**Analysis of data for Air Astana. General Report**

**Author of the overall report**

**Kim Anton Sergeevich**

(a second-year student of the International Information Technologies University)

**Contacts:**

**8-707-555-82-10**

[**anton99.kim@gmail.com**](mailto:anton99.kim@gmail.com)

**anton.kim99@mail.ru**

**Introduction**

This is a report on the results of an analysis of data on flights of Air Astana employees and their beneficiaries. The primary task of analysis is to find and illustrate the answers to certain questions.

To answer these questions, I used several tools for:

* **Java and Python** - data mining and processing as well as for uploading to the database
* **PostgreSQL** – data storage and writing queries
* **Tableau** – data visualization

**Methodology**

***Data preparation***

The initial data consisted of 5 files of various types. The first task was to build a database and organize all the information.

The database consists of 6 tables:

* **Departments** – data from file “departments.csv”
* **BenefTypes** – types of beneficiaries, data from file “beneficiaries.xml”
* **FlightInfo** – all necessary information about flights, data from file “flighttime.xlsx” + “distances.csv” (a file which I generated to obtain data on distances between airports)
* **Transactions** – data from file “transactions.csv”
* **Employees** – data from file “employees”
* **Beneficiaries** – data from file “beneficiaries.xml”

***Query building***

1. **The first information to get was most popular destinations for families.**

SQL:

**select families.arr\_airport\_code, count(\*) from**

**(**

**select arr\_airport\_code from transactions**

**group by booking\_time, arr\_airport\_code**

**having count(traveler\_id)>1**

**) as families**

**group by families.arr\_airport\_code**

**order by count(\*) desc**

Logic:

In the first step, I find simultaneous transactions, grouping them by the time of booking and selecting only those with more than one passenger. Thus, I get a list of people who are not flying alone. Then, I designate the resulting table as families, and group them by place of expected destination. I sort by the number of families, in descending order. As a result, I get the top 5 most popular destinations for flying with family:

1. Almaty
2. Nur-Sultan
3. Istanbul
4. Shymkent
5. Dubai

If we consider only foreign airports, then the top will look like this:

1. Istanbul – Turkey
2. Dubai – United Arab Emirates
3. Tbilisi – Georgia
4. Seoul – South Korea
5. Kuala Lumpur – Malaysia
6. **The following statistics were on the total distance covered for each employee and his beneficiary for the period**

SQL:

**select traveler\_id, round(sum(f.distance)+sum(total\_benef\_distance)/count(\*)) as total\_distance from transactions t**

**inner join employees e**

**on staff\_number = traveler\_id**

**inner join flightinfo f**

**on t.dep\_airport\_code = f.dep\_airport\_code and t.arr\_airport\_code = f.arr\_airport\_code**

**inner join**

**(**

**select staff\_number, round(sum(distance)) as total\_benef\_distance from beneficiaries**

**inner join transactions t**

**on benef\_id=traveler\_id**

**inner join flightinfo f**

**on t.dep\_airport\_code = f.dep\_airport\_code and t.arr\_airport\_code = f.arr\_airport\_code**

**group by staff\_number**

**) as b**

**on b.staff\_number=e.staff\_number**

**group by traveler\_id**

Logic:

The first thing I do is join the tables: Transactions, Beneficiaries, FlightInfo. Then I group them by employee (staff\_number) and summarize the total distance of the employee's beneficiaries. In the second step, you need to calculate how much the employee flew by himself. To do this, I combine the tables: Transactions, Employees, FlightInfo and the previous table. Now it remains only to group by traveler (traveler\_id) and calculate the total distance of the employee plus the distance of his beneficiaries.

1. **The third statistics on the trends of the best booking time / ticket purchase in each direction based on the booking time**

SQL:

1 Popular destination.

**select t.dep\_airport\_code, t.arr\_airport\_code, EXTRACT(HOUR FROM booking\_time) as vacant\_hour, count(\*) from transactions t**

**inner join**

**(**

**select dep\_airport\_code, arr\_airport\_code from transactions**

**group by dep\_airport\_code, arr\_airport\_code**

**having count(\*) >**

**(**

**select avg(s) from**

**(**

**select count(\*) as s from transactions**

**group by dep\_airport\_code, arr\_airport\_code) as t**

**)**

**) as f**

**on t.dep\_airport\_code = f.dep\_airport\_code and t.arr\_airport\_code = f.arr\_airport\_code**

**group by t.dep\_airport\_code, t.arr\_airport\_code, EXTRACT(HOUR FROM booking\_time)**

**order by (t.dep\_airport\_code, t.arr\_airport\_code, count(\*))**

2 General statistics:

**select EXTRACT(HOUR FROM booking\_time) as vacant\_hour, count(\*) from transactions t**

**inner join**

**(**

**select dep\_airport\_code, arr\_airport\_code from transactions**

**group by dep\_airport\_code, arr\_airport\_code**

**having count(\*) >**

**(**

**select avg(s) from**

**(**

**select count(\*) as s from transactions**

**group by dep\_airport\_code, arr\_airport\_code**

**) as tab2**

**)**

**) as tab1**

**on t.dep\_airport\_code = tab1.dep\_airport\_code and t.arr\_airport\_code = tab1.arr\_airport\_code**

**group by (vacant\_hour)**

**order by count(\*)**

Logic 1:

For this table, first I select those directions in which the number of flights exceeds the average in all directions.

Then I extract the hour from the booking\_time column and group it. As a result, I get a table containing: Destination, booking hour and how much bookings were made at that hour

Logic 2:

To derive general statistics for all popular destinations, I also select those that have more than average flights, then group by hours and then sort them by repetition. As a result, I get a table of the number of bookings and hours

On average, a free interval for booking a ticket is in the morning (from about 5 to 9 am)

1. **The next statistics were for the total time spent in flight for each passenger for the period.**

SQL:

**select traveler\_id, sum(flight\_time) from transactions t**

**inner join flightinfo f**

**on t.dep\_airport\_code = f.dep\_airport\_code and t.arr\_airport\_code = f.arr\_airport\_code**

**group by traveler\_id**

**order by sum(flight\_time)**

Logic:

The first step, I combine the flight time table with the transaction table, so I get a table in which there is an estimated flight time. After that, I group all transactions by passengers for the entire period and summarize the time of all flights for each passenger. Then just sort them in ascending order.

1. **The fifth task was to identify the most frequently flying employees in each department**

SQL:

**select dep\_name, traveler\_id, count(\*) as flights from transactions t**

**inner join employees e**

**on e.staff\_number = t.traveler\_id**

**inner join departments d**

**on d.dep\_id = e.dep\_id**

**group by traveler\_id, dep\_name**

**order by (dep\_name, count(\*))**

Logic:

First you need to join the tables (inner join): Employees, Transactions, Departments, so I get a list of flights of employees. After that, I group the resulting table by employee and his department and count how many flights each employee of the department made. At last, I sort everyone by department and number of flights.

At the output, I get the name of the department, an indicator of the employee, as well as how much he flew this season.

1. **The sixth task was to draw up a diagram of the dependence of the passenger's age on the number of flights**

SQL:

**select age, count(\*) from**

**(**

**select 2020-extract(year from dob) as age from transactions**

**inner join beneficiaries**

**on traveler\_id = benef\_id**

**union all**

**select 2020-extract(year from dob) as age from transactions**

**inner join employees**

**on traveler\_id = staff\_number**

**) as tab**

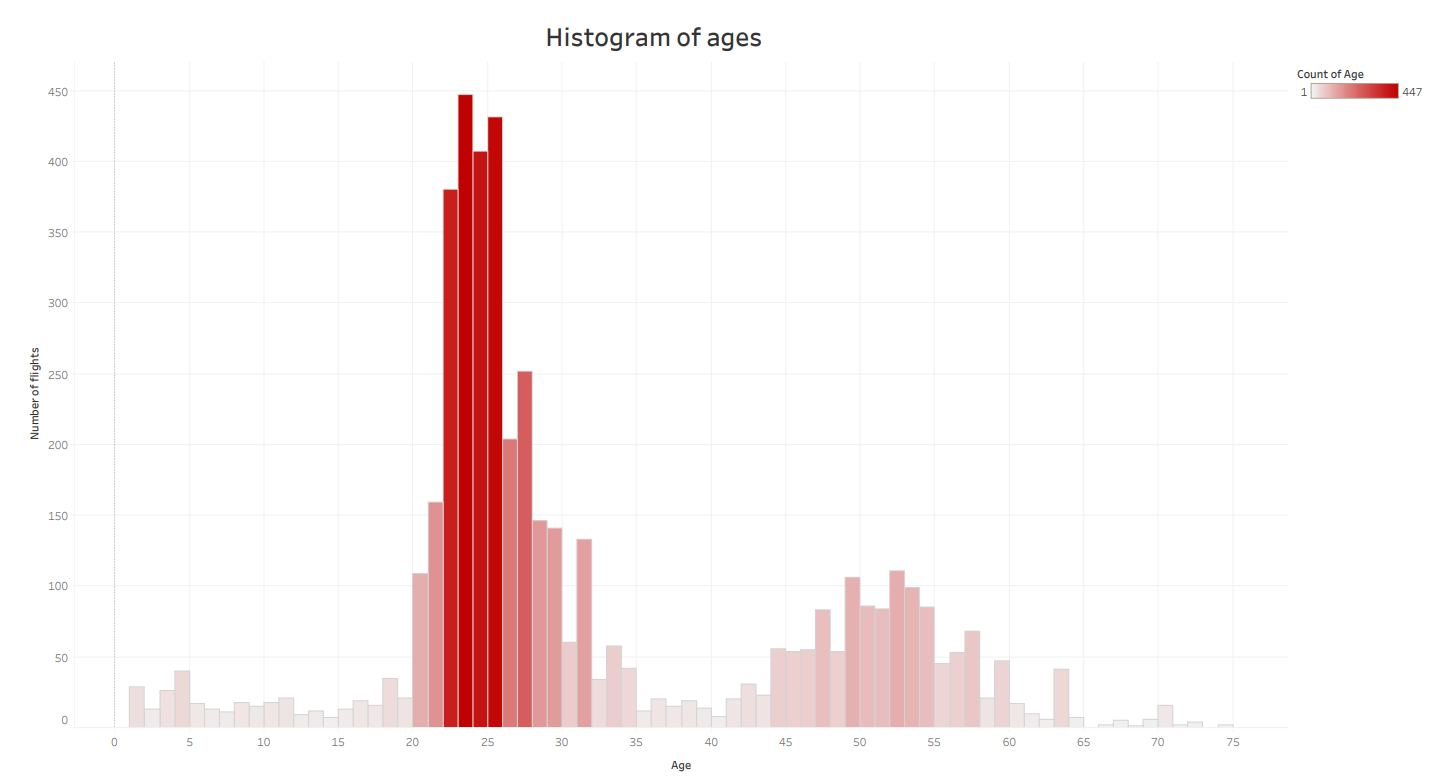
**group by age**

**order by age**

Logic:

First, I compile a table of the ages of all passengers. To do this, I first combine the employee table, from it I take the year of birth, then I do the same with the beneficiary table. After that, both tables are combined and grouped by age. In the end, I just sort by age. As a result, I get a table of ages and the number of passengers

For visualization I used Tableau. The color intensity on the graph depends on the number of passengers



1. **The next task - create a chart of TOP 5 most popular destinations depending on the season**

SQL:

**select \* from**

**(**

**select 'winter' as month, dep\_airport\_code || ' - ' || arr\_airport\_code as direction, count(\*)**

**from transactions**

**group by extract (month from booking\_time), dep\_airport\_code, arr\_airport\_code**

**having extract(month from booking\_time) in (12,1,2) and count(\*) > 27**

**union all**

**select 'spring' as month, dep\_airport\_code || ' - ' || arr\_airport\_code as direction, count(\*)**

**from transactions**

**group by extract(month from booking\_time), dep\_airport\_code, arr\_airport\_code**

**having extract(month from booking\_time) in (3,4,5) and count(\*) > 5**

**union all**

**select 'summer' as month, dep\_airport\_code || ' - ' || arr\_airport\_code as direction, count(\*) from transactions**

**group by extract(month from booking\_time), dep\_airport\_code, arr\_airport\_code**

**having extract(month from booking\_time) in (6,7,8) and count(\*) > 47**

**union all**

**select 'autumn' as month, dep\_airport\_code || ' - ' || arr\_airport\_code as direction, count(\*)**

**from transactions**

**group by extract(month from booking\_time), dep\_airport\_code, arr\_airport\_code**

**having extract(month from booking\_time) in (9,10,11) and count(\*) > 78**

**) as tab**

**order by month, count desc**

Logic:

I break the data into four seasons. To do this, take a month from booking\_time and, depending on the value, mark one season. For each season, I select directions in top 5 of number of flights



1. **The eighth request was to identify the most popular and unpopular destinations for employees of both airlines**

SQL:

**select line, dep\_airport\_code, arr\_airport\_code, count(\*) as flights from transactions**

**inner join employees e**

**on staff\_number = traveler\_id**

**group by dep\_airport\_code, arr\_airport\_code, line**

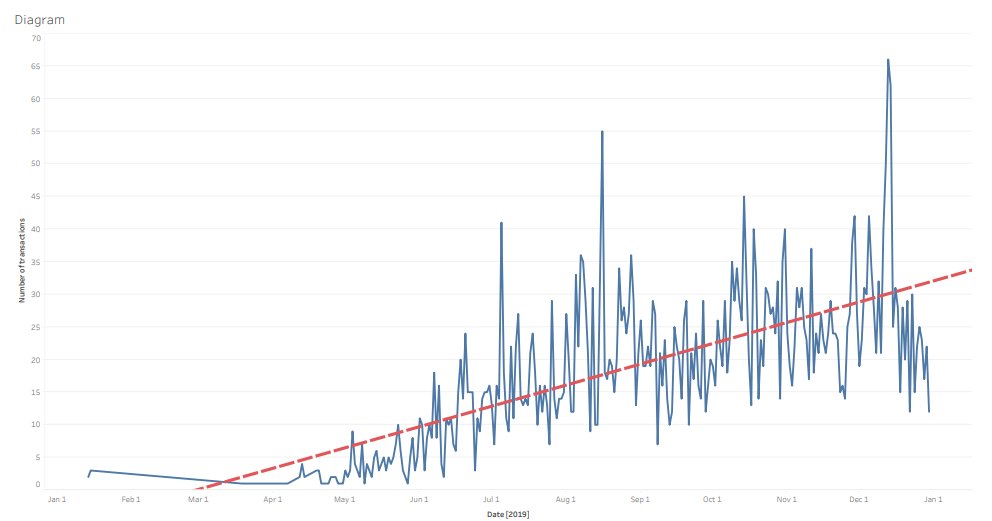
**order by (line, count(\*))**

Logic:

At the beginning, I select the flights of employees (inner join). I group the resulting table by directions and airlines. I count flights in the resulting directions and sort.

At the end, I get the names of the airlines, directions and the numbers of flights.

1. **For the last assignment, I built a curve of the frequency of booking airline tickets for the entire period, then plotted a trend line (linear regression)**



**Summary**

A small summary of the study. A large number of domestic flights: Almaty, Nur-Sultan, Shimkent, etc. Among international flights, the most frequent were Istanbul, Turkey, Tbilisi, etc. The best time to book tickets is early morning, before 10 a.m. Basically, the age of passengers lies in the range from 20 to 30 years. In autumn there was the largest number of flights, but mainly domestic flights, the smallest number of tickets sold in the spring.

**Used scripts can be seen at this link:**

**https://github.com/Kolmuss/ML/tree/master/AirCompany**