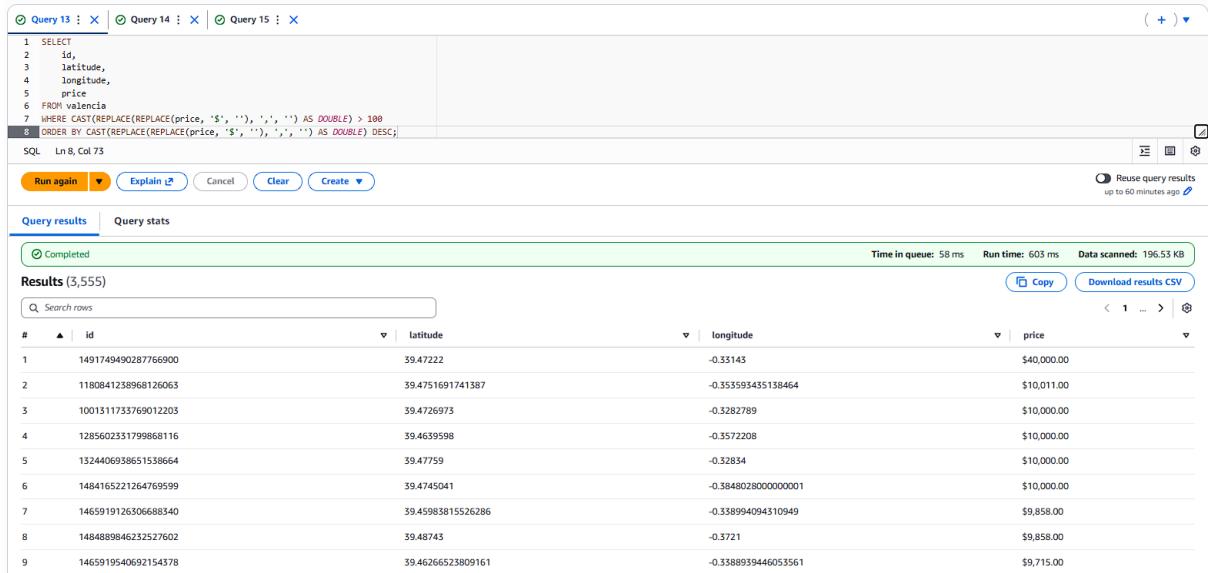


Imágenes de referencia en pasos del examen

Paso 2, inciso 1: Filtrado de los precios superiores a 100 dolares en Valencia.
Query tanto resultado de la query en athena.



```
1 SELECT
2     id,
3     latitude,
4     longitude,
5     price
6 FROM valencia
7 WHERE CAST(REPLACE(REPLACE(price, '$', ''), ',', '') AS DOUBLE) > 100
8 ORDER BY CAST(REPLACE(REPLACE(price, '$', ''), ',', '') AS DOUBLE) DESC;
```

SQL Ln 8, Col 3

Run again Explain Cancel Clear Create

Query results Query stats

Completed

Results (3,555)

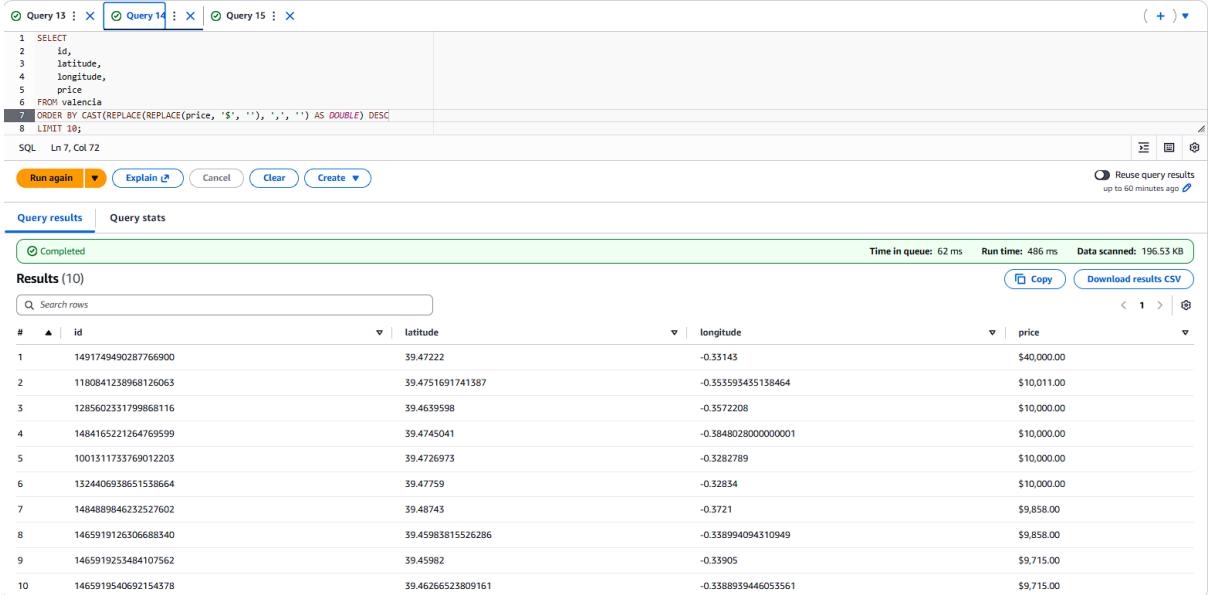
Search rows

#	id	latitude	longitude	price
1	1491749490287766900	39.47222	-0.33143	\$40,000.00
2	1180841238968126063	39.4751691741387	-0.353593435138464	\$10,011.00
3	1001311733769012203	39.4726973	-0.3282789	\$10,000.00
4	1285602331799868116	39.4639598	-0.3572208	\$10,000.00
5	1324406938651538664	39.47759	-0.32834	\$10,000.00
6	1484165221264769599	39.4745041	-0.3848028000000001	\$10,000.00
7	1465919126306688340	39.45983815526286	-0.338994094310949	\$9,858.00
8	1484889846232527602	39.48743	-0.3721	\$9,858.00
9	1465919540692154378	39.46266523809161	-0.3388939446053561	\$9,715.00

Time in queue: 58 ms Run time: 603 ms Data scanned: 196.53 KB

Copy Download results CSV

Paso 2, inciso 2: Selección de los 10 primeros id con mayor precio en Valencia.
Query tanto resultado de la query en athena.



```
1 SELECT
2     id,
3     latitude,
4     longitude,
5     price
6 FROM valencia
7 ORDER BY CAST(REPLACE(REPLACE(price, '$', ''), ',', '') AS DOUBLE) DESC
8 LIMIT 10;
```

SQL Ln 7, Col 2

Run again Explain Cancel Clear Create

Query results Query stats

Completed

Results (10)

Search rows

#	id	latitude	longitude	price
1	1491749490287766900	39.47222	-0.33143	\$40,000.00
2	1180841238968126063	39.4751691741387	-0.353593435138464	\$10,011.00
3	1285602331799868116	39.4639598	-0.3572208	\$10,000.00
4	1484165221264769599	39.4745041	-0.3848028000000001	\$10,000.00
5	1001311733769012203	39.4726973	-0.3282789	\$10,000.00
6	1324406938651538664	39.47759	-0.32834	\$10,000.00
7	1484889846232527602	39.48743	-0.3721	\$9,858.00
8	1465919126306688340	39.45983815526286	-0.338994094310949	\$9,858.00
9	1465919253484107562	39.45982	-0.33905	\$9,715.00
10	1465919540692154378	39.46266523809161	-0.3388939446053561	\$9,715.00

Time in queue: 62 ms Run time: 486 ms Data scanned: 196.53 KB

Copy Download results CSV

Paso 2, inciso 3: Join de Valencia y Madrid para ver que ids tienen el mismo precio. Query tanto resultado de la query en athena.

```

1 SELECT
2     v.id AS valencia_id,
3     m.id AS madrid_id,
4     v.price AS shared_price,
5     v.latitude AS valencia_lat,
6     v.longitude AS valencia_lon,
7     m.latitude AS madrid_lat,
8     m.longitude AS madrid_lon
9 FROM valencia v
10 INNER JOIN madrid m
11 ON v.price = m.price
12 ORDER BY CAST(REPLACE(REPLACE(v.price, '$', ''), ',', '') AS DOUBLE) DESC;

```

SQL Ln 10, Col 18

Run again Explain Cancel Clear Create

Reuse query results up to 60 minutes ago

Query results | Query stats

Completed

Results (659,427)

Copy Download results CSV

#	valencia_id	madrid_id	shared_price	valencia_lat	valencia_lon	madrid_lat	madrid_lon
1	1001511733769012203	1250450625722567310	\$10,000.00	39.4726975	-0.3282789	40.420472	-3.7177158
2	1484165221264769599	1250450623722567310	\$10,000.00	39.4745041	-0.384802800000001	40.420472	-3.7177158
3	1324406938651538664	1250450623722567310	\$10,000.00	39.47759	-0.32834	40.420472	-3.7177158
4	128560235179968116	1250450623722567310	\$10,000.00	39.4639598	-0.3572208	40.420472	-3.7177158
5	1387381043217817309	909558935732387134	\$8,000.00	39.467162675716	-0.391310603119	40.49812	-3.62324
6	1502801638501812293	1274641476444843678	\$1,500.00	39.45017	-0.39142	40.42179	-3.6995

Paso 3: ingestá de datos del archivo de valencia de S3 a snowflake:

```

UEV_MU_ADMIN.PUBLIC <-- Open in Workspaces Code Version

1 --> ----CREAR FILE FORMAT-----
2 CREATE OR REPLACE FILE FORMAT UEV_MU_ADMIN.MODULO_2.PARQUET_FORMAT
3   TYPE = 'PARQUET';
4
5 --> ----COPIAR DATOS A LA TABLA-----
6 COPY INTO UEV_MU_ADMIN.MODULO_2.VALENCIA_LISTINGS (ID, LATITUDE, LONGITUDE, PRICE)
7 FROM (
8   SELECT
9     $1:id::VARCHAR,
10    $1:latitude::NUMBER(10,6),
11    $1:longitude::NUMBER(10,6),
12    TO_NUMBER(REPLACE(REPLACE($1:price::VARCHAR, '$', ''), ',', '')), 10, 2)
13   FROM @UEV_MU_ADMIN.MODULO_2. VALENCIA_STAGE/FileYear=2026/FileMonth=01/FileDay=16/Valencia_data.parquet
14 )
15 FILE_FORMAT = UEV_MU_ADMIN.MODULO_2.PARQUET_FORMAT;
16
17 --> ----VERIFICAR DATOS CARGADOS-----
18 SELECT COUNT(*) FROM UEV_MU_ADMIN.MODULO_2.VALENCIA_LISTINGS;
19
20 --> Ver los primeros 10 registros
21 SELECT * FROM UEV_MU_ADMIN.MODULO_2.VALENCIA_LISTINGS
22 ORDER BY LOAD_TIMESTAMP DESC
23 LIMIT 10;

```

Results | Chart

ID	LATITUDE	LONGITUDE	PRICE	LOAD_TIMESTAMP
48154	39.483750	-0.375020	83.00	2026-01-16 09:38:58.290
137143	39.363350	-0.319320	390.00	2026-01-16 09:38:58.290
149715	39.467460	-0.328130	245.00	2026-01-16 09:38:58.290
165971	39.467900	-0.382060	124.00	2026-01-16 09:38:58.290

Query Details | Query duration | Rows | Query ID 01c1ca | Ask

Paso 4: transient table del 90% de los apartamentos mas caros de valencia, mediante una tarea:

Se hizo un select para ver los registros cargados:

```
-- Ver cuántos registros se cargaron (debe ser ~90% del total)
SELECT COUNT(*) as total_top_90 FROM UEV_MU_ADMIN.MODULO_2.VALENCIA_TOP_90_PRICES;

-- Comparar con el total original
SELECT
    (SELECT COUNT(*) FROM UEV_MU_ADMIN.MODULO_2.VALENCIA_LISTINGS) as total_original,
    (SELECT COUNT(*) FROM UEV_MU_ADMIN.MODULO_2.VALENCIA_TOP_90_PRICES) as total_top_90,
    ROUND((SELECT COUNT(*) FROM UEV_MU_ADMIN.MODULO_2.VALENCIA_TOP_90_PRICES) * 100.0 /
```

Se hizo otro para comparar si si es el 90%:

```
128  
129 -- Comparar con el total original  
130 SELECT  
131     (SELECT COUNT(*) FROM UEV_MU_ADMIN.MODULO_2.VALENCIA_LISTINGS) as total_original,  
132     (SELECT COUNT(*) FROM UEV_MU_ADMIN.MODULO_2.VALENCIA_TOP_90_PRICES) as total_top_90,  
133     ROUND((SELECT COUNT(*) FROM UEV_MU_ADMIN.MODULO_2.VALENCIA_TOP_90_PRICES) * 100.0 /  
134             (SELECT COUNT(*) FROM UEV_MU_ADMIN.MODULO_2.VALENCIA_LISTINGS), 2) as porcentaje;  
135  
136 -- Ver los precios mínimo y máximo del top 90%  
137 SELECT  
138     MIN(PRICE) as precio_minimo_top90,  
139     MAX(PRICE) as precio_maximo_top90,  
140     AVG(PRICE) as precio_promedio_top90  
141 FROM UEV_MU_ADMIN.MODULO_2.VALENCIA_TOP_90_PRICES;  
142  
143 -- Ver los primeros 10 registros  
144 SELECT * FROM UEV_MU_ADMIN.MODULO_2.VALENCIA_TOP_90_PRICES  
145 ORDER BY PRICE DESC  
146 LIMIT 10;
```

Otro para los minimos y máximos dentro de ese 90%

```
36 -- Ver los precios mínimo y máximo del top 90%
37 SELECT
38     MIN(PRICE) as precio_minimo_top90,
39     MAX(PRICE) as precio_maximo_top90,
40     AVG(PRICE) as precio_promedio_top90
41 FROM UEV_MU_ADMIN.MODULO_2. VALENCIA_TOP_90_PRICES;
42
43 -- Ver los primeros 10 registros
44 SELECT * FROM UEV_MU_ADMIN.MODULO_2. VALENCIA_TOP_90_PRICES
45 ORDER BY PRICE DESC
46 LIMIT 10;
47
48 ----- MONITOREAR LA TAREA -----
49 -- Ver el historial de ejecución de la tarea
50 SELECT *
51 FROM TABLE(INFORMATION_SCHEMA.TASK_HISTORY(
52     TASK_NAME => 'TASK_I_NAD_TOP_90_PRICES')
```

Paso 7. Creacion de parquet particionado en base a los datos logrados en los pasos anteriores:

uev-mu-admin-modulo-2-snow > kaggle/ > c_1_0_0/ > 1_0_0/ > valencia/ > FileYear=2026/ > FileMonth=01/ > FileDay=16/

FileDay=16/

[Copy S3 URI](#)

[Objects](#) [Properties](#)

Objects (2) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) [Upload](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix 1

Name	Type	Last modified	Size	Storage class
data_01c1c69b-0308-d759-0003-3c0600554dd_007_1_0.snap.parquet	parquet	January 16, 2026, 20:07:49 (UTC+01:00)	182.8 KB	Standard