Data Management and Business Intelligence

2nd Assignment

Christos Kallaras, p2822009

Clayton Preston, p2822025

Athens, 14/12/2020

Contents

1. Abstract

The purpose of this project is to build and design a Data Warehouse based on a dataset of our choice. Then build a data cube on top of it, develop OLAP reports and visualize our results. For this project we used SQL Server Database, SQL Server Analysis Services and Power BI.

2. Description of the case

Covid-19 pandemic has been the biggest event in the entire world for this year and has affected many areas in a lot of ways. In this project we will present how Covid-19 pandemic has evolved over the past year and how has affected air traffic in order to be able to have a better understanding of how the following months will evolve.

3. Datasets

For the purposes of this analysis, we needed data for both air traffics in 2020 and the number of Covid-19 cases of each country around the world.

3.1 Covid-19 Dataset

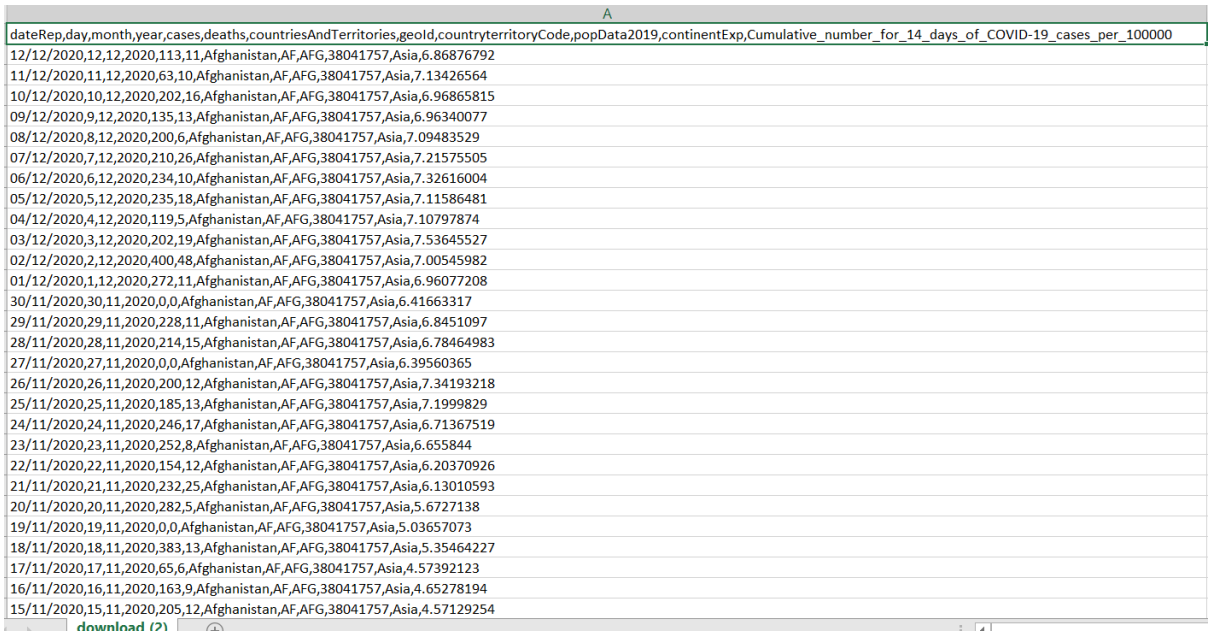
The Covid-19 Dataset was retrieved from the site data.europa.eu and it shows data of the geographic distribution of Covid-19 cases worldwide for the period 31/12/2019 to 28/11/2020. The European Union Open Data Portal (EU ODP) provides access to an expanding range of data from the European Union (EU) institutions and other EU bodies. It is mentioned in the site info that “all these data are freely available. They can be reused in databases, reports or projects”.

The dataset consists of:

- *dateRep*: date the measurement (cases and deaths) was recorded
- *day*: day the measurement (cases and deaths) was recorded
- *month*: month the measurement (cases and deaths) was recorded
- *year*: year the measurement (cases and deaths) was recorded
- *cases*: number of official cases regarding Covid-19 infection for the given date
- *deaths*: number of official deaths due to Covid-19 infection
- *countriesAndTerritories*: country or territory which the number of cases and deaths was recorded
- *geoId*: country geographical code
- *countryterritoryCode*: country or territory code
- *popData2019*: population of the country or territory based on 2019 records
- *continentExp*: continent within which the country is located

- *Cumulative_number_for_14_days_of_COVID-19_cases_per_100000*: sum of the previous 14 days number of cases for 100000 people

Figure 1 shows a sample of the dataset as founded on web.



A
dateRep,day,month,year,cases,deaths,countriesAndTerritories,geoId,countryterritoryCode,popData2019,continentExp,Cumulative_number_for_14_days_of_COVID-19_cases_per_100000
12/12/2020,12,12,2020,113,11,Afghanistan,AF,AFG,38041757,Asia,6.86876792
11/12/2020,11,12,2020,63,10,Afghanistan,AF,AFG,38041757,Asia,7.13426564
10/12/2020,10,12,2020,202,16,Afghanistan,AF,AFG,38041757,Asia,6.96865815
09/12/2020,9,12,2020,135,13,Afghanistan,AF,AFG,38041757,Asia,6.96340077
08/12/2020,8,12,2020,200,6,Afghanistan,AF,AFG,38041757,Asia,7.09483529
07/12/2020,7,12,2020,210,26,Afghanistan,AF,AFG,38041757,Asia,7.21575505
06/12/2020,6,12,2020,234,10,Afghanistan,AF,AFG,38041757,Asia,7.32616004
05/12/2020,5,12,2020,235,18,Afghanistan,AF,AFG,38041757,Asia,7.11586481
04/12/2020,4,12,2020,119,5,Afghanistan,AF,AFG,38041757,Asia,7.10797874
03/12/2020,3,12,2020,202,19,Afghanistan,AF,AFG,38041757,Asia,7.53645527
02/12/2020,2,12,2020,400,48,Afghanistan,AF,AFG,38041757,Asia,7.00545982
01/12/2020,1,12,2020,272,11,Afghanistan,AF,AFG,38041757,Asia,6.96077208
30/11/2020,30,11,2020,0,0,Afghanistan,AF,AFG,38041757,Asia,6.41663317
29/11/2020,29,11,2020,228,11,Afghanistan,AF,AFG,38041757,Asia,6.8451097
28/11/2020,28,11,2020,214,15,Afghanistan,AF,AFG,38041757,Asia,6.78464983
27/11/2020,27,11,2020,0,0,Afghanistan,AF,AFG,38041757,Asia,6.39560365
26/11/2020,26,11,2020,200,12,Afghanistan,AF,AFG,38041757,Asia,7.34193218
25/11/2020,25,11,2020,185,13,Afghanistan,AF,AFG,38041757,Asia,7.1999829
24/11/2020,24,11,2020,246,17,Afghanistan,AF,AFG,38041757,Asia,6.71367519
23/11/2020,23,11,2020,252,8,Afghanistan,AF,AFG,38041757,Asia,6.655844
22/11/2020,22,11,2020,154,12,Afghanistan,AF,AFG,38041757,Asia,6.20370926
21/11/2020,21,11,2020,232,25,Afghanistan,AF,AFG,38041757,Asia,6.13010593
20/11/2020,20,11,2020,282,5,Afghanistan,AF,AFG,38041757,Asia,5.6727138
19/11/2020,19,11,2020,0,0,Afghanistan,AF,AFG,38041757,Asia,5.03657073
18/11/2020,18,11,2020,383,13,Afghanistan,AF,AFG,38041757,Asia,5.35464227
17/11/2020,17,11,2020,65,6,Afghanistan,AF,AFG,38041757,Asia,4.57392123
16/11/2020,16,11,2020,163,9,Afghanistan,AF,AFG,38041757,Asia,4.65278194
15/11/2020,15,11,2020,205,12,Afghanistan,AF,AFG,38041757,Asia,4.57129254

download (2)

Figure 1 : Sample of Covid-19 dataset

3.2 Flights Dataset

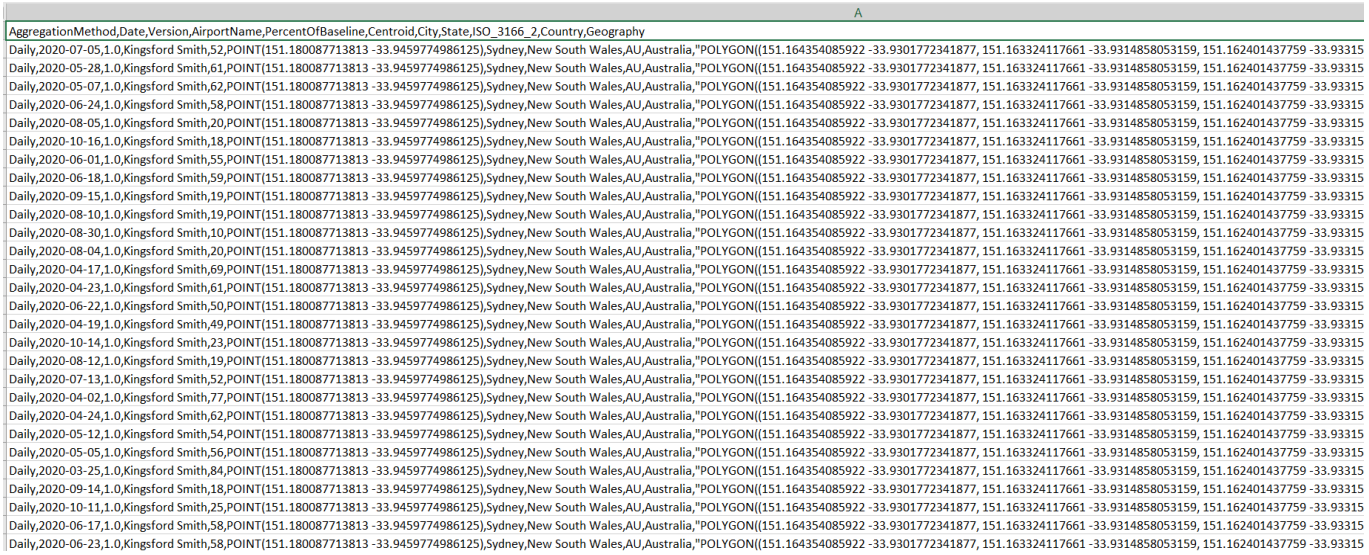
The flights dataset was retrieved from kaggle.com and shows traffic to and from an airport as a percentage of the traffic volume during the baseline period. The baseline period used for computing this metric is from 1st February to 15th March 2020.The data are from Australia, Chile, Canada and United States of America and it covers the period from 16/3/2020 to 16/10/2020.

The dataset consists of:

- *AggregationMethod*: aggregation period used to compute this metric (has Daily as unique value)
- *Date*: date of flight logging
- *Version*: version of the data
- *AirportName*: name of the airport which the the given flight occurred
- *PercentOfBaseline*: proportion of trips on this date as compared to average number of trips on the same day of week in baseline period i.e., 1st
- *Centroid*: geography representing centroid of the airport polygon

- *City*: City within which the Airport is located
- *State*: State within which the Airport is located
- *ISO_3166_2*: ISO-3166-2 code representing Country and Subdivision
- *Country*: Country within which the Airport is located
- *Geography*: Polygon of the airport that is used to compute this metric

Figure 2 shows a sample of the dataset as founded on web.



A

AggregationMethod,Date,Version,AirportName,PercentOfBaseline,Centroid,City,State,ISO_3166_2,Country,Geography

Daily,2020-07-05,1.0,Kingsford Smith,52,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-05-28,1.0,Kingsford Smith,61,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-05-07,1.0,Kingsford Smith,62,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-06-24,1.0,Kingsford Smith,58,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-08-05,1.0,Kingsford Smith,20,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-10-16,1.0,Kingsford Smith,18,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-06-01,1.0,Kingsford Smith,55,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-06-18,1.0,Kingsford Smith,59,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-09-15,1.0,Kingsford Smith,19,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-08-10,1.0,Kingsford Smith,19,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-08-30,1.0,Kingsford Smith,10,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-08-04,1.0,Kingsford Smith,20,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-04-17,1.0,Kingsford Smith,69,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-04-23,1.0,Kingsford Smith,61,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-06-22,1.0,Kingsford Smith,50,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-04-19,1.0,Kingsford Smith,49,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-10-14,1.0,Kingsford Smith,23,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-08-12,1.0,Kingsford Smith,19,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-07-13,1.0,Kingsford Smith,52,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-04-02,1.0,Kingsford Smith,77,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-04-24,1.0,Kingsford Smith,62,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-05-12,1.0,Kingsford Smith,54,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-05-05,1.0,Kingsford Smith,56,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-03-25,1.0,Kingsford Smith,84,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-09-14,1.0,Kingsford Smith,18,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-10-11,1.0,Kingsford Smith,25,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-06-17,1.0,Kingsford Smith,58,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315
Daily,2020-06-23,1.0,Kingsford Smith,58,POINT(151.180087713813 -33.9459774986125),Sydney,New South Wales,AU,Australia,"POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93315

Figure 2 : Sample of flights dataset

4. Data Warehouse

In this sector we will show how we designed and built a data warehouse using the datasets of the previous sectors and following the ETL (Extract – Transform – Load) procedure.

4.1 Extract Data

After scouting the web, we found the datasets we needed from the sources we mentioned in 3.1 and 3.2. The data opened in excel was as one column like in Figure 1 and Figure 2, so we delimited by comma and we got the Figure 3 and Figure 4.

dateRep	day	month	year	cases	deaths	countriesAndTerritories	geoId	countryter	popData2(	continentE	Cumulative_number_for_14_days_of_COVID-19_cases_per_100000
28/11/2020	28	11	2020	214	15	Afghanistan	AF	AFG	38041757	Asia	678.464.983
27/11/2020	27	11	2020	0	0	Afghanistan	AF	AFG	38041757	Asia	639.560.365
26/11/2020	26	11	2020	200	12	Afghanistan	AF	AFG	38041757	Asia	734.193.218
25/11/2020	25	11	2020	185	13	Afghanistan	AF	AFG	38041757	Asia	71.999.829
24/11/2020	24	11	2020	246	17	Afghanistan	AF	AFG	38041757	Asia	671.367.519
23/11/2020	23	11	2020	252	8	Afghanistan	AF	AFG	38041757	Asia	6.655.844
22/11/2020	22	11	2020	154	12	Afghanistan	AF	AFG	38041757	Asia	620.370.926
21/11/2020	21	11	2020	232	25	Afghanistan	AF	AFG	38041757	Asia	613.010.593
20/11/2020	20	11	2020	282	5	Afghanistan	AF	AFG	38041757	Asia	56.727.138
19/11/2020	19	11	2020	0	0	Afghanistan	AF	AFG	38041757	Asia	503.657.073
18/11/2020	18	11	2020	383	13	Afghanistan	AF	AFG	38041757	Asia	535.464.227
17/11/2020	17	11	2020	65	6	Afghanistan	AF	AFG	38041757	Asia	457.392.123
16/11/2020	16	11	2020	163	9	Afghanistan	AF	AFG	38041757	Asia	465.278.194
15/11/2020	15	11	2020	205	12	Afghanistan	AF	AFG	38041757	Asia	457.129.254
14/11/2020	14	11	2020	66	10	Afghanistan	AF	AFG	38041757	Asia	423.219.148
13/11/2020	13	11	2020	360	14	Afghanistan	AF	AFG	38041757	Asia	447.140.231
12/11/2020	12	11	2020	146	4	Afghanistan	AF	AFG	38041757	Asia	384.840.269
11/11/2020	11	11	2020	0	0	Afghanistan	AF	AFG	38041757	Asia	34.646.139
10/11/2020	10	11	2020	224	12	Afghanistan	AF	AFG	38041757	Asia	376.165.591
9/11/2020	9	11	2020	80	3	Afghanistan	AF	AFG	38041757	Asia	369.593.865
8/11/2020	8	11	2020	126	6	Afghanistan	AF	AFG	38041757	Asia	365.650.829
7/11/2020	7	11	2020	58	2	Afghanistan	AF	AFG	38041757	Asia	353.821.723

Figure 3 : Covid-19 dataset after delimitation

Aggregatic	Date	Version	AirportNa	PercentOf	Centroid	City	State	ISO_3166_	Country	Geography
Daily	5/7/2020	1.0	Kingsford	52	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	28/5/2020	1.0	Kingsford	61	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	7/5/2020	1.0	Kingsford	62	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	24/6/2020	1.0	Kingsford	58	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	5/8/2020	1.0	Kingsford	20	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	16/10/2020	1.0	Kingsford	18	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	1/6/2020	1.0	Kingsford	55	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	18/6/2020	1.0	Kingsford	59	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	15/9/2020	1.0	Kingsford	19	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	10/8/2020	1.0	Kingsford	19	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	30/8/2020	1.0	Kingsford	10	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	4/8/2020	1.0	Kingsford	20	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	17/4/2020	1.0	Kingsford	69	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	23/4/2020	1.0	Kingsford	61	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	22/6/2020	1.0	Kingsford	50	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	19/4/2020	1.0	Kingsford	49	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	14/10/2020	1.0	Kingsford	23	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	12/8/2020	1.0	Kingsford	19	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	13/7/2020	1.0	Kingsford	52	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	2/4/2020	1.0	Kingsford	77	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	24/4/2020	1.0	Kingsford	62	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	12/5/2020	1.0	Kingsford	54	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	5/5/2020	1.0	Kingsford	56	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	25/3/2020	1.0	Kingsford	84	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	14/9/2020	1.0	Kingsford	18	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	11/10/2020	1.0	Kingsford	25	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	17/6/2020	1.0	Kingsford	58	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93
Daily	23/6/2020	1.0	Kingsford	58	POINT(151.180087713813 -33.94	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.9301772341877, 151.163324117661 -33.9314858053159, 151.162401437759 -33.93

Figure 4 : Flight dataset after delimitation

Next step is transforming the data but first we need to build a database and the staging area.

4.2 Build Database

Using MSSQL we created a new database under our local server called **covid19_impact**.

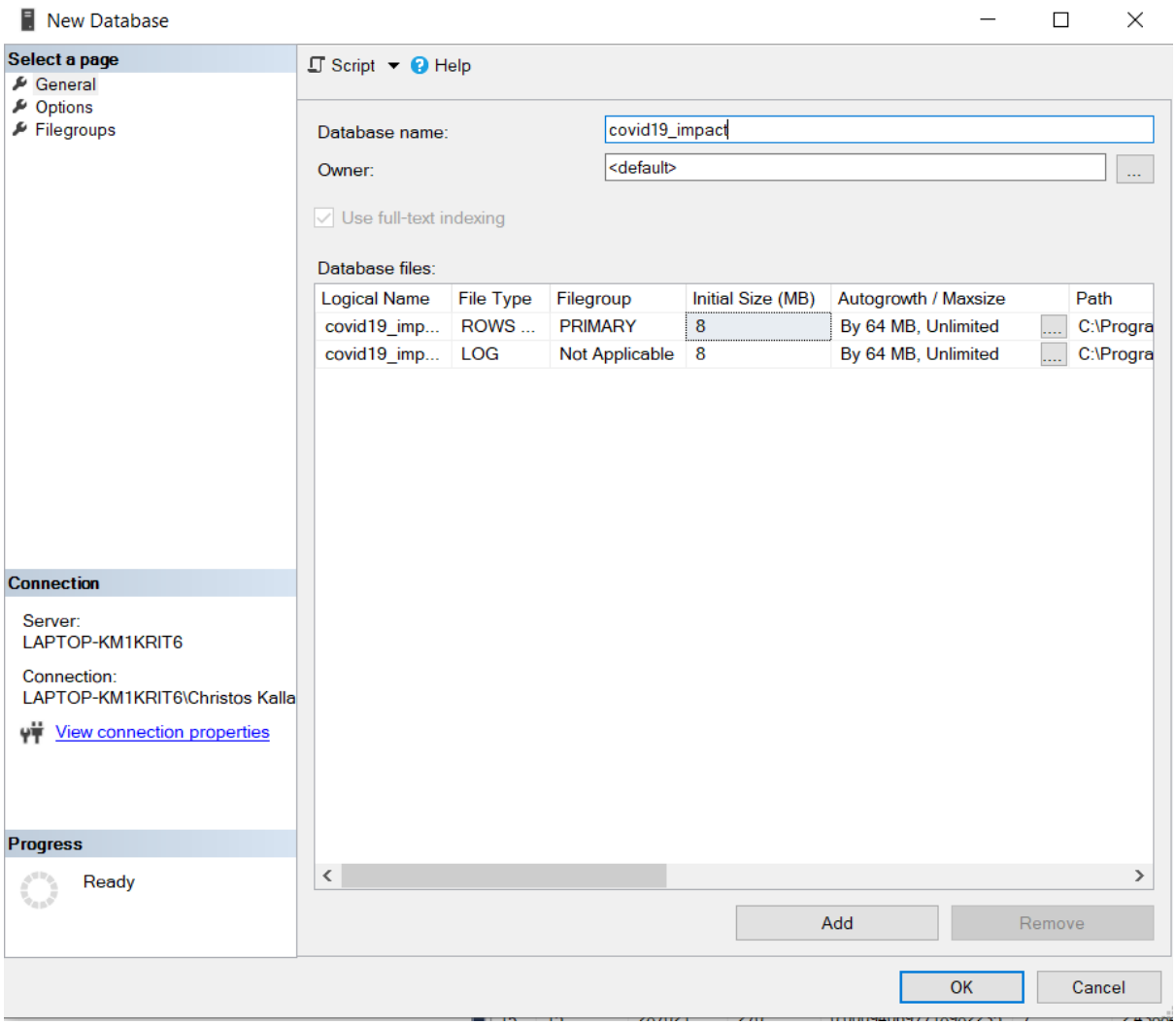


Figure 5 : Creation of covid19_impact database

This database will have tables for the staging area and the table needed for the Data Warehouse.

4.3 Transform Data - Create Staging Area

The data were quite clean from the start as they were prepared datasets; not something we scraped from the internet, etc. But still they needed some modifications.

Before we create the staging area, we can make some transformations to our data through excel.

In the flights dataset there is a column called Centroid that represents the centroid of the airport polygon. They are of the following form: POINT(latitude longitude) , for example POINT(151.180087713813 -33.9459774986125). We are only interest for the latitude and longitude, and it is better to have them as two separate columns in order to be able to use them more efficiently late on. So, we split it by the space in the middle and removed the strings at either end so we can use the Latitude and Longitude within. Figure 4 shows the excel at the beginning of the process, Figure 6 and Figure 7 are during the transformation when we delimitated column G(Centroid) with "(" and then

again with spaces, and Figure 8 shows the final result after we removed column F(since it only has the string “POINT”) and delimited longitude with “)” .

Aggregatic	Date	Version	AirportNar	PercentOfBaseline	Centroid		City	State	ISO_3166_	Country	Geography
Daily	5/7/2020	1.0	Kingsford S	52	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	28/5/2020	1.0	Kingsford S	61	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	7/5/2020	1.0	Kingsford S	62	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	24/6/2020	1.0	Kingsford S	58	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	5/8/2020	1.0	Kingsford S	20	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	16/10/2020	1.0	Kingsford S	18	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	1/6/2020	1.0	Kingsford S	55	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	18/6/2020	1.0	Kingsford S	59	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	15/9/2020	1.0	Kingsford S	19	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	10/8/2020	1.0	Kingsford S	19	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	30/8/2020	1.0	Kingsford S	10	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	4/8/2020	1.0	Kingsford S	20	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	17/4/2020	1.0	Kingsford S	69	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	23/4/2020	1.0	Kingsford S	61	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	22/6/2020	1.0	Kingsford S	50	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	19/4/2020	1.0	Kingsford S	49	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	14/10/2020	1.0	Kingsford S	23	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	12/8/2020	1.0	Kingsford S	19	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	13/7/2020	1.0	Kingsford S	52	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	2/4/2020	1.0	Kingsford S	77	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	24/4/2020	1.0	Kingsford S	62	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	12/5/2020	1.0	Kingsford S	54	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	5/5/2020	1.0	Kingsford S	56	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	25/3/2020	1.0	Kingsford S	84	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	14/9/2020	1.0	Kingsford S	18	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	11/10/2020	1.0	Kingsford S	25	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	17/6/2020	1.0	Kingsford S	58	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.
Daily	23/6/2020	1.0	Kingsford S	58	POINT	151.180087713813 -33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((151.164354085922 -33.

Figure 6 : Transformation in csv 1

A	B	C	D	E	F	G	H	I	J	K	L	M
Aggregatic	Date	Version	AirportNar	PercentOfBaseline	Centroid			City	State	ISO_3166_	Country	Geography
Daily	5/7/2020	1.0	Kingsford S	52	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	28/5/2020	1.0	Kingsford S	61	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	7/5/2020	1.0	Kingsford S	62	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	24/6/2020	1.0	Kingsford S	58	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	5/8/2020	1.0	Kingsford S	20	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	16/10/2020	1.0	Kingsford S	18	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	1/6/2020	1.0	Kingsford S	55	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	18/6/2020	1.0	Kingsford S	59	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	15/9/2020	1.0	Kingsford S	19	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	10/8/2020	1.0	Kingsford S	19	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	30/8/2020	1.0	Kingsford S	10	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	4/8/2020	1.0	Kingsford S	20	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	17/4/2020	1.0	Kingsford S	69	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	23/4/2020	1.0	Kingsford S	61	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	22/6/2020	1.0	Kingsford S	50	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	19/4/2020	1.0	Kingsford S	49	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	14/10/2020	1.0	Kingsford S	23	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	12/8/2020	1.0	Kingsford S	19	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	13/7/2020	1.0	Kingsford S	52	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	2/4/2020	1.0	Kingsford S	77	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	24/4/2020	1.0	Kingsford S	62	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	12/5/2020	1.0	Kingsford S	54	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	5/5/2020	1.0	Kingsford S	56	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	25/3/2020	1.0	Kingsford S	84	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	14/9/2020	1.0	Kingsford S	18	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	11/10/2020	1.0	Kingsford S	25	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	17/6/2020	1.0	Kingsford S	58	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((
Daily	23/6/2020	1.0	Kingsford S	58	POINT	151.180.087.713.813	-33.9459774986125)	Sydney	New South	AU	Australia	POLYGON((

Figure 7 : Transformation in csv 2

A	B	C	D	E	F	G	H	I	J	K	
AggregationMethod	Date	Version	AirportNar	PercentOfBaseline	Latitude	Longitude	City	State	ISO_3166_	Country	Geogra
Daily	5/7/2020	1.0	Kingsford S	52	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	28/5/2020	1.0	Kingsford S	61	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	7/5/2020	1.0	Kingsford S	62	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	24/6/2020	1.0	Kingsford S	58	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	5/8/2020	1.0	Kingsford S	20	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	16/10/2020	1.0	Kingsford S	18	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	1/6/2020	1.0	Kingsford S	55	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	18/6/2020	1.0	Kingsford S	59	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	15/9/2020	1.0	Kingsford S	19	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	10/8/2020	1.0	Kingsford S	19	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	30/8/2020	1.0	Kingsford S	10	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	4/8/2020	1.0	Kingsford S	20	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	17/4/2020	1.0	Kingsford S	69	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	23/4/2020	1.0	Kingsford S	61	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	22/6/2020	1.0	Kingsford S	50	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	19/4/2020	1.0	Kingsford S	49	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	14/10/2020	1.0	Kingsford S	23	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	12/8/2020	1.0	Kingsford S	19	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	13/7/2020	1.0	Kingsford S	52	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	2/4/2020	1.0	Kingsford S	77	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	24/4/2020	1.0	Kingsford S	62	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	12/5/2020	1.0	Kingsford S	54	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	5/5/2020	1.0	Kingsford S	56	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	25/3/2020	1.0	Kingsford S	84	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	14/9/2020	1.0	Kingsford S	18	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	11/10/2020	1.0	Kingsford S	25	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	17/6/2020	1.0	Kingsford S	58	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO
Daily	23/6/2020	1.0	Kingsford S	58	151.180.087.713.813	-339.459.774.986.125	Sydney	New South Wales	AU	Australia	POLYGO

Figure 8 : Final csv form

Since we have 2 datasets, we can create 2 new tables for our staging area. These tables will be used to transform our data if needed before we load them into the data warehouse.

The first table will be **staging_covid** and it will have the dataset for the Covid-19 worldwide cases. We will use the import data wizard of MSSQL as in Figure 9 and Figure 10. Our source file is the excel with the data and the destination is a new table in our database.

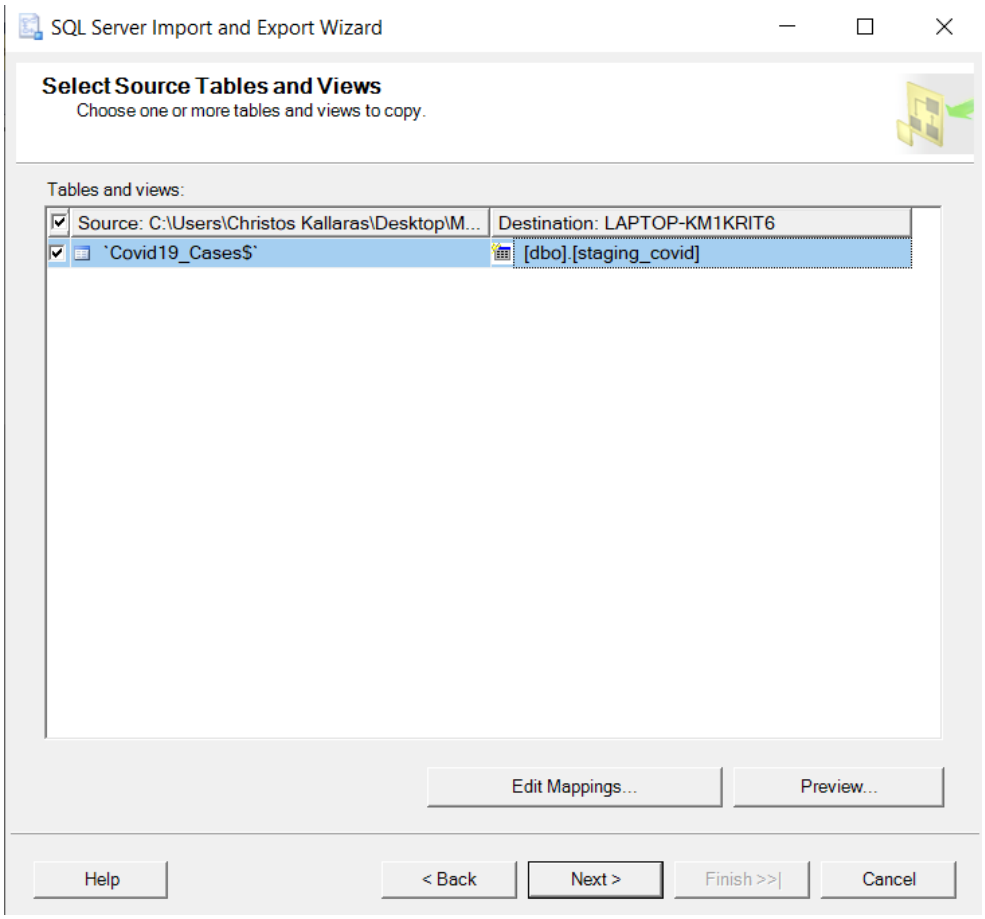


Figure 9 : MSSQS import data wizard for staging_covid

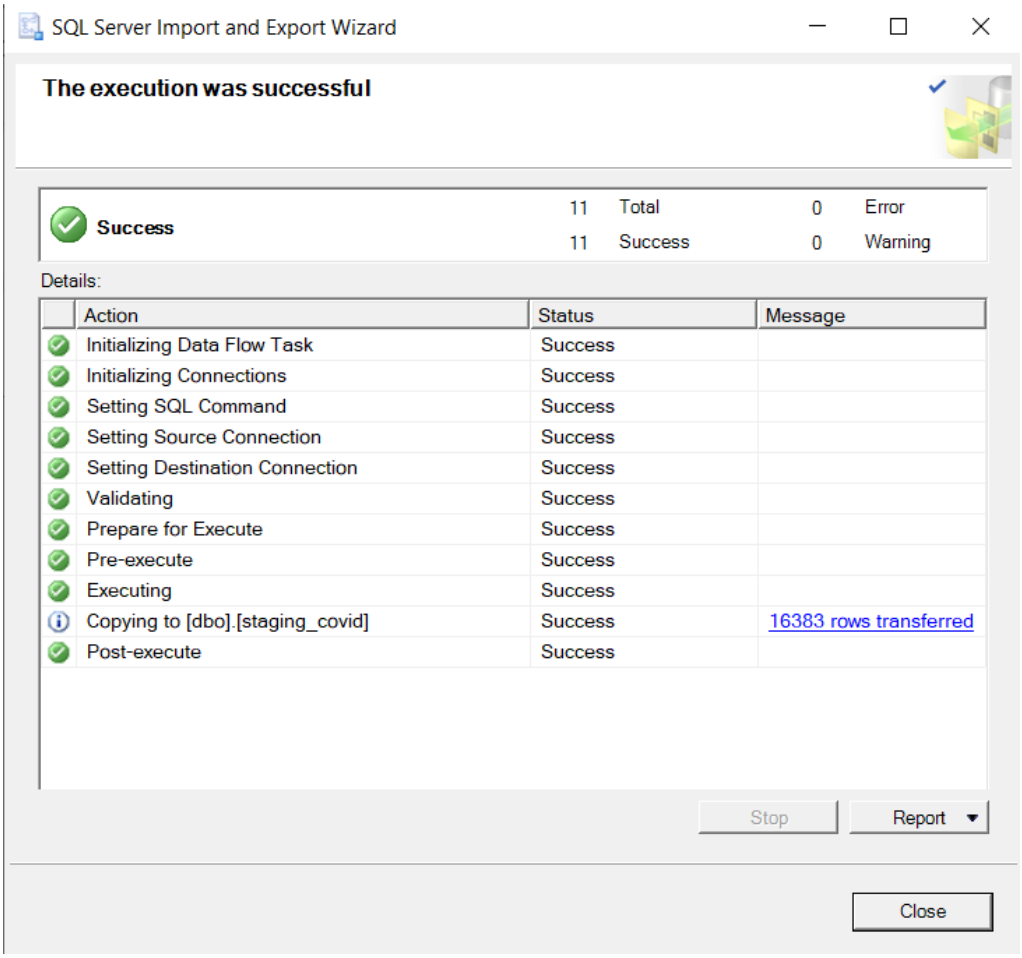


Figure 10 : MSSQS import data successfully for staging_covid

Every column of the excel source file will be represented with a table column in the **staging_covid** table. The design of the table after the import is showing in Figure 11

Column Name	Data Type	Allow Nulls
dateRep	datetime	☑
day	float	☑
month	float	☑
year	float	☑
cases	float	☑
deaths	float	☑
countriesAndTerritories	nvarchar(255)	☑
geoId	nvarchar(255)	☑
countryterritoryCode	nvarchar(255)	☑
popData2019	float	☑
continentExp	nvarchar(255)	☑
[Cumulative_number_for_14_days...	float	☑

Figure 11 : staging_covid table design

MSSQS has automatically declared dateRep as datetime which means that the date will be in the following form: *date timestamp.* Figure 12 shows a query execution with the dateRep as datetime type.

	dateRep	day	month	year	cases	deaths	countriesAndTerritories	geoId	countryterritoryCode	popData2019	continentExp	Cumulative_number_for_14_days_of_COVID-19_cases_per_1
1	2020-11-28 00:00:00.000	28	11	2020	214	15	Afghanistan	AF	AFG	38041757	Asia	678464983
2	2020-11-27 00:00:00.000	27	11	2020	0	0	Afghanistan	AF	AFG	38041757	Asia	639560365
3	2020-11-26 00:00:00.000	26	11	2020	200	12	Afghanistan	AF	AFG	38041757	Asia	734193218
4	2020-11-25 00:00:00.000	25	11	2020	185	13	Afghanistan	AF	AFG	38041757	Asia	71999829
5	2020-11-24 00:00:00.000	24	11	2020	246	17	Afghanistan	AF	AFG	38041757	Asia	671367519
6	2020-11-23 00:00:00.000	23	11	2020	252	8	Afghanistan	AF	AFG	38041757	Asia	6655844
7	2020-11-22 00:00:00.000	22	11	2020	154	12	Afghanistan	AF	AFG	38041757	Asia	620370926
8	2020-11-21 00:00:00.000	21	11	2020	232	25	Afghanistan	AF	AFG	38041757	Asia	613010593
9	2020-11-20 00:00:00.000	20	11	2020	282	5	Afghanistan	AF	AFG	38041757	Asia	56727138
10	2020-11-19 00:00:00.000	19	11	2020	0	0	Afghanistan	AF	AFG	38041757	Asia	503657073
11	2020-11-18 00:00:00.000	18	11	2020	383	13	Afghanistan	AF	AFG	38041757	Asia	535464227
12	2020-11-17 00:00:00.000	17	11	2020	65	6	Afghanistan	AF	AFG	38041757	Asia	457392123

Figure 12 : dateRep datetime format

In order to use dateRep in our calculations and reports we have to change its type to date (there is no need for the sql ALTER command, we can easily change it with MSSQS UI) and so we have the new design of staging_covid as shows in Figure 13.

Column Name	Data Type	Allow Nulls
dateRep	date	☑
day	float	☑
month	float	☑
year	float	☑
cases	float	☑
deaths	float	☑
countriesAndTerritories	nvarchar(255)	☑
geoId	nvarchar(255)	☑
countryterritoryCode	nvarchar(255)	☑
popData2019	float	☑
continentExp	nvarchar(255)	☑
[Cumulative_number_for_14_days...	float	☑

Figure 13 : New staging_covid design

Since we created the staging table for the Covid-19 dataset, we have to create the staging table for the flights dataset. We will use again the MSSQS import data wizard. Our source file is the excel with the data and the destination is a new table in our database. The process is depicted in Figure 14 and Figure 15 .

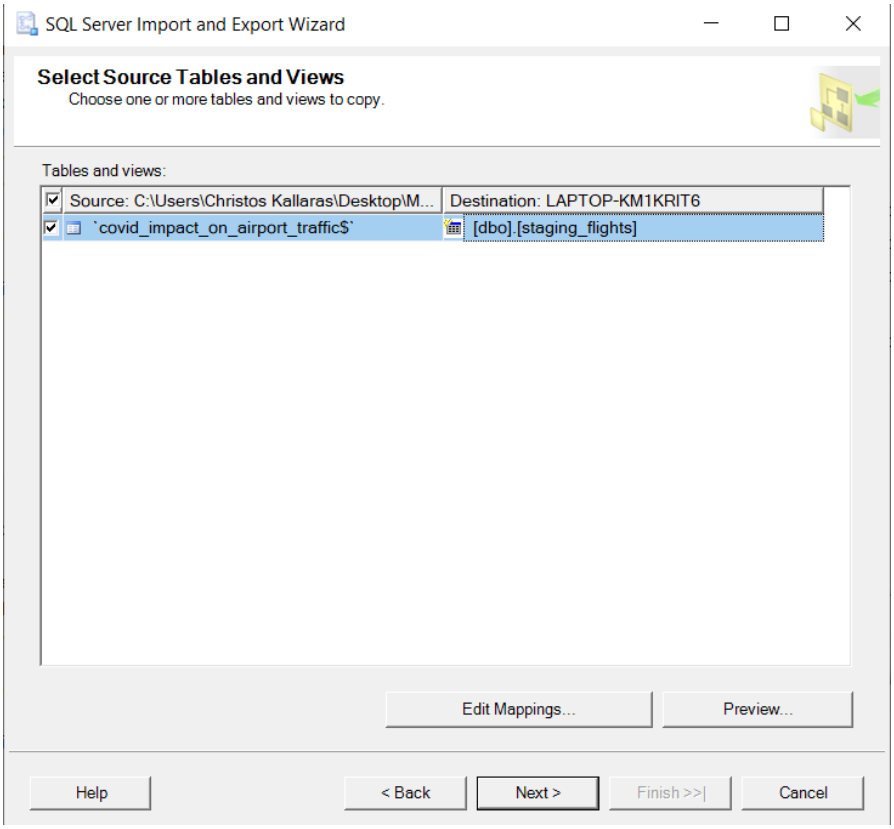


Figure 14 : MSSQS import data wizard for staging_flights

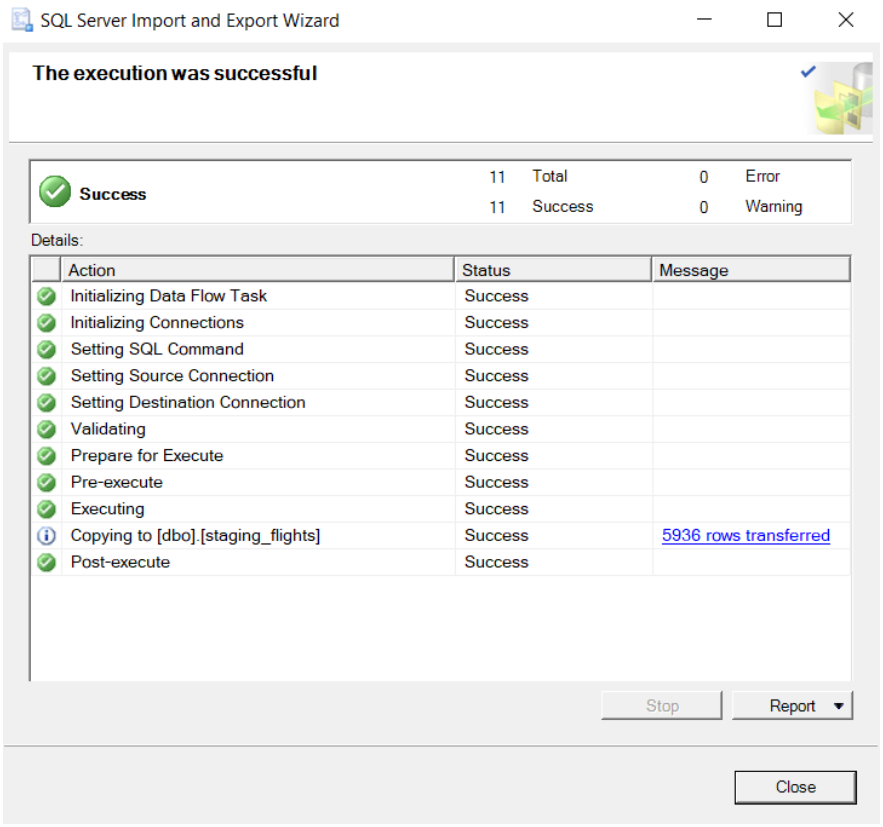


Figure 15 : MSSQS import data successfully for staging_flights

Every column of the excel source file will be represented with a table column in the **staging_flights** table. As with **staging_covid** we also need to change the type of Date to date, so the new design of the table is depicted in Figure 16 .

Column Name	Data Type	Allow Nulls
AggregationMethod	nvarchar(255)	☑
Date	date	☑
Version	nvarchar(255)	☑
AirportName	nvarchar(255)	☑
PercentOfBaseline	float	☑
Latitude	float	☑
Longitude	float	☑
City	nvarchar(255)	☑
State	nvarchar(255)	☑
ISO_3166_2	nvarchar(255)	☑
Country	nvarchar(255)	☑
Geography	nvarchar(MAX)	☑

Figure 16 : New staging_flights design

Now that both staging tables are complete, we can continue with modifications to the data.

1. Date type: we already change the dates type to date in order to be more useful for us when we create the dimensions table later. The change can also happen with sql commands like in Query 1 .

```
ALTER TABLE [covid19_impact].[dbo].[staging_covid]
ALTER COLUMN [dateRep] date;

ALTER TABLE [covid19_impact].[dbo].[staging_flights]
ALTER COLUMN [Date] date;
```

Query 1 : Change type of dates to date

2. Delete unnecessary columns: not all columns have information that we need for this analysis. Those columns are

    a. *AggregationMethod* from staging_flights: the column has the unique value Daily for all records and it does not depict information of importance for the dataset since it is everywhere the same.

    b. *Version* from staging_flights: the column shows the version of the dataset which also is not of significant.

    c. *Geography* from staging_flights:we already have the latitude and longitude so we don't need the polygon of the Airport.

d. *Cumulative_number_for_14_days_of_COVID-19_cases_per_100000* from staging_covid: we will not use the values from this column for our analysis, we have the number of cases and population so we can use metrics to achieve that result.

The SQL commands are showing in Query 2 .

```
ALTER TABLE [covid19_impact].[dbo].[staging_flights]
DROP COLUMN [AggregationMethod];

ALTER TABLE [covid19_impact].[dbo].[staging_flights]
DROP COLUMN [Version];


ALTER TABLE [covid19_impact].[dbo].[staging_flights]
DROP COLUMN [Geography];

ALTER TABLE [covid19_impact].[dbo].[staging_covid]
DROP COLUMN [Cumulative_number_for_14_days_of_COVID-19_cases_per_100000];
```

Query 2 : Delete unnecessary columns

3. Null values: we need to search the tables for possible null values. Executing query of the form of Query 3 for all columns of staging_covid we get the results of Result 1 .

```
SELECT distinct [dateRep]
 FROM [covid19_impac].[dbo].[staging_covid]
 WHERE [dateRep] IS null;
```

Query 3 : Null search query

dateRep
day
month
year
cases
deaths
countriesAndTerritories
geoId
countryterritoryCode
NULL
popData2019
NULL
continentExp

Result 1 : Null search results

By executing Query 4 we can see that the country “Wallis_and_Futuna” and the “Cases_on_an_international_conveyance_Japan” both have null in the population column (popData2019) and in countryterritoryCode. By searching the web, we found that the **population of Wallis and Futuna is 12.067** according to 2019 records (source: Wikipedia) and so we can updated it. For the “Cases_on_an_international_conveyance_Japan” there are no records since it was a case of ship in Japan where the crew got infected with Covisd-19, and therefore we will **update it with 0**. Since both countryterritoryCode and geoId shows the same information (country code) we will update the **records with null countryterritoryCode with the value of geoId**. Query 5 shows the update process.

```
SELECT distinct countriesAndTerritories
 FROM [covid19_impac].[dbo].[staging_covid]
 where [countryterritoryCode] is null;

SELECT distinct countriesAndTerritories
FROM [covid19_impac].[dbo].[staging_covid]
where [popData2019] is null;
```

Query 4 : Countries with null values

```
UPDATE [covid19_impact].[dbo].[staging_covid]
SET [popData2019] = 12.067
WHERE countriesAndTerritories = 'Wallis_and_Futuna'


UPDATE [covid19_impact].[dbo].[staging_covid]
SET [popData2019] = 0
WHERE countriesAndTerritories = 'Cases_on_an_international_conveyance_Japan'

  UPDATE [covid19_impact].[dbo].[staging_covid]
  SET countryterritoryCode = geoId
  where countryterritoryCode  is null
```

Query 5 : Update countries with null values

By doing the same for the staging_flights we can see that there are not any null values. Query 6 Shows the query.

```
SELECT *
  FROM [covid19_impact].[dbo].[staging_flights]
  where
   [Date] is null or
  [AirportName] is null or
  [PercentOfBaseline] is null or
  [Latitude] is null or
  [Longitude] is null or
  [City] is null or
  [State] is null or
  [State] is null or
  [ISO_3166_2] is null or
  [Country] is null
```

Query 6 : Null search in staging_flights

4. Different countries names: besides date, the other value that connects the two tables is country, which is represented by countriesAndTerritories column in staging_covid and Country column in stagin_flights. So, we need to make sure that the names of the countries match. Performing Query 7 we get Result 2.

```
select distinct Country from [covid19_impac].[dbo].[staging_flights]
where Country not in (select countriesAndTerritories from [covid19_impac].[dbo].[staging_covid])
```

Query 7 : Search for missing countries

	Country
1	United States of America (the)

Result 2 : Missing countries results

Since it seems impossible that United States of America is not in the covid dataset we perform Query 8, and we get Result 3. That means that the USA exist in both tables but with different names, as “United States of America (the)” in staging_flights and as “United_States_of_America” in staging_covid. So, we execute Query 9 in order to have the same name in both tables.

```
select distinct countriesAndTerritories from staging_covid
where countriesAndTerritories like '%merica%'
```

Query 8 : Find the name of USA in staging_covid

	countriesAndTerritories
1	United_States_of_America

Result 3 : USA name in staging_covid

```
UPDATE staging_flights
set Country = 'United_States_of_America'
where Country = 'United States of America (the)'
```

Query 9 : Update USA in staging_flights

After this, the final design of our staging tables is as showing in Figure 17 and Figure 18.

dateRep	date	☑
day	float	☑
month	float	☑
year	float	☑
cases	float	☑
deaths	float	☑
countriesAndTerritories	nvarchar(255)	☑
geoId	nvarchar(255)	☑
countryterritoryCode	nvarchar(255)	☑
popData2019	float	☑
continentExp	nvarchar(255)	☑

Figure 17 : Final design of staging_covid

Column Name	Data Type	Allow Nulls
Date	date	☑
AirportName	nvarchar(255)	☑
PercentOfBaseline	float	☑
Latitude	float	☑
Longitude	float	☑
City	nvarchar(255)	☑
State	nvarchar(255)	☑
ISO_3166_2	nvarchar(255)	☑
Country	nvarchar(255)	☑

Figure 18 : Final design of staging_flights

Figure 19 shows how the database looks in staging area.

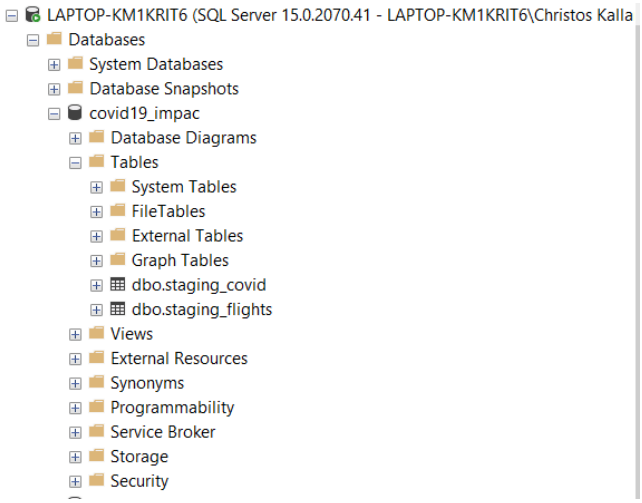


Figure 19 : Database in staging area

Now that our data have been cleaned and transformed in a usable way, we can design the data warehouse and load them into it.

4.4 Create Data Warehouse

A Data Warehouse consists of facts and dimension tables. So, we must define those tables based on our datasets. Columns that have measure in our datasets are column *cases* and *deaths* from staging_covid and *PercentOfBaseline* from staging_flights. That means we can create **two fact tables**, one with information about flights with *PercentOfBaseline* column as its measurement and the other with informations about Covid-19 cases and deaths with *cases* and *deaths* columns as measurements. The rest of the columns will have information needed for the analysis. The fact table for the flights will be called **f_flights** and the one for the Covid-19 cases will be called **f_covid19.**

- **f_flights**: the information we need for this fact table are in staging_flights table and they consist of date, name of the airport, latitude and longitude of the airport, city within the airport is located, state of that city, ISO code for that airport and country. So, the fact table will consist of the following columns

    - *date*: date of the flight
    - *city*: city within the airport is located
    - *airport*: airport from which the flight occurred
    - *coordinate*: latitude and longitude of airport
    - *country*: country within the city is located
    - *percentofbaseline*: proportion of trips on this date as compared to average number of trips on the same day of week in baseline period i.e., 1st

    The percentofbaseline column is the measurement. The rest of the columns will be type int and will be a code showing to the real information using another table as reference. That table will be the dimension table and it will relate to the fact table using the respective column as key. So, based on the above **we need 5 dimension tables for date, city, airport, coordinate and country**.

- **f_covid19**: the information we need for this fact table are in staging_flights table and they consist of date,country,casesand deaths.We can also create a new measurement that shows the deats per cases. So, the fact table will consist of the following columns

    - *date*: date of the cases and deaths logging
    - *country*: country which the cases and deaths were reported
    - *cases*: number of cases
    - *deaths*: number of deaths
    - *deathspercase*: number of deaths per cases

    The *cases,deaths* and *deathspercase* columns are the measurement. The rest of the columns will be type int and will be a code showing to the real information using another table as reference like the f_flights. So, based on the above **we need 2 dimension tables for date, and country**.

Having found that, the fact tables will be we can design the dimensions tables. We already mentioned that **f_flights needs 5 dimension tables** and **f_covid19 needs 2 dimension tables**. The dimension tables that f_covid19 needs are date and country and they are also needed from the f_flights. So those 2 can be common dimensions. That means **that for the two fact tables the dimension will be 5** and will be called **d_airports**, **d_cities**, **d_coordinates**, **d_countries** and **d_date**. Since we also have the information of continent, we will also create a 6th dimension called **d_continent** that it will only relate to the countries table.

- **d_airports:** the table will have information about the airports, specifically its name. It will relate to the fact table **f_flights** with the column *airport_id* (column *airport* will be the foreign key in f_flights). It will consist of
    - *airport_id*: id for the airport (primary key)
    - *airport_name*: name of the airport
- **d_cities**: the table will have information regarding cities, specifically city and state. It will relate to the fact table **f_flights** using *city_id* (column *city* will be the foreign key in f_flights). It will consist of
    - *city_id*: id for the city (primary key)
    - *city*: name of the city
    - *state*: state within the city is located
- **d_continent**: the table will have information regarding the continents, specifically the name of the continents. It will relate to **d_countries** using *continent_id* (column *continent_id* will be the foreign key in d_countries). It will consist of
    - continent_id: id for the continent (primary key)
    - continent: name of the continent
- **d_coordinates:** the table will have information regarding the coordinates of the airport from which the flight occurred, specifically its longitude and latitude. It will relate to the fact table **f_flights** with the column *coordinate_id* (column coordinate will be the foreign key in f_flights). It will consist of
    - *loc_id:* id for the coordinates (primary key)
    - *latitude:* latitude of the airport
    - *longitude:* longitude of the airport
- **d_countries:** the table will have information regarding the countries which the flight, cases of Covid-19 and deaths from Covid-19 were recorded. It will relate to the fact tables **f_flights** and **f_covid19** using column *country_id* (column *country* will be foreign key in both fact tables). It will consist of
    - *country_id:* id for the country (primary key)
    - *country:* name of the country

- *char_id:* geographical character id for the country
- *population:* population of the country
- *continent_id:* id of the continent within the country exists (foreign key)

➢ **d_date:** the table will have information about the date which the flight,cases of Covid-19 and deaths from Covid-19 were recorded. It will relate to the fact tables **f_flights** and **f_covid19** using column *date_id* (column *date* will be foreign key in both fact tables). It will consist of

- *date_id:* id for the date (primary key)
- *date:* date of the logging (format YYYY-MM-DD)
- *day:* day of date
- *month:* month of date
- *year:* year of date

So, with all that in mind a possible schema for the data warehouse would be in the form of Figure 20 (this is not an official star schema, just something we draw in order to better understand the database. Star schema will be presented later)

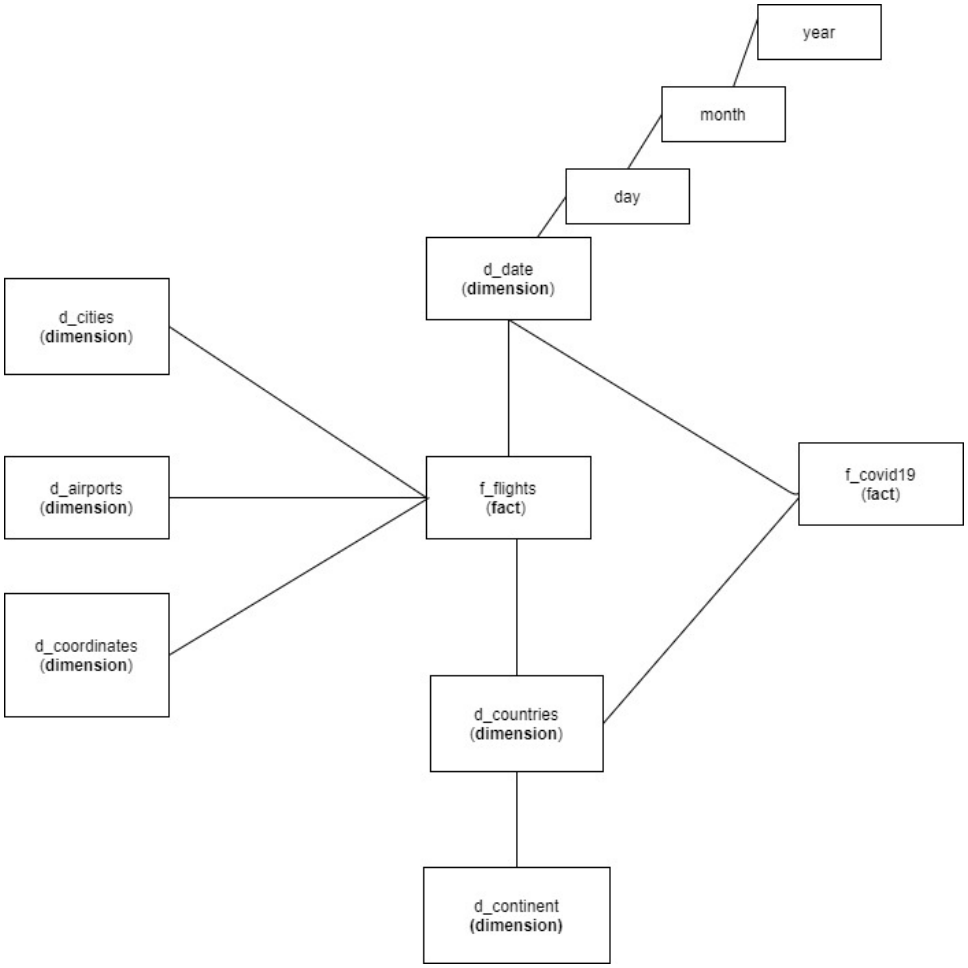


Figure 20 : Diagram of the DWH

After this we can create the tables in the database with the columns we analyzed. Table 1 shows the design for dimension tables and Table 2 for the fact tables.

d_airports

Column Name	Data Type
airport_id	int
airportname	nchar(64)

d_continent

Column Name	Data Type	Allow
continent_id	int	
continent	nchar(64)	

d_cities

Column Name	Data Type	Allow Nulls
city_id	int	
city	nchar(64)	
state	nchar(64)	

d_countries

Column Name	Data Type	Allow Nulls
country_id	int	
country	nchar(255)	
char_id	varchar(64)	
population	float	
continent_id	int	

d_coordinates

Column Name	Data Type	Allow Nulls
loc_id	int	
latitude	float	
longitude	float	

d_date

Column Name	Data Type	Allow Null
date_id	int	
date	date	
day	int	
month	int	
year	int	

Table 1 : Dimension tables design

f_flights

Column Name	Data Type	Allow Nulls
date	int	
city	int	
airport	int	
cooordinate	int	
country	int	
percentofbaseline	float	

f_covid19

Column Name	Data Type	Allow Nulls
date	int	
country	int	
cases	float	
deaths	float	
deathspercase	float	

Table 2 : Fact tables design

The tables have been created and the database is ready (Figure 21). So, we can proceed to the next step which is load the data.

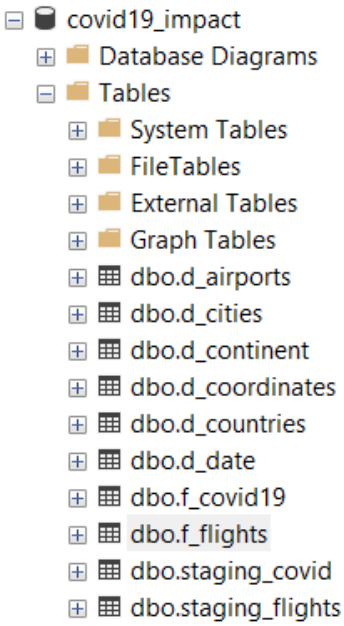


Figure 21 : Database after the dimension and fact tables creation

4.5 Load Data

We need to populate every fact and dimension table. We will do that using SQL commands. First the dimension tables will be filled using the staging tables.

a. For the **d_airports** table we need to insert all the airports from staging_flights and assign an id to each one of them. So, we will execute Figure 22. As showing in Result 4 , the table has successfully been filled.

```
INSERT INTO d_airports (airportname) select distinct AirportName from [covid19_impact].[dbo].[staging_flights];

    DECLARE @id INT
    SET @id = 0
    UPDATE [covid19_impact].[dbo].[d_airports]
    SET @id = airport_id = @id + 1
    GO
```

Figure 22 : Load d_airports

	airport_id	airportname
1	1	John F. Kennedy International
2	2	Detroit Metropolitan Wayne County
3	3	Newark Liberty International
4	4	Kingsford Smith
5	5	San Francisco International
6	6	Santiago International Airport
7	7	Chicago OHare International
8	8	LaGuardia
9	9	Hartsfield-Jackson Atlanta International
10	10	Dallas/Fort Worth International
11	11	Daniel K. Inouye International
12	12	Edmonton International
13	13	Hamilton International
14	14	Los Angeles International
15	15	Miami International
16	16	Montreal Mirabel
17	17	Winnipeg International

Result 4 : Populated d_airports

b. For the **d_cities** table we need to insert all the cities from staging_flights and assign an id to each one of them. So, we will execute Figure 23. As showing in Result 5Result 4 , the table has successfully been filled.

```
INSERT INTO d_cities (city,state) select distinct City,State from [covid19_impact].[dbo].[staging_flights];

DECLARE @id INT
    SET @id = 0
    UPDATE [covid19_impact].[dbo].[d_cities]
    SET @id = city_id = @id + 1
    GO
```

Figure 23 : Load d_cities

	city_id	city	state
1	1	Boston	Massachusetts
2	2	Calgary	Alberta
3	3	Charlotte	North Carolina
4	4	Chicago	Illinois
5	5	College Park	Georgia
6	6	Denver	Colorado
7	7	Dorval	Quebec
8	8	Floris	Virginia
9	9	Grapevine	Texas
10	10	Halifax	Nova Scotia
11	11	Hamilton	Ontario
12	12	Leduc County	Alberta
13	13	Los Angeles	California
14	14	Miami Springs	Florida
15	15	Mirabel	Quebec
16	16	Mississauga	Ontario
17	17	New York	New York

Result 5 : Populated d_cities

c. For the **d_coordinates** table we need to insert all the latitude and longitude from staging_flights and assign an id to each one of them. So, we will execute Figure 24. As showing in Result 6 the table has successfully been filled.

```
INSERT INTO d_coordinates (latitude,longitude) select distinct latitude,longitude from [covid19_impact].[dbo].[staging_flights];
UPDATE d_coordinates SET longitude = x.longitude FROM (select distinct longitude from [covid19_impact].[dbo].[staging_flights]) AS x;
    DECLARE @id INT
    SET @id = 0
    UPDATE [covid19_impact].[dbo].[d_coordinates]
    SET @id = loc_id = @id + 1
    GO
```

Figure 24 : Load d_coordinates

	loc_id	latitude	longitude
1	1	-833537314720423	422129725988552
2	2	-802887147460152	257957243107604
3	3	-970394983968728	328940590356408
4	4	-741751246689879	406915033838306
5	5	-799266930701955	431720360844936
6	6	-74048379779338	456815027038316
7	7	-972190621861971	499024712565967
8	8	-118404993180627	33941369379328
9	9	-635116720134583	448830168353507
10	10	-12317754124324	491935788600694
11	11	-737424121584688	454678436940192
12	12	-738732455278797	407738834966785
13	13	-12238393507603	376211875471696
14	14	-879105952039514	419804600429329
15	15	-113576260685047	53308783097969
16	16	-844279188822754	336410758198944

Result 6 : Populated d_coordinates

d. For the **d_date** table we need to insert all the dates from staging_covid (staging covid has the most dates and every date of staging_flights exist on staging_covid) and assign an id to each one of them. So, we will execute Figure 25. As showing in Result 7 the table has successfully been filled.

```
INSERT INTO d_date (date) select distinct dateRep from [covid19_impact].[dbo].[staging_covid];
UPDATE d_date SET day = DAY(date)
UPDATE d_date SET month = MONTH(date)
UPDATE d_date SET year = YEAR(date)

DECLARE @id INT
        SET @id = 0
        UPDATE [covid19_impact].[dbo].[d_date]
        SET @id = date_id = @id + 1
        GO
```

Figure 25 : Load d_date

	date_id	date	day	month	year
1	1	2020-07-29	29	7	2020
2	2	2020-07-06	6	7	2020
3	3	2020-09-13	13	9	2020
4	4	2020-08-21	21	8	2020
5	5	2020-10-29	29	10	2020
6	6	2020-10-06	6	10	2020
7	7	2020-11-21	21	11	2020
8	8	2020-08-04	4	8	2020
9	9	2020-06-05	5	6	2020
10	10	2020-06-19	19	6	2020
11	11	2020-04-20	20	4	2020
12	12	2020-01-25	25	1	2020
13	13	2020-03-11	11	3	2020
14	14	2020-03-05	5	3	2020
15	15	2020-09-19	19	9	2020
16	16	2020-04-26	26	4	2020

Result 7 : Populated d_date

e. For the **d_continent** table we need to insert all the continents from staging_covid and assign an id to each one of them. So, we will execute Figure 26. As showing in Result 8 the table has successfully been filled.

```
INSERT INTO d_continent (continent) select distinct continentExp from [covid19_impact].[dbo].[staging_covid];


DECLARE @id INT
    SET @id = 0
    UPDATE [covid19_impact].[dbo].[d_continent]
    SET @id = continent_id = @id + 1
    GO
```

Figure 26 : Load d_continent

	continent_id	continent
1	1	America
2	2	Asia
3	3	Africa
4	4	Oceania
5	5	Other
6	6	Europe

Result 8 : Populated d_continent

f. For the **d_countries** table we need to insert all the countries from staging_covid and assign an id to each one of them. Also we must join with the d_continent in order to get the continent_id. So, we will execute Figure 27. As showing in Result 9Result 8 the table has successfully been filled

```
INSERT INTO d_countries (country,char_id,population,continent_id)
select distinct countriesAndTerritories,countryterritoryCode,popData2019,b.continent_id from [covid19_impact].[dbo].[staging_covid] a,d_continent b
where a.continentExp = b.continent group by countryterritoryCode,countriesAndTerritories,popData2019,b.continent_id
order by countryterritoryCode,countriesAndTerritories,popData2019,b.continent_id ;

DECLARE @id INT
    SET @id = 0
    UPDATE [covid19_impact].[dbo].[d_countries]
    SET @id = country_id = @id + 1
    GO
```

Figure 27 : Load d_countries

	country_id	country	char_id	population	continent_id
1	1	Antigua_and_Barbuda	ATG	97115	1
2	2	Cayman_Islands	CYM	64948	1
3	3	Belize	BLZ	390351	1
4	4	Argentina	ARG	44780675	1
5	5	Anguilla	AIA	14872	1
6	6	Aruba	ABW	106310	1
7	7	Bolivia	BOL	11513102	1
8	8	Canada	CAN	37411038	1
9	9	Bonaire, Saint Eustatius and Saba	BES	25983	1
10	10	Dominican_Republic	DOM	10738957	1
11	11	Haiti	HTI	11263079	1
12	12	Bahamas	BHS	389486	1
13	13	Colombia	COL	50339443	1
14	14	Ecuador	ECU	17373657	1
15	15	Barbados	BRB	287021	1
16	16	Costa_Rica	CRI	5047561	1

Result 9 : Populated d_countries

g. For the **f_flights** table we need to insert all the information regarding airports,cities,countries,dates and percent of baseline. In order to do this, we will perform a join with the dimension tables and staging_flights. So, we will execute Figure 28. As showing in Result 10Result 8 the table has successfully been filled.

```
insert into f_flights (city,airport,cooordinate,country,date,percentofbaseline)
select a.city_id,b.airport_id,c.loc_id,d.country_id,f.date_id,e.PercentOfBaseline
FROM d_cities a,d_airports b,d_coordinates c,d_countries d ,d_date f, staging_flights e
where a.city = e.City
and b.airportname = e.AirportName
and c.latitude = e.Latitude
and c.longitude = e.longitude
and d.country = e.Country
and f.date = e.date
```

Figure 28 : Load f_flights

Results | Messages

	date	city	airport	cooordinate	country	percentofbaseline
1	10	12	12	15	8	96
2	10	2	21	26	8	88
3	10	22	6	22	28	28
4	10	23	28	24	46	68
5	10	8	24	19	46	56
6	11	7	23	11	8	68
7	11	15	16	6	8	68
8	11	11	13	5	8	100
9	11	16	22	28	8	59
10	11	10	26	9	8	41
11	11	28	17	7	8	100
12	11	6	19	23	46	31
13	11	24	5	13	46	46
14	11	13	14	8	46	59
15	11	25	4	18	148	60
16	11	21	2	1	46	47
17	11	1	18	21	46	48

Result 10 : Populated f_flights

h. For the **f_covid19** table we need to insert all the information regarding date,countries,cases,deaths and deaths per case. In order to do this, we will perform a join with the dimension tables and staging_covid19 with special care for the division deaths/cases. If cases are 0 then we fill with null and then update null values with 0. So, we will execute Figure 29. As showing in Result 11Result 8 the table has successfully been filled.

```
insert into f_covid19(country,date,cases,deaths,deathspercase)
   select a.country_id,b.date_id,c.cases,c.deaths,(c.deaths/NULLIF (cases,0))*100
   FROM d_countries a ,d_date b, staging_covid c
   where a.country = c.countriesAndTerritories
   and b.date = c.dateRep

   --FIX NULL VALUES
   UPDATE f_covid19
   set deathspercase = 0
   where deathspercase IS NULL
```

Figure 29 : Load f_covid19

	date	country	cases	deaths	deathspercase
1	1	76	48	0	0
2	1	197	54	1	1,85185185185185
3	1	8	383	11	2,87206266318538
4	1	186	6	0	0
5	1	203	56	2	3,57142857142857
6	1	180	0	0	0
7	1	122	7232	190	2,62721238938053
8	1	45	5288	194	3,66868381240545
9	1	99	29	0	0
10	1	94	653	11	1,68453292496172
11	1	190	8	0	0
12	1	135	0	0	0
13	2	135	129	2	1,55038759689922
14	2	190	5	0	0
15	2	94	0	0	0
16	2	99	32	0	0
17	2	45	3638	177	4,8653106102254

Result 11 : Populated f_covid19

With all the tables being filled, that concludes the creation of the Data Warehouse. That means we are ready to create the cube on top of it.

5. Data Cube

We are going to use SQL Server Analysis Services with Visual Studio to create the cube. So, we created an analysis services project in Visual Studio and then imported the database. We can see the solution that was created in Figure 30. The measurements are the ones from the fact tables plus a measurement for count for each fact table that we added in Visual Studio (Figure 31). After that, the cube was created and can be seen as a star schema in Figure 32 .

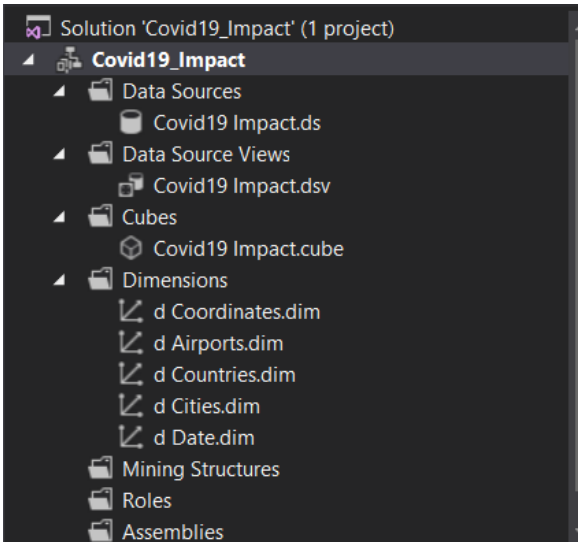


Figure 30 : Covid19_Impact solution

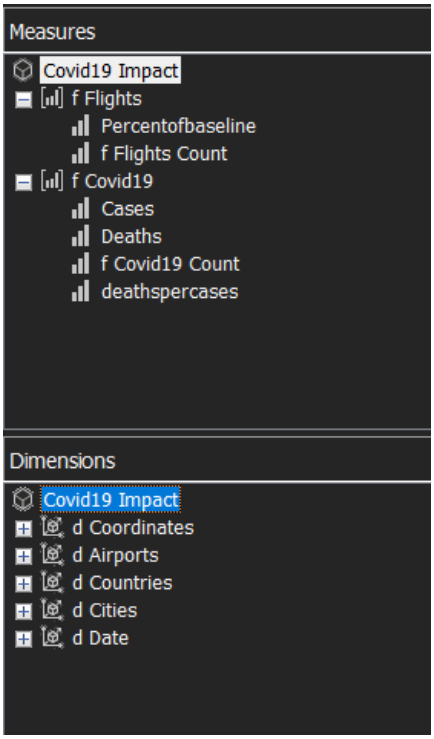


Figure 31 : Cube measures and dimensions

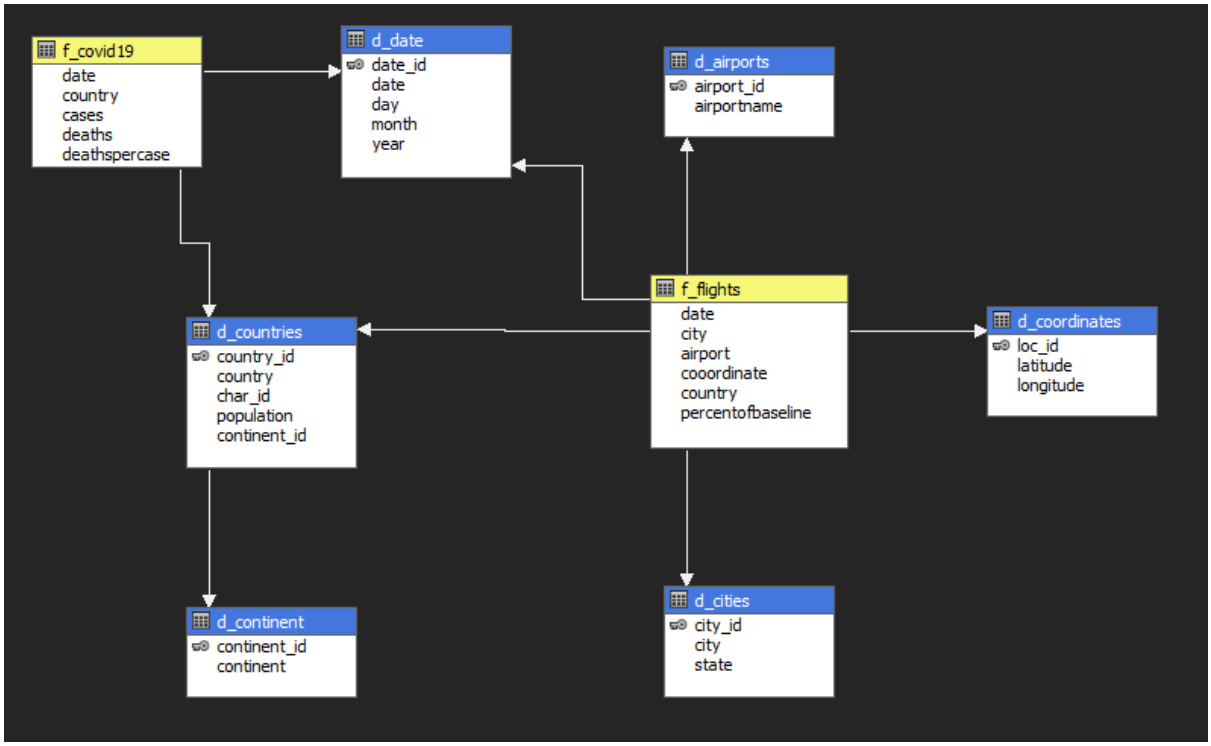


Figure 32 : Cube/Star Schema

Since date consists of day, month and year we created a hierarchy and change the default order by name to order by key in order to be able to sort it based on the value (Figure 33).

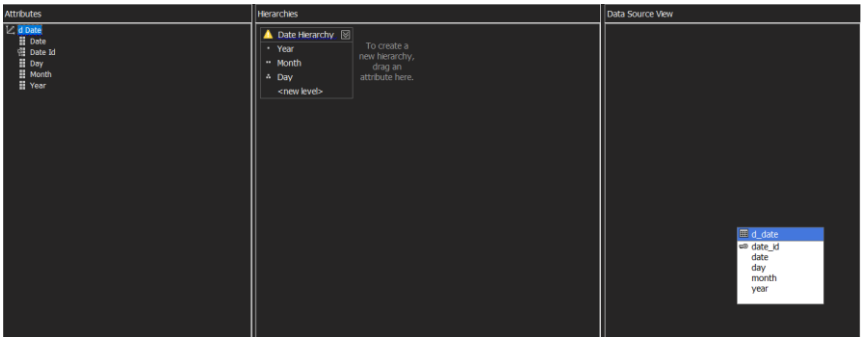


Figure 33 : Date Hierarchy

In order to be able to use every column of the tables, we defined as attributes every column like in Figure 33 , where we also put date, day, month and year as attributes. We did this for every dimension so that we can use all the information for the analysis.

Browsing the cube, we can create small queries like showing the total number of Covid-19 cases in each country (Figure 34) or the number of flights for each city (Figure 35).

Country	Cases
Afghanistan	45616
Albania	36245
Algeria	79110
Andorra	6610
Angola	14821
Anguilla	4
Antigua_and_Barbuda	141
Argentina	1407264
Armenia	133594
Aruba	4808
Australia	27874
Austria	269510
Azerbaijan	109813
Bahamas	7496

Figure 34 : Number of cases per country

City	f Flights Count
Boston	214
Calgary	214
Charlotte	214
Chicago	214
College Park	215
Denver	214
Dorval	215
Floris	214
Grapevine	215
Halifax	205
Hamilton	213
Leduc County	189
Los Angeles	215

Figure 35 : Number of flights per city

In order to be able to use the cube in Power BI we need to deploy it. The deployment process can be seen in Figure 36 and Figure 37.

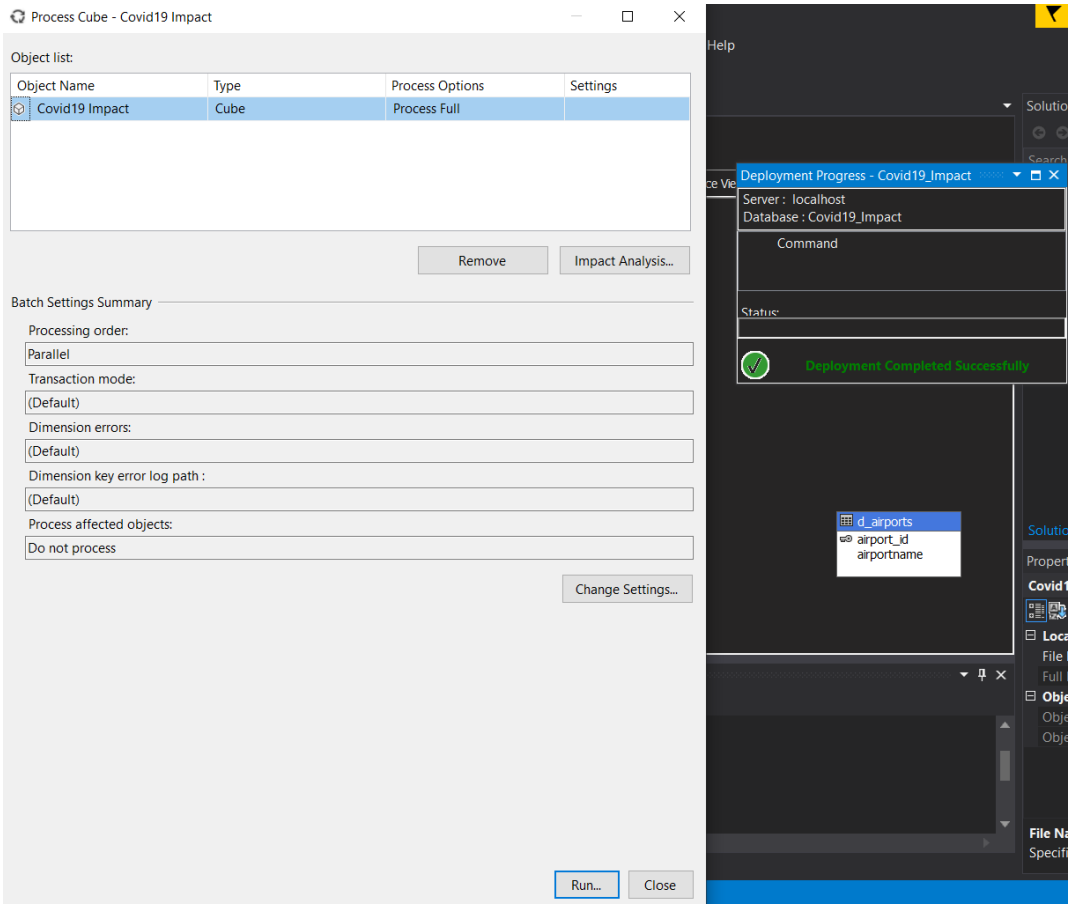


Figure 36 : Deploy the cube

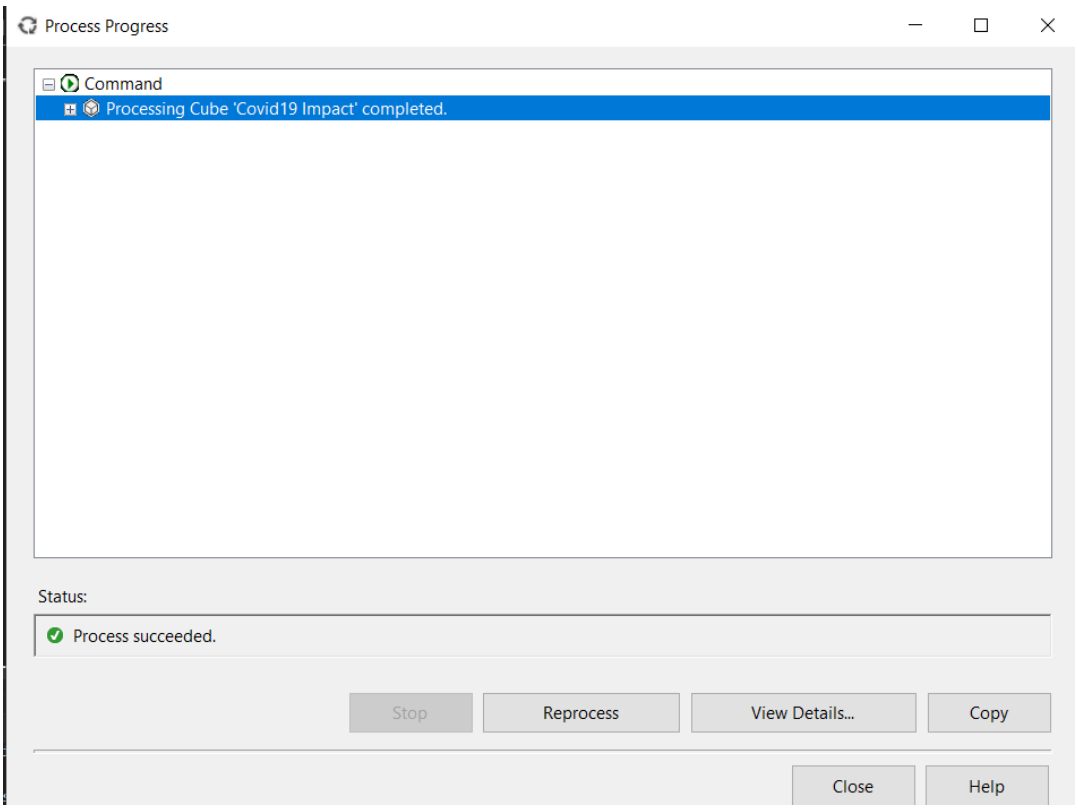


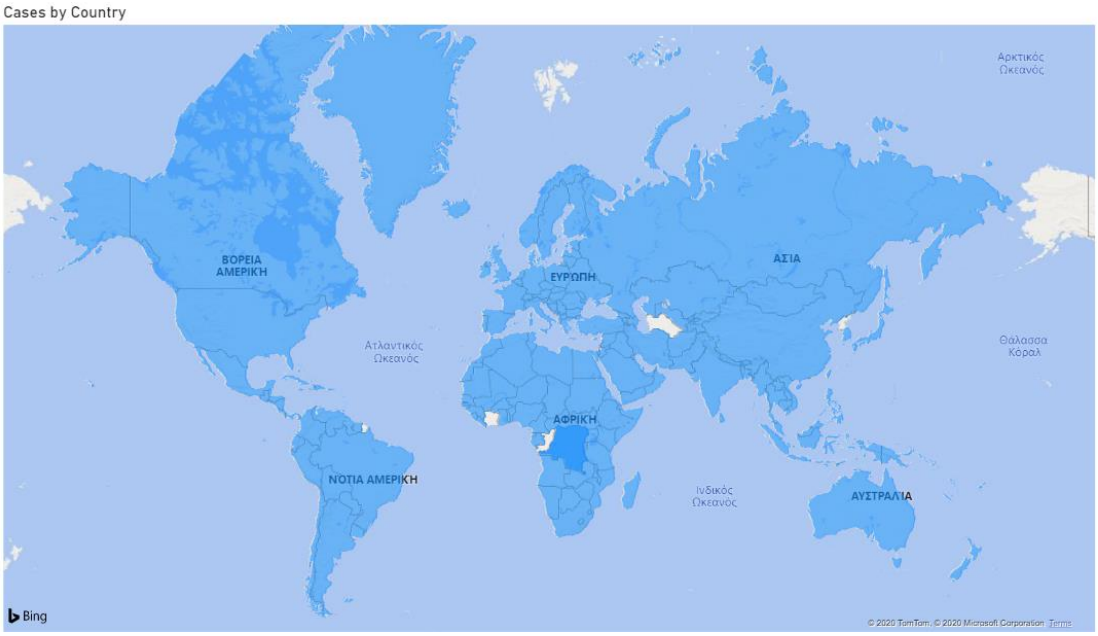
Figure 37 : Successful deployment

6. Visualized Reports

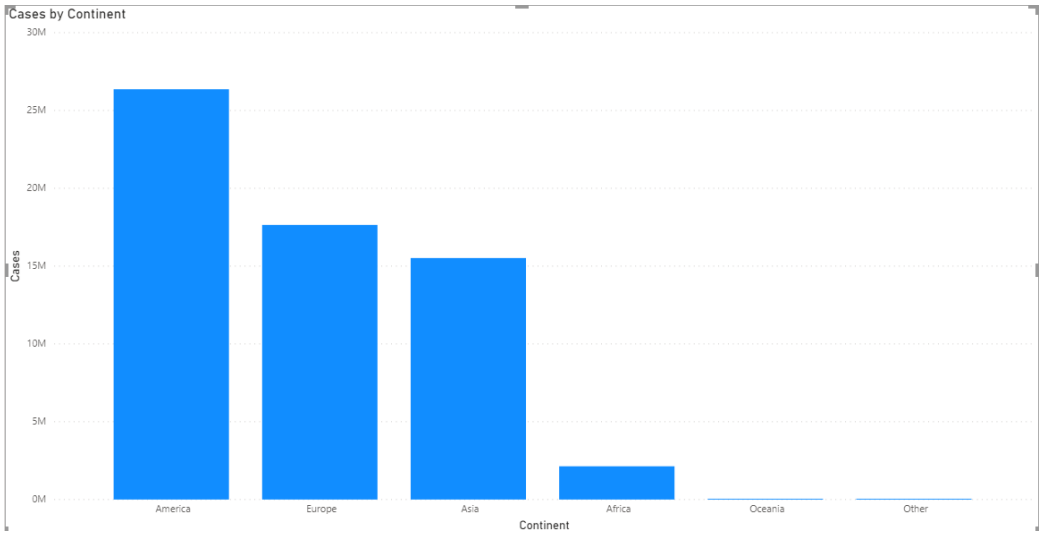
6.1 Using the deployed Cube

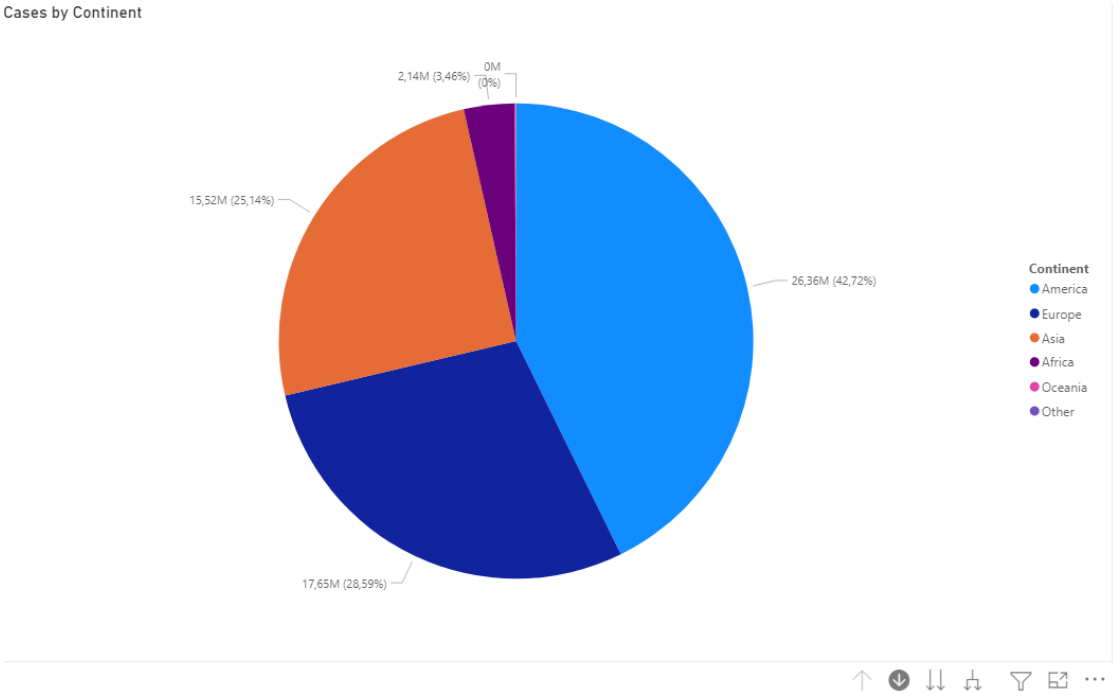
With the Cube deployed we can open it using Power BI creating a connection with the SQL Analysis Server.

Firstly, we will show the cases and deaths of Covid-19 around the world. The following depicts the cases around the world in a map. We can see that almost every country in the world had cases of Covid-19

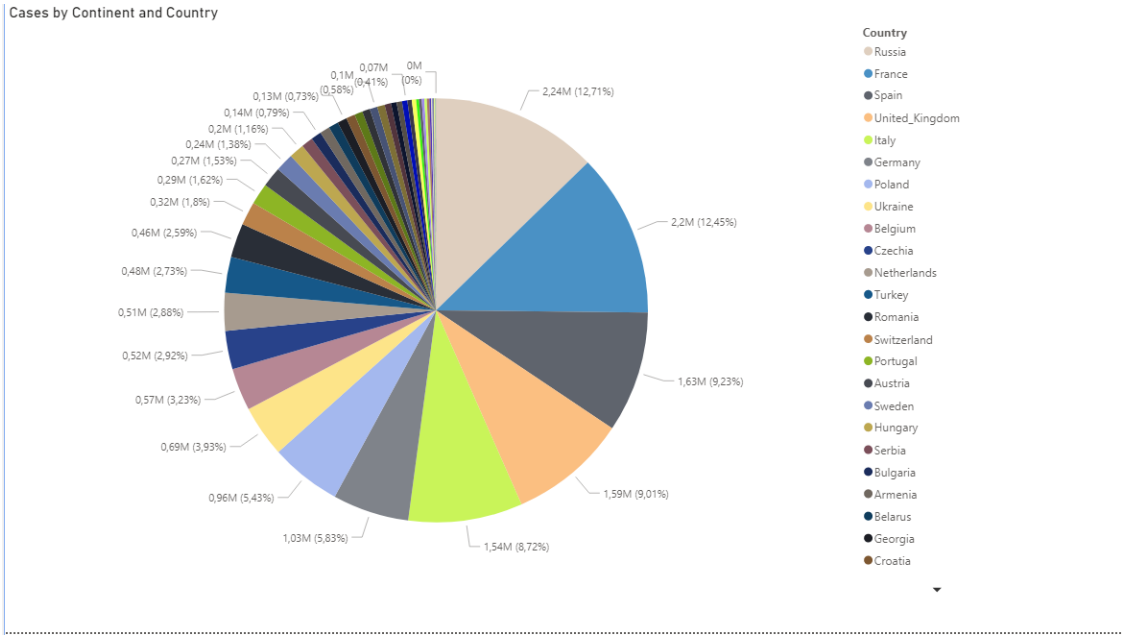


In a continent level we can see from the following that America has the most cases followed by Europe and then Asia with Oceania having the least.



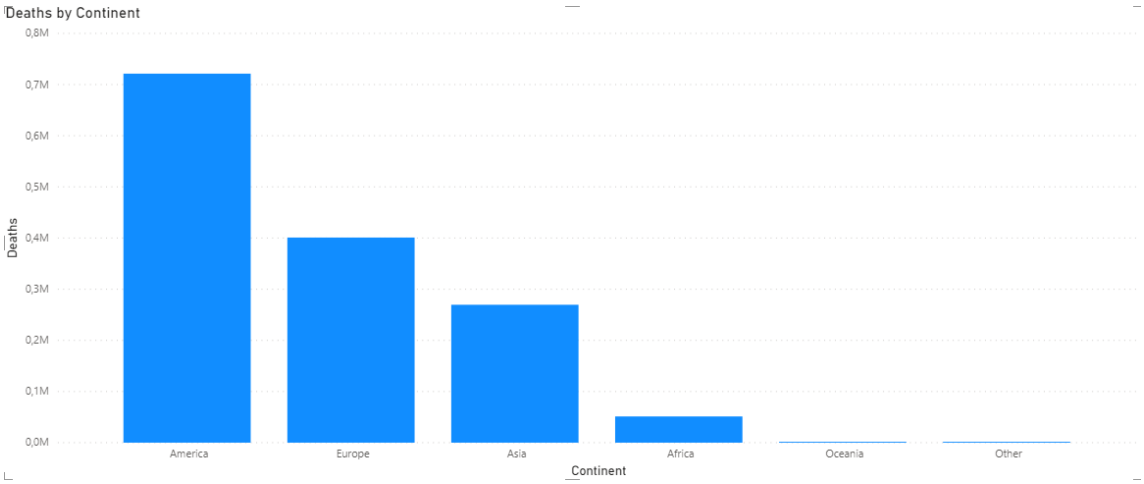


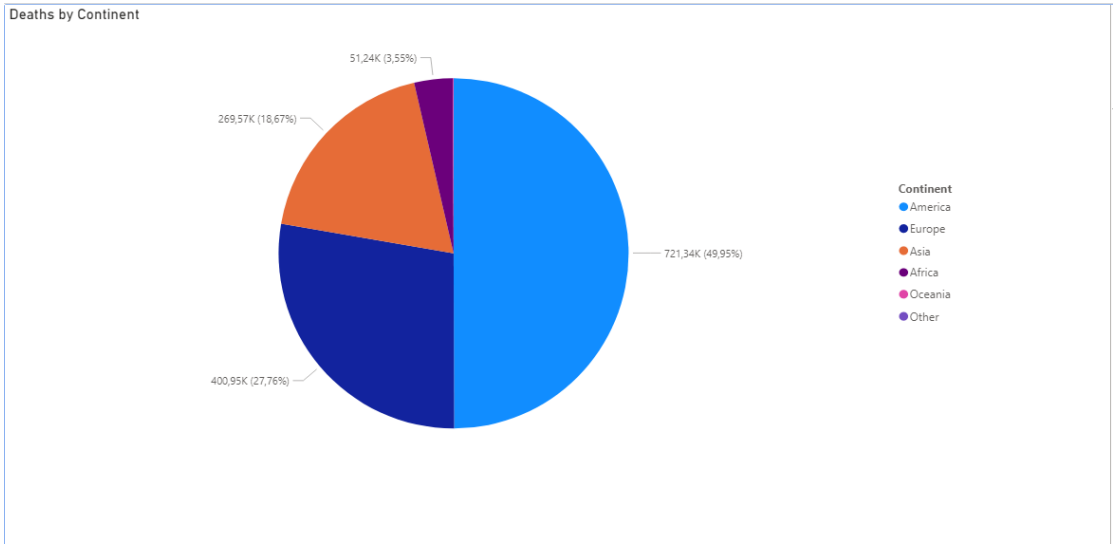
Activating the drill down function and by double clicking the continent of our choice we can see the cases for the countries in that continent. For example, Europe:



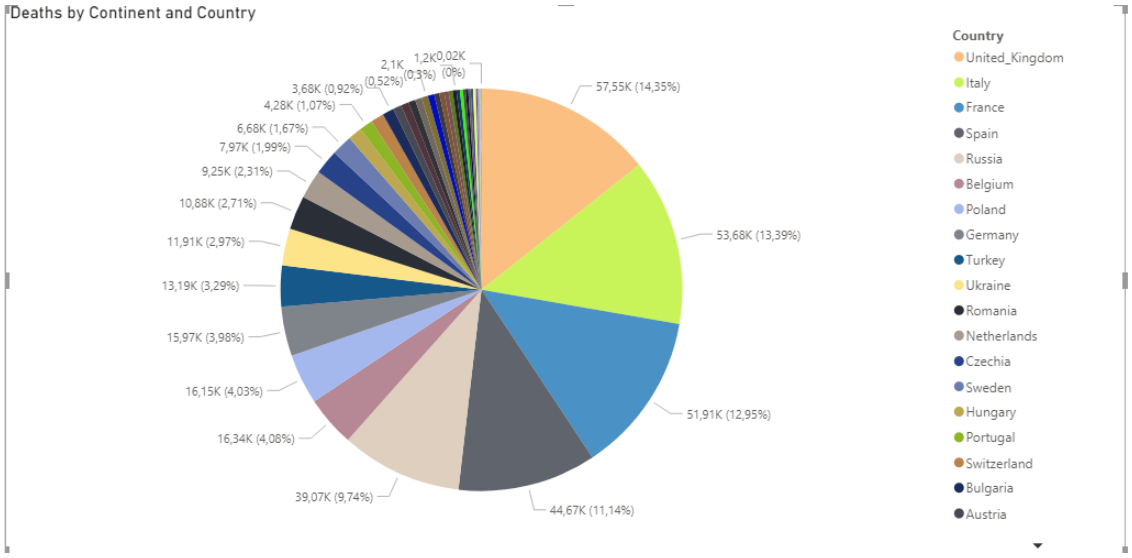
Russia appears to have most cases followed by France, Spain and the United Kingdom.

Doing the same for the deaths we can see that the order is the same with America reporting most deaths.



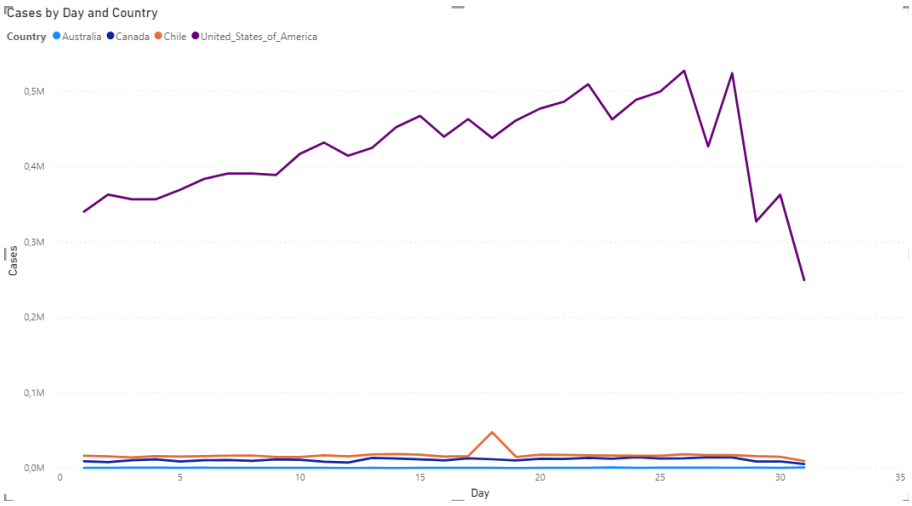


Activating the drill down function and by double clicking the continent of our choice we can see the deaths for the countries in that continent. For example, Europe:

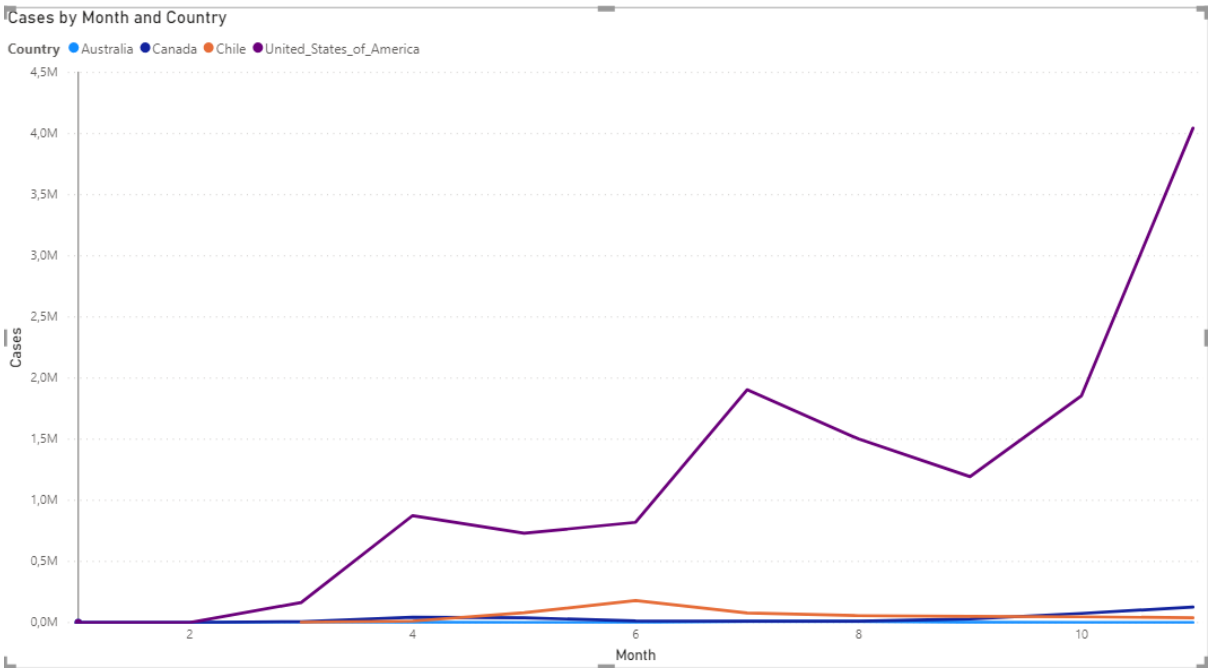


The order changes here with the United Kingdom having the most deaths followed by Italy, France and Spain.

Since the data for flights are about Australia, Chile, Canada and United States of America we will focus on these countries. The following graph depicts the number of cases per the above countries first by day and then by month. Since date has a hierarchy (day,month,year) we can use the “drill up” and the “go to next level of hierarchy” button to change the axis.

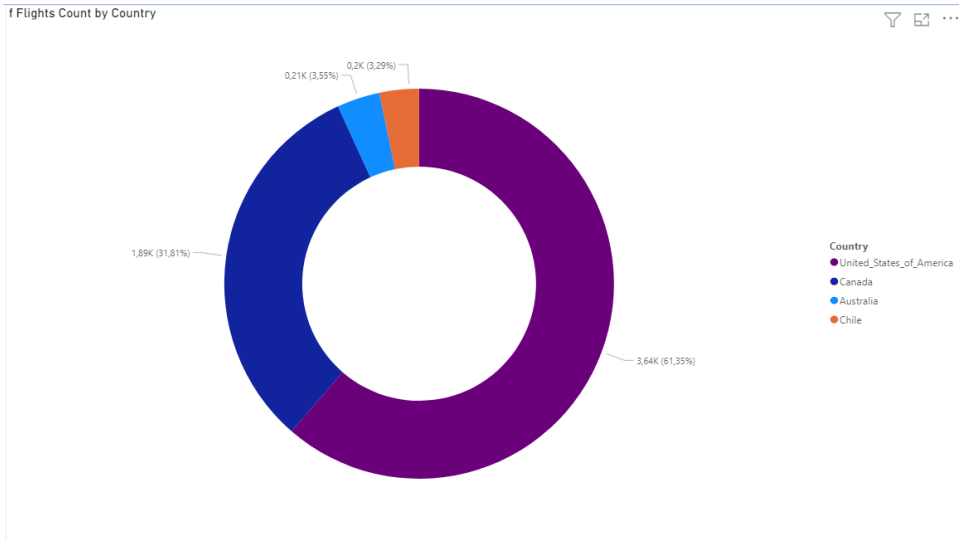


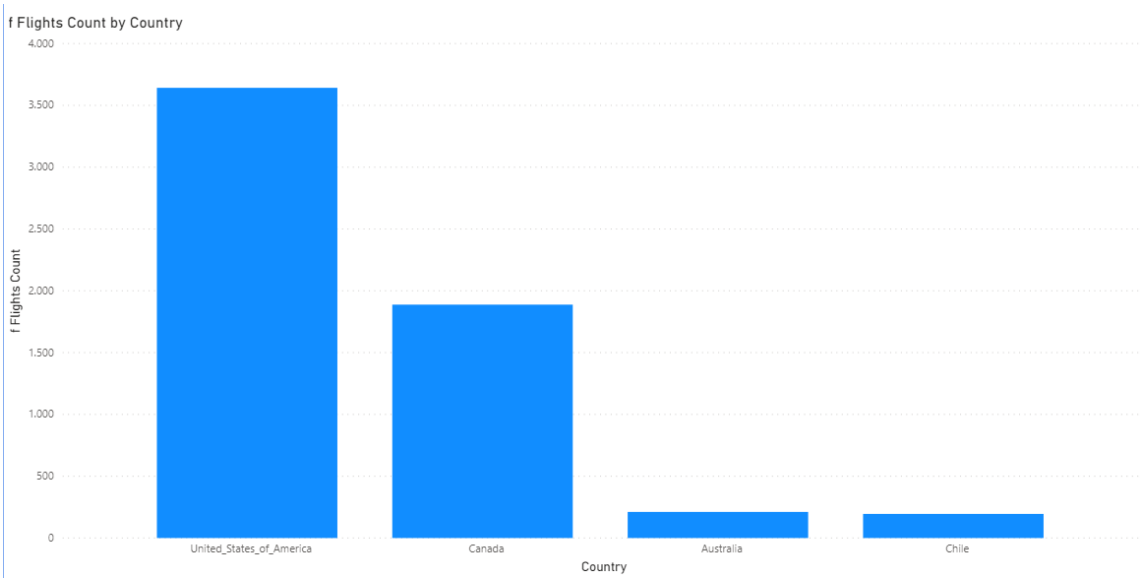
From the above graph we can see that USA has most of the cases on the 26 and 28 day. Bellow it’s the same graph with the month in the axis that give us a better look at how the cases evolved through the year 2020 for these countries.



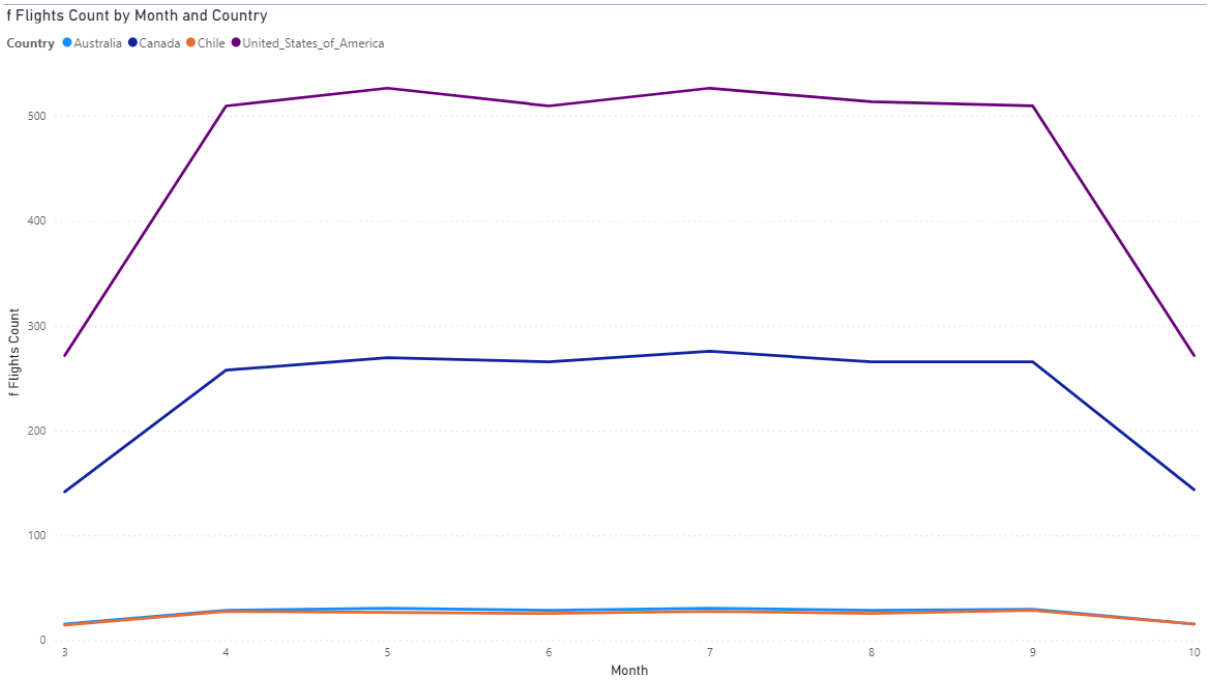
We can see that USA have the most cases with most cases being reported in November.

Moving on to flight data we can see from the following graphs that the USA has the most flights followed by Canada and then Australia and Chile.





We can also see the number of flights for these countries by date. Since date has a hierarchy we can “go to the next level” by drill down.

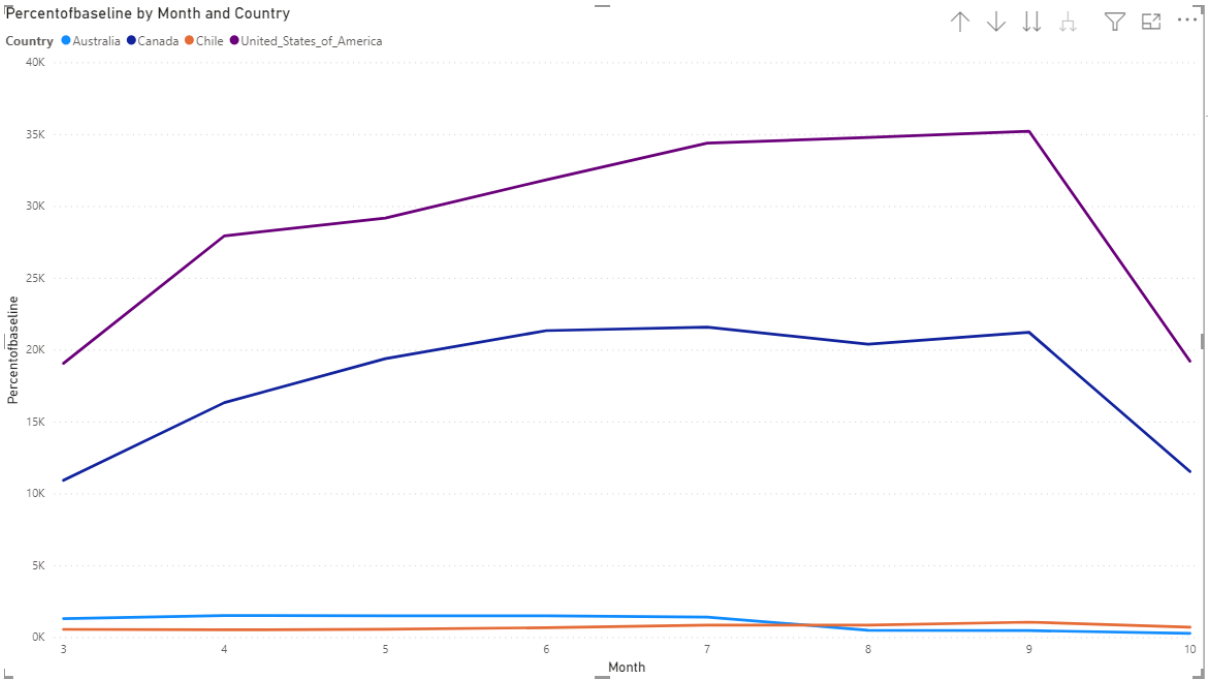


As it seems, the number of flights is stable over the course of the year following the same pattern for all countries, rising from March to April and then being steady for the rest of the year (the lack of data is the reason for that unexpected drop in October). The number of flights from these countries and their airports can been seen bellow.

Country	Airportname	f Flights Count
United_States_of_America	Dallas/Fort Worth International	215
United_States_of_America	Hartsfield-Jackson Atlanta International	215
United_States_of_America	John F. Kennedy International	215
United_States_of_America	Los Angeles International	215
Canada	Montreal Trudeau	215
United_States_of_America	Boston Logan International	214
Canada	Calgary International	214
United_States_of_America	Charlotte Douglas International	214
United_States_of_America	Chicago OHare International	214
United_States_of_America	Daniel K. Inouye International	214
United_States_of_America	Denver International	214
United_States_of_America	Detroit Metropolitan Wayne County	214
United_States_of_America	LaGuardia	214
United_States_of_America	McCarran International	214
United_States_of_America	Miami International	214
United_States_of_America	Newark Liberty International	214
United_States_of_America	San Francisco International	214
United_States_of_America	Seattle-Tacoma International	214
Canada	Toronto Pearson	214
Canada	Vancouver International	214
United_States_of_America	Washington Dulles International	214
Canada	Winnipeg International	214
Canada	Hamilton International	213
Australia	Kingsford Smith	211
Canada	Montreal Mirabel	210
Canada	Halifax International	205
Chile	Santiago International Airport	195
Canada	Edmonton International	189
Total		**5936**

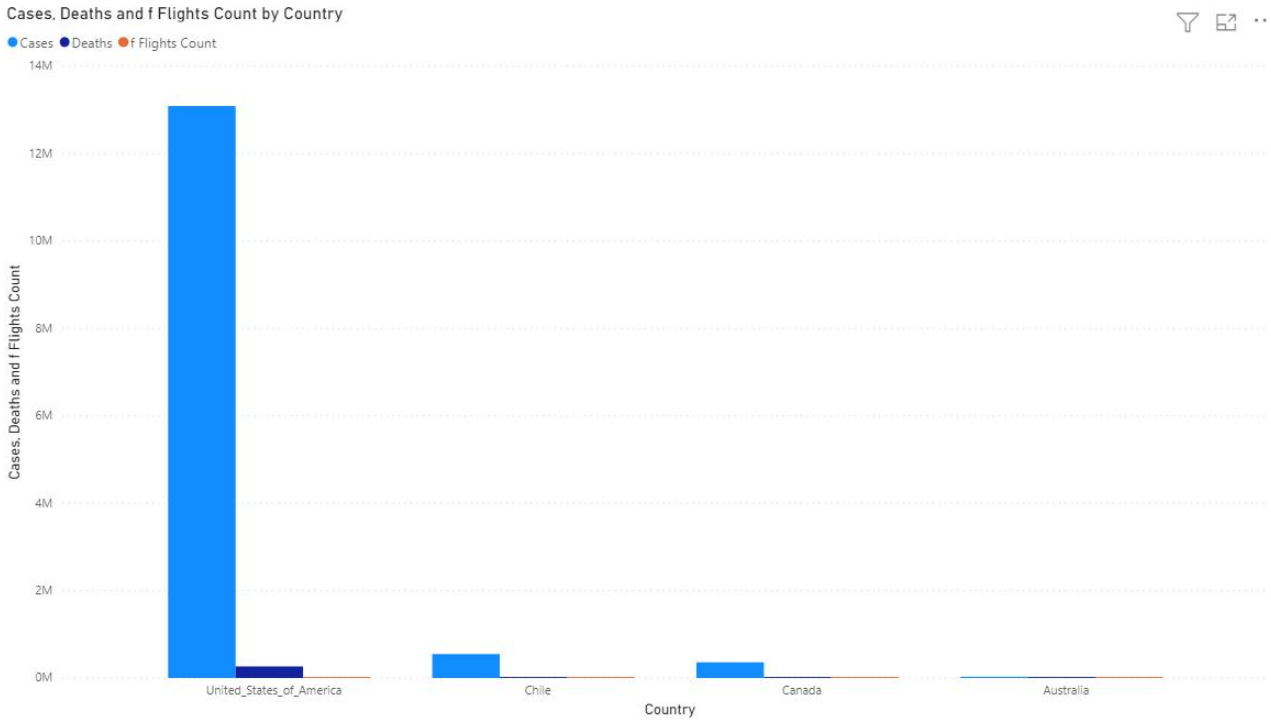
Country	Cases	Deaths	f Flights Count
United_States_of_America	13.091.758,00	264.859,00	3642
Chile	547.223,00	15.278,00	195
Canada	359.064,00	11.894,00	1888
Australia	27.874,00	907,00	211
Total	**14.025.919,00**	**292.938,00**	**5936**

In addition to the number of flights, the percent of baseline does not seem to be steady although USA and Canada follows a kind of same pattern and Chile is almost a straight line.

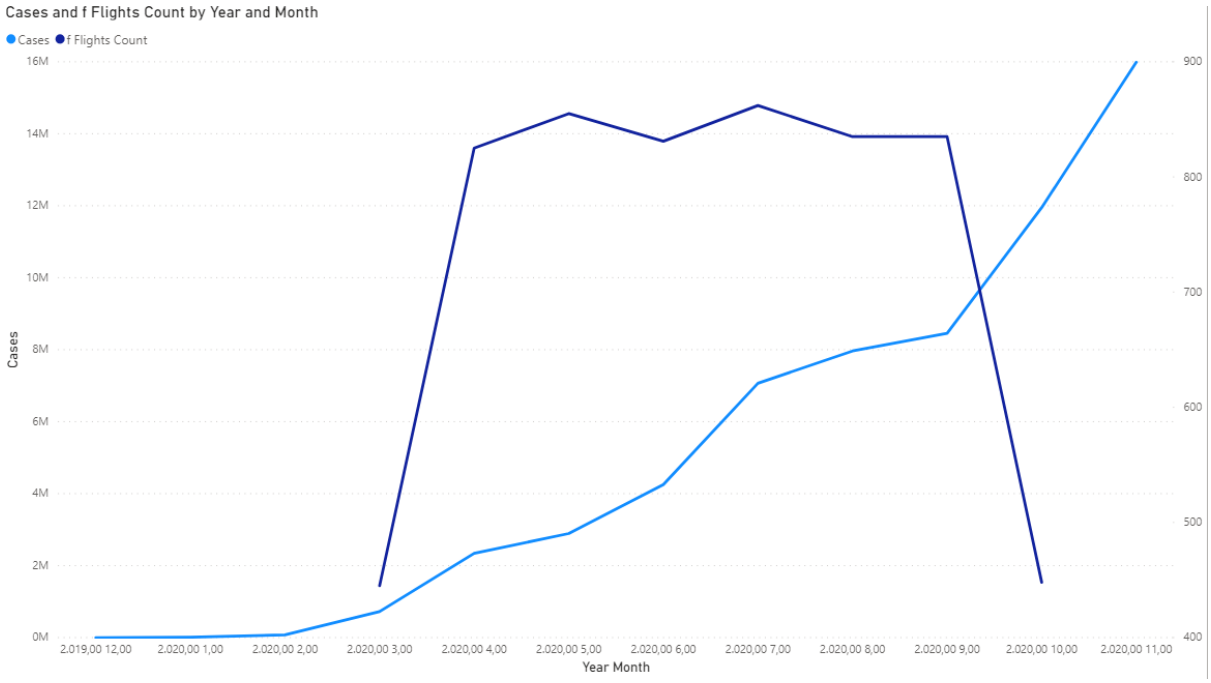


From this we can conclude that although from the previous graph the number of flights appeared to be the same across the year, the proportion of trips on this date as compared to average number of trips on the same day of week in baseline period i.e., 1st has raised, which means that through March the number of flights have been steadily increased over the year (again the drop in October is expected since there is no data for the whole month).

Now let's see the number of cases, deaths and number of flights.



And finally, the cases for these countries with the number of flights on the same date.



We can see that as the number of cases rise, there is no significant change to the number of flights, they appear to be steady. We can guess however that there should be an increase in flights during the summer period (months 5-8) but as mentioned they appear to be steady and there are no available data for year 2019 in order to see the number of flights at that period.

6.2 Creating the Cube through Power BI

The implementation to Power BI through SSAS is very simple. Power BI allows you to create a live connection directly into your cube. For simple analysis, this is ideal, i.e., creating new columns which are calculations of existing columns in your table, showing data over time through the fact tables and date dimension. However, when you need to make more complex analysis such as creating measures that pull data from multiple tables or scale data at an enterprise level, SSAS cubes fail and Power BI alone succeeds!

We created and implemented a second cube directly within Power BI (see Figure 38). This allows you to utilize the strength and reliability of a SSAS cube while leveraging the flexibility of Microsoft's newest data technology. Cubes implemented within Power BI not only give you the ability to create new columns and derived tables like in SSAS, but additionally you can create measures that sit on top of the entire cube and run dynamically based on visualization and relationship to other data. Further, utilizing this approach, you can create cubes in Power BI directly and upload them into the Power BI Pro cloud service which then enables others within your organization to use the data warehouse and cube you've created to make reports and dashboards of their own and share them throughout the organization while maintaining the integrity of the data you've given them access to. This approach leverages the ***power*** of SQL and SSAS, the ***flexibility*** of data warehousing and cubes, and the ability to ***scale*** data at an enterprise level through cloud services.

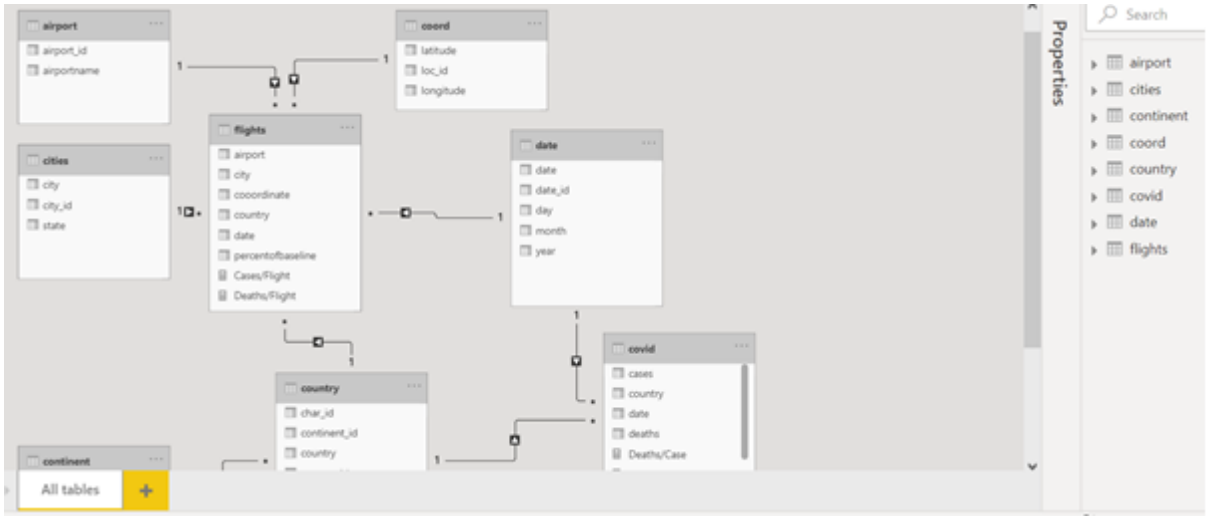


Figure 38 : Fact and Dimension Tables within Power BI

Once the data has been imported into Power BI, it's drag and drop interface makes report and visualization creation extremely easy. In Figure 39 below, we analyse data strictly from the f_covid19 table; looking primarily at Cases and Deaths over time. To display the data over time, we also use the date dimension table. On this page, we have filtered (using the filter pane) just the countries in which we are interested.

We can see Cases trend upwards while Deaths spike then seem to plateau. Additionally, we've created a measure Deaths/Case by dividing the sum of all deaths (in the given period) by the sum of all cases (in the given period). It seems to mimic the trajectory of Deaths more so than Cases.

In figure 40 we analyse this same data, broken out by country. We use country from the d_country dimension table.

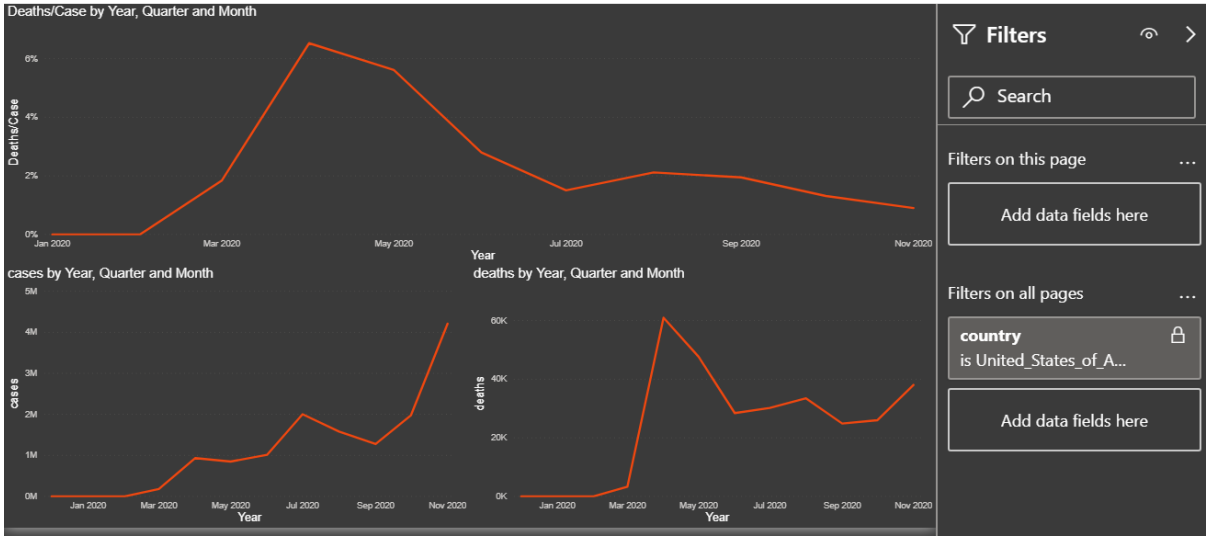


Figure 39 : Analysis of f_covid19 table

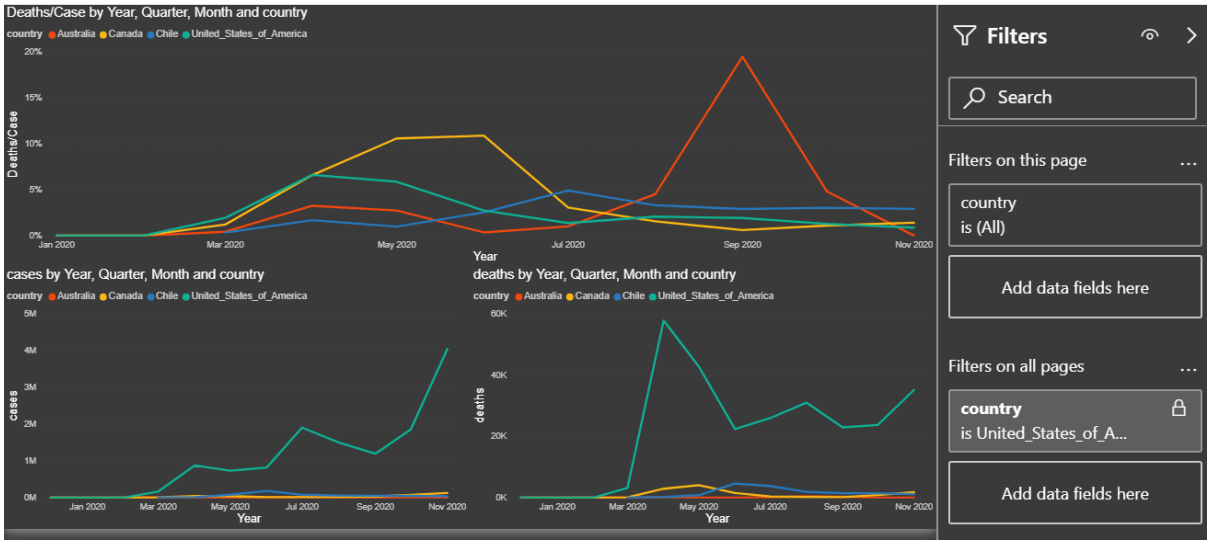


Figure 40 : Analysis of f_covid19 table by country

In figure 41 we normalize the f_covid19 Cases and Deaths using population from the d_country dimension table. Normalized Cases here we are defining as the total number of Cases in the period divided by that country's population. Normalized Deaths are calculated the same way. These were created using Measures in Power BI. Finally, we created a measure using the two previous measures. Normalized Deaths/Case takes the Normalized Cases measure and divides it by the Normalized Deaths measure. These visualizations cannot be read practically as normalizing the data takes the context out; I.e. 0.07 Normalized Cases doesn't make sense. You can interpret these visualizations only relatively when comparing the stats between other countries.

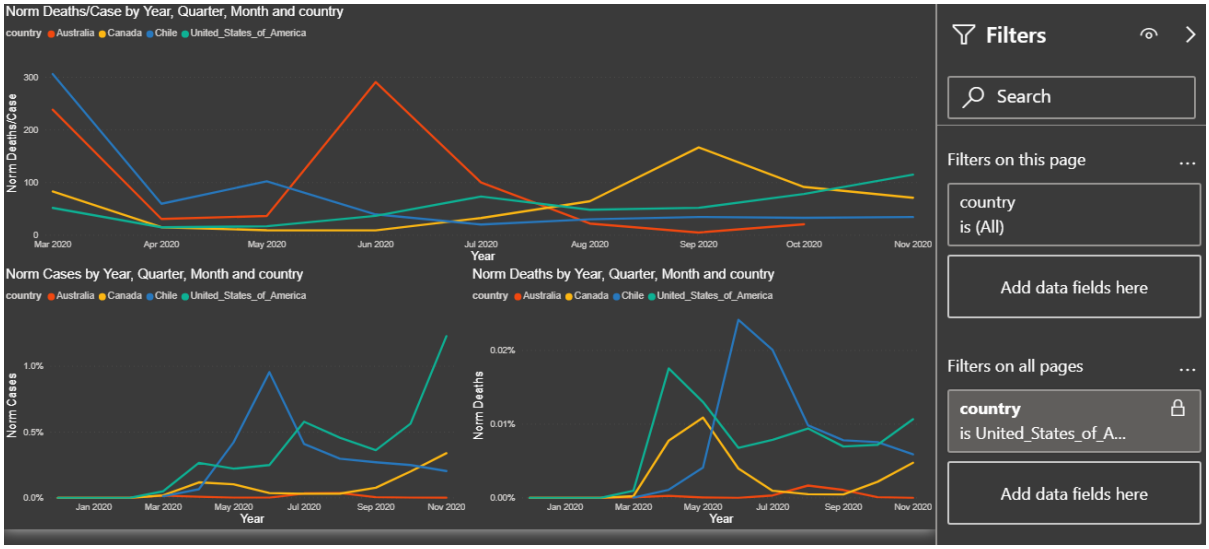


Figure 41 : Normalized Analysis of f_covid19 table by country

In figure 42 we begin to analyze the flight data from the f_flights table. The graph at the top shows the Percent of Baseline for flights which illustrates the change in quantity of flights by country. We can see that despite the covid19 pandemic, flights seemed to overall stay the same; Australia decreasing slightly, but not dramatically. We can affirm this theory by looking at the raw flight count numbers in the graph at the bottom left. Clearly there is no major change between months. The graph in the

bottom right has drill down enabled so you can analyze the number of flights from continent, country, city, and specific airport.

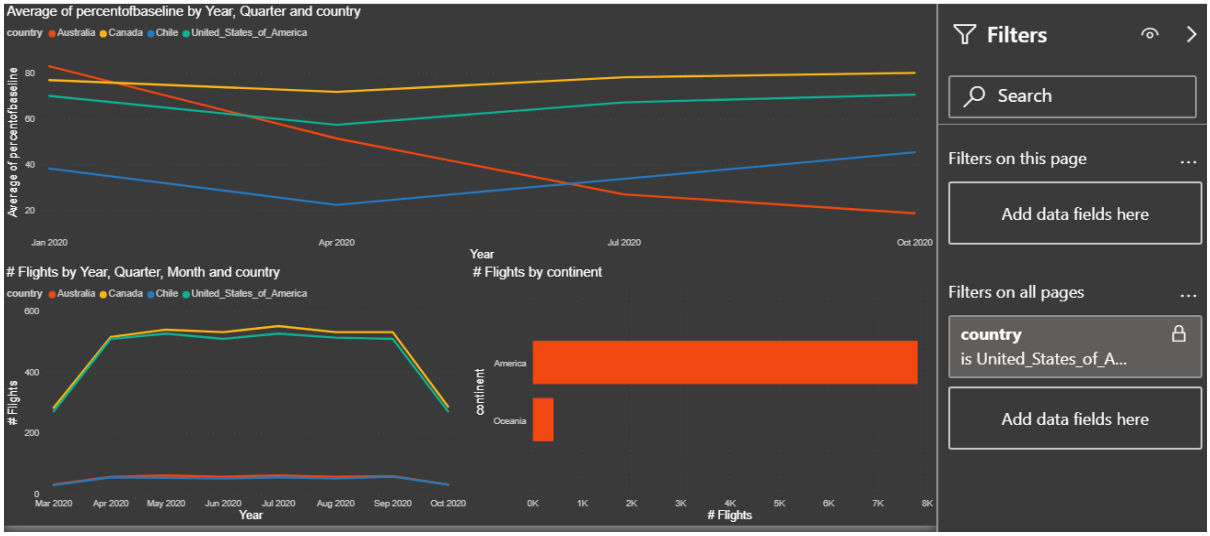


Figure 42 : Analysis of f_flights table by country

Figure 43 illustrates both Cases and Flight data for each of the countries in the study. Here we use a bar chart and simply place various data points side by side on the same graph in order to compare them. We can see that both Canada and the USA have a considerable number of flights, but only the USA has a considerable number of COVID cases. When we normalize the data, the US and Canada appear to have similar standings. We will look more closely at this in figure 45.

Examining the two bottom graphs in figure 43, Cases/Flight and Deaths/Flight, we see that they look very similar to the non-standardized Cases and Deaths graphs. This is further evidence that flights has no considerable effect on COVID numbers. These are both measures created in Power BI. Cases/Flight is the sum of all cases divided by the sum of all flights; Deaths/Flight is calculated similarly.

Figure 44 is the same data except we break out Cases/Flight and Deaths/Flight by country.

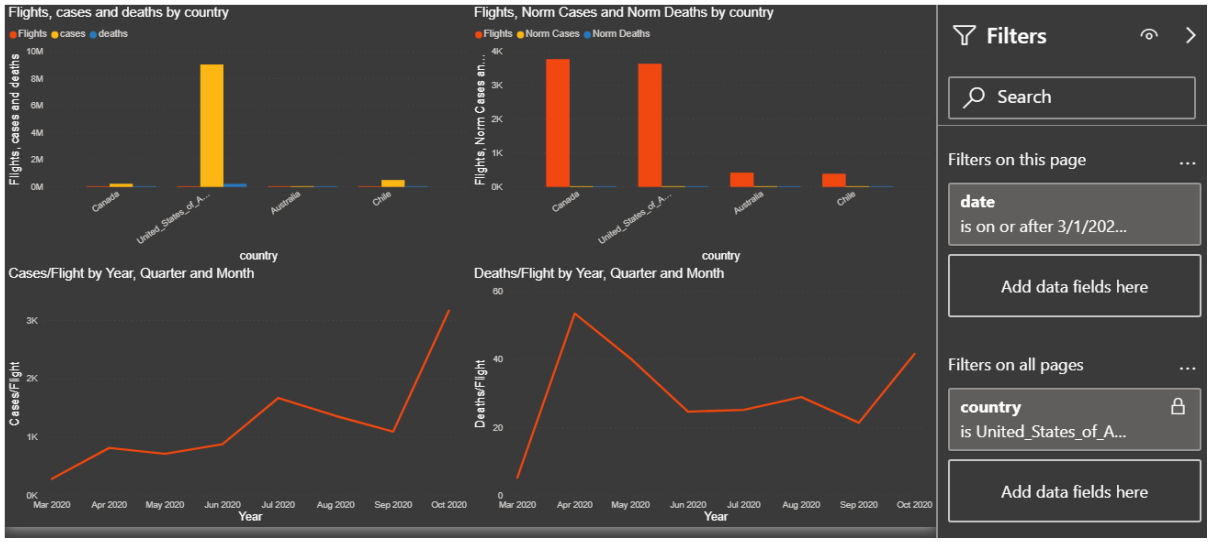


Figure 43 : Analysis of both f_covid19 and f_flights tables

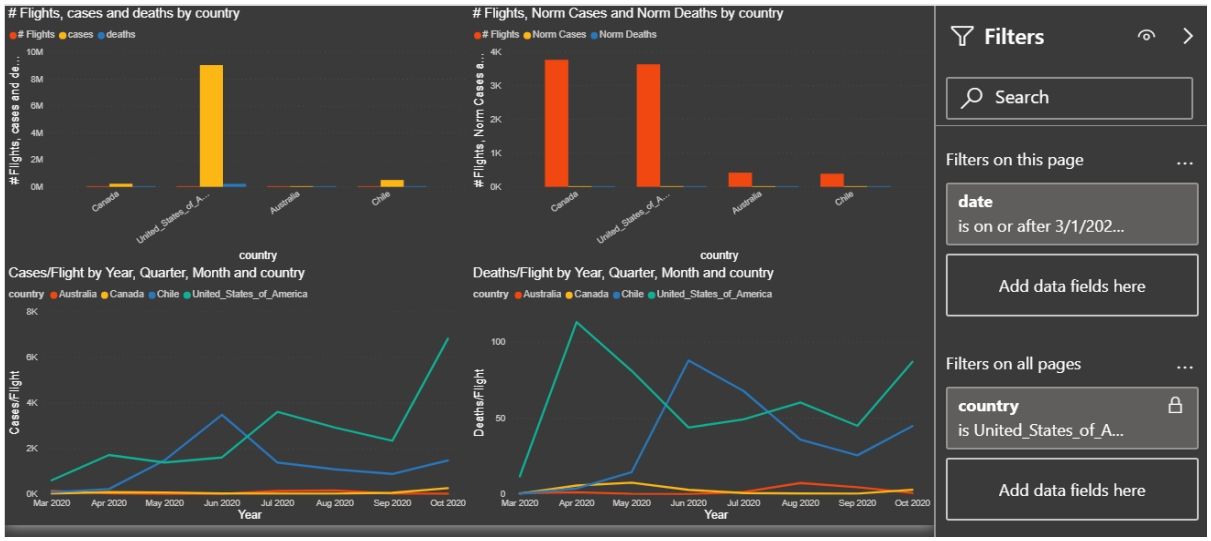


Figure 44 : Analysis of both f_covid19 and f_flights tables by country

Finally, in figure 45 we take a close look at flights, cases/deaths and normalized cases/deaths to see if flights have a significant impact. Here we use a bar and line chart to compare two different types of data. We can see that the US and Canada both have similar numbers of flights, but both their COVID cases/deaths and normalized cases/deaths looks very different. Similarly, Australia and Chile have a similar number of flights for the period and their COVID situation looks much different. We can conclude our analysis by saying we don't believe number of flights from a country contributes to the number of COVID19 cases or deaths in that country.

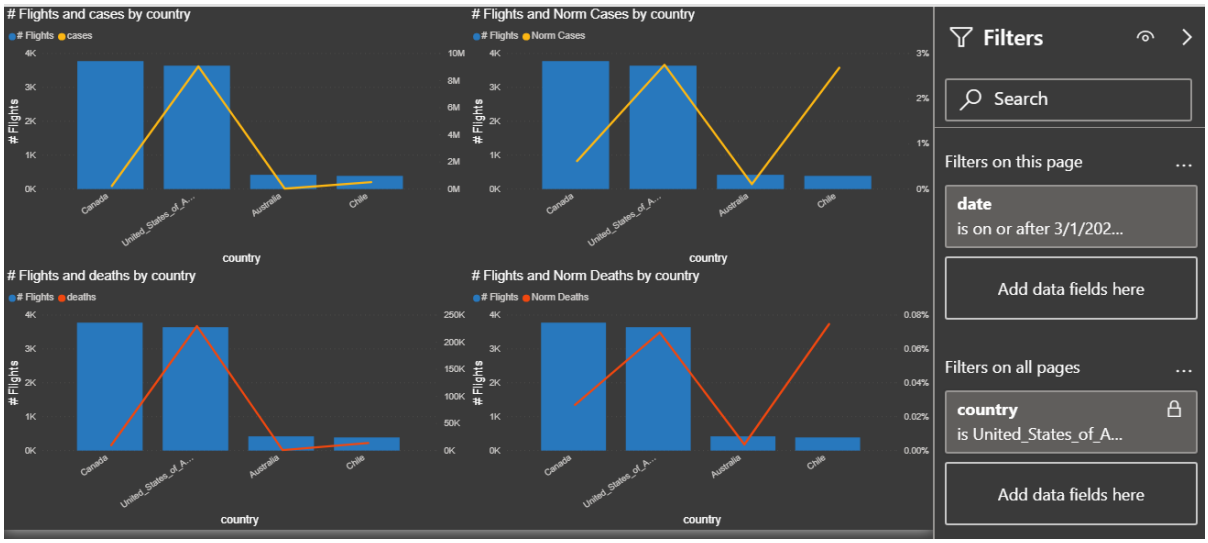


Figure 45 : Summary analysis comparing flights to COVID19 cases

7. Conclusion

The number of flights is not increasing or decreasing, it appears to be steady during the whole year despite the rise of Covid-19 cases. The analysis happened for the countries Australia, Chile, Canada and USA, so it does not apply to every country. Interesting facts that came up with this analysis is that USA is the country with the most cases and deaths making America the continent with most cases and deaths. In Europe, while Russia has the most cases, the United Kingdom seems to have the most deaths followed by Italy. USA is also the lead in the number of flights followed by Canada. So, in conclusion it appears that **there is no connection between the cases of Covid-19 and number of flights**.