**Insights into airfares offered from Belgium**

The online travel opportunities is giving a massive amounts of data, ready for scraping and analyzing. So from April 2023 to July 2023 the prices and available seats of flights were scraped daily for 3 airlines: Ryanair, Tui Fly Belgium and Transavia

Those 3 airlines operate in Belgium, offering flights from multiple airports:

* Ryanair => Zaventem + Charleroi
* Tui fly => Antwerpen + Brugge-Oostende + Brussel + Luik
* Transavia => Brussel

The destinations that were scraped:

* Greece
  + Corfu
  + Kreta (Heraklion)
  + Rhodos
* Italy
  + Brindisi
  + Napels
  + Palermo
* Portugal
  + Faro
* Spain
  + Alicante
  + Ibiza
  + Malaga
  + Palma
  + Tenerife

Type of flight that was scraped:

* One-way
* 1 adult
* Economy class
* Between April, 8 and October, 1 2023

Using the retrieved data, we wanted to find out how the pricing strategy and the rates compare from one airline to another? Are there differences based on the destinations, on the day of booking? Are prices changing when the departure date comes nearer? Are there other factors that influence the price?

**Draw a star schema**

The schema of the OLTP database Airfares is shown below. It isn’t too hard to understand

* Airport: contains information about the 17 airports that are involved
* Airline: contains information about the 3 airlines that are involved
* Flight: flightnumber (this isn’t unique! That’s why a unique flight\_id was introduced), date of departure, date of arrival, departure time, arrival time, duration of the flight, number of stops, iata\_code of the airline, iata\_code of the departure airport and iata\_code of the arrival airport
* Scraped: scrape\_id is a surrogate primary key. The combination of flight\_id and scrape\_date is also unique. This table contains the number of available seats and the price of a particular flight on a specific scrape date. Available\_seats = -1 means 11 or more seats are available.

Afbeelding met tekst, Lettertype, nummer, lijn

Automatisch gegenereerde beschrijving

1. Download and run the script Create and Fill Airfares.sql from Chamilo.
2. Take a look at the database. Make sure to understand the database.
3. Draw a star schema of the DWH to be developed

* Almost every DWH has a Date Dimension.
* You can reuse the date dimension from Northwind
* Which other dimensions are you going to create?
* Which dimension will be Slowly Changing Dimensions? It was assumed that the info about the airports or airlines wouldn’t change. Although the information of 1 flight could change, e.g. the departure\_time, arrival\_time, duration and number\_of\_stops

1. What are the facts? How is the Fact table going to look like?

**Create the database AirfaresDWH and the dimenson tables and the fact table**

1. Write a .sql script to create the database AirfaresDWH, the dimension tables and the fact table. Use the following code to start from.

IF NOT EXISTS (SELECT \* FROM sys.databases WHERE name = 'AirfaresDWH')

BEGIN

CREATE DATABASE AirfaresDWH;

END;

GO

USE [AirfaresDWH]

CREATE TABLE [dbo].[DimAirline](

airline\_key INT PRIMARY KEY NOT NULL IDENTITY(1,1),

airline\_iata\_code [nchar](10) NOT NULL,

airline\_name [nvarchar](75),

airline\_country [nvarchar](75)

)

CREATE TABLE [dbo].[DimDate](

[date\_key] [int] NOT NULL PRIMARY KEY,

[FullDateAlternateKey] [date] NOT NULL,

[DayOfMonth] [varchar](2) NULL,

[EnglishDayNameOfWeek] [varchar](10) NOT NULL,

[DutchDayNameOfWeek] [varchar](10) NOT NULL,

[DayOfWeek] [char](1) NULL,

[DayOfWeekInMonth] [varchar](2) NULL,

[DayOfWeekInYear] [varchar](2) NULL,

[DayOfQuarter] [varchar](3) NULL,

[DayOfYear] [varchar](3) NULL,

[WeekOfMonth] [varchar](1) NULL,

[WeekOfQuarter] [varchar](2) NULL,

[WeekOfYear] [varchar](2) NULL,

[Month] [varchar](2) NULL,

[EnglishMonthName] [varchar](10) NOT NULL,

[DutchMonthName] [varchar](10) NOT NULL,

[MonthOfQuarter] [varchar](2) NULL,

[Quarter] [char](1) NULL,

[QuarterName] [varchar](9) NULL,

[Year] [char](4) NULL,

[MonthYear] [char](10) NULL,

[MMYYYY] [char](6) NULL

) …

1. Add the Foreign Key constraints

**DimDate**

Almost every data warehouse has a date dimension.

1. Download from Chamilo the file Documenten > Scripts > Fill\_DimDate.sql. In this file you find the code of the stored procedure FillDimDate
2. Make sure you understand this script. This script is retrieved from <https://www.codeproject.com/Articles/647950/Create-and-Populate-Date-Dimension-for-Data-Wareho> and will be used to fill DimDate
3. Write a stored procedure CheckAndUpdateDimDate
   * For ease of use, we are going to use the minimum date and maximum in Flight. Calculate both.
   * Calculate the minimum date in DimDate.
   * Calculate the maximum date in DimDate.
   * If DimDate is empty or (the minimum date in Flight < minimum date in DimDate) or (maximum date in Flight > maximum date in DimDate)
     + Delete everything from DimDate
     + Fill DimDate with the appropriate dates (from januari, 1 of the minimum year until December, 31 of the maximum year).   
       Use the stored procedure FillDimDate.
4. Execute the stored procedure CheckAndUpdateDimDate.
5. Check if AirfaresDWH > DimDate contains the correct data (from 01/01/2023 until 31/12/2023)

**Fill DimAirport**

1. We need to fill DimAirport in AirfaresDWH using the data from Airfares.
2. Create a stored procedure CheckAndUpdateDimAirport that contains 1 MERGE – statement
   * If an airport disappeared out of Airfares, nothing should be done.
   * If the properties of an airport change, the changes should be transferred to AirfaresDWH
   * If a new airport was inserted into Airfares, the new airport should be inserted into AirfaresDWH

-- Fill DimAirport

-- Add airports to DimAirport that are already in the OLTP database (Airport)

-- but that aren't yet in the DWH (DimAiport)

-- It's possible that there are airports in DimAirport that aren't in Airport any more,

-- but that is no problem.

CREATE OR ALTER PROCEDURE CheckAndUpdateDimAirport

AS

BEGIN

MERGE AirfaresDWH.dbo.DimAiport as t -- t = target

USING Airfares.dbo.Airport as s -- s = source

ON (t.airport\_iata\_code = s.airport\_iata\_code)

-- Which rows are in source and have different values for airportname, place or country?

-- Update those rows in target with the values coming from source

WHEN MATCHED AND -- to be completed: there are some changes to airportname, …

THEN UPDATE SET -- to be completed: the new values should be transferred to DimAirport

-- Which rows are in Airport and not in DimAirport?

-- Insert those rows from source into target

WHEN NOT MATCHED BY target --> rows to insert

THEN INSERT VALUES -- to be completed

~~-- Which rows are in DimAirport and not any more in Airport?~~

~~-- Do nothing~~

~~WHEN NOT MATCHED BY source~~

END

1. Execute the stored procedure CheckAndUpdateDimAirport.
2. Check if AirfaresDWH > DimAirport contains the correct data

Afbeelding met tekst, schermopname, nummer, menu

Automatisch gegenereerde beschrijving

1. Make some changes to Airport in Airfares

UPDATE Airport

SET airport\_name = 'Brussels-Zaventem'

WHERE airport\_iata\_code = 'BRU'

1. Execute the stored procedure CheckAndUpdateDimAirport.
2. Check if the changes are visible in DimAirport

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijving

**Fill DimAirline**

1. We need to fill DimAirline in AirfaresDWH using the data from Airfares.
2. Create a stored procedure CheckAndUpdateDimAirline that contains 1 MERGE – statement
   * If an airline disappeared out of Airfares, nothing should be done.
   * If the properties of an airline change, the changes should be transferred to AirfaresDWH
   * If a new airline was inserted into Airfares, the new airline should be inserted into AirfaresDWH
3. This is completely similar to Fill DimAirport
4. Execute the stored procedure CheckAndUpdateDimAirline.
5. Check if AirfaresDWH > DimAirline contains the correct data

Afbeelding met tekst, Lettertype, lijn, nummer

Automatisch gegenereerde beschrijving

**Fill DimFlight 🡪 Slowly changing dimension**

1. DimFlight is a Slowly Changing Dimension Type 2. The data of 1 flight could change, e.g. the departure\_time, arrival\_time, duration and number\_of\_stops
2. We need to fill DimFlight in AirfaresDWH using the data from Airfares.
   * If a flight disappeared out of Airfares, nothing should be done.
   * If a new flight was inserted into Airfares, the new flight should be inserted into AirfaresDWH. The Startdate is today.
   * If the column values of a flight have changed in Airfares
     + The Enddate in DimFlight should get the date of yesterday
     + A new record is inserted into DimFlight with the new values and the Startdate is today
3. Create a new stored procedure CheckAndUpdateDimFlight and do the following
   * Create a temporary table FlightTempTableType to contain flight\_key, flight\_id, departure\_time, arrival\_time, duration, number\_of\_stops
   * Fill this temporary table with the new values, out of Airfares.dbo.Flight, for already existing records in AirfaresDWH.dbo.DimFlight
   * Insert these records (from the temporary table) into AirfaresDWH.dbo.DimFlight. Use Today as StartDate.
   * Change the EndDate of the old records to yesterday
   * Insert the entirely new flights from Airfares.dbo.Flight into DimFlight.   
     This will also garantee that DimFlight get’s filled for the first time (if it was still empty)
     + If DimFlight was empty => Startdate is the minimum date of Flight for the newly inserted records: current flight data is valid since the start of the company (since we have no history yet).
     + If DimFlight already contains values => Startdate = today for the newly inserted records

-- Fill DimFlight -- Slowly Changing Dimension

CREATE OR ALTER PROCEDURE CheckAndUpdateDimFlight AS

BEGIN

-- Create a temporary table #FlightTempTable to contain the values that have changed

-- Change the EndDate of those records which contain old values to yesterday

-- Insert these records (from the temporary table) into AirfaresDWH.dbo.DimFlight.

-- Use Today as StartDate.

-- Insert the entirely new flights from Airfares.dbo.Flight into DimFlight.

-- This will also guarantee that DimFlight get’s filled for the first time (if it was still empty)

-- If DimFlight was empty => Startdate is the minimum date of Flight for the newly inserted records: current flight data is valid since the start of the company (since we have no history yet).

-- If DimFlight already contains values => Startdate = today for the newly inserted records

END

1. Execute the stored procedure CheckAndUpdateDimFlight.
2. Check if AirfaresDWH > DimFlight contains the correct data

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijving

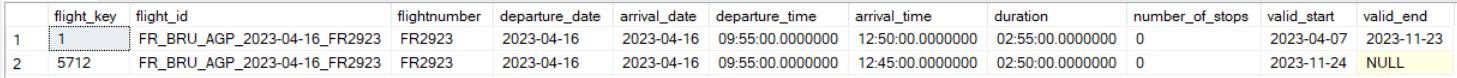
1. Make some changes to Flight in Airfares

UPDATE Flight

SET arrival\_time = '12:45', duration = '02:50'

WHERE flight\_id = 'FR\_BRU\_AGP\_2023-04-16\_FR2923'

1. Execute the stored procedure CheckAndUpdateDimFlight.
2. Check if the changes are visible in DimFlight



**Fill FactFares 🡪 fill fact table**

1. Write a stored procedure UpdateFactFares that contains the INSERT statement to fill the table FactFares repeatedly. This is absolutely non trivial
2. The INSERT statement is analogous to the INSERT statement for NorthwindDWH.
3. Start with the SELECT statement. FactFares contains the following fields. Try to find the appropriate way to get the correct value for each of these fields:
   * airline\_key
   * airport\_departure\_key
   * airport\_arrival\_key
   * flight\_key
   * scrape\_date\_key
   * departure\_date\_key
   * arrival\_date\_key
   * available\_seats
   * price
4. Add the WHERE clause to the SELECT statement
5. Add the INSERT at the front
6. We can wrap the 5stored procedures into 1 stored procedure

CREATE OR ALTER PROCEDURE UpdateDWH

AS

BEGIN

EXEC CheckAndUpdateDimDate

EXEC CheckAndUpdateDimAirline

EXEC CheckAndUpdateDimAirport

EXEC CheckAndUpdateDimFlight

EXEC UpdateFactFares

END

**Queries**

Try to find the answers to the following questions using the data warehouse

1. What is the number of unique flights per airline?
2. What is the average price per destination?
3. Which is the most expensive weekday to depart?
4. Give the number of flights that depart for every weekday.
5. What is the average price per destination per airline? Use cte’s to create the following overview. The PIVOT operator would have been more appropriate to solve this.

Afbeelding met tekst, schermopname, nummer, Lettertype

Automatisch gegenereerde beschrijving

1. On which weekday most flights are booked? -1 means there are more 10 seats available, but the exact number is unknown
   * Use a cte to add an extra column that contains the number of seats available for the next day
   * Drop the records for which both columns (seats available today and seats available tomorrow) both contain -1
   * Drop the records for which the column seats available tomorrow contains -1
   * If seats available today = -1, replace is by 11 using IIF
   * Drop the records for which the column seats available tomorrow < seats available today
   * Calculate the difference between the seats available today and the seats available tomorrow. Make the sum
2. Ryanair does a lot to earn money. Do they also slightly raise their prices on the day on which most flights are booked? Calculate the average price per weekday for Ryanair flights.
3. Create a visual showing the price evolution of flights departing in a specified month for a specified airline based on the number of days before departure
   * Create a stored procedure price\_evolution with 2 parameters: the name of the airline and the name of the month
   * Check if the airline exists. If not print an error message
   * Check if the month exists. If not print an error message
   * Write an SQL statement that calculates the average price of all flights of the specific airline for the specific month, with number of days before departure being less than 90. A cte can be used to add an extra column to calculate the number of days before departure
   * Use a cursor to loop through this SQL statement and use REPLICATE('x',@avg\_price / 2) to create some kind of visual
   * Write testcode