

ASSIGNMENT 5.4
on
Data Architecture and Data Lakes in AWS

Submitted by:
Haseebullah Shaikh (2303.KHI.DEG.015)
and
Faiza Gulzar Ahmed (2303.khi.deg.001)

Dated: 19th May 2023

Solution :

Simulated the Earning predictions for two more days, each day contains 100 entries or rows, created 200 new random entries for two days 100 for each day. We have used np.random function of NumPy library of python for generating random rows. Furthermore, random data is converted in to parquet format and stored in local directory.

Manipulation file

```
[2] import pandas as pd
import numpy as np

directory = 'output_data/employee_earnings/earnings_date=2022-02-'
file= 'employee_earnings.parquet'

df_earnings1 = pd.read_parquet(f'{directory}10/{file}')
df_earnings2 = pd.read_parquet(f'{directory}11/{file}')
df_earnings3 = pd.read_parquet(f'{directory}12/{file}')
df_earnings4 = pd.read_parquet(f'{directory}13/{file}')
df_earnings5 = pd.read_parquet(f'{directory}14/{file}')

df_earnings = pd.concat([df_earnings1, df_earnings2, df_earnings3, df_earnings4, df_earnings5], axis=0)

[8]
```

```
[8] ... <bound method DataFrame.count of      emp_id first_name middle_initial last_name
0    526540  Angelique           K    Goodwin  angelique.goodwin@gmail.com
1    859327      Jeni           S    Shaffer   jeni.shaffer@gmail.com
2    887387   Donald           T    Farris  donald.farris@bellsouth.net
3    779497   Steven           D    Rendon  steven.rendon@gmail.com
4    896517   Jenell           L  Almanza   jenell.almanza@yahoo.com
..      ...      ...      ...      ...
95   549389  Clemente           M    Gould   clemente.gould@hotmail.com
96   466832    Chang           K    Roden   chang.roden@yahoo.com
97   203380   Marvin           R    Nickel   marvin.nickel@ibm.com
98   915991   Eldora           Y  Tribble  eldora.tribble@earthlink.net
99   289172    Azzie           L    Layman  azzie.layman@hotmail.co.uk
```

```

    date_of_birth date_of_joining      ssn phone_number user_name \
0    1964-05-15    2001-03-24  471-57-0359  212-884-7146  akgoodwin
1    1962-01-13    2015-12-10  624-85-4146  205-665-7020  jsshaffer
2    1958-04-11    1979-11-12  097-02-3315  205-959-7879  dtfarris
3    1982-04-04    2008-09-18  134-98-6566  217-858-0054  sdrendon
4    1958-07-01    1993-07-14  599-92-7345  314-893-2590  jlalmanza
..      ...      ...      ...      ...      ...
95   1961-12-31    1992-10-02  271-17-5467  228-485-0919  cmgould
96   1988-09-07    2010-08-06  074-02-9202  316-256-7851  ckroden
97   1986-11-25    2012-10-06  552-99-5545  270-750-7760  mrnickel
98   1995-05-29    2016-10-17  763-12-2082  236-584-1916  eytribble
99   1961-09-06    2004-03-26  637-29-1007  503-456-5899  allayman
...
97      8*E[g-_X|      Scranton      7741
98      z>ms?;$8-u      Nashua      8080
99      k<%?%TML.].1ZY      New York      5868

```

[500 rows x 13 columns]>

Output is truncated. View as a [scrollable element](#) or open in a [text editor](#). Adjust cell output [settings...](#)

```

num_rows = 100 # Number of rows to add
df_predict_1 = pd.DataFrame(
    {col: np.random.choice(df_earnings[col], num_rows) for col in df_earnings.columns}
)

df_predict_1

```

Python Python

	emp_id	first_name	middle_initial	last_name	email	date_of_birth	date_of_joining	ssn	phone_number	us
0	633636	Rex	B	Mullis	dolores.begley@hotmail.com	1968-08-11	2015-12-10	235-57-9815	207-532-1710	
1	886060	Adalberto	C	Harwell	winfred.gonzales@aol.com	1965-05-21	1998-04-14	704-18-2387	217-297-4629	
2	403534	Myron	W	Burk	angelique.goodwin@gmail.com	1958-04-11	2015-04-23	345-11-4018	219-802-6287	
3	432820	Adalberto	V	Goad	nam.schmidt@yahoo.co.uk	1980-12-13	1987-01-02	599-92-7345	239-814-5354	
4	819367	Allan	Q	Trotter	verna.michaud@gmail.com	1992-06-15	2016-01-27	074-02-0202	262-835-1119	n

```
num_rows = 100 # Number of rows to add
df_predict_2 = pd.DataFrame(
    {col: np.random.choice(df_earnings[col], num_rows) for col in df_earnings.columns}
)

df_predict_2
```

	emp_id	first_name	middle_initial	last_name	email	date_of_birth	date_of_joining	ssn	phone_number	use
0	550206	Chastity	Z	Knox	nam.schmidt@yahoo.co.uk	1986-10-09	1992-07-14	345-11-4018	503-456-5899	cc
1	317987	Humberto	O	Wilson	benjamin.doss@gmail.com	1982-10-26	1991-04-14	622-85-5122	239-814-5354	c
2	242388	Jake	U	Lumpkin	michale.colson@comcast.net	1992-07-26	2005-09-03	044-15-4027	217-348-2881	mo
3	530134	Arlena	G	Goolsby	ramiro.conover@earthlink.net	1970-07-27	2017-04-28	019-94-6803	316-256-7851	
4	721091	Virgil	S	Farris	vicente.dawkins@gmail.com	1988-09-07	2016-01-18	005-13-9026	605-487-4676	rz
...

```
df_predict_1.to_parquet(f'{directory}15/{file}', index=False)
df_predict_2.to_parquet(f'{directory}16/{file}', index=False)
```

Uploaded created random files in existing bucket which were created in task

Amazon S3 > Buckets > faiza-module5-day4 > data/ > output_data/ > employee_earnings/

employee_earnings/ Copy S3 URI

Objects Properties

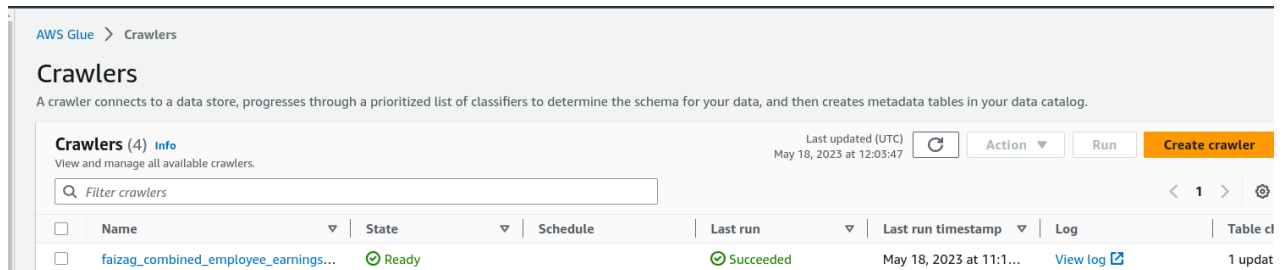
Objects (7)

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](#)

Refresh Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	earnings_date=2022-02-10/	Folder	-	-	-
<input type="checkbox"/>	earnings_date=2022-02-11/	Folder	-	-	-
<input type="checkbox"/>	earnings_date=2022-02-12/	Folder	-	-	-
<input type="checkbox"/>	earnings_date=2022-02-13/	Folder	-	-	-
<input type="checkbox"/>	earnings_date=2022-02-14/	Folder	-	-	-
<input type="checkbox"/>	earnings_date=2022-02-15/	Folder	-	-	-
<input type="checkbox"/>	earnings_date=2022-02-16/	Folder	-	-	-

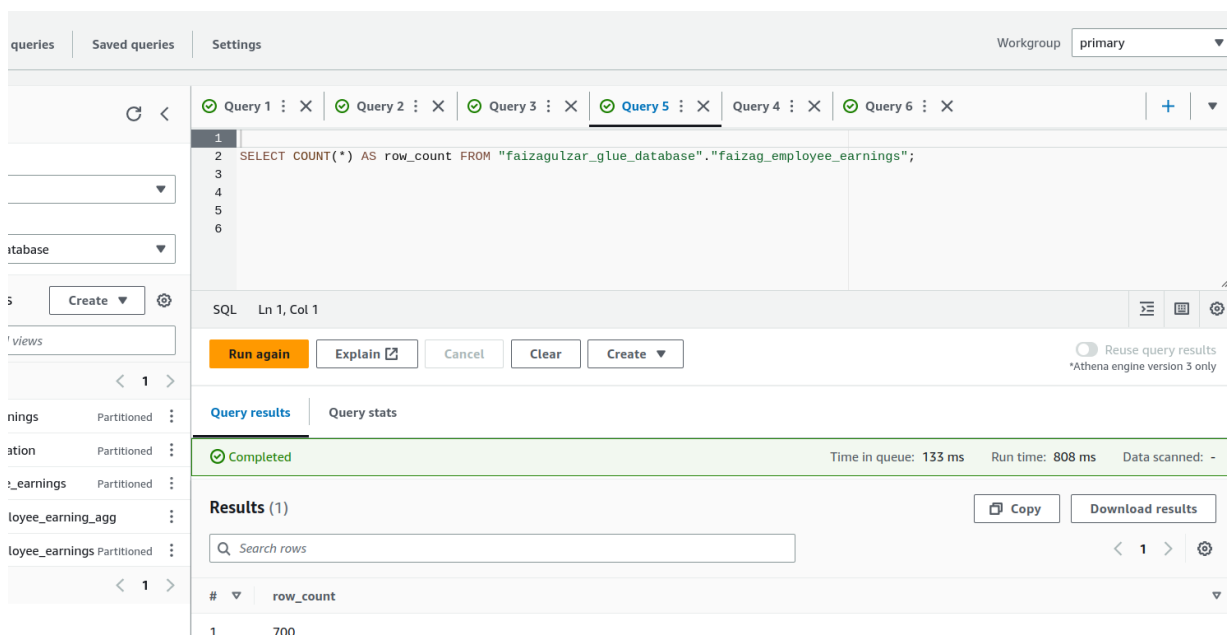
We ran again the crawler named `faizag_combined_employee_earnings` which were created in the task for extracting the new uploaded data.



The screenshot shows the AWS Glue Crawlers console. At the top, it says 'Crawlers' and 'A crawler connects to a data store, progresses through a prioritized list of classifiers to determine the schema for your data, and then creates metadata tables in your data catalog.' Below this, there's a section for 'Crawlers (4)' with an 'Info' link. It says 'View and manage all available crawlers.' To the right, it shows 'Last updated (UTC) May 18, 2023 at 12:03:47' and buttons for 'Action', 'Run', and 'Create crawler'. A search bar labeled 'Filter crawlers' is present. Below the search bar is a table with columns: Name, State, Schedule, Last run, Last run timestamp, Log, and Table count. The table lists one crawler: 'faizag_combined_employee_earnings...' with a state of 'Ready', a 'Succeeded' status, a last run timestamp of 'May 18, 2023 at 11:1...', a 'View log' link, and a table count of '1 update'.

Name	State	Schedule	Last run	Last run timestamp	Log	Table count
faizag_combined_employee_earnings...	Ready		Succeeded	May 18, 2023 at 11:1...	View log	1 update

We ran the given SQL queries again in Athena service to see the changes in data. As it can be seen there are 700 rows, previously there were 500 as we have now add 2 days random data which are 200 hundred rows.



The screenshot shows the AWS Athena console. At the top, there are tabs for 'queries', 'Saved queries', and 'Settings'. The 'Workgroup' is set to 'primary'. Below the tabs, there's a section for 'Query 1' through 'Query 6'. The 'Query 5' tab is selected. The SQL query is: `SELECT COUNT(*) AS row_count FROM "faizagular_glue_database"."faizag_employee_earnings";`. Below the query, there's a 'Run again' button, an 'Explain' button, a 'Cancel' button, a 'Clear' button, and a 'Create' button. To the right, there's a 'Reuse query results' toggle and a note '*Athena engine version 3 only'. Below the query, there's a 'Query results' tab and a 'Query stats' tab. The 'Query results' tab is selected. It shows a 'Completed' status, 'Time in queue: 133 ms', 'Run time: 808 ms', and 'Data scanned: -'. Below this, there's a 'Results (1)' section with a 'Copy' button and a 'Download results' button. A search bar labeled 'Search rows' is present. Below the search bar, there's a table with columns: #, row_count. The table has one row: '1 700'.

#	row_count
1	700

Amazon Athena > Query editor

Editor Recent queries Saved queries Settings Workgroup primary

Data

Data source: AwsDataCatalog

Database: faizagulzar_glue_database

Tables and views

Filter tables and views

Tables (5)

- faiza_gulzar_earnings Partitioned
- faiza_gulzar_location Partitioned
- faizag_employee_earnings Partitioned
- faizagulzar_employee_earning_agg
- faizagulzar_employee_earnings Partitioned

Views (0)

Query 1

```
1 SELECT DISTINCT emp_id, email, office_branch, (date_diff('year', DATE(date_of_birth), current_date))
2 AS age
3 FROM "faizagulzar_glue_database"."faizag_employee_earnings"
4 WHERE office_branch IN ('New York', 'Scranton')
5 AND
6 (date_diff('year', DATE(date_of_birth), current_date)) > 30;
```

SQL Ln 5, Col 63

Run again Explain Cancel Clear Create

Reuse query results
*Athena engine version 3 only

Query results Query stats

Completed Time in queue: 170 ms Run time: 857 ms Data scanned: 24.70 KB

Results (134)

Search rows

#	emp_id	email	office_branch	age
1	403534	angelique.goodwin@gmail.com	New York	65
2	439483	alfred.beamon@gmail.com	New York	42
3	496541	steven.rendon@gmail.com	New York	42
4	235295	sofia.poole@earthlink.net	New York	38

Data

Data source: AwsDataCatalog

Database: faizagulzar_glue_database

Tables and views

Filter tables and views

Tables (5)

- faiza_gulzar_earnings Partitioned
- faiza_gulzar_location Partitioned
- faizag_employee_earnings Partitioned
- faizagulzar_employee_earning_agg
- faizagulzar_employee_earnings Partitioned

Views (0)

Query 2

```
1
2 SELECT office_branch, MIN(earnings) as min_earnings, MAX(earnings) as max_earnings, AVG(earnings)
3 as avg_earnings, SUM(earnings) as total_earnings, earnings_date
4 FROM "faizagulzar_glue_database"."faizag_employee_earnings"
5 GROUP BY office_branch, earnings_date
6 ORDER BY SUM(earnings) desc;
```

SQL Ln 4, Col 40

Run again Explain Cancel Clear Create

Reuse query results
*Athena engine version 3 only

Query results Query stats

Completed Time in queue: 192 ms Run time: 1.068 sec Data scanned: 5.19 KB

Results (28)

Search rows

#	office_branch	min_earnings	max_earnings	avg_earnings	total_earnings	earnings_date
1	Nashua	2124	9916	5874.4857142857145	205607	2022-02
2	New York	2660	9916	6991.793103448276	202762	2022-02
3	Nashua	2098	9728	6099.8387096774195	189095	2022-02

Data

Data source

AwsDataCatalog

Database

faizagulzar_glue_database

Tables and views

Create

Filter tables and views

Tables (5)

faiza_gulzar_earnings

Partitioned

faiza_gulzar_location

Partitioned

faizag_employee_earnings

Partitioned

faizagulzar_employee_earning_agg

faizagulzar_employee_earnings

Partitioned

Views (0)

Query 1Query 2Query 3Query 4Query 5

```
1 SELECT DISTINCT office_branch, (MAX(avg_earnings.value) - MIN(avg_earnings.value)) as earnings_range
2 FROM (
3   SELECT office_branch as ob, AVG(earnings) AS value FROM "faizagulzar_glue_database"
4     ."faizag_employee_earnings" GROUP BY office_branch, earnings_date
5   ) avg_earnings, "faizagulzar_glue_database"."faizag_employee_earnings"
6   WHERE office_branch = avg_earnings.ob
   GROUP BY office_branch;
```

SQLLn 6, Col 26

Run again

Explain

Cancel

Clear

Create

Reuse query results

Athena engine version 3 only

Query results

Query stats

Completed

Time in queue: 153 ms

Run time: 1.457 sec

Data scanned: 6.13 KB

Results (4)

Copy

Download results

Search rows

#	office_branch	earnings_range
1	Scranton	2037.3307692307699
2	Nashua	978.9813895781635
3	New York	1376.2573891625616
4	Stanford	1129.880434782609

Created new query which calculates the % change in earnings for every employee from a given day compared to the previous day

Query 1 : X
Query 2 : X
Query 3 : X
Query 5 : X
Query 4 : X
Query 6 : X

+

```

1 SELECT
2   previous_day.earnings_date AS previous_date,
3   previous_day.earnings AS previous_amount,
4   current_day.earnings_date AS date_current,
5   current_day.earnings AS current_amount,
6   ((current_day.earnings - CAST(previous_day.earnings as double)) / previous_day.earnings) * 100 AS percentage_change
7 FROM
8   "faizagulzar_glue_database"."faizag_employee_earnings" AS previous_day
9 JOIN
10  "faizagulzar_glue_database"."faizag_employee_earnings" AS current_day ON previous_day.earnings_date < current_day.earnings_date
11 WHERE
12  previous_day.earnings_date = '2022-02-14'
13  AND current_day.earnings_date = '2022-02-15';
14

```

SQL Ln 12, Col 33

Run again
Explain
Cancel
Clear
Create

Reuse query results
*Athena engine version 3 only

Query results
Query stats

Completed
Time in queue: 119 ms
Run time: 998 ms
Data scanned: 1.18 KB

#	previous_date	previous_amount	date_current	current_amount	percentage_change
1	2022-02-14	2716	2022-02-15	6883	153.4241531664212
2	2022-02-14	8357	2022-02-15	6883	-17.63790834031351
3	2022-02-14	8123	2022-02-15	6883	-15.265296072879478
4	2022-02-14	8297	2022-02-15	6883	-17.042304447390624
5	2022-02-14	2057	2022-02-15	6883	234.61351482741856
6	2022-02-14	9378	2022-02-15	6883	-26.60481979100021
7	2022-02-14	3557	2022-02-15	6883	93.50576328366601
8	2022-02-14	8353	2022-02-15	6883	-17.598467616425236
9	2022-02-14	4977	2022-02-15	6883	38.296162346795256
10	2022-02-14	7272	2022-02-15	6883	-5.34928492849285
11	2022-02-14	4101	2022-02-15	6883	67.83711289929285
12	2022-02-14	4843	2022-02-15	6883	42.12265124922568
13	2022-02-14	6157	2022-02-15	6883	11.791456878349846
14	2022-02-14	8794	2022-02-15	6883	-21.73072549465545

The End 😊