

Configuración de un data lake simple.

Configuramos nuestra Data Lake exitosamente con los archivos otorgados,

Databases (1)

A database is a set of as

Filter databases

☐

Name

☐

ebac-dl-sales

Tables

A table is the metadata definition that represents your data, including its schema. A table can be used as a source or target in a job definition.

Tables (3)

View and manage all available tables.

Filter tables

<input type="checkbox"/>	Name	Database	Location	Classific...	Depreca...	View data	Data quality	Column statistics
<input type="checkbox"/>	analista_de_dat	ebac-dl-sales	s3://rubens-buc	CSV	-	Table data	View data qual	View statistics
<input type="checkbox"/>	analista_de_dat	ebac-dl-sales	s3://rubens-buc	CSV	-	Table data	View data qual	View statistics
<input type="checkbox"/>	analista_de_dat	ebac-dl-sales	s3://rubens-buc	CSV	-	Table data	View data qual	View statistics

Por desgracia las Querys de SQL que estamos arrojando no nos están funcionando

Completado

Resultados (0)

Filas de búsqueda

#	invoice id	branch	city	customer type	gender	product line	unit price	quantity
---	------------	--------	------	---------------	--------	--------------	------------	----------

No hay resultados

Ejecute una consulta para ver los resu

Como podemos observar, nos detecta correctamente las columnas

analista_de_datos_m54__kc_house_d

ata_csv

id

bigint

date

string

price

string

bedrooms

bigint

bathrooms

double

sqft_living

bigint

sqft_lot

bigint

floors

double

waterfront

bigint

view

bigint

condition

bigint

grade

bigint

sqft_above

bigint

Exploración de la información de housing utilizando Python, y obteniendo la información del ejercicio anterior.

```
● df = pd.read_csv('DataLake/Analista de datos M54 - kc_house_data.csv')
  df.head(3)
✓ 0.0s
```

	id	date	price	bedrooms	bathrooms	sqft_living	sqft
0	7129300520	20141013T000000	221900.0	3	1.00	1180	
1	6414100192	20141209T000000	538000.0	3	2.25	2570	
2	5631500400	20150225T000000	180000.0	2	1.00	770	1

3 rows × 21 columns

Cree una interfaz de SQL para continuar con la actividad

```
# Crear base de datos en memoria
conexion = sql.connect(":memory:")

# Pasar df a SQL
df.to_sql("kc", conexion, index=False, if_exists="replace")
✓ 0.1s

613

# Realizar consulta SQL
query = """
SELECT avg(bedrooms) as AvgHabitaciones, max(bedrooms) as MaxHabitaciones, min(bedrooms) as MinHabitaciones
FROM kc
"""
✓ 0.0s

# imprimir resultado
resultado = pd.read_sql_query(query, conexion)
print(resultado)
✓ 0.0s
```

Incluir 3 análisis adicionales seleccionados por el estudiante, que respondan a preguntas que el negocio quisiera hacer.

```
# 1. Precio promedio de una casa
query = """
SELECT round(avg(price),2) AS PrecioPromedio
FROM kc;
"""

resultado = pd.read_sql_query(query, conexion)
print(resultado)
✓ 0.0s
```

PrecioPromedio
540088.14

```
# 2. Superficie promedio (sqft_living)
query = """
SELECT round(avg(sqft_living),2) AS SuperficiePromedio
FROM kc;
"""

resultado = pd.read_sql_query(query, conexion)
print(resultado)
✓ 0.0s
```

SuperficiePromedio
2079.9

```
# 3. Promedio de recámaras y baños
query = """
SELECT round(avg(bedrooms),2) AS PromedioRecamaras, round(avg(bathrooms),2) AS PromedioBaños
FROM kc;
"""

resultado = pd.read_sql_query(query, conexion)
print(resultado)
✓ 0.0s
```

PromedioRecamaras	PromedioBaños
3.37	2.11

Incluir KPIs y datos que permitan a una persona sin conocer el negocio a fondo, darse cuenta de sus magnitudes

```
# ¿Qué meses del año tienen los precios más altos? (Abril)
query = """
SELECT
    substr(date, 5, 2) AS month,
    avg(price) AS avg_price
FROM kc
GROUP BY month
ORDER BY avg_price DESC;
"""

resultado = pd.read_sql_query(query, conexion)
print(resultado)
```

✓ 0.0s

	month	avg_price
0	04	561837.774989
1	06	558002.199541
2	05	550768.785833
3	07	544788.764360
4	03	543977.187200
5	10	539026.971778
6	08	536445.276804
7	09	529253.821871
8	01	525870.889571
9	12	524461.866757

```
# ¿Las renovaciones (yr_renovated) impactan el precio? (La respuesta es que si, las casas renovadas tienen un precio promedio más alto)
query = """
SELECT
    CASE WHEN yr_renovated > 0 THEN 'Renovada' ELSE 'No renovada' END AS status,
    avg(price) AS avg_price
FROM kc
GROUP BY status;
"""

resultado = pd.read_sql_query(query, conexion)
print(resultado)
```

✓ 0.0s

Python

	status	avg_price
0	No renovada	530360.818155
1	Renovada	760379.029540

```
# ¿Qué tipo de casas ofrecen el mejor retorno entre precio y metros cuadrados?
query = """
SELECT round(avg(price / sqft_living), 2) || ' dlls/ft²' AS PrecioPromedioPorMetroCuadrado
FROM kc;
"""

resultado = pd.read_sql_query(query, conexion)
print(resultado)
```

✓ 0.0s

PrecioPromedioPorMetroCuadrado
264.16 dlls/ft²