



*CARGO POR EVALUAR:*

**Desarrollador**

*OBJETIVO:*

El objetivo de esta prueba es evaluar el nivel de técnico que la persona tiene sobre Terraform

*NOTA:*

Favor enviar el resultado de la prueba a los siguientes correos, antes de la fecha indicada en el correo donde se realizó el envío.

- [JMolinaB@bancooccidente.com.co](mailto:JMolinaB@bancooccidente.com.co)
- [zcastaneda@bancooccidente.com.co](mailto:zcastaneda@bancooccidente.com.co)
- [CBernalP@bancooccidente.com.co](mailto:CBernalP@bancooccidente.com.co)

En el asunto relacionar: **Prueba practica desarrollador célula de datos/ Nicolás Adolfo Castillo Betancourt**

---

Nombre: Nicolás Adolfo Castillo Betancourt

Fecha: 31/01/2024

**Solución Prueba:** [https://github.com/nicomurci/Prueba\\_Desarrollador\\_DataCycle](https://github.com/nicomurci/Prueba_Desarrollador_DataCycle)

A continuación, encontrará una serie de descripciones de recursos de AWS, usted como parte de esta prueba técnica debe resolver el cómo desplegarlos, para cada uno de estos tenga en cuenta el uso de variables junto con cualquier otra buena práctica de terraform.

## 1. IAM ROLE

Haciendo uso de uno o más recursos, cree un rol que cuente con los siguientes atributos:

- Todos los permisos de List
- Todos los permisos de Get
- Permiso de replicacion de etiquetas
- Permiso de creacion y eliminacion del bucket
- No debe tener acceso a la lectura de objetos almacenados

## 2. Bucket S3

Haciendo uso de uno o más recursos cree un bucket de S3 que cuente con los siguientes atributos:

- Las lists de control de acceso deben ser privadas
- El bucket debe contar con versionamiento
- Debe contar con cifrado
- El ciclo de vida del bucket debe ser de 15 días

## 3. EC2

Haciendo uso de uno o más recursos cree una instancia de EC2 que cuente con los siguientes atributos:

- Debe ser una t2.micro
- La region debe ser ohio
- El AMI debe ser el amazon linux 2 4.14
- Almacenamiento de 30 GB SSD
- Una vpc y subnet (configuraciones de su elección) relacionadas a la instancia
- Un grupo de seguridad con entrada por el Puerto 80 para tcp, cualquier ip, vinculado al ec2



## 4. Desarrollo

Dados los nombres y calificaciones de cada estudiante en una clase de  $N$  estudiantes, guárdelos en una lista anidada e imprima los nombres de los estudiantes que tengan la segunda calificación más baja.

**Nota:** Si hay varios estudiantes con la segunda calificación más baja, ordene sus nombres alfabéticamente e imprima cada nombre en una nueva línea.

**Ejemplo:** Records = `[["chi", 20.0], ["beta", 50.0], ["alpha", 50.0]]`

La lista ordenada de puntuaciones es `[20.0, 50.0, 50.0]`, por lo que la segunda puntuación más baja es 50,0.

Hay dos estudiantes con esa puntuación: `["beta", "alfa"]`.

### Restricciones

- $2 \leq N \leq 5$
- Siempre habrá uno o más estudiantes con la segunda nota más baja.

### Formato de salida

Imprima el nombre de cualquier estudiante que tenga la segunda calificación más baja. Si hay varios estudiantes, ordene sus nombres alfabéticamente e imprima cada uno en una nueva línea.

### Entrada de muestra

1. Harry = 37.21
2. Berry = 37.21
3. Tina = 37.2
4. Akriti = 41
5. Harsh = 39

### Ejemplo Salida

Berry  
Harry

### Explicación

Hay 5 estudiantes en esta clase cuyos nombres y calificaciones se reúnen para crear la siguiente lista:

`estudiantes = [['Harry', 37.21], ['Berry', 37.21], ['Tina', 37.2], ['Akriti', 41], ['Harsh', 39]]`

La nota más baja de 37,2 pertenece a Tina. La segunda calificación más baja de 37,21 pertenece tanto a Harry como a Berry, por lo que ordenamos sus nombres alfabéticamente e imprimimos cada nombre en una nueva línea.



Solución

ONLINE PYTHON BETA

main.py

```
12
13 while True:
14     try:
15         calificacion = float(input(f'Ingrese la calificación para {nombre}: '))
16         break
```

Ln: 37, Col: 1

Run

Share

Command Line Arguments

Ingrese la cantidad de estudiantes (entre 2 y 5):

5

Ingrese el nombre del estudiante 1:

Harry

Ingrese la calificación para Harry:

37.21

Ingrese el nombre del estudiante 2:

berry

Ingrese la calificación para berry:

37.21

Ingrese el nombre del estudiante 3:

Tina

Ingrese la calificación para Tina :

37.2

Ingrese el nombre del estudiante 4:

akiti

Ingrese la calificación para akiti:

41

Ingrese el nombre del estudiante 5:

harsh

Ingrese la calificación para harsh:

39

Estudiantes con la segunda peor nota (37.21):

Harry

berry

ESP 12:37 a. m.

LAA 31/01/2024

ONLINE PYTHON BETA

main.py

```
12
13 while True:
14     try:
15         calificacion = float(input(f'Ingrese la calificación para {nombre}: '))
16         break
```

Ln: 37, Col: 1

Run

Share

Command Line Arguments

Ingrese la cantidad de estudiantes (entre 2 y 5):

3

Ingrese el nombre del estudiante 1:

Nicolas

Ingrese la calificación para Nicolas:

4.8

Ingrese el nombre del estudiante 2:

luisa

Ingrese la calificación para luisa:

2.6

Ingrese el nombre del estudiante 3:

camilo

Ingrese la calificación para camilo:

1.2

Estudiantes con la segunda peor nota (2.6):

luisa

\*\* Process exited - Return Code: 0 \*\*

Press Enter to exit terminal

|

ESP 12:38 a. m.

LAA 31/01/2024



## 5. SQL

### Caso 1

La puntuación total de un hacker es la suma de sus puntuaciones máximas en todos los desafíos. Escriba una consulta para imprimir el hacker\_id, el nombre y la puntuación total de los piratas informáticos ordenados por puntuación descendente. Si más de un hacker logró la misma puntuación total, ordene el resultado en forma ascendente hacker\_id. Excluya de su resultado a todos los piratas informáticos con una puntuación total de 0.

Formato de entrada

Las siguientes tablas contienen datos del concurso:

- Hackers: hacker\_id es la identificación del hacker y name es el nombre del hacker.

Column	Type
hacker_id	Integer
name	String

- Envíos: el id\_envío es el id. del envío, el id.hacker\_id es el id. del hacker que realizó el envío, el id\_desafío es el id. del desafío al que pertenece el envío y la puntuación es la puntuación del envío.

Column	Type
submission_id	Integer
hacker_id	Integer
challenge_id	Integer
score	Integer

### Ejemplo de entrada

#### Tabla Hackers

hacker_id	name
4071	Rose
4806	Angela
26071	Frank
49438	Patrick
74842	Lisa
80305	Kimberly
84072	Bonnie
87868	Michael
92118	Todd
95895	Joe



Tabla Envios

submission_id	hacker_id	challenge_id	score
67194	74842	63132	76
64479	74842	19797	98
40742	26071	49593	20
17513	4806	49593	32
69846	80305	19797	19
41002	26071	89343	36
52826	49438	49593	9
31093	26071	19797	2
81614	84072	49593	100
44829	26071	89343	17
75147	80305	49593	48
14115	4806	49593	76
6943	4071	19797	95
12855	4806	25917	13
73343	80305	49593	42
84264	84072	63132	0
9951	4071	49593	43
45104	49438	25917	34
53795	74842	19797	5
26363	26071	19797	29
10063	4071	49593	96

Ejemplo salida:

- 1. 4071 Rose 191
- 2. 74842 Lisa 174
- 3. 84072 Bonnie 100
- 4. 4806 Angela 89
- 5. 26071 Frank 85
- 6. 80305 Kimberly 67
- 7. 49438 Patrick 43

Explicación

El hacker 4071 envió soluciones para los desafíos 19797 y 49593, por lo que la puntuación total = 95 + max(43,96) = 191.

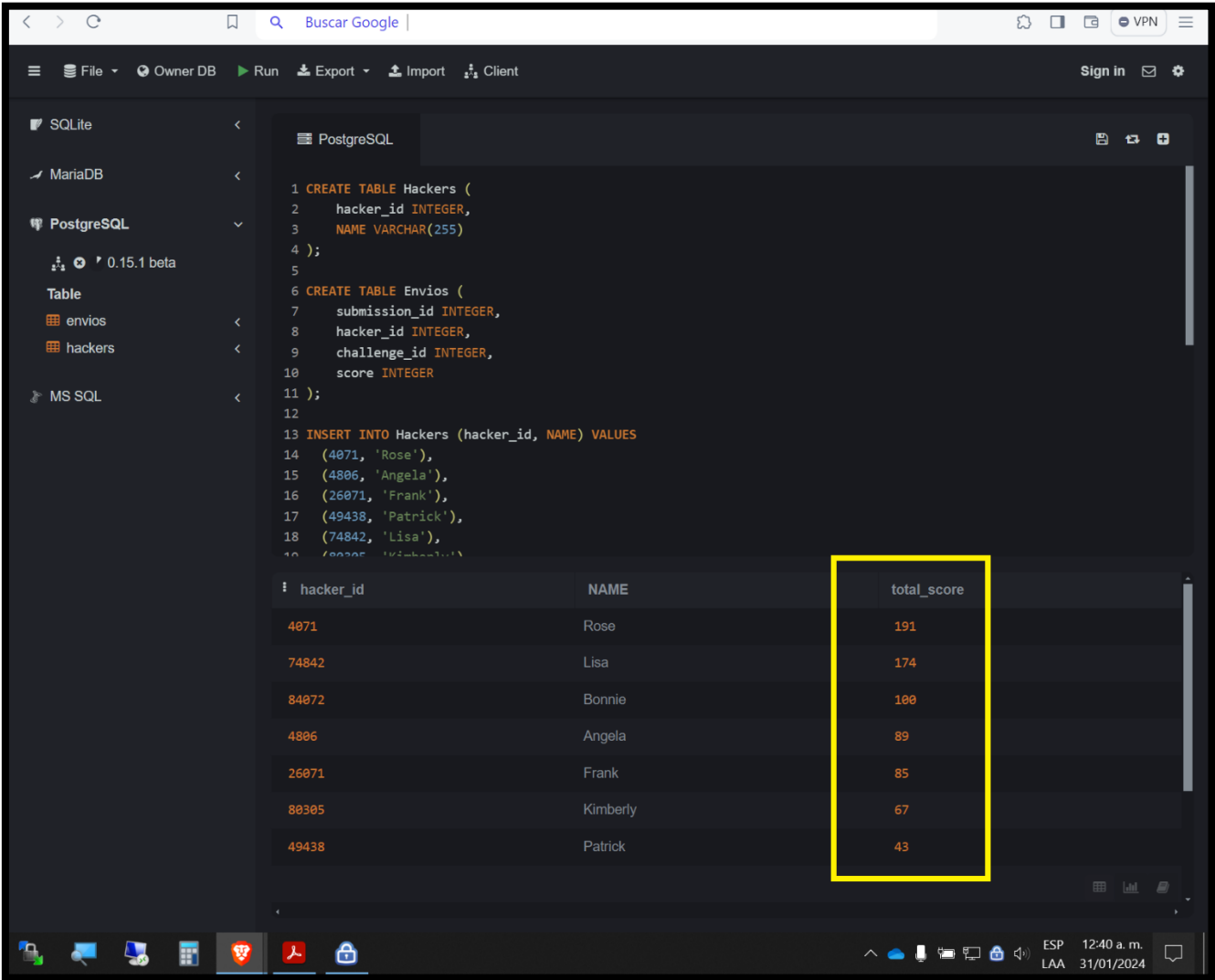
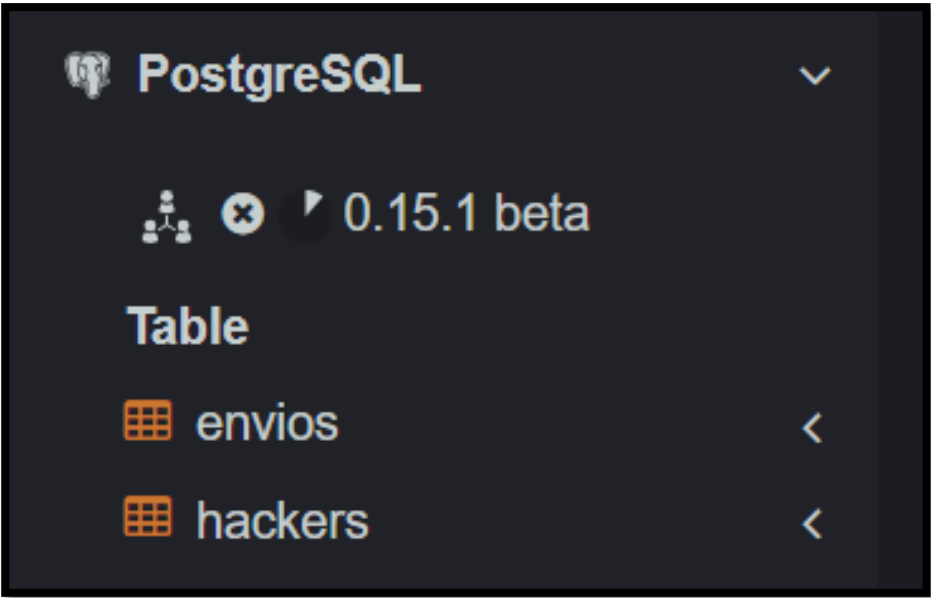
El hacker 74842 envió soluciones para los desafíos 19797 y 63132, por lo que la puntuación total = max(98,5) + 76 = 174

El hacker 84072 envió soluciones para los desafíos 49593 y 63132, por lo que la puntuación total = 100 + 0

Las puntuaciones totales de los hackers 4806, 26071, 80305 y 49438 se pueden calcular de manera similar.



Solución

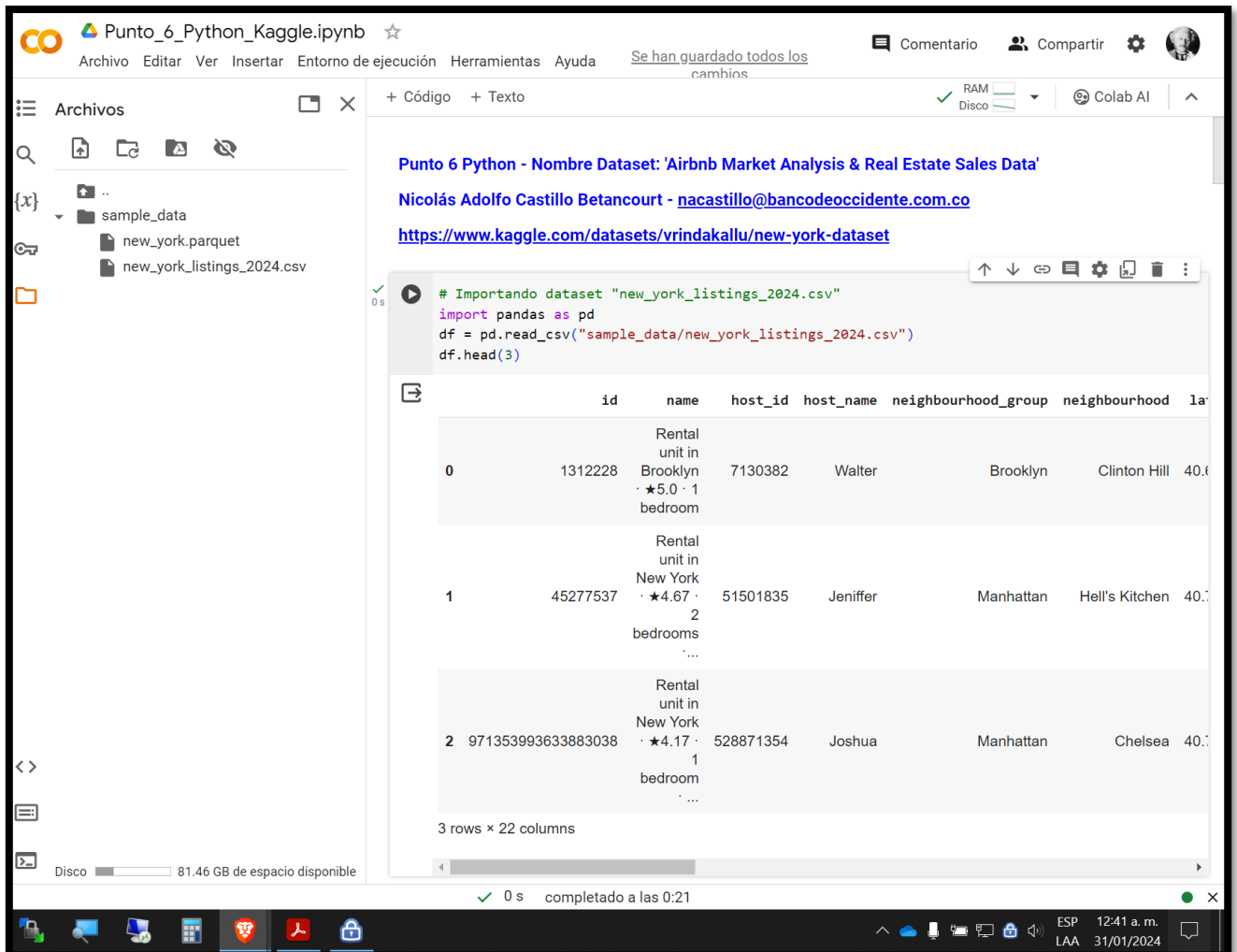




## 6. Python

### Solución

Tomar un dataset de <https://www.kaggle.com/> (a su elección), realizar una limpieza del conjunto de datos con python, procesar los archivos y convertirlos en formato parquet agregando una nueva columna llamada “AVG” donde calcule el promedio de dos columnas numéricas que tenga dentro de su dataset.



The screenshot shows a Jupyter Notebook titled "Punto\_6\_Python\_Kaggle.ipynb". The left sidebar displays a file explorer with a folder named "sample\_data" containing two files: "new\_york.parquet" and "new\_york\_listings\_2024.csv". The main area shows a code cell with the following Python code:

```
# Importando dataset "new_york_listings_2024.csv"
import pandas as pd
df = pd.read_csv("sample_data/new_york_listings_2024.csv")
df.head(3)
```

The output of the code is a table with 3 rows and 22 columns. The columns are: id, name, host\_id, host\_name, neighbourhood\_group, neighbourhood, and la. The first three rows are:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	la
0	1312228	Rental unit in Brooklyn · ★5.0 · 1 bedroom	7130382	Walter	Brooklyn	Clinton Hill	40.6
1	45277537	Rental unit in New York · ★4.67 · 2 bedrooms · ...	51501835	Jeniffer	Manhattan	Hell's Kitchen	40.7
2	971353993633883038	Rental unit in New York · ★4.17 · 1 bedroom · ...	528871354	Joshua	Manhattan	Chelsea	40.7

The bottom status bar indicates "3 rows x 22 columns" and "completado a las 0:21". The system tray at the bottom shows the date and time: "ESP 12:41 a. m. LAA 31/01/2024".



Punto\_6\_Python\_Kaggle.ipynb

Archivos

- sample\_data
  - new\_york.parquet
  - new\_york\_listings\_2024.csv

Proceso de creación de la columna AVG

```
# Asignando columnas a promediar
columna1 = df['price']
columna2 = df['availability_365']
df['AVG'] = (columna1 + columna2) / 2
print(df.head())
```

id	name
1312228	Rental unit in Brooklyn · ★5.0 · 1 bedroom
45277537	Rental unit in New York · ★4.67 · 2 bedrooms · ...
971353993633883838	Rental unit in New York · ★4.17 · 1 bedroom · ...
3857863	Rental unit in New York · ★4.64 · 1 bedroom · ...
40896611	Condo in New York · ★4.91 · Studio · 1 bed · 1...

host_id	host_name	neighbourhood_group	neighbourhood
7130382	Walter	Brooklyn	Clinton Hill
51501835	Jeffer	Manhattan	Hell's Kitchen
528871354	Joshua	Manhattan	Chelsea
19902271	John And Catherine	Manhattan	Washington Heights
61391963	Stay With Vibe	Manhattan	Murray Hill

latitude	longitude	room_type	price	last_review
40.683710	-73.964610	Private room	55.0	2015-12-20
40.766610	-73.988100	Entire home/apt	144.0	2023-05-01
40.750764	-73.994605	Entire home/apt	187.0	2023-12-18
40.835600	-73.942500	Private room	120.0	2023-09-17
40.751120	-73.978600	Entire home/apt	85.0	2023-12-03

reviews_per_month	calculated_host_listings_count	availability_365
0.03	1	0
0.24	139	364
1.67	1	343
1.38	2	363
0.24	133	335

colab.research.google.com/drive/1z5X0zTWbsm2yNmJLnYzY8-FRqzk8v#scr...

Punto\_6\_Python\_Kaggle.ipynb

Archivos

- sample\_data
  - new\_york.parquet
  - new\_york\_listings\_2024.csv

```
2 40.750764 -73.994605 Entire home/apt 187.0 ... 2023-12-18
3 40.835600 -73.942500 Private room 120.0 ... 2023-09-17
4 40.751120 -73.978600 Entire home/apt 85.0 ... 2023-12-03
```

reviews_per_month	calculated_host_listings_count	availability_365
0.03	1	0
0.24	139	364
1.67	1	343
1.38	2	363
0.24	133	335

number_of_reviews_ltm	license	bedrooms	beds	baths	AVG
0	No License	1	1	1.177995	27.5
1	No License	2	1	1.000000	254.0
2	Exempt	1	2	1.000000	265.0
3	No License	1	1	1.000000	241.5
4	No License	Studio	1	1.000000	210.0

[5 rows x 22 columns]

Conversión de DataFrame a Formato Parquet

```
[21] # Conversión a formato parquet
df.to_parquet("sample_data/new_york.parquet", index=False)
```

Disco 81.46 GB de espacio disponible

completado a las 0:21

ESP 12:42 a.m.  
LAA 31/01/2024