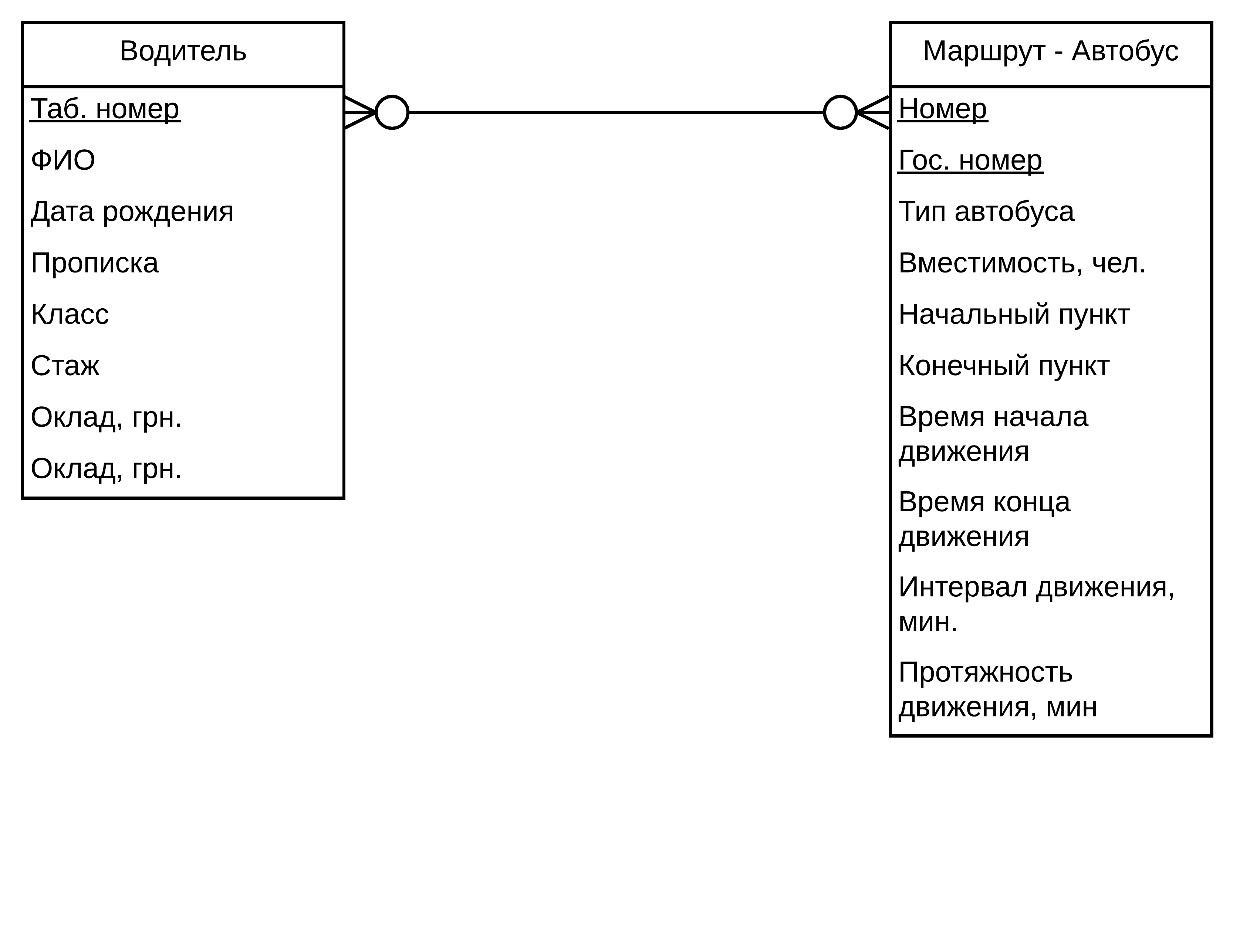


2.2. Побудувати концептуальну модель (ER-модель):

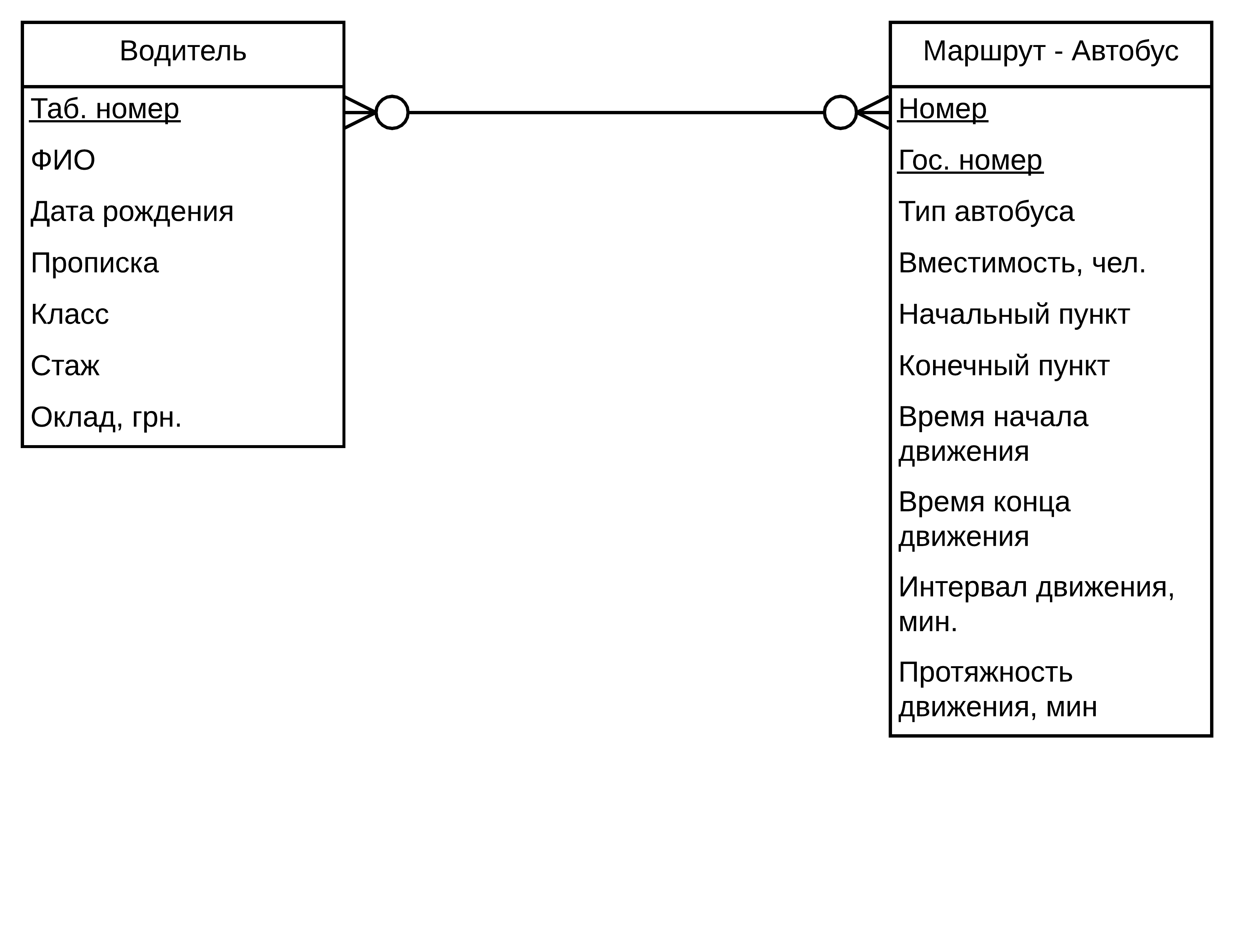


**Приведення до третьої нормальної форми**

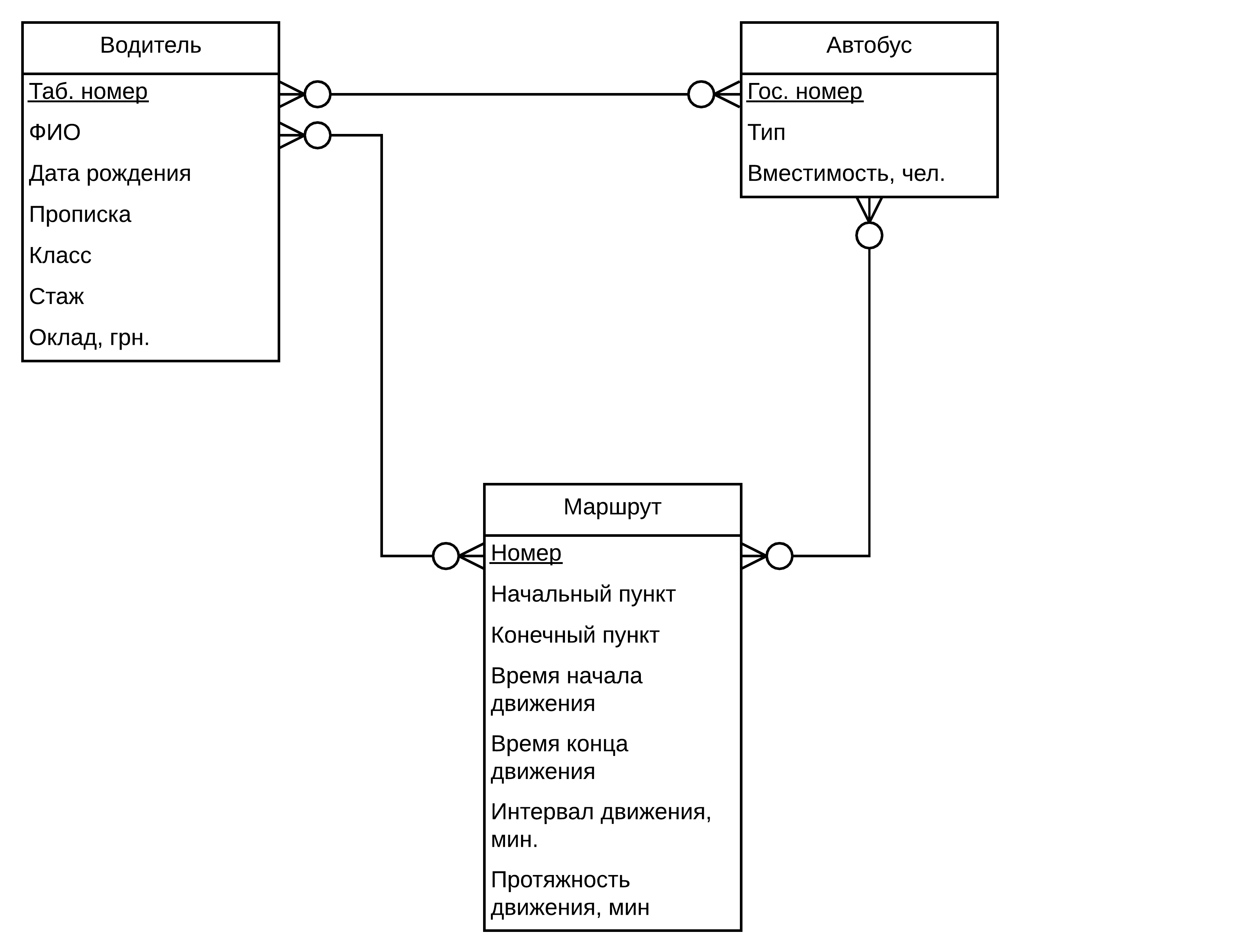
Маємо:



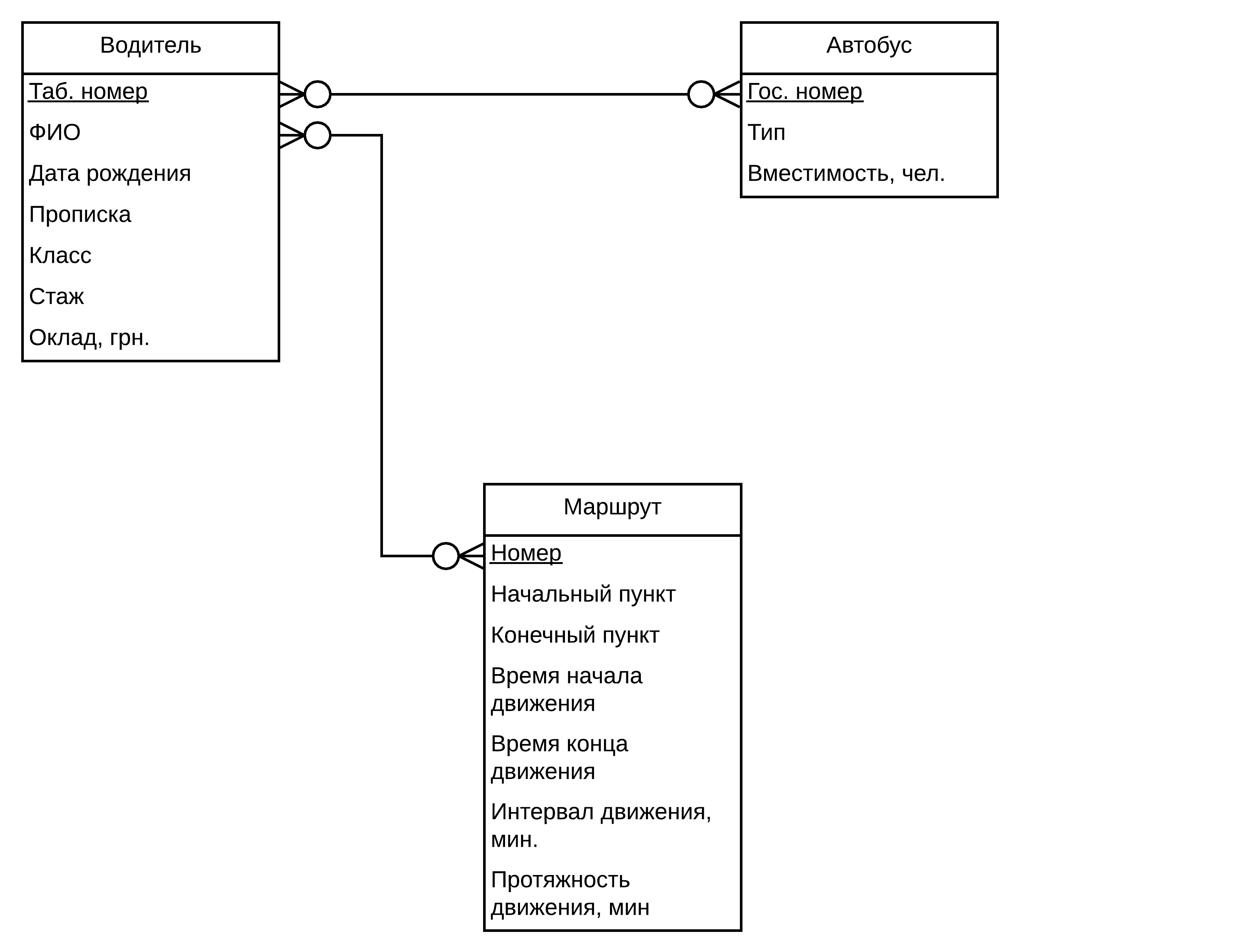
Приведення до першої нормальної форми: треба щоб всі атрибути сутностей були атомарними. Ми маємо не атомарні атрибути в сутності Водій. Отримуємо:



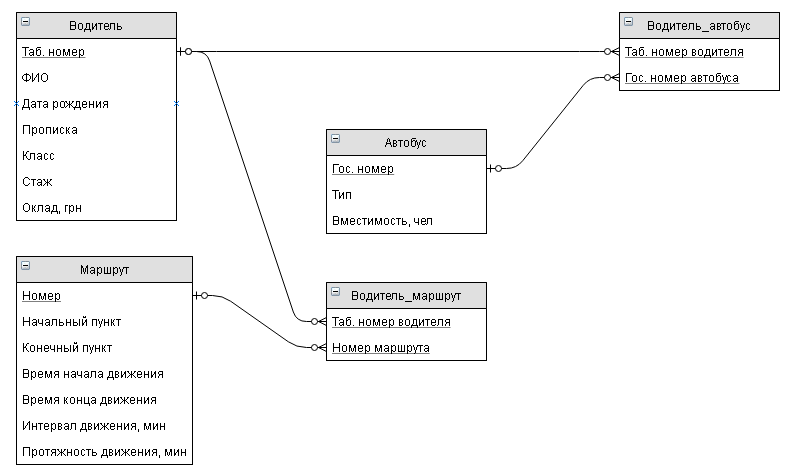
Приведення до другої нормальної форми: треба щоб кожна сутність не містила неповних функціональних залежностей не ключових атрибутів від атрибутів первинного ключа. На даний момент ми маємо в сутності Мершрут — Автобус два первинних ключа, але можна помітити, що атрибути Тип автобуса, Місткість залежать лише від автобуса, тобто від первинного ключа Держ. номер. Всі ж інші атрибути навпаки залежать від маршруту, тобто від первинного ключа Номер маршруту. Отже можна розділити цю сутність на дві сутності. Отримуємо:



Приведення до третьої нормальної форми: треба забезпечити відсутність транзитивних зв’язків між сутностями. На даний момент ми маємо транзитивний зв’язок між сутностями. Наприклад: ми маємо доступ з таблиці Водій до таблиці Автобус, з таблиці Автобус ми маємо доступ до таблиці Маршрут, але при цьому ми маємо прямий доступ з таблиці Водій до таблиці Маршрут. Це є транзитивним зв’язком. Виправимо це:



2.3. Отримати реляційну схему з ER-моделі:



2.4. Використовуючи СУБД MS SQL Server створити спроектовану базу даних.

USE master;

GO

CREATE DATABASE autopark

GO

USE autopark

GO

CREATE TABLE Driver (

id int IDENTITY PRIMARY KEY,

fullName nvarchar(50) NOT NULL,

dateBirth date NOT NULL,

placeOfLive nvarchar(100) NOT NULL,

class nvarchar(2),

experience tinyint,

salary decimal(8, 2),

ready nvarchar(50) NOT NULL,

CONSTRAINT DriverCheck

CHECK (fullName LIKE ('\_%[ ]\_%[ ]\_%')),

CHECK (DATEDIFF(day, dateBirth, GetDate()) >= 18 \* 365),

CHECK (Len(RTrim(placeOfLive)) > 5),

CHECK (ISNULL(experience, 0) >= 0),

CHECK (ISNULL(salary, 0) >= 0),

);

CREATE TABLE Bus (

plateNumber nvarchar(25) PRIMARY KEY,

capacity tinyint NOT NULL,

ready nvarchar(50) NOT NULL,

vehicleType nvarchar(1),

CONSTRAINT BusCheck

CHECK (ISNULL(capacity, 0) >= 0),

);

GO

IF OBJECT\_ID ('BusInsert', 'P') IS NOT NULL

DROP PROC BusInsert;

GO

CREATE PROCEDURE BusInsert

@plateNumber nvarchar(25),

@capacity tinyint,

@ready nvarchar(50)

AS

DECLARE @vehicleType nvarchar(25)

IF @capacity < 22 (

SELECT @vehicleType = N'B'

);

ELSE (

SELECT @vehicleType = N'A'

);

INSERT INTO Bus VALUES (@plateNumber,

@capacity,

@ready,

@vehicleType);

GO

CREATE TABLE WayRoute (

number nvarchar(10) PRIMARY KEY,

startPoint nvarchar(100) NOT NULL,

finishPoint nvarchar(100) NOT NULL,

startTime time(0) NOT NULL,

finishTime time(0) NOT NULL,

interval tinyint,

extent tinyint,

CONSTRAINT WayRouteCheck

CHECK (ISNULL(interval, 0) >= 0),

CHECK (ISNULL(extent, 0) >= 0),

);

CREATE TABLE DriverBus (

DriverId int,

BusPlateNumber nvarchar(25),

primary key (DriverId, BusPlateNumber),

CONSTRAINT DriverBusFK

FOREIGN KEY (DriverId)

REFERENCES Driver(id)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY (BusPlateNumber)

REFERENCES Bus(plateNumber)

ON UPDATE CASCADE

ON DELETE CASCADE,

);

CREATE TABLE DriverWayRoute (

DriverId int,

WayRouteNumber nvarchar(10),

primary key (DriverId, WayRouteNumber),

CONSTRAINT DriverWayRouteFK

FOREIGN KEY (DriverId)

REFERENCES Driver(id)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY (WayRouteNumber)

REFERENCES WayRoute(number)

ON UPDATE CASCADE

ON DELETE CASCADE,

);

2.5. На мові SQL записати вирази для зазначених у варіанті завдання типів запитів. Перевірити працездатність написаних запитів в інтерактивному режимі.

USE autopark

GO

INSERT INTO Driver VALUES

(N'Lyovkin Nikita Sergeevich', '1995-05-08', N'Kyiv, Ukraine', N'A', 6, 60000, N'yes'),

(N'Pupkin Ivan Ivanovich', '1994-05-08', N'Kyiv, Ukraine', N'B', 7, 6000, N'yes'),

(N'Vasilyev Ivan Petrovich', '1993-05-08', N'Kyiv, Ukraine', N'C', 10, 9000, N'yes'),

(N'Kravchuk Leonid Makarovich', '1992-05-08', N'Kyiv, Ukraine', N'C', 15, 15000, N'yes'),

(N'Kuchma Leonid Danilovich', '1991-05-08', N'Kyiv, Ukraine', N'C', 17, 16000, N'yes'),

(N'Ushenko Viktor Andreyovich', '1990-05-08', N'Kyiv, Ukraine', N'A', 2, 8000, N'yes'),

(N'Yanukovich Viktor Federovich', '1996-05-08', N'Rostov, Russia', N'A', 4, 10000, N'yes'),

(N'Poroshenko Petr Alekseyevich', '1996-05-08', N'Kyiv, Ukraine', N'A', 4, 12000, N'vacation'),

(N'Tymoshenko Uliya Volodimirovna', '1995-05-08', N'Dnepr, Ukraine', N'B', 4, 10000, N'yes'),

(N'Parubiy Andrey Vlodimirovich', '1996-05-08', N'Kyiv, Ukraine', N'D', 1, 2000, N'yes');

EXEC BusInsert N'AA0505CX', 7, N'yes';

EXEC BusInsert N'AA1111XX', 22, N'yes';

EXEC BusInsert N'AA1121TT', 100, N'yes';

EXEC BusInsert N'AA1122TT', 40, N'yes';

EXEC BusInsert N'AA1222TT', 23, N'yes';

EXEC BusInsert N'AA2222TT', 3, N'yes';

EXEC BusInsert N'AA1123TT', 2, N'yes';

EXEC BusInsert N'AA3333TT', 120, N'yes';

EXEC BusInsert N'AA4444TT', 88, N'yes';

EXEC BusInsert N'AA1010TT', 14, N'ill';

INSERT INTO WayRoute VALUES

(N'16A', N'Kyiv, Ukraine', N'Kyiv, Ukraine', '12:15:04', '22:15:04', 10, 60),

(N'18', N'Kyiv, Ukraine', N'Kyiv, Ukraine', N'10:15:04', N'23:15:04', 20, 45),

(N'20', N'Kharkiv, Ukraine', N'Kharkiv, Ukraine', '12:15:04', '22:15:04', 30, 120),

(N'118', N'Kyiv, Ukraine', N'Kyiv, Ukraine', '09:15:04', '20:15:04', 40, 160),

(N'55', N'Kyiv, Ukraine', N'Kyiv, Ukraine', '07:15:04', '21:15:04', 10, 180),

(N'62', N'Odessa, Ukraine', N'Odessa, Ukraine', '06:15:04', '22:15:04', 5, 20),

(N'38', N'Kyiv, Ukraine', N'Kyiv, Ukraine', '12:15:04', '23:15:04', 25, 50),

(N'62B', N'Kyiv, Ukraine', N'Kyiv, Ukraine', '12:15:04', '00:15:04', 21, 100),

(N'17', N'Kyiv, Ukraine', N'Kyiv, Ukraine', '13:15:04', '21:15:04', 5, 60),

(N'553', N'Kyiv, Ukraine', N'Kyiv, Ukraine', '15:15:04', '18:15:04', 15, 15);

INSERT INTO DriverBus VALUES (1, N'AA0505CX'),

(2, N'AA1111XX'),

(3, N'AA1121TT'),

(4, N'AA1122TT'),

(5, N'AA1222TT'),

(6, N'AA2222TT'),

(7, N'AA1123TT'),

(7, N'AA3333TT'),

(9, N'AA4444TT'),

(10, N'AA4444TT');

INSERT INTO DriverWayRoute VALUES (1, N'16A'),

(2, N'18'),

(3, N'20'),

(4, N'118'),

(5, N'55'),

(6, N'62'),

(7, N'38'),

(7, N'62B'),

(9, N'17'),

(10, N'553');

USE autopark

GO

-- Список водителей, работающих на определенном маршруте с указанием графика их работы?

SELECT D.fullName [Full Name], D.dateBirth [Date of Birth], D.placeOfLive [Place of Live], D.class [Class], D.experience [Experience], D.salary [Salary], D.ready [Ready], WR.startTime [Time of route start], WR.finishTime [Time of route finish] FROM Driver D

JOIN DriverWayRoute DWR ON D.id = DWR.DriverId

JOIN WayRoute WR ON DWR.WayRouteNumber = WR.number

-- Какие автобусы обслуживают данный маршрут?

SELECT B.plateNumber [Plate Number], B.capacity [Capacity], B.vehicleType [Type], B.ready [Ready] FROM Bus B

JOIN DriverBus DB ON DB.BusPlateNumber = B.plateNumber

JOIN Driver D ON DB.DriverId = D.id

JOIN DriverWayRoute DWR ON DWR.DriverId = D.id

JOIN WayRoute WR ON DWR.WayRouteNumber = WR.number

WHERE WR.number = N'62'

-- Какие маршруты начинаются или заканчиваются в пункте с заданным названием?

SELECT WR.number [Number], WR.startPoint [Start Point], WR.finishPoint [Finish Point], WR.startTime [Time of route start], WR.finishTime [Time of route finish], WR.interval [Interval], WR.extent [Extent] FROM WayRoute WR

WHERE WR.startPoint = N'Odessa, Ukraine' OR WR.finishPoint = N'Odessa, Ukraine'

-- Когда начинается и заканчивается движение автобусов на каждом маршруте?

SELECT WR.number [Number], WR.startTime [Time of route start], WR.finishTime [Time of route finish] FROM WayRoute WR

-- Какова протяженность определенного маршрута?

SELECT WR.number [Number], WR.extent [Extent] FROM WayRoute WR

WHERE WR.number = N'62'

-- Какова общая протяженность маршрутов, обслуживаемых автопарком?

SELECT SUM(WR.extent) [Total Extent] FROM WayRoute WR

-- Какие автобусы не вышли на линию, и по какой причине (неисправность, отсутствие водителя)?

SELECT B.plateNumber [Plate Number], B.capacity [Capacity], B.vehicleType [Type], B.ready [Reason] FROM Bus B

LEFT JOIN DriverBus DB ON DB.BusPlateNumber = B.plateNumber

WHERE B.ready <> N'yes' OR DB.BusPlateNumber IS NULL

--WHERE B.ready <> N'yes' OR NOT EXISTS(SELECT 1 FROM DriverBus DB WHERE DB.BusPlateNumber = B.plateNumber)

-- Сколько водителей каждого класса работает в автопарке?

SELECT D.class [Class], COUNT(D.fullName) [Drivers] FROM Driver D

GROUP BY D.class

-- Водитель с самым большим стажем работы?

SELECT D.fullName [Full Name], D.dateBirth [Date of Birth], D.placeOfLive [Place of Live], D.class [Class], D.experience [Experience], D.salary [Salary], D.ready [Ready] FROM Driver D

WHERE D.experience = (

SELECT MAX(D.experience) FROM Driver D

)

-- Справки о маршруте (протяженность, время и интервал движения, конечные пункты, кто обслуживает)

SELECT WR.number [Number], WR.extent [Extent], WR.interval [Interval], WR.startPoint [Start Point], WR.finishPoint [Finish Point],

(

SELECT D.fullName + ', ' FROM Driver D

JOIN DriverWayRoute DWR ON DWR.WayRouteNumber = WR.number

WHERE DWR.DriverId = D.id

ORDER BY D.fullName

FOR XML PATH('')

) AS Serve

FROM WayRoute WR

WHERE number = N'20'

-- Отчет по автопарку, сгруппированный по типам автобусов, с указанием маршрутов, обслуживаемых автобусами каждого типа

SELECT DISTINCT B.vehicleType [Bus Type],

(

SELECT DISTINCT WR.number + ', ' FROM Bus B2

JOIN DriverBus DB ON DB.BusPlateNumber = B2.plateNumber

JOIN Driver D ON D.id = DB.DriverId

JOIN DriverWayRoute DWR ON DWR.DriverId = D.id

JOIN WayRoute WR ON WR.number = DWR.WayRouteNumber

WHERE B.vehicleType = B2.vehicleType

FOR XML PATH('')

) [Routes]

FROM Bus B

-- Для маршрутов должны быть указаны все характеристики, включая списки автобусов и водителей, обслуживающих каждый маршрут

SELECT DISTINCT WR.number [Route Number], WR.startPoint [Start Point], WR.finishPoint [Finish Point], WR.startTime [Time of route start], WR.finishTime [Time of route finish], WR.interval [Interval], WR.extent [Extent],

(

SELECT DISTINCT B.plateNumber + ', ' FROM WayRoute WR2

JOIN DriverWayRoute DWR ON DWR.WayRouteNumber = WR2.number

JOIN Driver D ON D.id = DWR.DriverId

JOIN DriverBus DB ON DB.DriverId = D.id

JOIN Bus B ON B.plateNumber = DB.BusPlateNumber

WHERE WR.number = WR2.number

FOR XML PATH('')

) [Buses],

(

SELECT D.fullName + ', ' FROM WayRoute WR3

JOIN DriverWayRoute DWR ON DWR.WayRouteNumber = WR3.number

JOIN Driver D ON D.id = DWR.DriverId

WHERE WR.number = WR3.number

FOR XML PATH('')

) [Drivers]

FROM WayRoute WR

-- Cуммарная протяженность обслуживаемых маршрутов

SELECT SUM(WR.extent) [Extend Sum] FROM WayRoute WR

-- Количество имеющихся в автопарке автобусов каждого типа

SELECT B.vehicleType [Bus Type], COUNT(B.plateNumber) [Buses Quantity] FROM Bus B

GROUP BY B.vehicleType

-- Kоличествo водителей, их средний возраст и стаж

SELECT COUNT(D.id) [Drivers Quantity],

AVG(DATEDIFF(hour,D.dateBirth,GETDATE())/8766) [Drivers Avarage Age],

AVG(D.experience) [Drivers Avarage Experience]

FROM Driver D

2.6.

Реалізувати закінчене додаток, що працює з створеною базою даних.

import pyodbc  
  
# Authentication  
server = 'SOMEONE\SQLEXPRESS'  
database = 'autopark'  
cnxn = pyodbc.connect('DRIVER={ODBC Driver 13 for SQL Server};SERVER='+server+  
 ';DATABASE='+database+';trusted\_connection=yes')  
cursor = cnxn.cursor()  
  
  
def execute\_and\_show(query):  
 cursor.execute(query)  
 row = cursor.fetchone()  
 print([column[0] for column in cursor.description]) # description  
 while row:  
 print(row)  
 row = cursor.fetchone()  
  
  
def process\_input(\*args):  
 res\_arg = []  
 for arg in args:  
 res\_arg.append('N\'' + str(arg) + '\'')  
  
 return len(res\_arg) > 1 and res\_arg or res\_arg[0]  
  
  
def drivers\_routes():  
 query =\  
'''SELECT D.fullName [Full Name], D.dateBirth [Date of Birth], D.placeOfLive [Place of Live], D.class [Class], D.experience [Experience], D.salary [Salary], D.ready [Ready], WR.startTime [Time of route start], WR.finishTime [Time of route finish] FROM Driver D  
JOIN DriverWayRoute DWR ON D.id = DWR.DriverId  
JOIN WayRoute WR ON DWR.WayRouteNumber = WR.number'''  
  
 execute\_and\_show(query)  
  
  
def get\_buses\_on\_route(number):  
 number = process\_input(number)  
 query =\  
f'''SELECT B.plateNumber [Plate Number], B.capacity [Capacity], B.vehicleType [Type], B.ready [Ready] FROM Bus B  
JOIN DriverBus DB ON DB.BusPlateNumber = B.plateNumber  
JOIN Driver D ON DB.DriverId = D.id  
JOIN DriverWayRoute DWR ON DWR.DriverId = D.id  
JOIN WayRoute WR ON DWR.WayRouteNumber = WR.number  
WHERE WR.number = {number}'''  
  
 execute\_and\_show(query)  
  
  
def get\_route\_by\_points(start\_point: str, finish\_point: str):  
 start\_point, finish\_point = process\_input(start\_point, finish\_point)  
 query =\  
f'''SELECT WR.number [Number], WR.startPoint [Start Point], WR.finishPoint [Finish Point], WR.startTime [Time of route start], WR.finishTime [Time of route finish], WR.interval [Interval], WR.extent [Extent] FROM WayRoute WR  
WHERE WR.startPoint = {start\_point} OR WR.finishPoint = {finish\_point}'''  
  
 execute\_and\_show(query)  
  
  
def routes\_time():  
 query =\  
'''SELECT WR.number [Number], WR.startTime [Time of route start], WR.finishTime [Time of route finish] FROM WayRoute WR'''  
  
 execute\_and\_show(query)  
  
  
def route\_extent(number):  
 number = process\_input(number)  
 query =\  
f'''SELECT WR.number [Number], WR.extent [Extent] FROM WayRoute WR  
WHERE WR.number = {number}'''  
  
 execute\_and\_show(query)  
  
  
def total\_extent():  
 query = 'SELECT *SUM*(WR.extent) [Total Extent] FROM WayRoute WR'  
  
 execute\_and\_show(query)  
  
  
def get\_buses\_no\_work():  
 query =\  
'''SELECT B.plateNumber [Plate Number], B.capacity [Capacity], B.vehicleType [Type], B.ready [Reason] FROM Bus B  
LEFT JOIN DriverBus DB ON DB.BusPlateNumber = B.plateNumber  
WHERE B.ready <> N'yes' OR DB.BusPlateNumber IS NULL'''  
  
 execute\_and\_show(query)  
  
  
def drivers\_each\_class\_qty():  
 query =\  
'''SELECT D.class [Class], *COUNT*(D.fullName) [Drivers] FROM Driver D  
GROUP BY D.class'''  
  
 execute\_and\_show(query)  
  
  
def get\_max\_exp\_driver():  
 query =\  
'''SELECT D.fullName [Full Name], D.dateBirth [Date of Birth], D.placeOfLive [Place of Live], D.class [Class], D.experience [Experience], D.salary [Salary], D.ready [Ready] FROM Driver D  
WHERE D.experience = (  
 SELECT MAX(D.experience) FROM Driver D  
)'''  
  
 execute\_and\_show(query)  
  
  
def route\_info(number):  
 number = process\_input(number)  
 query =\  
f'''SELECT WR.number [Number], WR.extent [Extent], WR.interval [Interval], WR.startPoint [Start Point], WR.finishPoint [Finish Point],  
(   
 SELECT D.fullName + ', ' FROM Driver D  
 JOIN DriverWayRoute DWR ON DWR.WayRouteNumber = WR.number  
 WHERE DWR.DriverId = D.id  
 ORDER BY D.fullName  
 FOR XML PATH('')   
) AS Serve  
FROM WayRoute WR  
WHERE number = {number}'''  
  
 execute\_and\_show(query)  
  
  
def bus\_types\_and\_routes():  
 query =\  
'''SELECT DISTINCT B.vehicleType [Bus Type],  
(   
 SELECT DISTINCT WR.number + ', ' FROM Bus B2  
 JOIN DriverBus DB ON DB.BusPlateNumber = B2.plateNumber  
 JOIN Driver D ON D.id = DB.DriverId  
 JOIN DriverWayRoute DWR ON DWR.DriverId = D.id  
 JOIN WayRoute WR ON WR.number = DWR.WayRouteNumber  
 WHERE B.vehicleType = B2.vehicleType  
 FOR XML PATH('')  
) [Routes]  
FROM Bus B'''  
  
 execute\_and\_show(query)  
  
  
def routes\_info():  
 query =\  
'''SELECT DISTINCT WR.number [Route Number], WR.startPoint [Start Point], WR.finishPoint [Finish Point], WR.startTime [Time of route start], WR.finishTime [Time of route finish], WR.interval [Interval], WR.extent [Extent],  
(   
 SELECT DISTINCT B.plateNumber + ', ' FROM WayRoute WR2  
 JOIN DriverWayRoute DWR ON DWR.WayRouteNumber = WR2.number  
 JOIN Driver D ON D.id = DWR.DriverId  
 JOIN DriverBus DB ON DB.DriverId = D.id  
 JOIN Bus B ON B.plateNumber = DB.BusPlateNumber  
 WHERE WR.number = WR2.number  
 FOR XML PATH('')  
) [Buses],  
(   
 SELECT D.fullName + ', ' FROM WayRoute WR3  
 JOIN DriverWayRoute DWR ON DWR.WayRouteNumber = WR3.number  
 JOIN Driver D ON D.id = DWR.DriverId  
 WHERE WR.number = WR3.number  
 FOR XML PATH('')  
) [Drivers]  
FROM WayRoute WR'''  
  
 execute\_and\_show(query)  
  
  
def routes\_extent\_sum():  
 query = 'SELECT *SUM*(WR.extent) [Extend Sum] FROM WayRoute WR'  
  
 execute\_and\_show(query)  
  
  
def get\_qty\_of\_buses\_by\_types():  
 query =\  
'''SELECT B.vehicleType [Bus Type], COUNT(B.plateNumber) [Buses Quantity] FROM Bus B  
GROUP BY B.vehicleType'''  
  
 execute\_and\_show(query)  
  
  
def drivers\_qty\_avg\_age\_and\_exp():  
 query =\  
'''SELECT *COUNT*(D.id) [Drivers Quantity],   
 AVG(DATEDIFF(hour,D.dateBirth,GETDATE())/8766) [Drivers Avarage Age],  
 AVG(D.experience) [Drivers Avarage Experience]  
 FROM Driver D'''  
  
 execute\_and\_show(query)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 drivers\_routes()  
 get\_buses\_on\_route(62)  
 get\_route\_by\_points('Odessa, Ukraine', 'Odessa, Ukraine')  
 routes\_time()  
 route\_extent(62)  
 total\_extent()  
 get\_buses\_no\_work()  
 drivers\_each\_class\_qty()  
 get\_max\_exp\_driver()  
 route\_info(20)  
 bus\_types\_and\_routes()  
 routes\_info()  
 routes\_extent\_sum()  
 get\_qty\_of\_buses\_by\_types()  
 drivers\_qty\_avg\_age\_and\_exp()