











The image shows a Visual Studio Code editor with a C# project named 'GarageConsoleApp'. The code is in 'Program.cs' and is divided into two main sections: 'ViewAddAndItineraries' and 'ViewFlights'.

ViewAddAndItineraries Method:

```
static void ViewAddAndItineraries()
{
    Console.WriteLine("Выберите действие:");
    Console.WriteLine("1. Посмотреть рейсы");
    Console.WriteLine("2. Добавить новый рейс");

    string actionChoice = Console.ReadLine();

    switch (actionChoice)
    {
        case "1":
            ViewFlights();
            break;
        case "2":
            AddFlight();
            break;
        default:
            Console.WriteLine("Неверный выбор действия.");
            break;
    }
}
```

ViewFlights Method:

```
static void ViewFlights()
{
    Console.WriteLine("Peйcu:");
    var querySql = "SELECT r.id, d.first_name || ' ' || d.last_name AS driver_name, c.name AS car_name, i.name AS itinerary_name, r.number_passengers " +
        "FROM route r " +
        "INNER JOIN driver d ON r.id_driver = d.id " +
        "INNER JOIN car c ON r.id_car = c.id " +
        "INNER JOIN itinerary i ON r.id_itinerary = i.id";

    using var cmd = new NpgsqlCommand(querySql, DatabaseService.GetSqlConnection());
    using var reader = cmd.ExecuteReader();

    while (reader.Read())
    {
    }
}
```

Database Schema:

The database schema is shown on the right side of the editor. It includes a 'public' schema with the following tables and sequences:

- Tables:
 - car
 - driver
 - driver_rights_category
 - itinerary
 - rights_category
 - route
 - type_car
- Sequences:
 - car_id_seq (integer)
 - driver_id_seq (integer)
 - itinerary_id_seq (integer)
 - rights_category_id_seq (integer)
 - route_id_seq (integer)
 - type_car_id_seq (integer)
- Database Objects:
 - postgres (1 of 3)
 - Server Objects

The image shows a Visual Studio Code editor with a C# project named 'GarageConsoleApp'. The code is split into two files: 'DatabaseService.cs' and 'Program.cs'. The 'DatabaseService.cs' file contains a method 'AddFlight()' that interacts with a PostgreSQL database. The 'Program.cs' file contains the main logic for the application, including reading user input and calling the 'AddFlight()' method.

```
259 INNER JOIN itinerary i ON i.id_itinerary = i.id;
260 using var cmd = new NpgsqlCommand(querySql, DatabaseService.GetSqlConnection());
261 using var reader = cmd.ExecuteReader();
262
263 while (reader.Read())
264 {
265     Console.WriteLine($"ID: {reader[0]}, Водитель: {reader[1]}, Машина: {reader[2]}, Маршрут: {reader[3]}, Количество пассажиров: {reader[4]}");
266 }
267 }
268
269 static void AddFlight()
270 {
271     Console.WriteLine("Выберите водителя (укажите ID из списка):");
272     //метод для просмотра водителей
273     ViewDrivers();
274
275     int driverId;
276     while (!int.TryParse(Console.ReadLine(), out driverId))
277     {
278         Console.WriteLine("Некорректный формат. Повторите ввод:");
279     }
280
281     Console.WriteLine("Выберите машину (укажите ID из списка):");
282     ViewCars();
283     int carId;
284     while (!int.TryParse(Console.ReadLine(), out carId))
285     {
286         Console.WriteLine("Некорректный формат. Повторите ввод:");
287     }
288
289     Console.WriteLine("Выберите маршрут (укажите ID из списка):");
290     ViewItineraries();
291     int itineraryId;
292     while (!int.TryParse(Console.ReadLine(), out itineraryId))
293     {
```

The right sidebar shows the PostgreSQL database schema. The database is named 'postgres@localhost'. The schema includes a 'public' schema with the following tables:

- car
- driver
- driver_rights_category
- itinerary
- rights_category
- route
- type_car

The 'sequences' section shows the following sequences:

- car_id_seq (integer)
- driver_id_seq (integer)
- itinerary_id_seq (integer)
- rights_category_id_seq (integer)
- route_id_seq (integer)
- type_car_id_seq (integer)

The bottom status bar shows the project is running on .NET 6.0, and the console output is visible.

```
GarageConsoleApp.csproj DatabaseService.cs Program.cs DatabaseRequests.cs rights_category Database

291 int itineraryId;
292 while (!int.TryParse(Console.ReadLine(), out itineraryId))
293 {
294     Console.WriteLine("Некорректный формат. Повторите ввод:");
295 }
296
297 Console.WriteLine("Введите количество пассажиров:");
298 int numberPassengers;
299 while (!int.TryParse(Console.ReadLine(), out numberPassengers))
300 {
301     Console.WriteLine("Некорректный формат. Повторите ввод:");
302 }
303
304 var querySql = $"INSERT INTO route (id_driver, id_car, id_itinerary, number_passengers) * +
305 $VALUES ((driverId), (carId), (itineraryId), (numberPassengers));";
306 using var cmd = new NpgsqlCommand(querySql, DatabaseService.GetSqlConnection());
307 var rowsAffected = cmd.ExecuteNonQuery();
308
309 if (rowsAffected > 0)
310 {
311     Console.WriteLine("Новый рейс успешно добавлен.");
312 }
313 else
314 {
315     Console.WriteLine("Ошибка при добавлении нового рейса.");
316 }
317
318
319
320 }
321
322 static void ViewCars()
323 {
324     Console.WriteLine("Машины:");
325     var querySql = "SELECT id, name, state_number, number_passengers FROM car";
326     using var cmd = new NpgsqlCommand(querySql, DatabaseService.GetSqlConnection());
327     using var reader = cmd.ExecuteReader();
328
329     while (reader.Read())
330     {
331         Console.WriteLine($"ID: {reader[0]}, Название: {reader[1]}, Гос. номер: {reader[2]}, Количество пассажиров: {reader[3]}");
332     }
333 }
334
335 static void ViewItineraries()
336 {
337     Console.WriteLine("Маршруты:");
338     var querySql = "SELECT id, name FROM itinerary";
339     using var cmd = new NpgsqlCommand(querySql, DatabaseService.GetSqlConnection());
340     using var reader = cmd.ExecuteReader();
341
342     while (reader.Read())
343     {
344         Console.WriteLine($"ID: {reader[0]}, Название: {reader[1]}");
345     }
346 }
347
348 static void ViewDrivers()
349 {
350     Console.WriteLine("Водители:");
351     var querySql = "SELECT id, first_name, last_name, birthdate FROM driver";
352     using var cmd = new NpgsqlCommand(querySql, DatabaseService.GetSqlConnection());
353     using var reader = cmd.ExecuteReader();
354
355     while (reader.Read())
356     {
357         Console.WriteLine($"ID: {reader[0]}, Имя: {reader[1]} {reader[2]}, Дата рождения: {reader[3]}");
358     }
359 }
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

```
GarageConsoleApp.csproj DatabaseService.cs Program.cs DatabaseRequests.cs rights_category Database

323 Console.WriteLine("Машины:");
324 var querySql = "SELECT id, name, state_number, number_passengers FROM car";
325 using var cmd = new NpgsqlCommand(querySql, DatabaseService.GetSqlConnection());
326 using var reader = cmd.ExecuteReader();
327
328 while (reader.Read())
329 {
330     Console.WriteLine($"ID: {reader[0]}, Название: {reader[1]}, Гос. номер: {reader[2]}, Количество пассажиров: {reader[3]}");
331 }
332
333
334 static void ViewItineraries()
335 {
336     Console.WriteLine("Маршруты:");
337     var querySql = "SELECT id, name FROM itinerary";
338     using var cmd = new NpgsqlCommand(querySql, DatabaseService.GetSqlConnection());
339     using var reader = cmd.ExecuteReader();
340
341     while (reader.Read())
342     {
343         Console.WriteLine($"ID: {reader[0]}, Название: {reader[1]}");
344     }
345 }
346
347 static void ViewDrivers()
348 {
349     Console.WriteLine("Водители:");
350     var querySql = "SELECT id, first_name, last_name, birthdate FROM driver";
351     using var cmd = new NpgsqlCommand(querySql, DatabaseService.GetSqlConnection());
352     using var reader = cmd.ExecuteReader();
353
354     while (reader.Read())
355     {
356         Console.WriteLine($"ID: {reader[0]}, Имя: {reader[1]} {reader[2]}, Дата рождения: {reader[3]}");
357     }
358 }
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

```
157 }
158 }
159
160 static void ViewAndAddDriversAndRights()
161 {
162     Console.WriteLine("Выберите действие:");
163     Console.WriteLine("1. Посмотреть водителей и их права");
164     Console.WriteLine("2. Добавить нового водителя и его права");
165
166     string actionChoice = Console.ReadLine();
167
168     switch (actionChoice)
169     {
170         case "1":
171             ViewDriversAndRights();
172             break;
173         case "2":
174             AddDriverAndRights();
175             break;
176         default:
177             Console.WriteLine("Неверный выбор действия.");
178             break;
179     }
180 }
181
182 static void ViewDriversAndRights()
183 {
184     Console.WriteLine("Водители и их права:");
185     var querySql = "SELECT dr.id, dr.first_name, dr.last_name, dr.birthdate, rc.name " +
186         "FROM driver dr " +
187         "INNER JOIN driver_rights_category drc ON dr.id = drc.id_driver " +
188         "INNER JOIN rights_category rc ON drc.id_rights_category = rc.id";
189     using var cmd = new NpgsqlCommand(querySql, DatabaseService.GetSqlConnection());
190     using var reader = cmd.ExecuteReader();
191 }
```

Database

- postgres@localhost 2
 - garage_db 1 of 3
 - public
 - tables 7
 - car
 - driver
 - driver_rights_category
 - itinerary
 - rights_category
 - route
 - type_car
 - sequences 6
 - car_id_seq integer
 - driver_id_seq integer
 - itinerary_id_seq integer
 - rights_category_id_seq integer
 - route_id_seq integer
 - type_car_id_seq integer
- Database Objects
 - postgres 1 of 3
 - Server Objects

GarageConsoleApp NET 6.0 ▾ GarageConsoleApp ▸ Program ▸ ViewDrivers

GarageConsoleApp ▸ GarageConsoleApp ▸ Program.cs

347:26 CRLF UTF-8 4 spaces

```
388 using var cmd = new NpgsqlCommand(querySql, DatabaseService.GetSqlConnection());
389 using var reader = cmd.ExecuteReader();
390
391 while (reader.Read())
392 {
393     Console.WriteLine($"ID: {reader[0]}, Имя: {reader[1]} {reader[2]}, Дата рождения: {reader[3]}, Права: {reader[4]}");
394 }
395
396 static void AddDriverAndRights()
397 {
398     Console.WriteLine("Введите имя водителя:");
399     string firstName = Console.ReadLine();
400     Console.WriteLine("Введите фамилию водителя:");
401     string lastName = Console.ReadLine();
402     Console.WriteLine("Введите дату рождения водителя в формате MM-dd-yyyy:");
403     DateTime birthdate;
404     while (!DateTime.TryParse(Console.ReadLine(), out birthdate))
405     {
406         Console.WriteLine("Некорректный формат даты. Повторите ввод:");
407     }
408
409     Console.WriteLine("Введите категории прав водителя (A, B, D):");
410     string rightsCategory = Console.ReadLine();
411     //добав инфо в бд
412     DatabaseService.AddDriverQuery(firstName, lastName, birthdate);
413
414     var driverId = DatabaseService.GetLastInsertedId();
415     // Добав инфо прав водителя в бд
416     DatabaseService.AddDriverRightsCategoryQuery(driverId, rightsCategory);
417 }
418
419 }
420
421 }
422 }
```

Database

- postgres@localhost 2
 - garage_db 1 of 3
 - public
 - tables 7
 - car
 - driver
 - driver_rights_category
 - itinerary
 - rights_category
 - route
 - type_car
 - sequences 6
 - car_id_seq integer
 - driver_id_seq integer
 - itinerary_id_seq integer
 - rights_category_id_seq integer
 - route_id_seq integer
 - type_car_id_seq integer
- Database Objects
 - postgres 1 of 3
 - Server Objects

GarageConsoleApp NET 6.0 ▾ GarageConsoleApp ▸ Program ▸ ViewAndAddDriversAndRights

GarageConsoleApp ▸ GarageConsoleApp ▸ Program.cs

375:25 CRLF UTF-8 4 spaces


```
Run GarageConsoleApp x
C# [ ] [ ] [ ] [ ]
'c:\Program Files\JetBrains\JetBrains Rider 2023.3\plugins\dpa\DotFiles\JetBrains.DPA.Runner.exe' --handle=21004 --backend-pid=7452 --etw-collect-flags=67108622 --detach-event-name=dpa.detach.21004 C:/Users/Пользователь/RiderP
objects/GarageConsoleApp/GarageConsoleApp/bin/Debug/net8.0/GarageConsoleApp.exe
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
0. Выход
Ваш Выбор:
1
Выберите действие:
1. Посмотреть типы машин
2. Добавить новый тип машины
1
Типы машин:
Id: 1 Название: Автобус
Id: 2 Название: Микроавтобус
Id: 3 Название: Легковой автомобиль
Id: 4 Название: Воздушный
Id: 5 Название: лимузин
Id: 6 Название: минивен
Id: 7 Название: Спорткар
Id: 8 Название: грузовая
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
0. Выход
Ваш Выбор:
1
Выберите действие:
GarageConsoleApp > GarageConsoleApp > C# Program.cs
```

```
Run GarageConsoleApp x
1
Выберите действие:
1. Посмотреть типы машин
2. Добавить новый тип машины
2
Введите название нового типа машины:
Крутая
Новый тип машины успешно добавлен.
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
0. Выход
Ваш Выбор:
2
Выберите действие:
1. Посмотреть водителей и их права
2. Добавить нового водителя и его права
1
Водители и их права:
Id: 1, Имя: Дмитрий Антипов, Дата рождения: 08.01.1958 0:00:00, Права: A
Id: 1, Имя: Дмитрий Антипов, Дата рождения: 08.01.1958 0:00:00, Права: B
Id: 2, Имя: Виктор Гаан, Дата рождения: 05.04.2009 0:00:00, Права: B
Id: 2, Имя: Виктор Гаан, Дата рождения: 05.04.2009 0:00:00, Права: D
Id: 3, Имя: Даниил Адаев, Дата рождения: 28.11.2001 0:00:00, Права: B
Id: 4, Имя: Иван Салманов, Дата рождения: 01.02.2000 0:00:00, Права: B
Id: 7, Имя: Петров Пётр, Дата рождения: 23.10.2004 0:00:00, Права: A
Id: 9, Имя: Полина Злобина, Дата рождения: 01.09.2004 0:00:00, Права: B
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
GarageConsoleApp > GarageConsoleApp > C# Program.cs
```

```
Run GarageConsoleApp x
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
0. Выход
Ваш Выбор:
2
Выберите действие:
1. Посмотреть водителей и их права
2. Добавить нового водителя и его права
2
Введите имя водителя:
Полинка
Введите фамилию водителя:
Злобина
Введите дату рождения водителя в формате MM-dd-yyyy:
01.02.2004
Введите категорию прав водителя (A, B, D):
A
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
0. Выход
Ваш Выбор:
2
Выберите действие:
1. Посмотреть водителей и их права
2. Добавить нового водителя и его права
1
Водители и их права:
GarageConsoleApp > GarageConsoleApp > C# Program.cs
```

```
Run GarageConsoleApp
3
Машины:
ID: 1, Название: PA3-32053, Гос. номер: K800MP70, Количество пассажиров: 25
ID: 2, Название: Hyundai Grand Starex 1, Гос. номер: C197KX70, Количество пассажиров: 9
ID: 3, Название: Audi A8, Гос. номер: C395MM70, Количество пассажиров: 3
ID: 4, Название: LADA Vesta, Гос. номер: K581X870, Количество пассажиров: 3
ID: 7, Название: Mercedes CLS, Гос. номер: K6S4XC70, Количество пассажиров: 5
ID: 8, Название: Lada 2107, Гос. номер: V004K070, Количество пассажиров: 4
ID: 9, Название: BMW, Гос. номер: K001MA70, Количество пассажиров: 10
Хотите добавить новую машину? (Да/Нет)
1f
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
0. Выход
Ваш Выбор:
3
Машины:
ID: 1, Название: PA3-32053, Гос. номер: K800MP70, Количество пассажиров: 25
ID: 2, Название: Hyundai Grand Starex 1, Гос. номер: C197KX70, Количество пассажиров: 9
ID: 3, Название: Audi A8, Гос. номер: C395MM70, Количество пассажиров: 3
ID: 4, Название: LADA Vesta, Гос. номер: K581X870, Количество пассажиров: 3
ID: 7, Название: Mercedes CLS, Гос. номер: K6S4XC70, Количество пассажиров: 5
ID: 8, Название: Lada 2107, Гос. номер: V004K070, Количество пассажиров: 4
ID: 9, Название: BMW, Гос. номер: K001MA70, Количество пассажиров: 10
Хотите добавить новую машину? (Да/Нет)
да
Введите название новой машины:
Баклажан
Введите государственный номер новой машины:
```

```
Run GarageConsoleApp
Выберите действие:
1. Посмотреть водителей и их права
2. Добавить нового водителя и его права
1
Водители и их права:
ID: 1, Имя: Дмитрий Антипов, Дата рождения: 08.01.1958 0:00:00, Права: A
ID: 1, Имя: Дмитрий Антипов, Дата рождения: 08.01.1958 0:00:00, Права: B
ID: 2, Имя: Виктор Гаан, Дата рождения: 05.04.2000 0:00:00, Права: B
ID: 2, Имя: Виктор Гаан, Дата рождения: 05.04.2000 0:00:00, Права: D
ID: 3, Имя: Даниил Андреев, Дата рождения: 28.11.2001 0:00:00, Права: B
ID: 4, Имя: Иван Саломонов, Дата рождения: 01.02.2000 0:00:00, Права: B
ID: 7, Имя: Петров Петр, Дата рождения: 23.10.2004 0:00:00, Права: A
ID: 8, Имя: Полина Злобина, Дата рождения: 01.09.2004 0:00:00, Права: B
ID: 9, Имя: Поленка Злобкина, Дата рождения: 01.02.2004 0:00:00, Права: A
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
0. Выход
Ваш Выбор:
Вы сделали неправильный выбор.
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
0. Выход
Ваш Выбор:
3
Машины:
```

```
Run GarageConsoleApp
Баклажан
Введите государственный номер новой машины:
K222NA70
Введите количество пассажиров новой машины:
55
Типы машин:
ID: 1 Название: Автобус
ID: 2 Название: Микроавтобус
ID: 3 Название: Легковой автомобиль
ID: 4 Название: Воздушный
ID: 5 Название: Пикап
ID: 6 Название: минивен
ID: 7 Название: Спорткар
ID: 8 Название: грузовая
ID: 9 Название: Крутая
Выберите тип машины(ID):
9
Новая машина успешно добавлена.
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
0. Выход
Ваш Выбор:
3
Машины:
ID: 1, Название: PA3-32053, Гос. номер: K800MP70, Количество пассажиров: 25
ID: 2, Название: Hyundai Grand Starex 1, Гос. номер: C197KX70, Количество пассажиров: 9
ID: 3, Название: Audi A8, Гос. номер: C395MM70, Количество пассажиров: 3
ID: 4, Название: LADA Vesta, Гос. номер: K581X870, Количество пассажиров: 3
ID: 7, Название: Mercedes CLS, Гос. номер: K6S4XC70, Количество пассажиров: 5
```

```
Run GarageConsoleApp
ID: 4, Название: LADA Vesta, Гос. номер: K581X870, Количество пассажиров: 3
ID: 7, Название: Mersdes CLS, Гос. номер: K654XC70, Количество пассажиров: 5
ID: 8, Название: Lada 2107, Гос. номер: V00AK070, Количество пассажиров: 4
ID: 9, Название: BMW, Гос. номер: K001MA70, Количество пассажиров: 10
ID: 10, Название: Баклан, Гос. номер: N222NA70, Количество пассажиров: 55
Хотите добавить новую машину? (Да/Нет)
нет
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
0. Выход
Ваш Выбор:
5
Выберите действие:
1. Посмотреть рейсы
2. Добавить новый рейс
1
Рейсы:
ID: 1, Водитель: Дмитрий Антипов, Машина: Audi A8, Маршрут: Томск-Новосибирск, Количество пассажиров: 1
ID: 2, Водитель: Виктор Гаан, Машина: LADA Vesta, Маршрут: Томск-Кемерово, Количество пассажиров: 2
ID: 3, Водитель: Даниил Азеев, Машина: Audi A8, Маршрут: Новосибирск-Томск, Количество пассажиров: 3
ID: 4, Водитель: Виктор Гаан, Машина: Hyundai Grand Starex 1, Маршрут: Молчаново-Томск, Количество пассажиров: 7
ID: 5, Водитель: Виктор Гаан, Машина: LADA Vesta, Маршрут: Томск-Крга, Количество пассажиров: 4
ID: 6, Водитель: Полина Злобина, Машина: Mersdes CLS, Маршрут: Томск-Москва, Количество пассажиров: 1
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
0. Выход
```

```
Run GarageConsoleApp
0. Выход
Ваш Выбор:
5
Выберите действие:
1. Посмотреть рейсы
2. Добавить новый рейс
2
Выберите водителя (укажите ID из списка):
Водители:
ID: 1, Имя: Дмитрий Антипов, Дата рождения: 08.01.1958 0:00:00
ID: 2, Имя: Виктор Гаан, Дата рождения: 05.04.2000 0:00:00
ID: 3, Имя: Даниил Азеев, Дата рождения: 28.11.2001 0:00:00
ID: 4, Имя: Иван Салманов, Дата рождения: 01.02.2008 0:00:00
ID: 5, Имя: Денис Иванов, Дата рождения: 12.01.1997 0:00:00
ID: 6, Имя: Петр Петров, Дата рождения: 23.10.2004 0:00:00
ID: 7, Имя: Петров Петр, Дата рождения: 23.10.2004 0:00:00
ID: 8, Имя: Полина Злобина, Дата рождения: 01.09.2004 0:00:00
ID: 9, Имя: Полина Злобина, Дата рождения: 01.02.2004 0:00:00
9
Выберите машину (укажите ID из списка):
Машины:
ID: 1, Название: PA3-32053, Гос. номер: K800MP70, Количество пассажиров: 25
ID: 2, Название: Hyundai Grand Starex 1, Гос. номер: C197KK70, Количество пассажиров: 9
ID: 3, Название: Audi A8, Гос. номер: C395MM70, Количество пассажиров: 3
ID: 4, Название: LADA Vesta, Гос. номер: K581X870, Количество пассажиров: 3
ID: 7, Название: Mersdes CLS, Гос. номер: K654XC70, Количество пассажиров: 5
ID: 8, Название: Lada 2107, Гос. номер: V00AK070, Количество пассажиров: 4
ID: 9, Название: BMW, Гос. номер: K001MA70, Количество пассажиров: 10
ID: 10, Название: Баклан, Гос. номер: N222NA70, Количество пассажиров: 55
10
Выберите маршрут (укажите ID из списка):
Маршруты:
ID: 1, Название: Томск-Новосибирск
ID: 2, Название: Новосибирск-Томск
ID: 3, Название: Томск-Кемерово
ID: 4, Название: Томск-Колпаево
ID: 5, Название: Молчаново-Томск
ID: 6, Название: Томск-Крга
ID: 7, Название: Томск-Москва
1
Введите количество пассажиров:
3
Новый рейс успешно добавлен.
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
0. Выход
Ваш Выбор:
5
Выберите действие:
1. Посмотреть рейсы
2. Добавить новый рейс
1
Рейсы:
ID: 1, Водитель: Дмитрий Антипов, Машина: Audi A8, Маршрут: Томск-Новосибирск, Количество пассажиров: 1
ID: 2, Водитель: Виктор Гаан, Машина: LADA Vesta, Маршрут: Томск-Кемерово, Количество пассажиров: 2
ID: 3, Водитель: Даниил Азеев, Машина: Audi A8, Маршрут: Новосибирск-Томск, Количество пассажиров: 3
ID: 4, Водитель: Виктор Гаан, Машина: Hyundai Grand Starex 1, Маршрут: Молчаново-Томск, Количество пассажиров: 7
```

```
Run GarageConsoleApp
ID: 10, Название: Баклан, Гос. номер: N222NA70, Количество пассажиров: 55
10
Выберите маршрут (укажите ID из списка):
Маршруты:
ID: 1, Название: Томск-Новосибирск
ID: 2, Название: Новосибирск-Томск
ID: 3, Название: Томск-Кемерово
ID: 4, Название: Томск-Колпаево
ID: 5, Название: Молчаново-Томск
ID: 6, Название: Томск-Крга
ID: 7, Название: Томск-Москва
1
Введите количество пассажиров:
3
Новый рейс успешно добавлен.
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
0. Выход
Ваш Выбор:
5
Выберите действие:
1. Посмотреть рейсы
2. Добавить новый рейс
1
Рейсы:
ID: 1, Водитель: Дмитрий Антипов, Машина: Audi A8, Маршрут: Томск-Новосибирск, Количество пассажиров: 1
ID: 2, Водитель: Виктор Гаан, Машина: LADA Vesta, Маршрут: Томск-Кемерово, Количество пассажиров: 2
ID: 3, Водитель: Даниил Азеев, Машина: Audi A8, Маршрут: Новосибирск-Томск, Количество пассажиров: 3
ID: 4, Водитель: Виктор Гаан, Машина: Hyundai Grand Starex 1, Маршрут: Молчаново-Томск, Количество пассажиров: 7
```

```
Run GarageConsoleApp >
Введите количество пассажиров:
3
Новый рейс успешно добавлен.
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
6. Выход
Ваш Выбор:
5
Выберите действие:
1. Посмотреть рейсы
2. Добавить новый рейс
1
Рейсы:
ID: 1, Водитель: Дмитрий Антипов, Машина: Audi A8, Маршрут: Томск-Новосибирск, Количество пассажиров: 1
ID: 2, Водитель: Виктор Гаан, Машина: LADA Vesta, Маршрут: Томск-Кемерово, Количество пассажиров: 2
ID: 3, Водитель: Даниил Азиев, Машина: Audi A8, Маршрут: Новосибирск-Томск, Количество пассажиров: 3
ID: 4, Водитель: Виктор Гаан, Машина: Hyundai Grand Starex 1, Маршрут: Колчано-Томск, Количество пассажиров: 7
ID: 5, Водитель: Виктор Гаан, Машина: LADA Vesta, Маршрут: Томск-Юга, Количество пассажиров: 4
ID: 6, Водитель: Полина Злобина, Машина: Mercedes CLS, Маршрут: Томск-Москва, Количество пассажиров: 1
ID: 7, Водитель: Полина Злобина, Машина: Баклажан, Маршрут: Томск-Новосибирск, Количество пассажиров: 3
Выберите действие из списка:
1. Посмотреть и добавить тип машины
2. Посмотреть и добавить водителя и их права
3. Посмотреть и добавить маршрут
5. Посмотреть и добавить рейс
6. Выход
Ваш Выбор:
```

GarageConsoleApp > GarageConsoleApp > C# Program.cs 407:14 CRLF UTF-8 4 spaces 601

```

using Npgsql;

namespace GarageConsoleApp;

/// <summary>
/// Класс DatabaseService
/// отвечает за подключение и открытие соединения с БД
/// </summary>
public static class DatabaseService
{
    /// <summary>
    /// Переменная _connection
    /// хранит открытое соединение с БД
    /// </summary>
    private static NpgsqlConnection? _connection;

    /// <summary>
    /// Метод GetConnectionString()
    /// возвращает строку подключения к БД
    /// </summary>
    private static string GetConnectionString()
    {
        return
@"Host=localhost;Port=5432;Database=garage_db;Username=postgres;Password=1234
";
    }

    /// <summary>
    /// Метод GetSqlConnection()
    /// проверяет есть ли уже открытое соединение с БД
    /// если нет, то открывает соединение с БД
    /// </summary>
    /// <returns></returns>
    public static NpgsqlConnection GetSqlConnection()
    {
        if (_connection is null)
        {
            _connection = new NpgsqlConnection(GetConnectionString());
            _connection.Open();
        }

        return _connection;
    }

    public static void AddDriverQuery(string firstName, string lastName,
DateTime birthdate)
    {
        var querySql =
            $"INSERT INTO driver(first name, last name, birthdate) VALUES
('{'firstName}', '{lastName}', '{birthdate.ToString("yyyy-MM-dd")}{'})'";
        using var cmd = new NpgsqlCommand(querySql, GetSqlConnection());
        cmd.ExecuteNonQuery();
    }

    public static void AddDriverRightsCategoryQuery(int driverId, string
rightsCategory)
    {
        var querySql = $"SELECT id FROM rights_category WHERE name =

```

```

    '{rightsCategory}';
    using var cmd = new NpgsqlCommand(querySql, GetSqlConnection());
    var rightsCategoryId = cmd.ExecuteScalar();

    if (rightsCategoryId == null)
    {
        Console.WriteLine("Указанная категория прав не найдена.");
        return;
    }

    querySql =
        $"INSERT INTO driver_rights_category(id_driver,
id_rights_category) VALUES ({driverId}, {rightsCategoryId})";
    cmd.CommandText = querySql;
    cmd.ExecuteNonQuery();
}

public static int GetLastInsertedId()
{
    var querySQL = "SELECT lastval()";
    using var cmd = new NpgsqlCommand(querySQL, GetSqlConnection());
    return Convert.ToInt32(cmd.ExecuteScalar());
}
}

```

```

using Microsoft.EntityFrameworkCore.Metadata.Internal;

namespace GarageConsoleApp;
using System;
using Npgsql;

/// <summary>
/// Класс Program
/// здесь описываем логику приложения
/// вызываем нужные методы пишем обработчики и тд
/// </summary>
public class Program
{
    public static void Main(string[] args)
    {
        while (true)
        {
            Console.WriteLine("Выберите действие из списка:");
            Console.WriteLine("1. Посмотреть и добавить тип машины");
            Console.WriteLine("2. Посмотреть и добавить водителя и их
права");
            Console.WriteLine("3. Посмотреть и добавить маршрут");
            Console.WriteLine("5. Посмотреть и добавить рейс");
            Console.WriteLine("0. Выход");

            Console.WriteLine("Ваш Выбор:");
            string A = Console.ReadLine();

            switch (A)
            {
                case "1" :
                    ViewAndTypeCars();
                    break;
                case "2" :
                    ViewAndAddDriversAndRights();
                    break;
                case "3":
                    ViewAndAddCars();
                    break;
                case "4":
                    ViewAndAddRoutes();
                    break;
                case "5":
                    ViewAndAddIteneraries();
                    break;
                case "0":
                    return;
                default:
                    Console.WriteLine("Вы сделали неправильны выбор.");
                    break;
            }
        }
        static void ViewAndTypeCars()
        {
            Console.WriteLine("Выберите действие:");
            Console.WriteLine("1. Посмотреть типы машин");
            Console.WriteLine("2. Добавить новый тип машины");

            //Выбор действия

```

```

        string actionChoice = Console.ReadLine();

        switch (actionChoice)
        {
            case "1":
                ViewTypeCars();
                break;
            case "2":
                AddTypeCar();
                break;
            default:
                Console.WriteLine("Неверный выбор действия.");
                break;
        }
    }

    static void AddTypeCar()
    {
        Console.WriteLine("Введите название нового типа машины:");
        string typeName = Console.ReadLine();

        var querySql = $"INSERT INTO type_car (name) VALUES ('{typeName}')";

        //выполнения SQL-запрос с использованием соединения с бд
        using var cmd = new NpgsqlCommand(querySql,
        DatabaseService.GetSqlConnection());
        var rowsAffected = cmd.ExecuteNonQuery();

        //использ строк
        if (rowsAffected > 0)
            Console.WriteLine("Новый тип машины успешно добавлен.");
        else
            Console.WriteLine("Ошибка при добавлении типа машины.");
    }

}

static void ViewTypeCars()
{
    Console.WriteLine("Типы машин:");
    // сбор записей
    var querySql = "SELECT * FROM type_car";
    using var cmd = new NpgsqlCommand(querySql,
    DatabaseService.GetSqlConnection());
    using var reader = cmd.ExecuteReader();

    while (reader.Read())
    {
        Console.WriteLine($"Id: {reader[0]} Название: {reader[1]}");
    }
}

static void ViewAndAddroutes()
{
    Console.WriteLine("Выберите действие:");
    Console.WriteLine("1. Посмотреть маршруты");
    Console.WriteLine("2. Добавить новый маршрут");

    string actionChoice = Console.ReadLine();

    switch (actionChoice)
    {

```



```

        case "1":
            ViewRoutes();
            break;
        case "2":
            AddRoute();
            break;
        default:
            Console.WriteLine("Неверный выбор действия.");
            break;
    }
}

static void ViewRoutes()
{
    Console.WriteLine("Маршруты:");
    var querySql = "SELECT id, name FROM itinerary";
    using var cmd = new NpgsqlCommand(querySql,
DatabaseService.GetSqlConnection());
    using var reader = cmd.ExecuteReader();

    while (reader.Read())
    {
        Console.WriteLine($"ID: {reader[0]}, Название: {reader[1]}");
    }
}

static void AddRoute()
{
    Console.WriteLine("Введите название нового маршрута:");
    string routeName = Console.ReadLine();

    //добавление нового маршрута
    var querySql = $"INSERT INTO itinerary (name) VALUES
('{routeName}')";
    using var cmd = new NpgsqlCommand(querySql,
DatabaseService.GetSqlConnection());
    var rowsAffected = cmd.ExecuteNonQuery();

    if (rowsAffected > 0)
    {
        Console.WriteLine("Новый маршрут успешно добавлен.");
    }
    else
    {
        Console.WriteLine("Ошибка при добавлении нового маршрута.");
    }
}

static void ViewAndAddCars()
{
    Console.WriteLine("Машины:");

    var querySql = "SELECT id, name, state_number, number_passengers FROM
car";

    using var cmd = new NpgsqlCommand(querySql,
DatabaseService.GetSqlConnection());
    using var reader = cmd.ExecuteReader();

    while (reader.Read())
    {
        Console.WriteLine($"ID: {reader[0]}, Название: {reader[1]}, Гос.

```

```

номер: {reader[2]}, Количество пассажиров: {reader[3]}");
    }

    reader.Close();

    Console.WriteLine("Хотите добавить новую машину? (Да/Нет)");
    string response = Console.ReadLine();

    if (response.ToLower() == "да")
    {
        Console.WriteLine("Введите название новой машины:");
        string name = Console.ReadLine();

        Console.WriteLine("Введите государственный номер новой машины:");
        string stateNumber = Console.ReadLine();

        Console.WriteLine("Введите количество пассажиров новой машины:");
        int numberPassengers;
        while (!int.TryParse(Console.ReadLine(), out numberPassengers))
        {
            Console.WriteLine("Некорректное значение. Повторите ввод:");
        }
        ViewTypeCars();
        Console.WriteLine("Выберите тип машины(ID):");
        string id_type_car = Console.ReadLine();

        int typeId = int.Parse(id_type_car);

        //Вызывает метод для добав машины с указанными параметрами
        AddCar(name, stateNumber, numberPassengers, typeId.ToString());
    }
}

static void AddCar(string name, string stateNumber, int numberPassengers,
string id_type_car)
{
    var querySql = $"INSERT INTO car (name, state_number,
number_passengers, id_type_car) VALUES ('{name}', '{stateNumber}',
{numberPassengers}, {id_type_car})";
    using var cmd = new NpgsqlCommand(querySql,
DatabaseService.GetSqlConnection());
    int rowsAffected = cmd.ExecuteNonQuery();

    if (rowsAffected > 0)
    {
        Console.WriteLine("Новая машина успешно добавлена.");
    }
    else
    {
        Console.WriteLine("Ошибка при добавлении новой машины.");
    }
}

static void ViewAndAddIteneraries()
{
    Console.WriteLine("Выберите действие:");
    Console.WriteLine("1. Посмотреть рейсы");
    Console.WriteLine("2. Добавить новый рейс");

    string actionChoice = Console.ReadLine();

    switch (actionChoice)
    {

```

```

        case "1":
            ViewFlights();
            break;
        case "2":
            AddFlight();
            break;
        default:
            Console.WriteLine("Неверный выбор действия.");
            break;
    }
}

static void ViewFlights()
{
    Console.WriteLine("Рейсы:");
    var querySql = "SELECT r.id, d.first_name || ' ' || d.last_name AS
driver_name, c.name AS car_name, i.name AS itinerary_name,
r.number_passengers " +
        "FROM route r " +
        "INNER JOIN driver d ON r.id driver = d.id " +
        "INNER JOIN car c ON r.id car = c.id " +
        "INNER JOIN itinerary i ON r.id itinerary = i.id";
    using var cmd = new NpgsqlCommand(querySql,
DatabaseService.GetSqlConnection());
    using var reader = cmd.ExecuteReader();

    while (reader.Read())
    {
        Console.WriteLine($"ID: {reader[0]}, Водитель: {reader[1]}, Машина:
{reader[2]}, Маршрут: {reader[3]}, Количество пассажиров: {reader[4]}");
    }
}

static void AddFlight()
{
    Console.WriteLine("Выберите водителя (укажите ID из списка):");
    //метод для просмотра водителей
    ViewDrivers();

    int driverId;
    while (!int.TryParse(Console.ReadLine(), out driverId))
    {
        Console.WriteLine("Некорректный формат. Повторите ввод:");
    }

    Console.WriteLine("Выберите машину (укажите ID из списка):");
    ViewCars();
    int carId;
    while (!int.TryParse(Console.ReadLine(), out carId))
    {
        Console.WriteLine("Некорректный формат. Повторите ввод:");
    }

    Console.WriteLine("Выберите маршрут (укажите ID из списка):");
    ViewItineraries();
    int itineraryId;
    while (!int.TryParse(Console.ReadLine(), out itineraryId))
    {
        Console.WriteLine("Некорректный формат. Повторите ввод:");
    }

    Console.WriteLine("Введите количество пассажиров:");
    int numberPassengers;
    while (!int.TryParse(Console.ReadLine(), out numberPassengers))

```

```

    {
        Console.WriteLine("Некорректный формат. Повторите ввод:");
    }

    var querySql = $"INSERT INTO route (id_driver, id_car, id_itinerary,
number_passengers) " +
        $"VALUES ({driverId}, {carId}, {itineraryId},
{numberPassengers})";
    using var cmd = new NpgsqlCommand(querySql,
DatabaseService.GetSqlConnection());
    var rowsAffected = cmd.ExecuteNonQuery();

    if (rowsAffected > 0)
    {
        Console.WriteLine("Новый рейс успешно добавлен.");
    }
    else
    {
        Console.WriteLine("Ошибка при добавлении нового рейса.");
    }
}

static void ViewCars()
{
    Console.WriteLine("Машины:");
    var querySql = "SELECT id, name, state_number, number_passengers FROM
car";
    using var cmd = new NpgsqlCommand(querySql,
DatabaseService.GetSqlConnection());
    using var reader = cmd.ExecuteReader();

    while (reader.Read())
    {
        Console.WriteLine($"ID: {reader[0]}, Название: {reader[1]}, Гос.
номер: {reader[2]}, Количество пассажиров: {reader[3]}");
    }
}

static void ViewItineraries()
{
    Console.WriteLine("Маршруты:");
    var querySql = "SELECT id, name FROM itinerary";
    using var cmd = new NpgsqlCommand(querySql,
DatabaseService.GetSqlConnection());
    using var reader = cmd.ExecuteReader();

    while (reader.Read())
    {
        Console.WriteLine($"ID: {reader[0]}, Название: {reader[1]}");
    }
}

static void ViewDrivers()
{
    Console.WriteLine("Водители:");
    var querySql = "SELECT id, first_name, last_name, birthdate FROM driver";
    using var cmd = new NpgsqlCommand(querySql,
DatabaseService.GetSqlConnection());
    using var reader = cmd.ExecuteReader();

    while (reader.Read())
    {

```

```

        Console.WriteLine($"ID: {reader[0]}, Имя: {reader[1]} {reader[2]},
Дата рождения: {reader[3]}");
    }
}

static void ViewAndAddDireversAndRights()
{
    Console.WriteLine("Выберите действие:");
    Console.WriteLine("1. Посмотреть водителей и их права");
    Console.WriteLine("2. Добавить нового водителя и его права");

    string actionChoice = Console.ReadLine();

    switch (actionChoice)
    {
        case "1":
            ViewDriversAndRights();
            break;
        case "2":
            AddDriverAndRights();
            break;
        default:
            Console.WriteLine("Неверный выбор действия.");
            break;
    }
}

static void ViewDriversAndRights()
{
    Console.WriteLine("Водители и их права:");
    var querySql = "SELECT dr.id, dr.first_name, dr.last_name,
dr.birthdate, rc.name " +
        "FROM driver dr " +
        "INNER JOIN driver_rights_category drc ON dr.id =
drc.id_driver " +
        "INNER JOIN rights_category rc ON
drc.id_rights_category = rc.id";
    using var cmd = new NpgsqlCommand(querySql,
DatabaseService.GetSqlConnection());
    using var reader = cmd.ExecuteReader();

    while (reader.Read())
    {
        Console.WriteLine($"ID: {reader[0]}, Имя: {reader[1]}
{reader[2]}, Дата рождения: {reader[3]}, Права: {reader[4]}");
    }
}

static void AddDriverAndRights()
{
    Console.WriteLine("Введите имя водителя:");
    string firstName = Console.ReadLine();
    Console.WriteLine("Введите фамилию водителя:");
    string lastName = Console.ReadLine();
    Console.WriteLine("Введите дату рождения водителя в формате ММ-
dd-yyyy:");
    DateTime birthdate;
    while (!DateTime.TryParse(Console.ReadLine(), out birthdate))
    {
        Console.WriteLine("Некорректный формат даты. Повторите
ввод:");
    }

    Console.WriteLine("Введите категорию прав водителя (A, B, D):");
    string rightsCategory = Console.ReadLine();
    //добав инфр в бд

```

```
        DatabaseService.AddDriverQuery(firstName, lastName, birthdate);

        var driverId = DatabaseService.GetLastInsertedId();
        // Добав информ прав водителя в бд
        DatabaseService.AddDriverRightsCategoryQuery(driverId,
rightsCategory);
    }
}
```