Airline case study: deployment and usage of an analytic system

José Luis Puente Bodoque

February, 2023

Abstract

In this report we design a date dimension to end up the design of our airline case of study. After that, we implement in SQL its create table definition sentence and we insert the date table in Oracle SQL Developer. Next we are thinking about four analytic questions (that include two KPIs) to reach four strategic objectives for the airline. Finally we are making a dashboard with a visualizations as answer to each question.

Contents

1 Deployment of an analytic system

The deployment of the analytic system for our airline case study was documented in the report of the task 2. We applied an ETL load process to every input file to fill up tables in Oracle SQL Developer. We were showing a screenshot for each table, too.

Now, as the teacher had suggested, we are implementing a date dimension in order to definitively complete our data warehouse design. First step is to create a new CSV input file with the data. Second step is to write a script to implement the date dimension table in SQL. Third step is to add foreign keys and references for the new date dimension in the fact table. Forth step is to load the data into the date dimension. Fifth step is to edit the data source to insert dates in `flight.csv` file (because no date fields in!). And finally seventh step is to modify the ETL load process to suit the updated fact table.

2 The date dimension

In the same way as we had designed an hour dimension for departure and arrival hours, now it's time to design a date dimension to record departure and arrival dates on the boarding pass.

2.1 Data input file

According to page 49 of Kimball and Ross' book, the primary key of our date dimension is an integer that represents the date in format "YYYYMMDD" instead of a sequentially-assigned surrogate key. Since we have *only* 100 flights in our database, we have considered a consecutive 7-days window from July 21 to July 27, 2023. In this way we'll get visualizations less scattered along time. So we have created a handmade CSV file containing 7 rows without taking into account the table header. Figure 1 shows the `date.csv` file content.

`id` stores an integer in format "YYYYMMDD" and it's the primary key. `date` stores the date in format "YYYY-MM-DD" according to ISO 8601 standard. `full_date_description` stores the date in American English. The rest of attributes are straightforward to comprehend.

Note: to be able to see the date according to ISO 8601 standard in Oracle SQL Developer we must change the date format in prefer-

	A	B	C	D	E	F	G
1	id	date	full_date_description	day_of_week	month	quarter	year
2	20230721	21/07/2023	July 21, 2023	viernes	julio	Q2	2023
3	20230722	22/07/2023	July 22, 2023	sábado	julio	Q2	2023
4	20230723	23/07/2023	July 23, 2023	domingo	julio	Q2	2023
5	20230724	24/07/2023	July 24, 2023	lunes	julio	Q2	2023
6	20230725	25/07/2023	July 25, 2023	martes	julio	Q2	2023
7	20230726	26/07/2023	July 26, 2023	miércoles	julio	Q2	2023
8	20230727	27/07/2023	July 27, 2023	jueves	julio	Q2	2023

Figure 1: Input for date dimension

ences: **Tools** → **Preferences**, then **Database** → **NLS** and set **Date Format** to "RRRR-MM-DD".

2.2 Implementation in SQL

The listing below shows the `CREATE TABLE` definition sentence for `fecha` dimension table.

```
CREATE TABLE fecha (
  id                        NUMBER(8) NOT NULL,
  fecha                     DATE DEFAULT NULL NULL,
  full_date_description     VARCHAR2(30) DEFAULT NULL NULL,
  day_of_week               VARCHAR2(20) DEFAULT NULL NULL,
  month                     VARCHAR2(10) DEFAULT NULL NULL,
  quarter                   VARCHAR2(2) DEFAULT NULL NULL,
  year                      NUMBER(4) DEFAULT NULL NULL,
  PRIMARY KEY (id))
;
```

2.3 Fact table modification

The creation of a date dimension implies adding two new foreign keys for departure and arrival dates and two references towards the date dimension table. The figure 2 shows the two new foreign keys in the fact table.

Note: we may use `ALTER TABLE` to insert only the new foreign keys attributes on the fact table already created.

2.4 ETL load process for date dimension

Our input data file contains values of the day of week column in Spanish, so we must export the XLSX file as CSV UTF-8 file to conserve

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ID_BOARDING_PASS	NUMBER(10,0)	No	(null)	1	(null)
2	ID_TRIP	NUMBER(10,0)	No	(null)	2	(null)
3	ID_LEG_ORIGIN	NUMBER(10,0)	No	(null)	3	(null)
4	ID_LEG_DESTINATION	NUMBER(10,0)	No	(null)	4	(null)
5	ID_TRIP_ORIGIN	NUMBER(10,0)	No	(null)	5	(null)
6	ID_TRIP_DESTINATION	NUMBER(10,0)	No	(null)	6	(null)
7	ID_DEPARTURE_HOUR	NUMBER(10,0)	No	(null)	7	(null)
8	ID_ARRIVAL_HOUR	NUMBER(10,0)	No	(null)	8	(null)
9	ID_DEPARTURE_DATE	NUMBER(10,0)	No	(null)	9	(null)
10	ID_ARRIVAL_DATE	NUMBER(10,0)	No	(null)	10	(null)
11	ID_AIRPLANE	NUMBER(10,0)	No	(null)	11	(null)
12	ID_PAYMENT_CHANNEL	NUMBER(10,0)	No	(null)	12	(null)
13	ID_FARE	NUMBER(10,0)	No	(null)	13	(null)
14	ID_PASSENGER	NUMBER(10,0)	No	(null)	14	(null)
15	BASE_FARE	BINARY_DOUBLE	Yes	NULL	15	(null)
16	MILES	NUMBER(10,0)	Yes	NULL	16	(null)
17	DELAY	NUMBER(10,0)	Yes	NULL	17	(null)
18	TICKET_NUMBER	NUMBER(10,0)	Yes	NULL	18	(null)

Figure 2: Fact table

tilde accents.

Then we step into the ETL load process (figure 3). The CSV File Input step reads the data in format "YYYY-MM-DD" and then the Output Table step loads the data received into the date dimension table.

Figure 3: ETL load process for date dimension

We have to make sure that UTF-8 is selected in "File encoding" select option in the CSV File Input step configuration (figure 4).

File encoding: UTF-8

#	Name	Type	Format
1	id	Number	#
2	date	Date	yyyy-MM-dd
3	full_date_description	String	
4	day_of_week	String	
5	month	String	
6	quarter	String	
7	year	Integer	#

Figure 4: Date CSV File Input configuration

Finally we set field the mapping in the Output Table step as shown in figure 5.

The result is shown in figure 6.

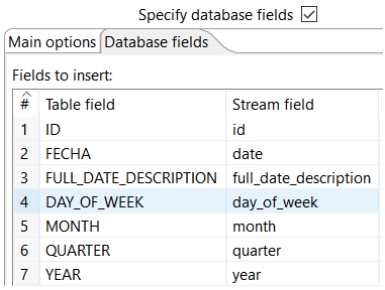

Figure 5: Field mapping for date dimension

	ID	FECHA	FULL_DATE_DESCRIPTION	DAY_OF_WEEK	MONTH	QUARTER	YEAR
1	20230721	2023-07-21	July 21, 2023	viernes	julio	Q2	2023
2	20230722	2023-07-22	July 22, 2023	sábado	julio	Q2	2023
3	20230723	2023-07-23	July 23, 2023	domingo	julio	Q2	2023
4	20230724	2023-07-24	July 24, 2023	lunes	julio	Q2	2023
5	20230725	2023-07-25	July 25, 2023	martes	julio	Q2	2023
6	20230726	2023-07-26	July 26, 2023	miércoles	julio	Q2	2023
7	20230727	2023-07-27	July 27, 2023	jueves	julio	Q2	2023

Figure 6: Date dimension table

2.5 Modification in `flight.csv`

Now we have to insert two columns in `flight.csv`: one for departure dates and another for arrival dates. Obviously we have to invent them in order to implement the date dimension. We have assigned randomly dates to every row in the 7-days windows from Friday, July 21 to Thursday, July 27, 2023. The dates are in a full date description format. Departure and arrival dates are the same for each row but, in order to provide more realism, some rows have consecutive departure and arrival dates.

2.6 Update of the ETL process for the fact table

We add the date keys into the main stream by adding two more steps to the ETL process (figure 3). Now the first Database Lookup step looks up in airport dimension table date values, too. Then, the fourth and fifth Database Lookup steps look up for each departure and arrival date the keys. Finally we have to add two correspondences for departure and arrival date keys in the Output Table step.

The fact table now looks like the figure 8 (zoom in to read it).

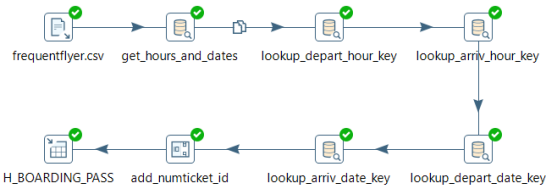

Figure 7: Updated ETL process for fact table

	ID_BOARDING_PASS	ID_TRIP	ID_LEG_ORIGIN	ID_LEG_DESTINATION	ID_TRIP_ORIGIN	ID_TRIP_DESTINATION	ID_DEPARTURE_HOUR	ID_ARRIVAL_HOUR	ID_DEPARTURE_DATE	ID_ARRIVAL_DATE	ID_AIRPLANE	ID_PAYMENT...	ID_FARE	ID_PASSENGER	BASE_FARE	MILES	DELAY	TICKET_NUMBER
1	1	18	12	1	12	6	581	702	20230724	20230724	20	4	2	1157,0	828	0	3019
2	2	18	1	6	12	6	951	1103	20230722	20230722	94	2	2	121,0	112	36	3019
3	3	24	6	12	6	12	511	623	20230727	20230727	6	2	3	1375,0	1972	22	3019
4	4	38	5	14	5	11	486	570	20230727	20230727	1	4	2	1303,0	1597	0	1063
5	5	38	14	11	5	11	861	1003	20230725	20230725	76	2	3	1360,0	1897	40	1063
6	6	41	11	4	11	5	556	664	20230726	20230726	15	1	4	199,0	522	0	1063
7	7	41	4	5	11	5	901	941	20230723	20230723	84	4	1	1238,0	1255	0	1063
8	8	32	14	19	14	19	536	636	20230722	20230722	11	2	2	1191,0	1004	0	8348
9	9	37	19	14	19	14	531	708	20230727	20230727	10	3	4	1239,0	1257	0	8348
10	10	51	14	11	14	11	561	720	20230723	20230723	16	4	1	1316,0	1665	0	3904
11	11	54	11	17	11	14	496	614	20230726	20230727	3	2	2	1136,0	714	0	3904
12	12	54	17	14	11	14	941	971	20230727	20230727	92	2	1	1288,0	1515	0	3904
13	13	44	5	4	5	7	601	665	20230727	20230727	24	1	4	1332,0	1745	0	954
14	14	44	4	7	5	7	886	957	20230723	20230723	81	4	3	112,0	61	0	954
15	15	52	7	5	7	5	571	703	20230724	20230724	18	4	3	1126,0	662	0	954
16	16	56	9	13	9	20	526	670	20230726	20230726	9	3	4	1378.0	1987	0	1146

Figure 8: Updated fact table

3 Analytic questions

Now let's imagine the marketing department need to explain how the business process is going. For that purpose, they are going to ask four analytic questions in our 7-days window of analysis.

3.1 First question

Let's set average delay as a KPI so that the first question is: **which is the average delay? Does the KPI in or out of target?** Let's set target value such as an average delay must be lesser than 15 minutes.

3.2 Second question

Let's set total revenues (in dollars) as a KPI. Total revenues are equal to the sum of base fares. So the second question is: **How much are our airline's total revenues? Does the KPI in or out of target?** Let's set target value such as revenues must be greater than 250 thousands of dollars.

3.3 Third and fourth question

Third question is: **how many passengers fly on each day?** And the forth is: **what is the passengers' favorite destination?** Both

are related each other by the day of week. We are going to watch the favorite destination based on day of week.

4 Visualizations on Power BI

Now the marketing department decides to display the answer to those questions by displaying them on a panel in a visual way. For that purpose, we are going to make a dashboard in Power BI Desktop.

4.1 Star schema

Once we have our tables got (or "Fields" as they are called here), we should may watch our star schema on the Model panel (figure 9).

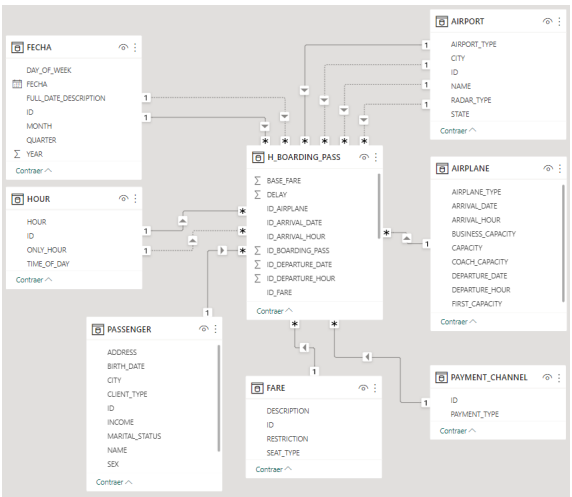

Figure 9: Star schema on Power BI

4.2 Dashboard: an overview

The figure 10 shows the dashboard overview. It's composed of four visualizations that respond to every question.

4.3 Answer No. 1: average delay

At the upper left of the dashboard we have displayed a gauge (from this point forward as "first visualization"). It compares the average

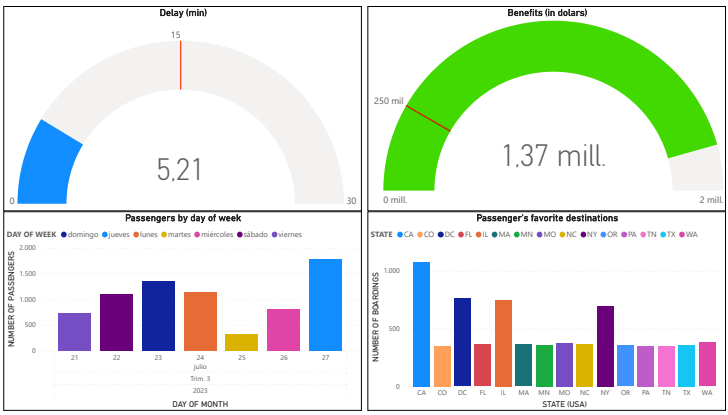

Figure 10: Dashboard overview

delay, which is 5.21 min, to the target value in red, which is set to 15 min. So we can say the average delay KPI is in target.

4.4 Answer No. 2: revenues

At the upper right of the dashboard we have displayed another gauge (from this point forward as "second visualization"). In this case, the target value, which we set to 250 thousands of dollars, is a minimum threshold, so we can see that benefits, which are equals to 1.37 millions of dollars are in target.

4.5 Answer No. 3: number of passengers by day of week

At the inner left of the dashboard we can see a bar chart (from this point forward as "third visualization") that shows the day with more passengers was on Thursday, July 27, 2023.

4.6 Answer No. 4: passenger's favorite destinations

At the inner right of the dashboard (from this point forward as "fourth visualization") we can see clearly that California (USA) is the passenger's favorite destination.

4.7 Increase of granularity

Now we are going to explore in depth the two inner visualizations. If we select a day of month at the third visualization the dashboard will suit the rest of them. For example, if we select Thursday, July 27, we'll see what the figure 11 shows. (Revenue KPI keeps being in target, by the way). We put the date hierarchy at the x-axis (year, quarter, month and day) and a count aggregation function for boarding passes at the y-axis. We set day of week as legend.

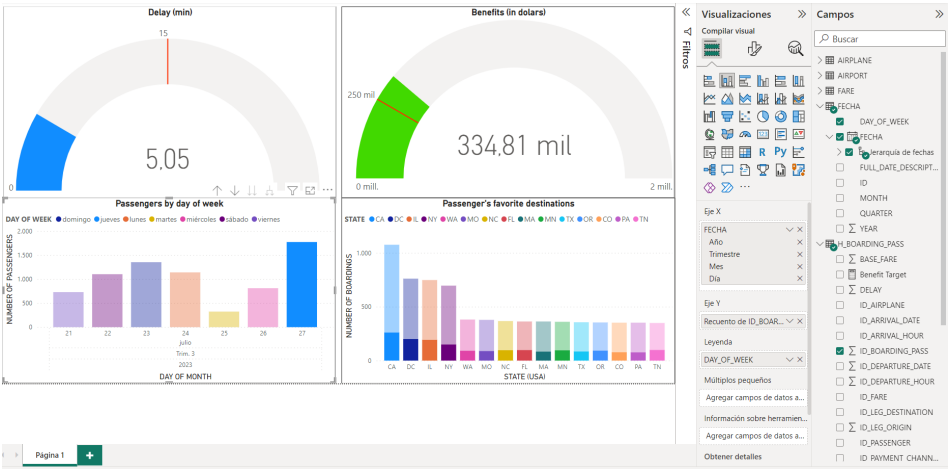

Figure 11: Dashboard particularized on Thursday, the 27th

The fourth visualization is configured as shown in figure 12. We have create a hierarchy to be able to navigate in depth across state, city and airport name. We have put it at x-axis. Also we have put a count aggregation function for boarding passes at y-axis. Finally we have added the city and the name at the additional fields to be able to read more information just by leaving the mouse on a bar.

From the fourth visualization we may also perform queries in depth. If we press click right on the corresponding bar for California, and then we select "Explore in depth", we'll walk down the hierarchy up to passenger's favorite destinations by city. If we change the current chart for a circular one, we'll see more clearly that San Diego is the winner, the city most visited by passengers (figure 13).

And if we now press on the 27th, in the third visualization, we'll see that Los Ángeles is the champion in this case (figure 14). We can see the other visualizations particularized on Thursday, the 27th, too.

Figure 12: Configuration of fourth visualization

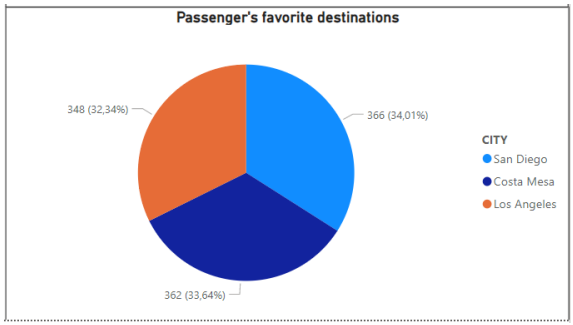

Figure 13: Bar chart of passenger's favorite destinations by city

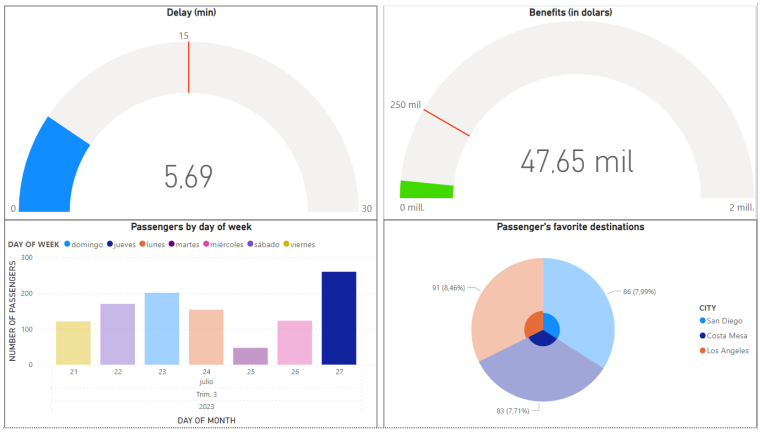

Figure 14: City most visited on Thursday, July, the 27th