

Установка Hadoop/Hive и подготовка данных источника

- Установка docker-container

docker-compose up -d

```
PS D:\YandexDisk\Education\spark-otus\homework\hw-06\docker_hive> docker-compose up -d
time="2025-06-17T01:07:27+03:00" level=warning msg="D:\YandexDisk\Education\spark-otus\homework\hw-06\docker_hive\
\docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confus
ion"
[+] Running 65/71
  ✓hive-metastore-postgresql Pulled 71.9s
  ✓hive-metastore Pulled 82.2s
  ✓presto-coordinator Pulled 94.7s
  ✓hive-server Pulled 82.2s
  ✓datanode Pulled 79.7s
  ✓namenode Pulled 79.7s
[+] Running 9/9
  ✓Network docker_hive_default Created 0.1s
  ✓Volume "docker_hive_namenode" Created 0.0s
  ✓Volume "docker_hive_datanode" Created 0.0s
  ✓Container docker_hive-namenode-1 Started 2.8s
  ✓Container docker_hive-hive-server-1 Started 3.1s
  ✓Container docker_hive-datanode-1 Started 2.7s
  ✓Container docker_hive-hive-metastore-postgresql-1 Started... 2.7s
  ✓Container docker_hive-presto-coordinator-1 Started 3.0s
  ✓Container docker_hive-hive-metastore-1 Started 3.0s
PS D:\YandexDisk\Education\spark-otus\homework\hw-06\docker_hive>
```

- Копирование данных с Kaggle

mkdir -p data

cd data

curl -L -o flights-and-airports-data.zip

"https://www.kaggle.com/api/v1/datasets/download/tylerx/flights-and-airports-data"

```
default@Main:~$ mkdir -p data
cd data
default@Main:~/data$ curl -L -o flights-and-airports-data.zip "https://www.kaggle.com/api/v1/datasets/download/tylerx/fl
ights-and-airports-data"
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left   Speed
  0     0    0     0    0     0      0  0 0:00:03  0:00:03  0:00:03  16.5M
```

- Распаковка архива

unzip flights-and-airports-data.zip

```
default@Main:~/data$ unzip flights-and-airports-data.zip
Archive:  flights-and-airports-data.zip
  inflating: airports.csv
  inflating: flights.csv
  inflating: raw-flight-data.csv
```

- Исправление кавычек в файлах

sed -i 's/"/@/g' airports.csv

sed -i 's/"/@/g' flights.csv

sed -i 's/"/@/g' raw-flight-data.csv

```
default@Main:~/data$ sed -i 's/"/@/g' airports.csv
default@Main:~/data$ sed -i 's/"/@/g' flights.csv
default@Main:~/data$ sed -i 's/"/@/g' raw-flight-data.csv
```

- Копирование файлов в контейнер namenode

docker cp airports.csv 1cb4c5799700:/tmp/

docker cp flights.csv 1cb4c5799700:/tmp/

```
default@Main:~/data$ docker cp airports.csv 1cb4c5799700:/tmp/
Successfully copied 17.9kB to 1cb4c5799700:/tmp/
default@Main:~/data$ docker cp flights.csv 1cb4c5799700:/tmp/
Successfully copied 72.1MB to 1cb4c5799700:/tmp/
```

- Создание директорий в HDFS

```
docker exec -it 1cb4c5799700 bash
```

```
hdfs dfs -mkdir -p /user/hive/warehouse/flight_analysis.db/airports
```

```
hdfs dfs -mkdir -p /user/hive/warehouse/flight_analysis.db/flights
```

```
root@1cb4c5799700:/# hdfs dfs -mkdir -p /user/hive/warehouse/flight_analysis.db/airports
root@1cb4c5799700:/# hdfs dfs -mkdir -p /user/hive/warehouse/flight_analysis.db/flights
```

- Загрузка данных в HDFS

```
hdfs dfs -put /tmp/airports.csv /user/hive/warehouse/flight_analysis.db/airports/
```

```
hdfs dfs -put /tmp/flights.csv /user/hive/warehouse/flight_analysis.db/flights/
```

```
root@1cb4c5799700:/# hdfs dfs -put /tmp/airports.csv /user/hive/warehouse/flight_analysis.db/airports/
root@1cb4c5799700:/# hdfs dfs -put /tmp/flights.csv /user/hive/warehouse/flight_analysis.db/flights/
```

- Создание БД

```
create database if not exists src;
```

```
create database if not exists ods;
```

```
create database if not exists dm;
```

Создание таблиц слоя SRC (слой источника)

- Создание таблицы «src.airports»

```
CREATE EXTERNAL TABLE src.airports (  
  airport_id INT,  
  city STRING,  
  state STRING,  
  name STRING  
)  
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY ','  
STORED AS TEXTFILE  
LOCATION '/user/hive/warehouse/flight_analysis.db/airports'  
TBLPROPERTIES ("skip.header.line.count"="1");
```

The screenshot shows a SQL IDE interface. The top pane displays the SQL code for creating the `src.airports` table and a query to select all data from it. The bottom pane shows the results of the query in a table format.

```
CREATE EXTERNAL TABLE src.airports (  
  airport_id INT,  
  city STRING,  
  state STRING,  
  name STRING  
)  
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY ','  
STORED AS TEXTFILE  
LOCATION '/user/hive/warehouse/flight_analysis.db/airports'  
TBLPROPERTIES ("skip.header.line.count"="1");  
  
select * from src.airports
```

Results 1 ×

select * from src.airports | Enter a SQL expression to filter results (use Ctrl+Space)

	airport_id	city	state	name
1	10,165	Adak Island	AK	Adak
2	10,299	Anchorage	AK	Ted Stevens Anchorage International
3	10,304	Aniak	AK	Aniak Airport
4	10,754	Barrow	AK	Wiley Post/Will Rogers Memorial
5	10,551	Bethel	AK	Bethel Airport
6	10,926	Cordova	AK	Merle K Mudhole Smith
7	14,709	Deadhorse	AK	Deadhorse Airport
8	11,336	Dillingham	AK	Dillingham Airport
9	11,630	Fairbanks	AK	Fairbanks International
10	11,997	Gustavus	AK	Gustavus Airport

- Создание таблицы «src.flights»

```
CREATE EXTERNAL TABLE src.flights (
  day_of_month INT,
  day_of_week INT,
  carrier STRING,
  origin_airport_id INT,
  dest_airport_id INT,
  dep_delay INT,
  arr_delay INT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/hive/warehouse/flight_analysis.db/flights'
TBLPROPERTIES ("skip.header.line.count"="1");
```

CREATE EXTERNAL TABLE src.flights (
 day_of_month INT,
 day_of_week INT,
 carrier STRING,
 origin_airport_id INT,
 dest_airport_id INT,
 dep_delay INT,
 arr_delay INT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/hive/warehouse/flight_analysis.db/flights'
TBLPROPERTIES ("skip.header.line.count"="1");

select * from src.flights

Results 1 x

select * from src.flights | Enter a SQL expression to filter results (use Ctrl+Space)

	123 day_of_month	123 day_of_week	A-Z carrier	123 origin_airport_id	123 dest_airport_id	123 dep_delay	123 arr_delay
1	19	5	DL	11,433	13,303	-3	1
2	19	5	DL	14,869	12,478	0	-8
3	19	5	DL	14,057	14,869	-4	-15
4	19	5	DL	15,016	11,433	28	24
5	19	5	DL	11,193	12,892	-6	-11
6	19	5	DL	10,397	15,016	-1	-19
7	19	5	DL	15,016	10,397	0	-1
8	19	5	DL	10,397	14,869	15	24
9	19	5	DL	10,397	10,423	33	34
10	19	5	DL	11,278	10,397	323	322

Создание таблиц слоя ODS (слой сырых данных)

- Создание таблицы «ods.airports»

```
CREATE TABLE ods.airports
```

```
(  
  airport_id INT,  
  city STRING,  
  state STRING,  
  name STRING  
)
```

```
stored as orc;
```

```
insert overwrite table ods.airports
```

```
select airport_id, city, state, name from src.airports;
```

The screenshot shows a SQL IDE with the following SQL code:

```
CREATE TABLE ods.airports  
(  
  airport_id INT,  
  city STRING,  
  state STRING,  
  name STRING  
)  
stored as orc;  
  
insert overwrite table ods.airports  
select airport_id, city, state, name from src.airports;  
  
select * from ods.airports
```

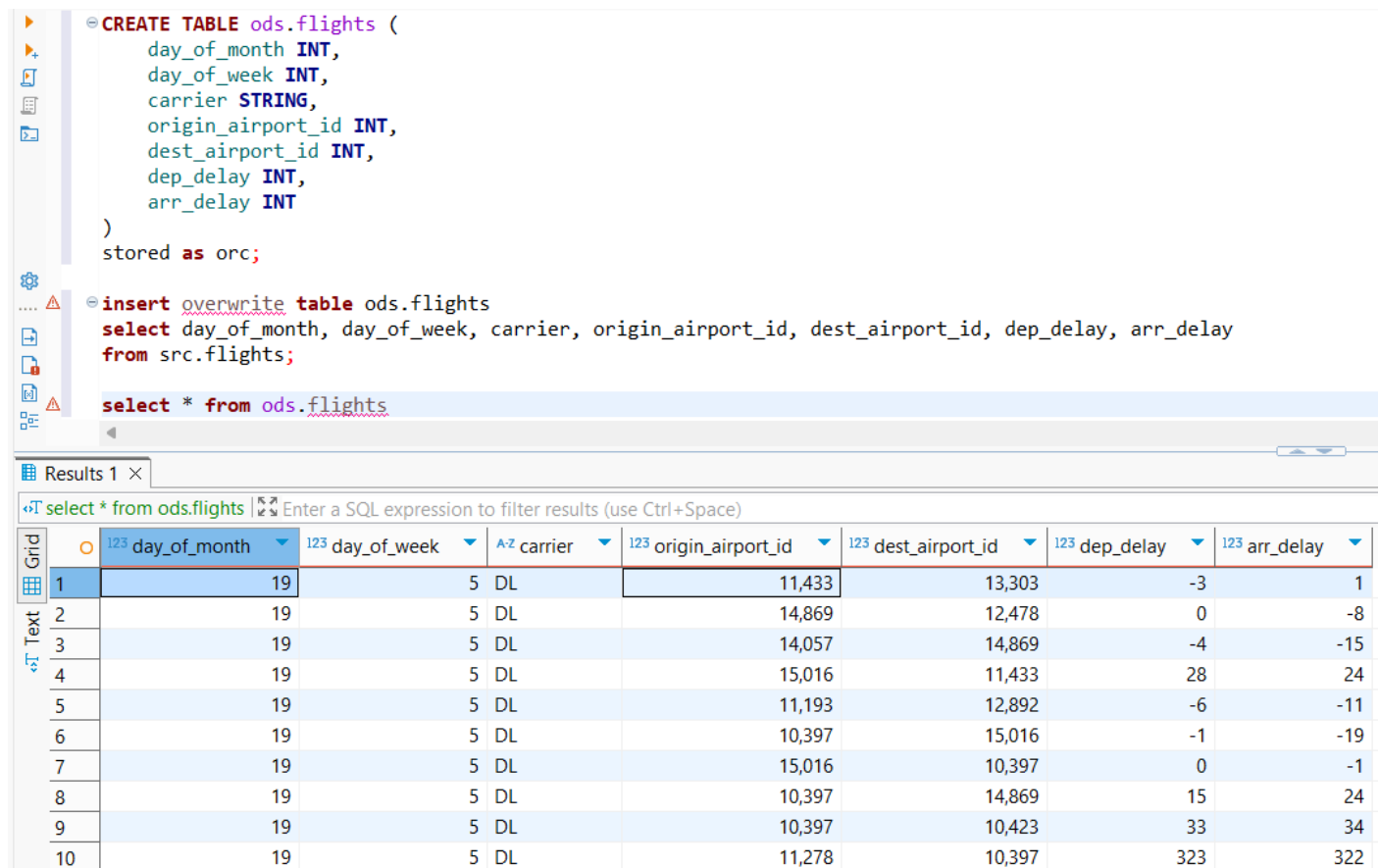
Below the code, the results of the query are displayed in a table with 5 columns: airport_id, city, state, and name. The table contains 10 rows of data.

airport_id	city	state	name
10,165	Adak Island	AK	Adak
10,299	Anchorage	AK	Ted Stevens Anchorage International
10,304	Aniak	AK	Aniak Airport
10,754	Barrow	AK	Wiley Post/Will Rogers Memorial
10,551	Bethel	AK	Bethel Airport
10,926	Cordova	AK	Merle K Mudhole Smith
14,709	Deadhorse	AK	Deadhorse Airport
11,336	Dillingham	AK	Dillingham Airport
11,630	Fairbanks	AK	Fairbanks International
11,997	Gustavus	AK	Gustavus Airport

- Создание таблицы «ods.flights»

```
CREATE TABLE ods.flights (
  day_of_month INT,
  day_of_week INT,
  carrier STRING,
  origin_airport_id INT,
  dest_airport_id INT,
  dep_delay INT,
  arr_delay INT
)
stored as orc;
```

```
insert overwrite table ods.flights
select day_of_month, day_of_week, carrier, origin_airport_id, dest_airport_id, dep_delay,
arr_delay
from src.flights;
```



The screenshot shows a SQL IDE interface. The top pane contains the SQL code for creating the 'ods.flights' table and inserting data from 'src.flights'. The bottom pane shows the results of the 'select * from ods.flights' query, displaying a table with 10 rows of flight data.

```
CREATE TABLE ods.flights (
  day_of_month INT,
  day_of_week INT,
  carrier STRING,
  origin_airport_id INT,
  dest_airport_id INT,
  dep_delay INT,
  arr_delay INT
)
stored as orc;

insert overwrite table ods.flights
select day_of_month, day_of_week, carrier, origin_airport_id, dest_airport_id, dep_delay, arr_delay
from src.flights;

select * from ods.flights
```

	day_of_month	day_of_week	carrier	origin_airport_id	dest_airport_id	dep_delay	arr_delay
1	19	5	DL	11,433	13,303	-3	1
2	19	5	DL	14,869	12,478	0	-8
3	19	5	DL	14,057	14,869	-4	-15
4	19	5	DL	15,016	11,433	28	24
5	19	5	DL	11,193	12,892	-6	-11
6	19	5	DL	10,397	15,016	-1	-19
7	19	5	DL	15,016	10,397	0	-1
8	19	5	DL	10,397	14,869	15	24
9	19	5	DL	10,397	10,423	33	34
10	19	5	DL	11,278	10,397	323	322

Создание витринного слоя DM

- Создание витрины с количеством рейсов по авиакомпаниям

```
CREATE TABLE dm.airline_flight_counts (  
    carrier string,  
    flight_count bigint  
)  
stored as orc;  
  
insert overwrite table dm.airline_flight_counts  
SELECT  
    carrier,  
    COUNT(*) as flight_count  
FROM  
    ods.flights  
GROUP BY  
    carrier  
ORDER BY  
    flight_count DESC;
```

The screenshot shows a SQL IDE interface. The top pane displays the SQL code for creating a table and inserting data. The bottom pane shows the results of a query executed against the table.

SQL Code:

```
CREATE TABLE dm.airline_flight_counts (  
    carrier string,  
    flight_count bigint  
)  
stored as orc;  
  
insert overwrite table dm.airline_flight_counts  
SELECT  
    carrier,  
    COUNT(*) as flight_count  
FROM  
    ods.flights  
GROUP BY  
    carrier  
ORDER BY  
    flight_count DESC;
```

Query:

```
select * from dm.airline_flight_counts
```

Results:

Grid	carrier	flight_count
1	WN	575,739
2	DL	381,657
3	AA	289,855
4	UA	286,418
5	US	233,321
6	OO	160,164
7	EV	157,928
8	B6	121,906
9	MQ	113,212
10	FL	92,702
11	9E	80,031
12	AS	68,555
13	YV	52,821
14	F9	35,738
15	VX	34,739
16	HA	17,432

- Создание витрины со средней задержкой по аэропортам отправления

```
CREATE TABLE dm.avg_departure_delay_by_airport
```

```
(
  airport_id int,
  airport_name string,
  city string,
  avg_dep_delay decimal
)
```

```
stored as orc;
```

```
insert overwrite table dm.avg_departure_delay_by_airport
```

```
SELECT
```

```
  a.airport_id,
  a.name as airport_name,
  a.city,
  AVG(f.dep_delay) as avg_dep_delay
```

```
FROM
```

```
  ods.flights f
```

```
INNER JOIN
```

```
  ods.airports a ON f.origin_airport_id = a.airport_id
```

```
GROUP BY
```

```
  a.airport_id, a.name, a.city
```

```
ORDER BY
```

```
  avg_dep_delay DESC;
```

The screenshot shows a SQL IDE with the following content:

```
CREATE TABLE dm.avg_departure_delay_by_airport
(
  airport_id int,
  airport_name string,
  city string,
  avg_dep_delay decimal
)
stored as orc;

insert overwrite table dm.avg_departure_delay_by_airport
SELECT
  a.airport_id,
  a.name as airport_name,
  a.city,
  AVG(f.dep_delay) as avg_dep_delay
FROM
  ods.flights f
INNER JOIN
  ods.airports a ON f.origin_airport_id = a.airport_id
GROUP BY
  a.airport_id, a.name, a.city
ORDER BY
  avg_dep_delay DESC;
```

Below the SQL code, there is a query result grid titled "Results 1". The grid shows the following data:

Grid	airport_id	airport_name	city	avg_dep_delay
1	13,232	Chicago Midway International	Chicago	16
2	13,930	Chicago O'Hare International	Chicago	16
3	11,618	Newark Liberty International	Newark	15
4	11,292	Denver International	Denver	14
5	11,298	Dallas/Fort Worth International	Dallas/Fort Worth	14
6	10,821	Baltimore/Washington International Thurgood Marshall	Baltimore	14
7	12,478	John F. Kennedy International	New York	14
8	14,771	San Francisco International	San Francisco	13
9	12,191	William P Hobby	Houston	13
10	12,264	Washington Dulles International	Washington	13

- Создание витрины с 10-ю самых популярных маршрутов

```
CREATE TABLE dm.popular_routes
```

```
(
  origin_airport string,
  dest_airport string,
  flight_count bigint
)
```

```
stored as orc;
```

```
insert overwrite table dm.popular_routes
```

```
SELECT
```

```
  a1.name as origin_airport,
  a2.name as dest_airport,
  COUNT(*) as flight_count
```

```
FROM
```

```
  ods.flights f
```

```
INNER JOIN
```

```
  ods.airports a1 ON f.origin_airport_id = a1.airport_id
```

```
INNER JOIN
```

```
  ods.airports a2 ON f.dest_airport_id = a2.airport_id
```

```
GROUP BY
```

```
  a1.name, a2.name
```

```
ORDER BY
```

```
  flight_count DESC
```

```
LIMIT 10;
```

The screenshot shows a SQL IDE with three SQL statements in the editor:

```
CREATE TABLE dm.popular_routes
(
  origin_airport string,
  dest_airport string,
  flight_count bigint
)
stored as orc;

insert overwrite table dm.popular_routes
SELECT
  a1.name as origin_airport,
  a2.name as dest_airport,
  COUNT(*) as flight_count
FROM
  ods.flights f
INNER JOIN
  ods.airports a1 ON f.origin_airport_id = a1.airport_id
INNER JOIN
  ods.airports a2 ON f.dest_airport_id = a2.airport_id
GROUP BY
  a1.name, a2.name
ORDER BY
  flight_count DESC
LIMIT 10;

SELECT origin_airport, dest_airport, flight_count
FROM dm.popular_routes;
```

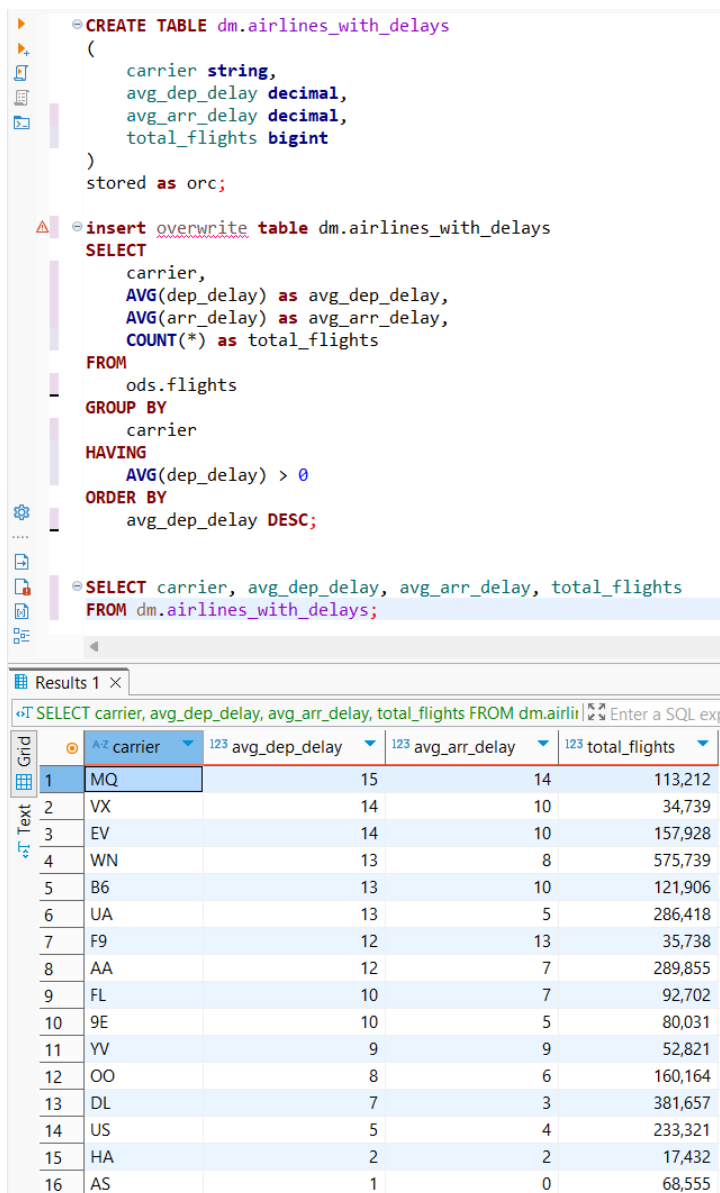
Below the editor, the 'Results 1' tab is active, displaying a grid of 10 rows and 3 columns. The columns are 'origin_airport', 'dest_airport', and 'flight_count'. The data is as follows:

	origin_airport	dest_airport	flight_count
1	San Francisco International	Los Angeles International	9,643
2	Los Angeles International	San Francisco International	9,583
3	Los Angeles International	McCarran International	6,864
4	McCarran International	Los Angeles International	6,835
5	John F. Kennedy International	Los Angeles International	6,643
6	Los Angeles International	John F. Kennedy International	6,620
7	Kahului Airport	Honolulu International	6,374
8	Honolulu International	Kahului Airport	6,306
9	LaGuardia	Hartsfield-Jackson Atlanta International	5,931
10	Hartsfield-Jackson Atlanta International	LaGuardia	5,900

- Создание витрины с авиакомпаниями, к которых наибольшие задержки

```
CREATE TABLE dm.airlines_with_delays
(
    carrier string,
    avg_dep_delay decimal,
    avg_arr_delay decimal,
    total_flights bigint
)
stored as orc;

insert overwrite table dm.airlines_with_delays
SELECT
    carrier,
    AVG(dep_delay) as avg_dep_delay,
    AVG(arr_delay) as avg_arr_delay,
    COUNT(*) as total_flights
FROM
    ods.flights
GROUP BY
    carrier
HAVING
    AVG(dep_delay) > 0
ORDER BY
    avg_dep_delay DESC;
```



Results 1 ×

SELECT carrier, avg_dep_delay, avg_arr_delay, total_flights FROM dm.airlii Enter a SQL exp

	carrier	avg_dep_delay	avg_arr_delay	total_flights
1	MQ	15	14	113,212
2	VX	14	10	34,739
3	EV	14	10	157,928
4	WN	13	8	575,739
5	B6	13	10	121,906
6	UA	13	5	286,418
7	F9	12	13	35,738
8	AA	12	7	289,855
9	FL	10	7	92,702
10	9E	10	5	80,031
11	YV	9	9	52,821
12	OO	8	6	160,164
13	DL	7	3	381,657
14	US	5	4	233,321
15	HA	2	2	17,432
16	AS	1	0	68,555

- Создание витрины с задержками по дням недели

```
CREATE TABLE dm.delays_by_weekday
(
    day_of_week int,
    avg_dep_delay decimal,
    avg_arr_delay decimal,
    total_flights bigint
)
stored as orc;

insert overwrite table dm.delays_by_weekday
SELECT
    day_of_week,
    AVG(dep_delay) as avg_dep_delay,
    AVG(arr_delay) as avg_arr_delay,
    COUNT(*) as total_flights
FROM
    ods.flights
GROUP BY
    day_of_week
ORDER BY
    day_of_week;
```

<pre>CREATE TABLE dm.delays_by_weekday (day_of_week int, avg_dep_delay decimal, avg_arr_delay decimal, total_flights bigint) stored as orc; insert overwrite table dm.delays_by_weekday SELECT day_of_week, AVG(dep_delay) as avg_dep_delay, AVG(arr_delay) as avg_arr_delay, COUNT(*) as total_flights FROM ods.flights GROUP BY day_of_week ORDER BY day_of_week; SELECT day_of_week, avg_dep_delay, avg_arr_delay, total_flights FROM dm.delays_by_weekday;</pre>				
Results 1 ×				
SELECT day_of_week, avg_dep_delay, avg_arr_delay, total_flights FROM dm.delays_by_weekday				
Grid	123 day_of_week	123 avg_dep_delay	123 avg_arr_delay	123 total_flights
1	1	11	7	407,837
2	2	9	4	397,594
3	3	10	7	403,072
4	4	14	11	406,563
5	5	12	9	396,387
6	6	7	2	318,537
7	7	10	5	372,228