**ETL – Project**

**Goal**

The objective for this project is to process flight details data from csv files into a Data Warehouse. Create an ETL pipeline in python and SQL to load the final tables. Also perform basic analysis on the final tables.

**Data**

The given data is of US flights for the year 2008.

1. L\_AIRPORT: Lookup table containing airport locations and airport code

|  |  |  |
| --- | --- | --- |
| **col\_name** | **data\_type** | **Description** |
| Code | string | Airport Code |
| Description | string | Airport description in the following format (city, country: name) |

1. L\_CANCELLATION: Lookup table containing cancellation reasons.

|  |  |  |
| --- | --- | --- |
| **col\_name** | **data\_type** | **Description** |
| Code | string | Cancelation code A to D |
| Description | string | Cancelation reason description |

1. L\_UNIQUE\_CARRIERS: Lookup table for unique carriers

|  |  |  |
| --- | --- | --- |
| **col\_name** | **data\_type** | **Description** |
| Code | string | Unique carrier code |
| Description | string | Carrier description |

1. L\_PLANE: Plane lookup table

|  |  |  |
| --- | --- | --- |
| **col\_name** | **data\_type** | **Description** |
| tailnum | string | Tail Number |
| type | string | Type of the plane |
| manufacturer | string | Manufacturer |
| issue\_date | date | Date of manufacturing |
| model | string | Model of the plane |
| status | string | Status of the plane (valid or not) |
| aircraft\_type | string | Air craft type |
| engine\_type | string | Engine Type |
| year | int | Year of manufacturing |

1. FLIGHT\_2008: Flight details along with departure times for the year 2008.

|  |  |  |
| --- | --- | --- |
| **col\_name** | **data\_type** | **Description** |
| Year | int | Year |
| Month | int | Month |
| DayofMonth | int | Day of Month |
| DayOfWeek | int | Day of Week |
| DepTime | double | Actual Departure Time (local time: hhmm) |
| CRSDepTime | int | CRS Departure Time (local time: hhmm) |
| ArrTime | double | Actual Arrival Time (local time: hhmm) |
| CRSArrTime | int | CRS Arrival Time (local time: hhmm) |
| UniqueCarrier | string | Unique Carrier Code. When the same code has been used by multiple carriers, a numeric suffix is used for earlier users, for example, PA, PA (1), PA (2). Use this field for analysis across a range of years |
| FlightNum | int | Flight Number |
| TailNum | string | Tail Number |
| ActualElapsedTime | double | Elapsed Time of Flight, in Minutes |
| CRSElapsedTime | int | CRS Elapsed Time of Flight, in Minutes |
| AirTime | string | Flight Time, in Minutes |
| ArrDelay | double | Difference in minutes between scheduled and actual arrival time. Early arrivals show negative numbers |
| DepDelay | double | Difference in minutes between scheduled and actual departure time. Early departures show negative numbers |
| Origin | string | Origin Airport |
| Dest | string | Destination Airport |
| Distance | double | Distance between airports (miles) |
| TaxiIn | string | Taxi In Time, in Minutes |
| TaxiOut | string | Taxi Out Time, in Minutes |
| Cancelled | int | Cancelled Flight Indicator (1=Yes) |
| CancellationCode | string | Specifies The Reason For Cancellation |
| Diverted | int | Diverted Flight Indicator (1=Yes) |
| CarrierDelay | string | Carrier Delay, in Minutes |
| WeatherDelay | string | Weather Delay, in Minutes |
| NASDelay | string | Weather Delay, in Minutes |
| SecurityDelay | string | Security Delay, in Minutes |
| LateAircraftDelay | string | Late Aircraft Delay, in Minutes |

The project should be (typically) executed in the following steps:

1. Scope and Gather data: The data set is already given in this case. No storage layer required.
2. Explore and assess the data: Explore each dataset and examine if every file is in usable state.

Examination of data should include the below:

* Data cleansing: NULL values and how to handle them, Check for any truncation of data, columns to be split etc.
* Relationships: Search for the unique rows in each table, if not present look to include one. Establish a relationship between all the tables.

1. Define the data model: After exploring the data, design the source and target table structures. And build and ER model.
2. Create the pipeline: Now comes the fun part of implementation! Break down the code into Extract; Transform and Load steps. Once loaded do data quality checks, eg: Compare NULL values (should they be present or not), compare unique values before and after transformations.
3. Documentation: Document all the steps followed in the project diligently and verbatim in your GitHub profile. You will thank yourself later for doing this.

**Analysis:** Once in the final DB run SQL queries on the data to answer the below questions.

1. Find the total flights by airline.
2. Find the number of total flights cancelled for each airline for which cancellation reason was not in weather or security related.
3. Check out the average delay on airline to see what airlines have the longest delays.
4. Can you see a trend in the delays by month and day of week?