



MEMORIA: PROCESAMIENTO DE DATOS MASIVOS

ETL PIPELINE BOOKING CON APACHE SPARK

Raul Alvaro Proleon

Índice

1. Introducción
2. Caso de Estudio
3. Requerimientos del usuario
4. Objetivos del Proyecto
5. Diseño Lógico y Físico del Proyecto
6. Proceso ETL
 - Extracción
 - Análisis Exploratorio de Datos (EDA)
 - Transformación
 - Carga
7. Desafíos Encontrados
8. Conclusiones y recomendaciones
9. Anexos

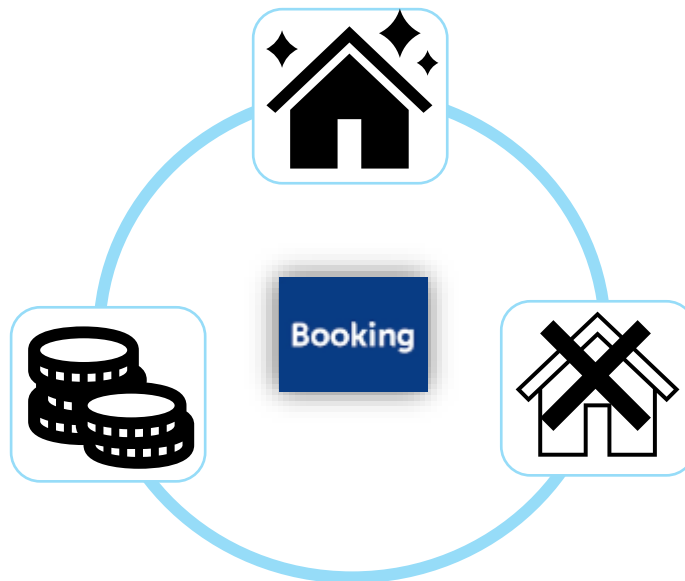
1. Introducción

Este documento describe el proceso de desarrollo de un pipeline ETL (Extract, Transform, Load) utilizando Apache Spark para el procesamiento de datos de reservas de hoteles. El proyecto se enfoca en la limpieza, transformación y carga de datos utilizando el lenguaje de PySpark, para su posterior análisis y toma de decisiones. El código proporcionado en el archivo Etl_ApacheSpark_Booking_RaulAP.ipynb es la base de este proyecto.



2. Caso de Estudio

El caso de negocio se centra en el análisis de datos de reservas de hoteles para mejorar la toma de decisiones en la gestión de reservas, cancelaciones y tarifas. Los datos provienen de un conjunto de datos que incluye información sobre reservas, cancelaciones, tipos de habitaciones, tarifas diarias, entre otros. El objetivo es proporcionar insights que permitan optimizar la gestión de reservas y reducir las cancelaciones.



3. Requerimientos del usuario

El equipo de Booking necesita tomar decisiones relacionadas a la operación del negocio, para lo cual requiere de tres reportes:

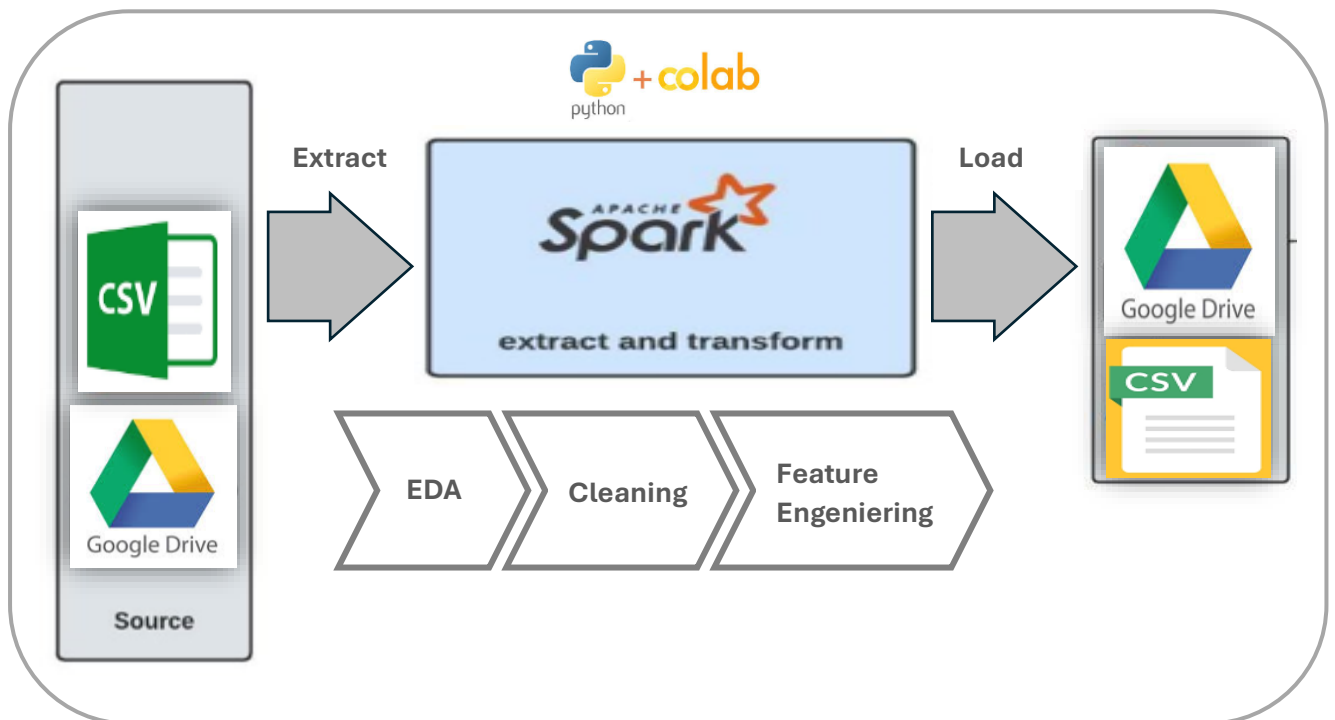
- Reservas y cancelaciones, porcentaje cancelados por mes y año.
- Promedio de días de espera y huéspedes por tipo de habitación.
- Promedio de tarifas diarias por tipo de habitación y año, excluyendo reservas canceladas.

Sin embargo, primero debe gestionar el procesamiento de la información cruda que se aloja en un repositorio.

4. Objetivos del Proyecto










- **Automatización:** Crear un pipeline ETL utilizando Apache Spark.
- **Extracción:** Obtener datos de reservas de hoteles desde un archivo CSV.
- **Análisis Exploratorio:** Realizar un análisis exploratorio de los datos para identificar patrones y tendencias.
- **Transformación:** Limpiar y transformar los datos para su análisis, incluyendo la imputación de valores faltantes, conversión de tipos de datos y creación de nuevas columnas.
- **Carga:** Almacenar los datos transformados o reportes en archivos CSV para su posterior análisis del equipo de negocio.

5. Diseño Lógico y Físico del Proyecto



Mi unidad > Colab Notebooks > Proyecto_Booking_Raul... ▾

Tipo ▾ Personas ▾ Modificado ▾ Fuente ▾

Nombre ▾	Propietario	Última modifi...
 src	 yo	23 feb 2025
 output	 yo	23 feb 2025
 input	 yo	23 feb 2025
 README.md 	 yo	4 mar 2025

6. Proceso ETL

Primero preparamos el entorno de pyspark y las librerías

```
[ ] !apt-get update

# install java
!apt-get install openjdk-8-jdk-headless -qq > /dev/null

# install spark (change the version number if needed)
!wget -q https://archive.apache.org/dist/spark/spark-3.5.4/spark-3.5.4-bin-hadoop3.tgz

# unzip the spark file to the current folder
!tar xf spark-3.5.4-bin-hadoop3.tgz

# install findspark using pip
!pip install -q findspark
```

```

# Importamos Librerías

import findspark
findspark.init()
from pyspark.sql import SparkSession
spark = SparkSession.builder.master("local[*]").getOrCreate()

from pyspark.sql import SparkSession # crear la session de spark
from pyspark.sql import SQLContext   # utilizar las funciones de spark

spark = SparkSession.builder.appName("Test_spark").master("local[*]").getOrCreate()
sqlCtx = SQLContext(sparkContext=spark.sparkContext, sparkSession=spark)
spark

from pyspark.sql.window import Window # manipulacion de ventana
from pyspark.sql import functions as F # operaciones para un dataframe
from pyspark.sql.types import IntegerType # convertir datos
from pyspark.sql.types import DateType   # convertir datos
from pyspark.sql.functions import to_date # convertir datos
import matplotlib.pyplot as plt          # Mostrar graficos
import os                                 # Funciones del Sistema Operativo

from google.colab import drive # conectarse a google drive
drive.mount('/content/drive')  # conectarse a google drive

```

El proceso seguido en el proyecto se divide en las siguientes etapas:

6.1. Extracción:

- Lectura del archivo CSV hotel_bookings.csv utilizando Spark.

```

# Declaramos la Ruta de la carpeta actual donde esta el Archivo
ruta_path = '/content/drive/MyDrive/Colab Notebooks'
#ruta_path = os.getcwd()
file_name='Proyecto_Booking_Raul_AlvaroProleon/input/hotel_bookings.csv'
file_path=os.path.join(ruta_path,file_name)

print("-----")
print("Inicio: Proceso ETL Iniciado...")
print("-----")
print("1.Extracción...")
df_hotel=f_extraccion(file_path)
print("2.Analisis Exploratorio de Datos...")
f_edat(df_hotel)

```

- Verificación de la estructura y calidad de los datos.

```

-----
Inicio: Proceso ETL Iniciado...
-----
1.Extracción...
Columnas: 33
Registros: 119390

```

6.2. Análisis Exploratorio:

Uso de gráficos y estadísticas descriptivas para entender los datos.

- Medidas de tendencia central:

2.Análisis Exploratorio de Datos...

2.1.Medidas de tendencia Central:

summary	index	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day
count	119390	119390	119390	119390	119390	119390	119390	119390
mean	59694.5	NULL	0.37041628277075134	104.01141636652986	2016.156554150264	NULL	27.16517296255968	15.798241
stddev	34465.06865653977	NULL	0.4829182265925987	106.86309704798795	0.7074759445202057	NULL	13.60513835549764	8.78882
min	0	City Hotel	0	0	2015	April	1	1
max	119389	Resort Hotel	1	737	2017	September	53	53

- Estructura del DataFrame:

```

2.2.Estructura del DataFrame:
root
|-- index: integer (nullable = true)
|-- hotel: string (nullable = true)
|-- is_canceled: integer (nullable = true)
|-- lead_time: integer (nullable = true)
|-- arrival_date_year: integer (nullable = true)
|-- arrival_date_month: string (nullable = true)
|-- arrival_date_week_number: integer (nullable = true)
|-- arrival_date_day_of_month: integer (nullable = true)
|-- stays_in_weekend_nights: integer (nullable = true)
|-- stays_in_week_nights: integer (nullable = true)
|-- adults: integer (nullable = true)
|-- children: double (nullable = true)
|-- babies: integer (nullable = true)
|-- meal: string (nullable = true)
|-- country: string (nullable = true)
|-- market_segment: string (nullable = true)
|-- distribution_channel: string (nullable = true)
|-- is_repeated_guest: integer (nullable = true)

```


- Análisis de Missing y Duplicidad

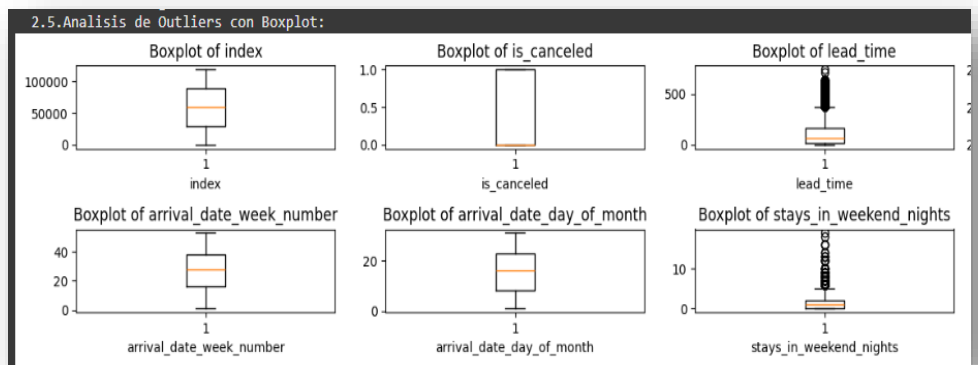
```

2.3.Analisis de missings:
+-----+-----+-----+-----+
|          children|          country|          agent|          company|
+-----+-----+-----+-----+
|3.350364352123293E-5|0.004087444509590418|0.13686238378423654|0.943068933746545|
+-----+-----+-----+-----+

2.4.Analisis de Duplicados:
-Cantidad de registros Totales: 119390
-Cantidad de registros por campo index: 119390
-Cantidad de registros distintos: 119390

```

- Análisis de Outliers



6.3. Transformación:

- Limpieza de datos: Imputación de valores faltantes, eliminación de duplicados y corrección de tipos de datos.

```

# tratamiento de missing
mi_dict = {'country': 'No Ubicado', 'children': 0}
df_tranf_sin_nan=f_tratar_missing(df_cast,mi_dict)
print(" 3.2.Tratamiento de missing: "+str(mi_dict) )
df_sin_nan=f_analisis_missings(df_tranf_sin_nan)
df_sin_nan.show(3)

```

```
# Casting de datos
num_cols=['children','agent','company']
fecha_cols=['reservation_status_date']
df_cast=f_casting_df(df,num_cols,fecha_cols)
print(" -Modificando los tipos de Datos Float a Int: "+str(num_cols))
print(" -Modificando los tipos de Datos String a Date: "+str(fecha_cols))
```

- Creación de nuevas columnas: Por ejemplo, la columna total_huespedes que suma adultos, niños y bebés.

```
# Creacion de nuevas columnas
print(" 3.3.Fase de Feature Engineerings:")
df_huspedes=df_tranf_sin_nan.withColumn('total_huespedes',F.col('adults')+F.col('children')+F.col('babies'))
print(" -Creacion de nuevas columna(s): "+str(['total_huespedes']) )
```

- Conversión de fechas y meses a formatos específicos.

```
# Convertimos el mes a Español
df_tranf_mes=df_huspedes.withColumn('arrival_date_month_esp',F.udf(f_convertir_mes)(F.col('arrival_date_month')))
print(" -Creacion del Mes en Español: "+str(['arrival_date_month_esp']) )
# Convertimos el mes a numerico
df_tranf_mes_num=df_tranf_mes.withColumn('arrival_date_month_num',F.udf(f_convertir_mes_num)(F.col('arrival_date_month')))
print(" -Creacion del Mes Numerico: "+str(['arrival_date_month_num']) )
```

6.4. Carga:

- Almacenamiento de los datos transformados en archivos CSV para su posterior análisis.

```
def f_carga(df,file_path_output1,file_path_output2,file_path_output3):
    # Carga de la Tabla 1
    df_hotel_reservas_pre=df.groupBy(F.col("arrival_date_year").alias('año'), F.col("arrival_date_month_esp").alias('mes'),
                                     F.col("arrival_date_month_num").alias('mes_num')).agg(
        F.count('index').alias('reservas'),
        F.sum('is_canceled').alias('cancelaciones'),
        F.round(F.sum('is_canceled')/F.count('index')*100,2).alias('porcentaje_cancelaciones')
    ).orderBy(F.col('año'),F.col('mes_num').desc() )
    df_hotel_reservas=df_hotel_reservas_pre.drop(F.col('mes_num'))
    df_hotel_reservas.write.mode("overwrite").option('header', 'true').csv(file_path_output1)
    print(" 4.1.Carga la Tabla1:Reservas y cancelaciones, porcentaje cancelados por mes y año")
    df_hotel_reservas.show(2)
    # Carga de la Tabla 2
    df_hotel_dia_espera=df.groupBy(F.col("reserved_room_type").alias('tipo_habitacion')).agg(
        F.round(F.avg('days_in_waiting_list'),2).alias('promedio_dias_espera'),
        F.round(F.avg('total_huespedes'),2).alias('total_huespedes'),
        F.min('total_huespedes').alias('minimo_huespedes'),
        F.max('total_huespedes').alias('maximo_huespedes')
    ).orderBy(F.col('promedio_dias_espera').desc())
    df_hotel_dia_espera.write.mode("overwrite").option('header', 'true').csv(file_path_output2)
    print(" 4.2.Carga la Tabla2:Promedio de días de espera y huespedes por tipo de habitación")
    df_hotel_dia_espera.show(2)
```

- Generación de tres tablas principales: Reservas y cancelaciones, días de espera y tarifas diarias.

Tabla 1:

```
4.Carga de Datos...
  4.1.Carga la Tabla1:Reservas y cancelaciones, porcentaje cancelados por mes y año
```

año	mes	reservas	cancelaciones	porcentaje_cancelaciones
2015	Septiembre	5114	2094	40.95
2015	Agosto	3889	1598	41.09

only showing top 2 rows

Tabla 2:

```
4.2.Carga la Tabla2:Promedio de días de espera y hspedes por tipo de habitación
```

tipo_habitacion	promedio_dias_espera	total_huespedes	minimo_huespedes	maximo_huespedes
A	3.12	1.82	0	55
B	0.58	2.13	0	5

only showing top 2 rows

Tabla 3:

```
4.3.Carga la Tabla3:Promedio de tarifas diarias por tipo de habitación y año, excluyendo reservas canceladas
```

tipo_habitacion	año	promedio_tarifa_diaria
H	2015	166.08
L	2015	151.0

only showing top 2 rows

Tablas Cargadas en Carpeta Output

6.5. Automatización:

- Creación de funciones reutilizables para cada etapa del proceso ETL.

2.Functions ETL

```
[ ] #####  
# Función de la Capa Extracción #  
#####  
def f_extraccion(path):  
    # Ruta archivo Input  
    file_path = path  
    # Creamos el DataFrame Spark  
    df_hotel = spark.read.csv(file_path, header=True, inferSchema=True)  
    # Dimensiones del Dataset  
    print('Columnas:', len(df_hotel.columns))  
    print('Registros:', df_hotel.count())  
    # Muestra las primeras 5 filas del DataFrame  
    #df_hotel.show( 5 , False )  
    return df_hotel  
  
# Función para hallar Missings  
def f_analisis_missings(df):  
    registros=df.count()  
    # Dataframe con cantidad de valores missing  
    df_nan=df.select([F.count(F.when(F.isnull(c), c)).alias(c) for c in df.columns])  
    # Columnas solo con valores missing  
    df_col_nan = [c for c in df_nan.columns if df_nan.select(F.col(c)).first()[0] > 0]
```

- Integración de las funciones en un pipeline principal.

3.Main ETL

```
def main():  
    # Declaramos la Ruta de la carpeta actual donde esta el Archivo  
    ruta_path = '/content/drive/MyDrive/Colab Notebooks'  
    #ruta_path = os.getcwd()  
    file_name='Proyecto_Booking_Raul_AlvaroProleon/input/hotel_bookings.csv'  
    file_path=os.path.join(ruta_path,file_name)  
  
    print("-----")  
    print("Inicio: Proceso ETL Iniciado...")  
    print("-----")  
    print("1.Extracción...")  
    df_hotel=f_extraccion(file_path)  
    print("2.Analisis Exploratorio de Datos...")  
    f_eda(df_hotel)  
    print("3.Transformación...")  
    df_processed=f_transformacion(df_hotel)  
    print("    Resultado de las transformaciones:")  
    df_processed.show(2)  
    print("4.Carga de Datos...")  
    file_name_out1 = 'Proyecto_Booking_Raul_AlvaroProleon/output/tabla_bkg_reservas_cancelacion.csv'  
    file_name_out2 = 'Proyecto_Booking_Raul_AlvaroProleon/output/tabla_bkg_dias_espera.csv'  
    file_name_out3 = 'Proyecto_Booking_Raul_AlvaroProleon/output/tabla_bkg_tarifa_diaria.csv'  
    file_path_output1=os.path.join(ruta_path,file_name_out1)
```

7. Desafíos Encontrados

- **Calidad de los Datos:** La presencia de valores faltantes y datos inconsistentes requirió un proceso de limpieza exhaustivo.
- **Cantidad de los Datos:** Gran volumen de datos con más de 119,000 registros.
- **Rendimiento:** El procesamiento de grandes volúmenes de datos con Spark requirió optimización para evitar cuellos de botella.
- **Conversión de Datos:** La conversión de tipos de datos y formatos de fechas presentó desafíos debido a la inconsistencia en los datos originales.
- **Automatización:** La creación de un pipeline completamente automatizado requirió la integración de múltiples funciones y la gestión de dependencias.

8. Conclusiones y Recomendaciones

- El proyecto demostró la eficacia de Apache Spark para el procesamiento de grandes volúmenes de datos en un pipeline ETL.
- Se logró limpiar, transformar y cargar los datos de reservas de hoteles de manera eficiente, lo que permitirá un análisis más profundo y la toma de decisiones informadas.
- Los principales insights obtenidos incluyen los siguientes reportes:
 - Tasa de cancelación por mes y año.
 - Promedio de días de espera para diferentes tipos de habitaciones.
 - Tarifas promedio diarias según la categoría del hotel.
- Los desafíos encontrados se superaron mediante la optimización del código y la implementación de buenas prácticas en la ingeniería de datos.
- Se logró construir un pipeline eficiente para analizar datos de reservas hoteleras.
- Se recomienda al negocio planificar estrategias en su política de cancelaciones, para reducir la tasa de cancelaciones mensual que aproximadamente ronda el 40%.

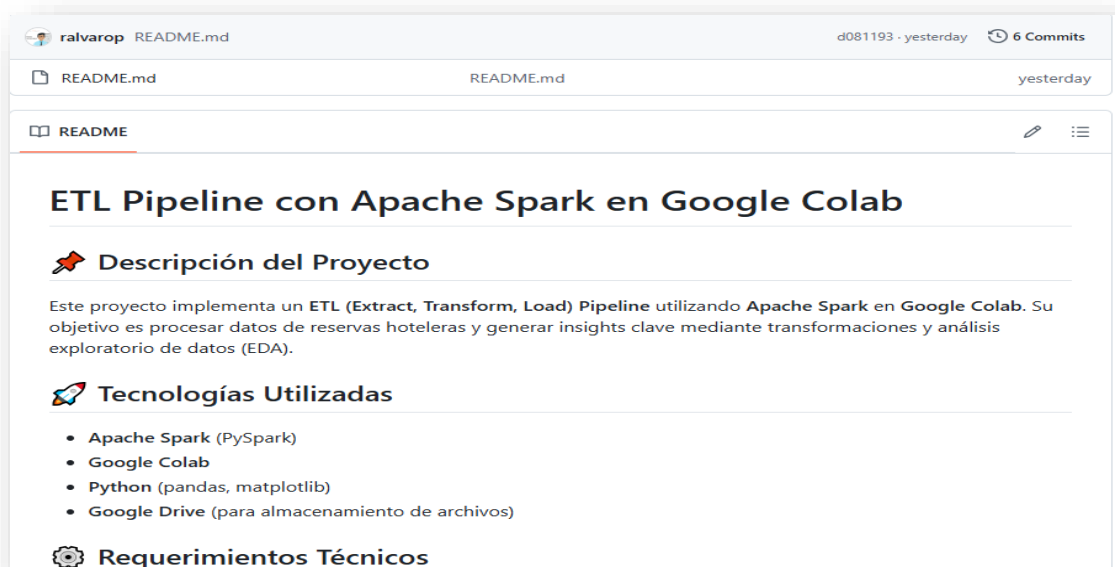
9. Anexos

- **Código Fuente y Base de Datos:**
Archivo Etl_ApacheSpark_Booking_RaulAP.ipynb.



Proyecto_Booking_
Raul_AlvaroProleon

- Archivo README.md





README.md

- Diccionario de las variables:**

RAUL	MÉTRICA	TIPO DE DATO	DESCRIPCIÓN
EJERCICIO 1	index	integer	Índice de la fila en el dataset.
	hotel	string	Tipo de hotel (City Hotel o Resort Hotel).
EJERCICIO 1	is_canceled	integer	Indica si la reserva fue cancelada (1 = Sí, 0 = No).
	lead_time	integer	Días entre la fecha de reserva y la llegada.
EJERCICIO 1	arrival_date_year	integer	Año de llegada del huésped.
EJERCICIO 1	arrival_date_month	string	Mes de llegada del huésped.
	arrival_date_week_number	integer	Número de la semana en el año en la que llega el huésped.
	arrival_date_day_of_month	integer	Día del mes en el que llega el huésped.
	stays_in_weekend_nights	integer	Número de noches de fin de semana en la reserva.
EJERCICIO 2	stays_in_week_nights	integer	Número de noches entre semana en la reserva.
EJERCICIO 2	adults	integer	Número de adultos en la reserva.
EJERCICIO 2	children	double	Número de niños en la reserva.
	babies	integer	Número de bebés en la reserva.
	meal	string	Tipo de comida (BB, FB, HB, SC, Undefined).
	country	string	País de origen del huésped.
	market_segment	string	Segmento de mercado de la reserva.
	distribution_channel	string	Canal por el que se hizo la reserva.
	is_repeated_guest	integer	Indica si el huésped ha reservado antes (1 = Sí, 0 = No).
	previous_cancellations	integer	Número de veces que este huésped ha cancelado reservas.
	previous_bookings_not_cancelled	integer	Número de reservas previas que no fueron canceladas.
EJERCICIO 2	reserved_room_type	string	Tipo de habitación reservada.
	assigned_room_type	string	Tipo de habitación asignada.
	booking_changes	integer	Número de cambios realizados en la reserva.
	deposit_type	string	Tipo de depósito (No Deposit, Non Refund, Refundable).
	agent	double	ID del agente de viaje que gestionó la reserva.
	company	double	ID de la compañía que gestionó la reserva.
EJERCICIO 2	days_in_waiting_list	integer	Días que la reserva estuvo en lista de espera.
	customer_type	string	Tipo de cliente (Transient, Contract, Group, etc.).
EJERCICIO 3	adr	double	Tarifa diaria promedio por habitación.
	required_car_parking_spaces	integer	Número de espacios de estacionamiento requeridos.
	total_of_special_requests	integer	Número de solicitudes especiales realizadas por el cliente.
	reservation_status	string	Estado final de la reserva (Canceled, Check-Out, No-Show).
	reservation_status_date	string	Fecha en la que se actualizó el estado de la reserva.

- Layout:**

GENERAL INFORMATION					
Path	tabla_bkg_reservas_cancelacion.csv				
Description	Muestra las cantidades de Reservas, Cancelaciones y % de Cancelaciones por mes y año.				
Particiones	NA				
SOURCE TABLES					
Source Data	Path	Particiones	Comentarios		
hotel_bookings.csv	/content/drive/MyDrive/Colab Notebooks/Proyecto_Booking_Raul_AhvaroProleon/input/	NA	Data input carga al Drive		
JOINS					
hotel_bookings	tabla_bkg_reservas_cancelacion	PK	Mandatorys	Formato campo	Table Z (Origen)
hotel_bookings.csv	arrival_date_year	X	X	DECIMAL(6)	tabla_bkg_reservas_cancelacion
hotel_bookings.csv	arrival_date_month	X	X	ALPHANUMERIC(50)	tabla_bkg_reservas_cancelacion
hotel_bookings.csv	index		X	DECIMAL(6)	tabla_bkg_reservas_cancelacion
hotel_bookings.csv	is_canceled		X	DECIMAL(6)	tabla_bkg_reservas_cancelacion
				ALPHANUMERIC(50)	tabla_bkg_reservas_cancelacion



Ejemplo Layout.xlsx