

--lab3

```
INSERT INTO Airport (airport_name, country, state, city)
```

```
VALUES
```

```
('Nursultan Nazarbayev International Airport', 'Kazakhstan', 'Akmola', 'Astana'),
```

```
('Almaty International Airport', 'Kazakhstan', 'Almaty', 'Almaty'),
```

```
('Heathrow Airport', 'United Kingdom', 'England', 'London'),
```

```
('airpekin', 'China', 'Pekin', 'Pekin');
```

```
INSERT INTO Passengers (first_name, last_name, date_of_birth, gender,
```

```
country_of_citizenship, country_of_residence, passport_number)
```

```
VALUES
```

```
('Aidos', 'Nursultanov', '1995-03-12', 'Male', 'Kazakhstan', 'Kazakhstan', 'KZ1234567'),
```

```
('Aliya', 'Serik', '2000-07-24', 'Female', 'Kazakhstan', 'Kazakhstan', 'KZ7654321'),
```

```
('John', 'Smith', '1988-11-02', 'Male', 'United Kingdom', 'United Kingdom', 'UK9876543'),
```

```
('Dana', 'Dana', '1999-06-15', 'Female', 'Kazakhstan', 'Kazakhstan', 'KZ2222333'); -- для  
задания 1 (имя=фамилия)
```

```
INSERT INTO Flights (sch_departure_time, sch_arrival_time, departing_airport_id,  
arriving_airport_id,
```

```
departing_gate, arriving_gate, airline_id, act_departure_time, act_arrival_time)
```

```
VALUES
```

```
('2025-09-23 08:00', '2025-09-23 09:30', 1, 2, 'A1', 'B3', 1, '2025-09-23 08:05', '2025-09-23  
09:25'),
```

```
('2025-09-24 14:00', '2025-09-24 19:00', 2, 3, 'C2', 'D5', 1, '2025-09-24 14:10', '2025-09-24  
18:50'),
```

```
('2025-09-26 12:00', '2025-09-26 22:00', 3, 1, 'E1', 'F2', 2, '2025-09-26 12:05', '2025-09-26  
21:55'),
```

```
('2025-09-27 14:00', '2025-09-27 18:00', 2, 4, 'D2', 'D5', 1, '2025-09-27 14:10', '2025-09-27 17:50');
```

```
INSERT INTO Booking (flight_id, passenger_id, booking_platform, status, ticket_price)  
VALUES
```

```
(1, 1, 'Website', 'Confirmed', 45000),  
(2, 2, 'Mobile App', 'Confirmed', 12000),  
(1, 3, 'Agent', 'Pending', 48000),  
(3, 4, 'Website', 'Confirmed', 30000);
```

```
INSERT INTO Boarding_pass (booking_id, seat, boarding_time)  
VALUES
```

```
(1, '12A', '2025-09-23 07:30'),  
(2, '20C', '2025-09-24 13:30'),  
(3, '7B', '2025-09-23 07:20'),  
(4, '15D', '2025-09-26 11:30');
```

```
INSERT INTO Baggage (weight_in_kg, booking_id)  
VALUES
```

```
(23.5, 1),  
(18.0, 2),  
(25.7, 3),  
(30.2, 4);
```

```
INSERT INTO Baggage_check (check_result, booking_id, passenger_id)  
VALUES
```

```
('OK', 1, 1),
```

('Extra screening', 2, 2),

('OK', 3, 3),

('Overweight', 4, 4);

INSERT INTO Security_check (check_result, passenger_id)

VALUES

('Cleared', 1),

('Secondary', 2),

('Cleared', 3),

('Cleared', 4);

INSERT INTO Booking_flight (booking_id, flight_id)

VALUES

(1, 1),

(2, 2),

(3, 1),

(4, 3);

--1)

SELECT * FROM Passengers

WHERE last_name = first_name;

--2)

SELECT DISTINCT last_name

FROM Passengers;

--3)

SELECT * FROM Passengers

WHERE gender = 'Male' AND date_of_birth BETWEEN '1990-01-01' AND '2000-12-31';

--4)

SELECT DATE_TRUNC('month', created_at) AS month,

SUM(ticket_price) AS total_sales

FROM Booking

GROUP BY month

ORDER BY month;

--5)

SELECT f. * FROM Flights f

JOIN Airport a on f.arriving_airport_id = a.airport_id

WHERE a.country = 'China';

--6)

SELECT * FROM Airline

WHERE airline_country in ('France', 'Portugal', 'Poland')

AND created_at BETWEEN '2023-11-01' AND '2024-03-31';

--7)

SELECT airline_name FROM Airline

WHERE airline_country = 'Kazakhstan';

--8)

UPDATE Booking

SET ticket_price = ticket_price * 0.9

WHERE created_at < '2023-11-01';

--9)

SELECT * FROM Baggage

WHERE weight_in_kg > 20

ORDER BY weight_in_kg DESC LIMIT 3;

--10)

SELECT first_name, last_name FROM Passengers

ORDER BY date_of_birth DESC LIMIT 1;

--11)

SELECT booking_platform, MIN(ticket_price) AS cheapest_price

FROM Booking

GROUP BY booking_platform;

--12)

SELECT * FROM Airline

WHERE airline_code ~ '[0-9]';

--13)

SELECT * FROM Airline

ORDER BY created_at DESC LIMIT 5;

--14)

SELECT * FROM Baggage_check

WHERE booking_id BETWEEN 1 AND 10

AND check_result <> 'OK';

--15)

SELECT * FROM Baggage_check

WHERE DATE_TRUNC('month', updated_at) = DATE_TRUNC('month', created_at)

AND updated_at < created_at;

1)

The screenshot shows a PostgreSQL IDE interface. The left sidebar displays the 'Object Explorer' with a tree view of databases, including 'airport_dbb' and 'postgres'. The main window is titled 'airport_dbb/postgres@local*' and contains a 'Query' editor with the following SQL code:

```
250 ('Extra screening', 2, 2),
251 ('OK', 3, 3),
252 ('Overweight', 4, 4);
253
254 INSERT INTO Security_check (check_result, passenger_id)
255 VALUES
256 ('Cleared', 1),
257 ('Secondary', 2),
258 ('Cleared', 3),
259 ('Cleared', 4);
260
261 INSERT INTO Booking_flight (booking_id, flight_id)
262 VALUES
263 (1, 1),
264 (2, 2),
265 (3, 1),
266 (4, 3);
267
268
269 --1)
270 SELECT * FROM Passengers
271 WHERE last_name = first_name;
272
```

Below the query editor, the 'Data Output' tab is active, showing a table with 8 columns: passenger_id, first_name, last_name, date_of_birth, gender, country_of_citizenship, country_of_residence, and passport_number. The table contains one row of data.

| passenger_id | first_name | last_name | date_of_birth | gender | country_of_citizenship | country_of_residence | passport_number |
|--------------|------------|-----------|---------------|--------|------------------------|----------------------|-----------------|
| 1 | Dana | Dana | 1999-06-15 | Female | Kazakhstan | Kazakhstan | KZ2222333 |

The status bar at the bottom indicates 'Total rows: 1' and 'Query complete 00:00:00.055'.

2)

The screenshot shows a database management tool interface with a dark theme. The left sidebar displays a tree view of databases, including 'airport_dbb' and 'postgres'. The main area is split into a 'Query' editor and a 'Data Output' pane. The 'Query' editor contains SQL code for inserting data into 'Booking_flight' and selecting distinct last names from 'Passengers'. The 'Data Output' pane shows the results of the second query, displaying four rows of last names: Dana, Serik, Smith, and Nursultanov.

Query Editor:

```
255 VALUES
256 ('Cleared', 1),
257 ('Secondary', 2),
258 ('Cleared', 3),
259 ('Cleared', 4);
260
261 INSERT INTO Booking_flight (booking_id, flight_id)
262 VALUES
263 (1, 1),
264 (2, 2),
265 (3, 1),
266 (4, 3);
267
268
269 --1)
270 SELECT * FROM Passengers
271 WHERE last_name = first_name;
272
273
274 2)
275 SELECT DISTINCT last_name
276 FROM Passengers;
277
```

Data Output:

| | last_name character varying (50) |
|---|-------------------------------------|
| 1 | Dana |
| 2 | Serik |
| 3 | Smith |
| 4 | Nursultanov |

Total rows: 4 Query complete 00:00:00.094 LF Ln 275, Col 1

3)

The screenshot shows the PostgreSQL IDE interface. The left sidebar displays the Object Explorer with the database structure. The main window shows the SQL editor with the following query:

```

259 ('Cleared', 4);
260
261 INSERT INTO Booking_flight (booking_id, flight_id)
262 VALUES
263 (1, 1),
264 (2, 2),
265 (3, 1),
266 (4, 3);
267
268
269
270 --1)
271 SELECT * FROM Passengers
272 WHERE last_name = first_name;
273
274 2)
275 SELECT DISTINCT last_name
276 FROM Passengers;
277
278 3)
279 SELECT * FROM Passengers
280 WHERE gender = 'Male' AND date_of_birth BETWEEN '1990-01-01' AND '2000-12-31';
281

```

The Data Output tab shows the results of the query, displaying a single row of data:

| passenger_id [PK] integer | first_name character varying (50) | last_name character varying (50) | date_of_birth date | gender character varying (50) | country_of_citizenship character varying (50) | country_of_residence character varying (50) | passport_number character varying (20) |
|------------------------------|--------------------------------------|-------------------------------------|-----------------------|----------------------------------|--|--|---|
| 1 | Aidos | Nursultanov | 1995-03-12 | Male | Kazakhstan | Kazakhstan | KZ1234567 |

Total rows: 1 Query complete 00:00:00.076

4)

The screenshot shows the PostgreSQL IDE interface. The left sidebar displays the Object Explorer with the database structure. The main window shows the SQL editor with the following query:

```

267
268
269
270 --1)
271 SELECT * FROM Passengers
272 WHERE last_name = first_name;
273
274 2)
275 SELECT DISTINCT last_name
276 FROM Passengers;
277
278 3)
279 SELECT * FROM Passengers
280 WHERE gender = 'Male' AND date_of_birth BETWEEN '1990-01-01' AND '2000-12-31';
281
282 4)
283 SELECT DATE_TRUNC('month', created_at) AS month,
284        SUM(ticket_price) AS total_sales
285 FROM Booking
286 GROUP BY month
287 ORDER BY month;
288
289

```

The Data Output tab shows the results of the query, displaying a single row of data:

| month timestamp without time zone | total_sales numeric |
|--------------------------------------|------------------------|
| 2025-09-01 00:00:00 | 135000.00 |

Total rows: 1 Query complete 00:00:00.056

5)

The screenshot shows the DBeaver interface with the 'airport_db' database selected. The SQL editor contains a query with five parts, and the 'Data Output' tab shows the results of the first part.

```

--1)
SELECT * FROM Passengers
WHERE last_name = first_name;

2)
SELECT DISTINCT last_name
FROM Passengers;

3)
SELECT * FROM Passengers
WHERE gender = 'Male' AND date_of_birth BETWEEN '1990-01-01' AND '2000-12-31';

4)
SELECT DATE_TRUNC('month', created_at) AS month,
SUM(ticket_price) AS total_sales
FROM Booking
GROUP BY month
ORDER BY month;

5)
SELECT f. * FROM Flights f
JOIN Airport a on f.arriving_airport_id = a.airport_id
WHERE a.country = 'China';
  
```

| flight_id | sch_departure_time | sch_arrival_time | departing_airport_id | arriving_airport_id | departing_gate | arriving_gate | airline_id |
|-----------|--------------------|---------------------|----------------------|---------------------|----------------|---------------|------------|
| 1 | 4 | 2025-09-27 14:00:00 | 2025-09-27 18:00:00 | 2 | 4 | D2 | D5 |

Total rows: 1 Query complete 00:00:00.065

6)

The screenshot shows the DBeaver interface with the 'airport_db' database selected. The SQL editor contains a query with six parts, and the 'Data Output' tab shows the results of the first part.

```

SELECT DISTINCT last_name
FROM Passengers;

3)
SELECT * FROM Passengers
WHERE gender = 'Male' AND date_of_birth BETWEEN '1990-01-01' AND '2000-12-31';

4)
SELECT DATE_TRUNC('month', created_at) AS month,
SUM(ticket_price) AS total_sales
FROM Booking
GROUP BY month
ORDER BY month;

5)
SELECT f. * FROM Flights f
JOIN Airport a on f.arriving_airport_id = a.airport_id
WHERE a.country = 'China';

6)
SELECT * FROM Airline
WHERE airline_country in ('France', 'Portugal', 'Poland')
AND created_at BETWEEN '2023-11-01' AND '2024-03-31';
  
```

| airline_id | airline_code | airline_name | airline_country | created_at | updated_at |
|------------|--------------|--------------|-----------------|------------|------------|
|------------|--------------|--------------|-----------------|------------|------------|

Total rows: 0 Query complete 00:00:00.080

7)

The screenshot shows the DBeaver interface with the 'Query' tab selected. The query is as follows:

```

283 SELECT * FROM Passengers
284 WHERE gender = 'Male' AND date_of_birth BETWEEN '1990-01-01' AND '2000-12-31';
285
286 4)
287 SELECT DATE_TRUNC('month', created_at) AS month,
288        SUM(ticket_price) AS total_sales
289 FROM Booking
290 GROUP BY month
291 ORDER BY month;
292
293 5)
294 SELECT f. * FROM Flights f
295 JOIN Airport a on f.arriving_airport_id = a.airport_id
296 WHERE a.country = 'China';
297
298 6)
299 SELECT * FROM Airline
300 WHERE airline_country in ('France', 'Portugal', 'Poland')
301 AND created_at BETWEEN '2023-11-01' AND '2024-03-31';
302
303 7)
304 SELECT airline_name FROM Airline
305 WHERE airline_country = 'Kazakhstan';

```

The 'Data Output' tab shows the results of the last query (Query 7):

| airline_name |
|--------------|
| Airastana |

Showing rows: 1 to 1 | Page No: 1 | of 1

Total rows: 1 | Query complete 00:00:00.054 | LF | Ln 305, Col 38

8)

The screenshot shows the DBeaver interface with the 'Query' tab selected. The query is as follows:

```

288        SUM(ticket_price) AS total_sales
289 FROM Booking
290 GROUP BY month
291 ORDER BY month;
292
293 5)
294 SELECT f. * FROM Flights f
295 JOIN Airport a on f.arriving_airport_id = a.airport_id
296 WHERE a.country = 'China';
297
298 6)
299 SELECT * FROM Airline
300 WHERE airline_country in ('France', 'Portugal', 'Poland')
301 AND created_at BETWEEN '2023-11-01' AND '2024-03-31';
302
303 7)
304 SELECT airline_name FROM Airline
305 WHERE airline_country = 'Kazakhstan';
306
307 8)
308 UPDATE Booking
309 SET ticket_price = ticket_price * 0.9
310 WHERE created_at < '2023-11-01';

```

The 'Data Output' tab shows the results of the last query (Query 8):

| UPDATE |
|--------|
| 0 |

Query returned successfully in 77 msec.

Total rows: | Query complete 00:00:00.077 | LF | Ln 310, Col 33

9)

The screenshot shows the DBeaver interface with the 'Query' tab selected. The query is as follows:

```

5)
SELECT f. * FROM Flights f
JOIN Airport a on f.arriving_airport_id = a.airport_id
WHERE a.country = 'China';

6)
SELECT * FROM Airline
WHERE airline_country in ('France', 'Portugal', 'Poland')
AND created_at BETWEEN '2023-11-01' AND '2024-03-31';

7)
SELECT airline_name FROM Airline
WHERE airline_country = 'Kazakhstan';

8)
UPDATE Booking
SET ticket_price = ticket_price * 0.9
WHERE created_at < '2023-11-01';

9)
SELECT * FROM Baggage
WHERE weight_in_kg > 20
ORDER BY weight_in_kg DESC LIMIT 3;

```

The 'Data Output' tab shows the results of the last query (Query 9):

| baggage_id [PK] integer | weight_in_kg numeric (4,2) | created_at timestamp without time zone | updated_at timestamp without time zone | booking_id integer | |
|----------------------------|-------------------------------|---|---|----------------------------|---|
| 1 | 4 | 30.20 | 2025-09-30 18:05:32.619902 | 2025-09-30 18:05:32.619902 | 4 |
| 2 | 3 | 25.70 | 2025-09-30 18:05:32.619902 | 2025-09-30 18:05:32.619902 | 3 |
| 3 | 1 | 23.50 | 2025-09-30 18:05:32.619902 | 2025-09-30 18:05:32.619902 | 1 |

Total rows: 3 Query complete 00:00:00.079

10)

The screenshot shows the DBeaver interface with the 'Query' tab selected. The query is as follows:

```

6)
SELECT * FROM Airline
WHERE airline_country in ('France', 'Portugal', 'Poland')
AND created_at BETWEEN '2023-11-01' AND '2024-03-31';

7)
SELECT airline_name FROM Airline
WHERE airline_country = 'Kazakhstan';

8)
UPDATE Booking
SET ticket_price = ticket_price * 0.9
WHERE created_at < '2023-11-01';

9)
SELECT * FROM Baggage
WHERE weight_in_kg > 20
ORDER BY weight_in_kg DESC LIMIT 3;

10)
SELECT first_name, last_name FROM Passengers
ORDER BY date_of_birth DESC LIMIT 1;

```

The 'Data Output' tab shows the results of the last query (Query 10):

| first_name character varying (50) | last_name character varying (50) |
|--------------------------------------|-------------------------------------|
| Aliya | Serik |

Total rows: 1 Query complete 00:00:00.075

11)

The screenshot shows the DBeaver SQL editor interface. The left sidebar displays the 'Object Explorer' with a tree view of the 'airport_db' database. The main editor area shows a SQL query with line numbers 302 to 324. The query includes a SELECT statement for 'airline_name' from the 'Airline' table, an UPDATE statement for 'Booking' table, and a SELECT statement for 'booking_platform' and 'MIN(ticket_price)' from the 'Booking' table. The 'Data Output' tab at the bottom shows the results of the query, displaying a table with 3 rows and 2 columns: 'booking_platform' and 'cheapest_price'.

```

302
303
304 SELECT airline_name FROM Airline
305 WHERE airline_country = 'Kazakhstan';
306
307
308
309 UPDATE Booking
310 SET ticket_price = ticket_price * 0.9
311 WHERE created_at < '2023-11-01';
312
313
314
315 SELECT * FROM Baggage
316 WHERE weight_in_kg > 20
317 ORDER BY weight_in_kg DESC LIMIT 3;
318
319
320
321
322 SELECT first_name, last_name FROM Passengers
323 ORDER BY date_of_birth DESC LIMIT 1;
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
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

```

| booking_platform | cheapest_price |
|------------------|----------------|
| Website | 30000.00 |
| Agent | 48000.00 |
| Mobile App | 12000.00 |

Total rows: 3 Query complete 00:00:00.071

12)

The screenshot shows the DBeaver SQL editor interface. The left sidebar displays the 'Object Explorer' with a tree view of the 'airport_db' database. The main editor area shows a SQL query with line numbers 306 to 328. The query includes an UPDATE statement for 'Booking' table, a SELECT statement for 'booking_platform' and 'MIN(ticket_price)' from the 'Booking' table, and a SELECT statement for 'airline_name' from the 'Airline' table. The 'Data Output' tab at the bottom shows the results of the query, displaying a table with 5 rows and 7 columns: 'airline_id', 'airline_code', 'airline_name', 'airline_country', 'created_at', and 'updated_at'.

```

306
307
308
309 UPDATE Booking
310 SET ticket_price = ticket_price * 0.9
311 WHERE created_at < '2023-11-01';
312
313
314
315 SELECT * FROM Baggage
316 WHERE weight_in_kg > 20
317 ORDER BY weight_in_kg DESC LIMIT 3;
318
319
320
321
322 SELECT first_name, last_name FROM Passengers
323 ORDER BY date_of_birth DESC LIMIT 1;
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
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

```

| airline_id | airline_code | airline_name | airline_country | created_at | updated_at |
|------------|--------------|--------------|-----------------|----------------------------|----------------------------|
| 1 | KZ001 | KazAir | Turkey | 2025-09-30 18:02:05.682893 | 2025-09-30 18:02:05.682893 |
| 2 | FR001 | AirEasy | France | 2025-09-30 18:02:25.204767 | 2025-09-30 18:02:25.204767 |
| 3 | BR001 | FlyHigh | Brazil | 2025-09-30 18:02:25.204767 | 2025-09-30 18:02:25.204767 |
| 4 | PL001 | FlyFly | Poland | 2025-09-30 18:02:25.204767 | 2025-09-30 18:02:25.204767 |
| 5 | KZ002 | Airastana | Kazakhstan | 2025-09-30 19:11:07.30127 | 2025-09-30 19:11:07.30127 |

Total rows: 5 Query complete 00:00:00.094

13)

The screenshot shows the DBeaver interface with the 'Query' tab selected. The query is as follows:

```

310 WHERE created_at < '2023-11-01';
311
312 9)
313 SELECT * FROM Baggage
314 WHERE weight_in_kg > 20
315 ORDER BY weight_in_kg DESC LIMIT 3;
316
317 10)
318 SELECT first_name, last_name FROM Passengers
319 ORDER BY date_of_birth DESC LIMIT 1;
320
321 11)
322 SELECT booking_platform, MIN(ticket_price) AS cheapest_price
323 FROM Booking
324 GROUP BY booking_platform;
325
326 12)
327 SELECT * FROM Airline
328 WHERE airline_code ~ '[0-9]';
329
330 13)
331 SELECT * FROM Airline
332 ORDER BY created_at DESC LIMIT 5;

```

The 'Data Output' tab shows the results of the query, displaying 5 rows of data from the 'Airline' table. The columns are: airline_id (PK) integer, airline_code character varying (30), airline_name character varying (50), airline_country character varying (50), created_at timestamp without time zone, and updated_at timestamp without time zone.

| airline_id (PK) integer | airline_code character varying (30) | airline_name character varying (50) | airline_country character varying (50) | created_at timestamp without time zone | updated_at timestamp without time zone |
|-------------------------|-------------------------------------|-------------------------------------|--|--|--|
| 1 | 5 KZ002 | Airastana | Kazakhstan | 2025-09-30 19:11:07.30127 | 2025-09-30 19:11:07.30127 |
| 2 | 2 FR001 | AirEasy | France | 2025-09-30 18:02:25.204767 | 2025-09-30 18:02:25.204767 |
| 3 | 3 BR001 | FlyHigh | Brazil | 2025-09-30 18:02:25.204767 | 2025-09-30 18:02:25.204767 |
| 4 | 4 PL001 | FlyFly | Poland | 2025-09-30 18:02:25.204767 | 2025-09-30 18:02:25.204767 |
| 5 | 1 KZ001 | KazAir | Turkey | 2025-09-30 18:02:05.682893 | 2025-09-30 18:02:05.682893 |

Total rows: 5 Query complete 00:00:00.184

14)

The screenshot shows the DBeaver interface with the 'Query' tab selected. The query is as follows:

```

316
317 10)
318 SELECT first_name, last_name FROM Passengers
319 ORDER BY date_of_birth DESC LIMIT 1;
320
321 11)
322 SELECT booking_platform, MIN(ticket_price) AS cheapest_price
323 FROM Booking
324 GROUP BY booking_platform;
325
326 12)
327 SELECT * FROM Airline
328 WHERE airline_code ~ '[0-9]';
329
330 13)
331 SELECT * FROM Airline
332 ORDER BY created_at DESC LIMIT 5;
333
334 14)
335 SELECT * FROM Baggage_check
336 WHERE booking_id BETWEEN 1 AND 10
337 AND check_result <> 'OK';
338

```

The 'Data Output' tab shows the results of the query, displaying 2 rows of data from the 'Baggage_check' table. The columns are: baggage_check_id (PK) integer, check_result character varying (50), created_at timestamp without time zone, updated_at timestamp without time zone, booking_id integer, and passenger_id integer.

| baggage_check_id (PK) integer | check_result character varying (50) | created_at timestamp without time zone | updated_at timestamp without time zone | booking_id integer | passenger_id integer |
|-------------------------------|-------------------------------------|--|--|--------------------|----------------------|
| 1 | 2 Extra screening | 2025-09-30 18:05:35.814478 | 2025-09-30 18:05:35.814478 | 2 | 2 |
| 2 | 4 Overweight | 2025-09-30 18:05:35.814478 | 2025-09-30 18:05:35.814478 | 4 | 4 |

Total rows: 2 Query complete 00:00:00.080

15)

The screenshot displays a database management tool interface with a dark theme. The left sidebar shows a tree view of databases, including 'airport_dbb' and 'postgres'. The main area is divided into a 'Query' editor and a 'Query History' pane. The 'Query' editor contains a SQL query with line numbers 320 through 342. The 'Query History' pane shows the same query. Below the query editor, there is a 'Data Output' section with a table of results. The table has six columns: 'baggage_check_id', 'check_result', 'created_at', 'updated_at', 'booking_id', and 'passenger_id'. The 'Data Output' section shows a single row of data. At the bottom, a status bar indicates 'Total rows: 0' and 'Query complete 00:00:00.070'. A green message box at the bottom right states 'Successfully run. Total query runtime: 70 msec. 0 rows affected.'

```
320
321
322 SELECT booking_platform, MIN(ticket_price) AS cheapest_price
323 FROM Booking
324 GROUP BY booking_platform;
325
326
327 SELECT * FROM Airline
328 WHERE airline_code ~ '[0-9]';
329
330
331 SELECT * FROM Airline
332 ORDER BY created_at DESC LIMIT 5;
333
334
335 SELECT * FROM Baggage_check
336 WHERE booking_id BETWEEN 1 AND 10
337 AND check_result <> 'OK';
338
339
340
341 SELECT * FROM Baggage_check
342 WHERE DATE_TRUNC('month', updated_at) = DATE_TRUNC('month', created_at)
AND updated_at < created_at;
```

| baggage_check_id | check_result | created_at | updated_at | booking_id | passenger_id |
|------------------|------------------------|-----------------------------|-----------------------------|------------|--------------|
| [PK] integer | character varying (50) | timestamp without time zone | timestamp without time zone | integer | integer |

Total rows: 0 Query complete 00:00:00.070

Successfully run. Total query runtime: 70 msec. 0 rows affected.