

A)

The screenshot shows a database console interface with a query editor and an output window. The query editor contains the following SQL code:

```
1 select airport_code, coordinates from airports
2 where city = 'Казань' or city = 'Москва'
3 ORDER BY airport_code DESC;
```

The output window displays the results of the query, showing 4 rows of data:

airport_code	coordinates
VK0	(37.2615013123, 55.5914993286)
SV0	(37.4146, 55.972599)
KZN	(49.278701782227, 55.606201171875)
DME	(37.90629959106445, 55.40879821777344)

B)

The screenshot shows a database console interface with a query editor and an output window. The query editor contains the following SQL code:

```
1 select airport_code, coordinates from airports
2 where city = 'Казань' or city = 'Москва'
3 ORDER BY airport_code DESC;
4
5 SELECT (airport_code || airport_name || city || coordinates || timezone) AS full_information
6 FROM airports_data
7 ORDER BY full_information;
```

The output window displays the results of the query, showing 104 rows of data. The first few rows are:

full_information
AAQ{"en": "Anapa Vityazevo Airport", "ru": "Витязево"}{"en": "Anapa", "ru": "Анапа"}(37.347301483154, 45.002101898193)Europe/...
ABA{"en": "Abakan Airport", "ru": "Абакан"}{"en": "Abakan", "ru": "Абакан"}(91.38500213623047, 53.7400016784668)Asia/Krasnoy...
AER{"en": "Sochi International Airport", "ru": "Сочи"}{"en": "Sochi", "ru": "Сочи"}(39.956600189209, 43.449901508011)Europe/...
ARH{"en": "Talagi Airport", "ru": "Талари"}{"en": "Arkhangel'sk", "ru": "Архангельск"}(40.71670150756836, 64.60030364990234)E...
ASF{"en": "Astrakhan Airport", "ru": "Астрахань"}{"en": "Astrakhan", "ru": "Астрахань"}(48.0063018799, 46.2832984924)Europe/...
BAX{"en": "Barnaul Airport", "ru": "Барнаул"}{"en": "Barnaul", "ru": "Барнаул"}(83.53849792480469, 53.363800048828125)Asia/K...
BQS{"en": "Ignatyev Airport", "ru": "Игнатьево"}{"en": "Blagoveschensk", "ru": "Благовещенск"}(127.41200256347656, 50.42539...
BTK{"en": "Bratsk Airport", "ru": "Братск"}{"en": "Bratsk", "ru": "Братск"}(101.697998046875, 56.370601654052734)Asia/Irkutsk
BZK{"en": "Bryansk Airport", "ru": "Брянск"}{"en": "Bryansk", "ru": "Брянск"}(34.176399231, 53.214199066199996)Europe/Moscow
CEE{"en": "Cherepovets Airport", "ru": "Череповец"}{"en": "Cherepovets", "ru": "Череповец"}(38.015800476100004, 59.273601532...
CEK{"en": "Chelyabinsk Balandino Airport", "ru": "Челябинск"}{"en": "Chelyabinsk", "ru": "Челябинск"}(61.5033, 55.305801)Asi...
CNN{"en": "Chulman Airport", "ru": "Чульман"}{"en": "Neryungri", "ru": "Нерюнгри"}(124.91400146484, 56.913898468018)Asia/Yak...

C)

Database Explorer

console [demo@localhost] console_1 [demo@localhost] x demo airports [demo@localhost] airports_data [demo@localhost] bocl

```

5 SELECT (airport_code || airport_name || city || coordinates || timezone) AS full_information
6 FROM airports_data
7 ORDER BY full_information;
8
9 ✓ SELECT airport_name, count(departure_airport) from airports_data a, flights f
10 where f.departure_airport = a.airport_code and a.airport_code in ('KZN', 'DME', 'OVB', 'IKT', 'LED', 'SVO')
11 group by airport_name
12 order by count(departure_airport) desc;

```

Services

TX

airports_data 207 ms

airports 207 ms

bookings 160 ms

bookings 160 ms

flights 223 ms

flights 223 ms

console_1 156 ms

Output Result 5

airport_name	count
{ "en": "Domodedovo International Airport", "ru": "Домодедово" }	6376
{ "en": "Sheremetyevo International Airport", "ru": "Шереметьево" }	5912
{ "en": "Pulkovo Airport", "ru": "Пулково" }	3769
{ "en": "Tolmachevo Airport", "ru": "Толмачёво" }	2891
{ "en": "Kazan International Airport", "ru": "Казань" }	934
{ "en": "Irkutsk Airport", "ru": "Иркутск" }	727

Database Consoles > demo@localhost > console_1 [demo@localhost]

12:40 CRLF UTF-8 4 spaces

17:14 07.12.2023

D)

Database Explorer

console [demo@localhost] console_1 [demo@localhost] x demo airports [demo@localhost] airports_data [demo@localhost] bocl

```

5 SELECT (airport_code || airport_name || city || coordinates || timezone) AS full_information
6 FROM airports_data
7 ORDER BY full_information;
8
9 ✓ SELECT airport_name, count(departure_airport) from airports_data a, flights f
10 where f.departure_airport = a.airport_code and a.airport_code not in ('KZN', 'DME', 'OVB', 'IKT', 'LED', 'SVO')
11 group by airport_name
12 order by count(departure_airport);

```

Services

TX

airports_data 207 ms

airports 207 ms

bookings 160 ms

bookings 160 ms

flights 223 ms

flights 223 ms

console_1 150 ms

Output Result 6

airport_name	count
{ "en": "Usinsk Airport", "ru": "Усинск" }	34
{ "en": "Komsomolsk-on-Amur Airport", "ru": "Хурба" }	35
{ "en": "Nyagan Airport", "ru": "Нягань" }	51
{ "en": "Polyarny Airport", "ru": "Полярный" }	51
{ "en": "Yelizovo Airport", "ru": "Елизово" }	52
{ "en": "Ivanovo South Airport", "ru": "Иваново-Южный" }	68
{ "en": "Sokol Airport", "ru": "Магадан" }	70
{ "en": "Ugolny Airport", "ru": "Анадырь" }	70
{ "en": "Kyzyl Airport", "ru": "Кызыл" }	86
{ "en": "Lipetsk Airport", "ru": "Липецк" }	86
{ "en": "Nefteyugansk Airport", "ru": "Нефтеюганск" }	87
{ "en": "Beloyarskiy Airport", "ru": "Белоярский" }	184

Database Consoles > demo@localhost > console_1 [demo@localhost]

12:35 CRLF UTF-8 4 spaces

17:17 07.12.2023

E)

Database Explorer

demo@localhost

- aircraft_code char(3)
- actual_departure timestamp with time zone
- actual_arrival timestamp with time zone
- keys 2
- foreign keys 3
- indexes 2
- checks 3
- seats
- ticket_flights
- tickets

console [demo@localhost] console_1 [demo@localhost] x demo airports [demo@localhost] airports_data [demo@localhost] book_1 [demo@localhost]

demo: bookings, public console_1

```

11 group by airport_name
12 order by count(departure_airport);
13
14 select flight_no, scheduled_departure, count(ticket_no) from flights f, ticket_flights t
15 where t.flight_id = f.flight_id
16 group by flight_no, scheduled_departure
17 having count(ticket_no) between 27 and 90
18 ORDER BY flight_no, scheduled_departure, count(ticket_no) DESC;
19

```

Output Result 7

flight_no	scheduled_departure	count
1 P60012	2017-07-11 07:55:00.000000 +00:00	31
2 P60013	2017-09-10 15:15:00.000000 +00:00	68
3 P60013	2017-09-11 15:15:00.000000 +00:00	50
4 P60013	2017-09-12 15:15:00.000000 +00:00	53
5 P60013	2017-09-13 15:15:00.000000 +00:00	43
6 P60014	2017-05-17 04:30:00.000000 +00:00	36
7 P60014	2017-05-18 04:30:00.000000 +00:00	40
8 P60014	2017-05-19 04:30:00.000000 +00:00	37
9 P60014	2017-05-20 04:30:00.000000 +00:00	39
10 P60014	2017-05-21 04:30:00.000000 +00:00	38
11 P60014	2017-05-22 04:30:00.000000 +00:00	38
12 P60014	2017-05-23 04:30:00.000000 +00:00	39
13 P60014	2017-05-24 04:30:00.000000 +00:00	39

Database Consoles > demo@localhost > console_1 [demo@localhost]

18:64 CRLF UTF-8 4 spaces

17:35 07.12.2023

F)

Database Explorer

demo@localhost

- aircraft_code char(3)
- actual_departure timestamp with time zone
- actual_arrival timestamp with time zone
- keys 2
- foreign keys 3
- indexes 2
- checks 3
- seats
- ticket_flights
- tickets

console [demo@localhost] console_1 [demo@localhost] x demo airports [demo@localhost] airports_data [demo@localhost] book_1 [demo@localhost]

demo: bookings, public console_1

```

16 group by flight_no, scheduled_departure
17 having count(ticket_no) between 27 and 90
18 ORDER BY flight_no, scheduled_departure, count(ticket_no) DESC;
19
20 select passenger_name as val from tickets
21 union
22 select airport_name from airports
23 order by val DESC;
24

```

Output val:text

val
1 Якутск
2 Элиста
3 Шереметьево
4 Чувшман
5 Чита
6 Череповец
7 Челябинск
8 Чебоксары
9 Хурба
10 Храброво
11 Хомутово
12 Ханты-Мансийск
13 Хабаровск

Database Consoles > demo@localhost > console_1 [demo@localhost]

22:34 CRLF UTF-8 4 spaces

17:33 07.12.2023

G)

The screenshot shows a database IDE with a dark theme. The left sidebar displays the 'Database Explorer' for a 'demo@localhost' connection, showing a schema with tables like 'aircrafts_data', 'airports', 'bookings', 'flights', 'ticket_flights', and 'tickets'. The main editor shows a SQL query in a 'console_1' window. The query is as follows:

```

group by flight_no, scheduled_departure
having count(ticket_no) between 27 and 90
ORDER BY flight_no, scheduled_departure, count(ticket_no) DESC;

select passenger_name as val, 'passenger' as type from tickets
union
select airport_name, 'airport' as type from airports
order by type, val DESC;

```

The 'Output' pane shows the results of the query, which are 12 rows of airport names and their type ('airport').

val	type
1 Якутск	airport
2 Элиста	airport
3 Шереметьево	airport
4 Чulьман	airport
5 Чита	airport
6 Череповец	airport
7 Челябинск	airport
8 Чебоксары	airport
9 Хурба	airport
10 Храброво	airport
11 Хомутово	airport
12 Ханты-Мансийск	airport

The bottom status bar indicates the current settings: 23:25, CRLF, UTF-8, 4 spaces.

Н) *у меня medium версия

The screenshot shows the same database IDE with a different SQL query in the 'console_1' window. The query is as follows:

```

select airport_name, 'airport' as type from airports
order by type, val DESC;

SELECT COUNT(f.flight_id) FROM flights f
LEFT JOIN ticket_flights AS t ON f.flight_id = t.flight_id
WHERE t.ticket_no IS NULL;

```

The 'Output' pane shows the results of the query, which is a single row with a count of 20490.

count
1 20490

The bottom status bar indicates the current settings: 27:27, CRLF, UTF-8, 4 spaces.

И)

Database Explorer

- demo@localhost
 - columns 3
 - book_ref char(6)
 - book_date timestamp with time zone
 - totalAmount numeric(10,2)
 - keys 1
 - indexes 1
 - flights
 - columns 10
 - flightId integer = nextval('flights_f...')
 - flight_no char(6)

console_1 [demo@localhost] x demo airports [demo@localhost] airports_data [demo@localhost] bookings [demo@localhost]

demo: bookings, public console_1

```

28
29
30
31
32
33
34
35
36
37
38

```

```

select distinct on (departure_airport) departure_airport, avg(a.range)
over (partition by f.departure_airport range_avg, avg(count(f.flight_id))
over (partition by f.departure_airport) count_avg
from flights f
join ticket_flights t on f.flight_id = t.flight_id
join aircrafts_data a on f.aircraft_code = a.aircraft_code
where scheduled_departure between '2017-06-01' and '2017-07-01'
and extract(MONTH from scheduled_departure) = 6
group by departure_airport, a.range
order by departure_airport, range_avg, count_avg desc;

```

Output Result 14

	departure_air...	range_avg	count_avg
1	AAQ	3600	3039.5
2	ABA	3950	581
3	AER	5257.1428571428571429	2891.1428571428571429
4	ARH	1950	1635.5
5	ASF	2700	1710
6	BAX	2850	1089
7	BQS	3000	1323
8	BTk	6700	1746
9	BZK	3000	10081
10	CEE	2100	43
11	CEK	2300	2059.3333333333333333

Database Consoles > demo@localhost > console_1 [demo@localhost]

36:48 CRLF UTF-8 4 spaces

21:58 07.12.2023

J)

Database Explorer

- demo@localhost
 - columns 3
 - book_ref char(6)
 - book_date timestamp with time zone
 - totalAmount numeric(10,2)
 - keys 1
 - indexes 1
 - flights
 - columns 10
 - flightId integer = nextval('flights_f...')
 - flight_no char(6)

console_1 [demo@localhost] x demo airports [demo@localhost] airports_data [demo@localhost] bookings [demo@localhost]

demo: bookings, public console_1

```

35
36
37
38
39
40
41
42
43
44
45

```

```

where scheduled_departure between '2017-06-01' and '2017-07-01'
and extract(MONTH from scheduled_departure) = 6
group by departure_airport, a.range
order by departure_airport, range_avg, count_avg desc;

select flight_no, min(amount) min_a, max(amount) max_a
from ticket_flights t, flights f
where t.flight_id = f.flight_id
group by flight_no;

```

Output Result 15

	flight_no	min_a	max_a
1	P60012	12300	13500
2	P60013	14000	42100
3	P60014	3300	9800
4	P60015	18700	20600
5	P60016	18700	20600
6	P60019	9500	10500
7	P60020	9500	10500
8	P60029	5300	5300
9	P60030	5300	5300
10	P60032	5300	5300
11	P60035	8700	8700

Database Consoles > demo@localhost > console_1 [demo@localhost]

43:20 CRLF UTF-8 4 spaces

22:01 07.12.2023