

Assignment 5.2 Submitted by Ali Nasir(2023-KHI-DEG-032) & Saif ur Rehman(2023-KHI-DEG-007):

Using the salary CSV as a base, prepare a new data file with employees' office locations. Make sure there are 5-6 distinct locations that are shared between employees.

Create a Glue job that aggregates the data based on the office location to calculate average salaries and raise percentages for these locations.

Solution:

Daily task:

- We have used S3 bucket input/location to extract the csv

The screenshot shows the AWS Glue console interface. On the left, a job graph is visible with the following components: two 'Data source - S3 bucket Amazon S3' nodes, a 'Transform - Join' node, a 'Transform - SQL Query' node, and a 'Data target - S3 bucket Amazon S3' node. The 'Data source properties - S3' tab is selected on the right, showing the following configuration:

- Name: Amazon S3
- S3 source type: S3 location (selected)
- S3 URL: s3://alinasir/input_data/location/locations.csv
- Recursive: ☒ Recursive
- Data format: CSV
- Delimiter: Comma (,)
- Escape character - optional: (empty)
- Quote character: Double quote (")
- First line of source file contains column headers: ☒

- We have used S3 bucket input/earning to extract the csv

The screenshot shows the AWS Glue console interface. On the left, a job graph is visible with the following components: two 'Data source - S3 bucket Amazon S3' nodes, a 'Transform - Join' node, a 'Transform - SQL Query' node, and a 'Data target - S3 bucket Amazon S3' node. The 'Data preview' tab is selected on the right, showing a preview of the data. The preview shows 100 rows of data with columns 'emp_id' and 'earnings'.

emp_id	earnings
526540	6227
859327	4437
887387	6228
779497	3127
896517	3930
220965	8693
721091	4820
633636	9040
823898	4648
413865	7636
439483	6722
809408	5850
748190	3187
397283	9688
553684	9252
495667	5201
936158	5636

- Here we have implemented join on the both dataset

The screenshot displays a data pipeline in a 'Visual' tab. Two 'Data source - S3 bucket Amazon S3' nodes are connected to a 'Transform - join Join' node. This node then connects to a 'Transform - SQL Query SQL Query' node, which finally connects to a 'Data target - S3 bucket Amazon S3' node. The right-hand panel shows the 'Data preview' for the 'Join' node, displaying a table with 100 rows and 4 columns: emp_id, .emp_id, location, and earnings.

emp_id	.emp_id	location	earnings
537591	537591	A	9106
886060	886060	B	7208
819367	819367	E	9801
526540	526540	A	6227
170637	170637	C	3193
709884	709884	C	4179
526254	526254	D	2066
530134	530134	D	5093
976422	976422	C	5327
820109	820109	E	2376
856379	856379	E	7610
505927	505927	D	9972
220965	220965	A	8693
550206	550206	B	9131
460226	460226	D	3714
397283	397283	A	9688
936158	936158	A	5636

- Here is the query that we have implemented to find the aggregate data.

The screenshot shows the configuration window for the 'SQL Query' node in the pipeline. The 'Name' field is set to 'SQL Query'. The 'Node parents' dropdown is set to 'Join - Transform'. The 'Input sources' section shows 'Join' as the source, with an associated 'myDataSource' alias. The 'SQL query' field contains the following SQL statement:

```

1 SELECT
2     location,
3     AVG(earnings) AS average_earnings,
4     (AVG(earnings) - MIN(earnings)) / MIN(earnings) * 100 AS raise_percentage
5 FROM
6     myDataSource
7 GROUP BY
8     location;

```

VisualScriptJob detailsRunsData quality NewSchedulesVersion Control

SourceActionTargetUndoRedoRemove

```
graph TD; S1[Data source - S3 bucket Amazon S3] --> J[Transform - Join]; S2[Data source - S3 bucket Amazon S3] --> J; J --> T[Transform - SQL Query]; T --> D[Data target - S3 bucket Amazon S3];
```

TransformOutput schemaData preview

Data preview (5) Info

Filter sample dataset

location	average_earnings	raise_percentage
B	6286.75	155.14407467532467
C	5576.95	129.78780387309433
A	5926.05	191.49286768322676
D	5889.7	185.07744433688285
E	5599.2	158.74306839186693

- Here is the output of the data from our crawler.

Amazon S3> Buckets> alinaslr> output_data/

output_data/

Copy S3 URI

ObjectsProperties

Objects (5)

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](#)

Copy S3 URICopy URLDownloadOpenDeleteActionsCreate folderUpload

Find objects by prefix

< 1 >

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	run-1684312012081-part-block-0-r-00002-snappy.parquet	parquet	May 17, 2023, 13:27:01 (UTC+05:00)	599.0 B	Standard
<input type="checkbox"/>	run-1684312012081-part-block-0-r-00014-snappy.parquet	parquet	May 17, 2023, 13:27:00 (UTC+05:00)	599.0 B	Standard
<input type="checkbox"/>	run-1684312012081-part-block-0-r-00021-snappy.parquet	parquet	May 17, 2023, 13:27:01 (UTC+05:00)	599.0 B	Standard
<input type="checkbox"/>	run-1684312012081-part-block-0-r-00025-snappy.parquet	parquet	May 17, 2023, 13:27:00 (UTC+05:00)	599.0 B	Standard
<input type="checkbox"/>	run-1684312012081-part-block-0-r-00031-snappy.parquet	parquet	May 17, 2023, 13:27:01 (UTC+05:00)	599.0 B	Standard