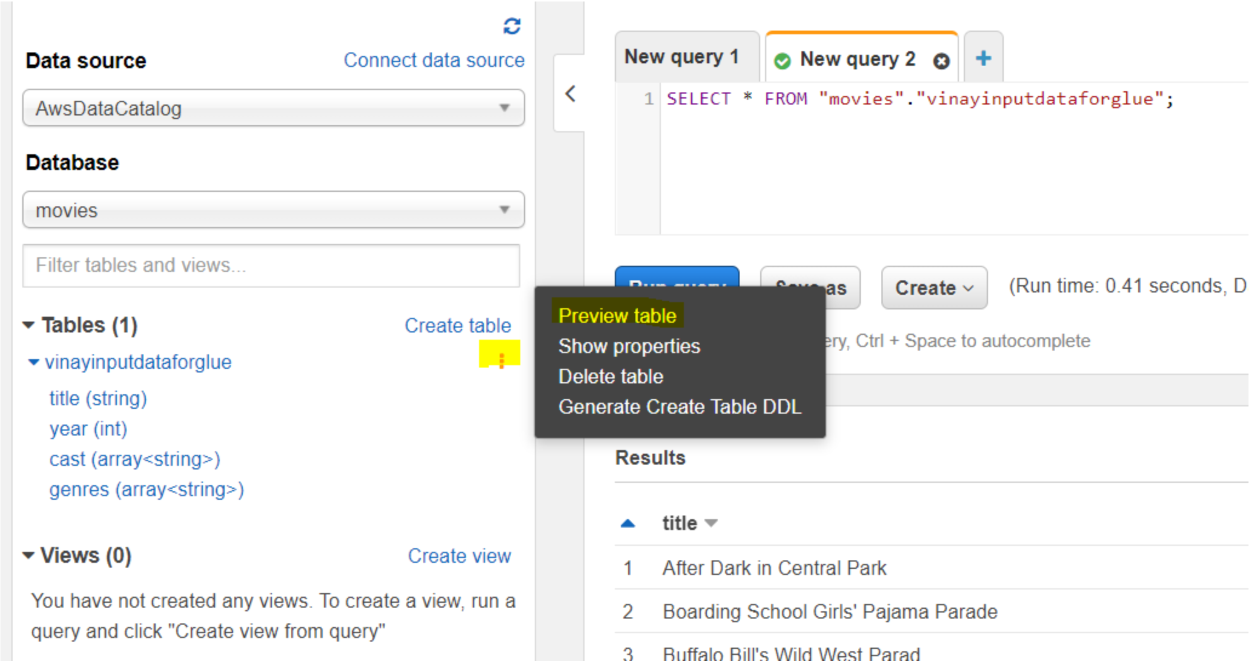
Prerequisites : Data catalog should be created in Glue. We have already done that in GlueDemo based on sample movies data (movies_sample.json).

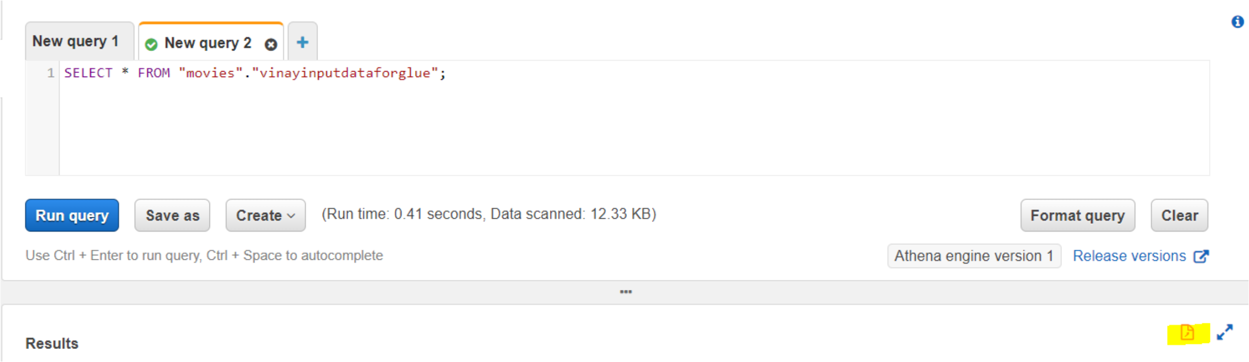
1. Go to Athena. Click on 3 vertical dots besides table name and click on preview table. It will generate a query and will execute and show the result.



```
SELECT * FROM "movies"."vinayinputdataforglue";
```

	title
1	After Dark in Central Park
2	Boarding School Girls' Pajama Parade
3	Buffalo Bill's Wild West Parad

2. You can also download the result by clicking on highlighted yellow button in below snapshot.



```
SELECT * FROM "movies"."vinayinputdataforglue";
```

(Run time: 0.41 seconds, Data scanned: 12.33 KB)

3. Run below query to see how many movies were released year wise.

SELECT year, COUNT(title) from "aug12-movies-database"."aug12moviestabledata"
GROUP BY year
ORDER BY COUNT(title);

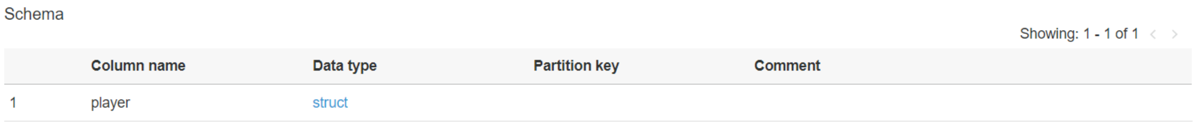
4. Modify above query to see percentage of movies released year wise.

```
SELECT year, COUNT(title), (COUNT(title) * 100.0/(SELECT COUNT(*) from "aug12-movies-database"."aug12moviestabledata")) from "aug12-movies-database"."aug12moviestabledata"
GROUP BY year
ORDER BY COUNT(title)
```

ASSIGNMENT:

5. Let's now look at NestedJSON.json file.Create a new bucket with the NestedJSON.json file uploaded in it.

6. Create a crawler and name it as *nested-crawler.* While creating the crawler, give the database name as *nesteddb.*
7. Once the crawler is created, Run the crawler.
8. Crawler will create database *nesteddb* and within *nesteddb*, there will be a table created.
9. After clicking on table name, you will get below schema:

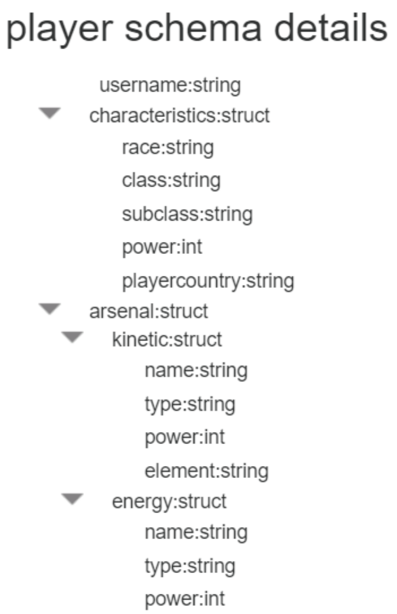


Schema

Showing: 1 - 1 of 1

	Column name	Data type	Partition key	Comment
1	player	struct		

This table cannot be queried by Athena as when we click on struct in above snapshot, we get below details:

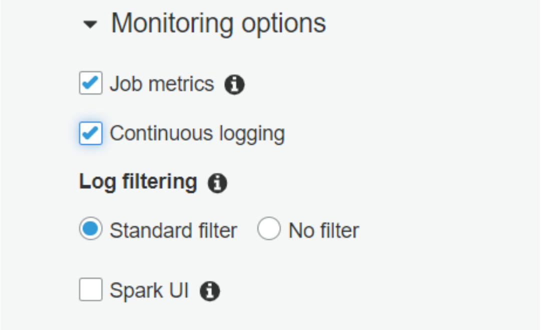


player schema details

- username:string
- characteristics:struct
  - race:string
  - class:string
  - subclass:string
  - power:int
  - playercountry:string
- arsenal:struct
  - kinetic:struct
    - name:string
    - type:string
    - power:int
    - element:string
  - energy:struct
    - name:string
    - type:string
    - power:int

10. We will now have to create Glue Job, which will convert NestedJSON file into a csv file.
11. Go to Glue jobs and click on Add Job. Name the job as flatten-data-job.

12. Expand Monitoring Options and select as shown below:



Monitoring options

☑ Job metrics

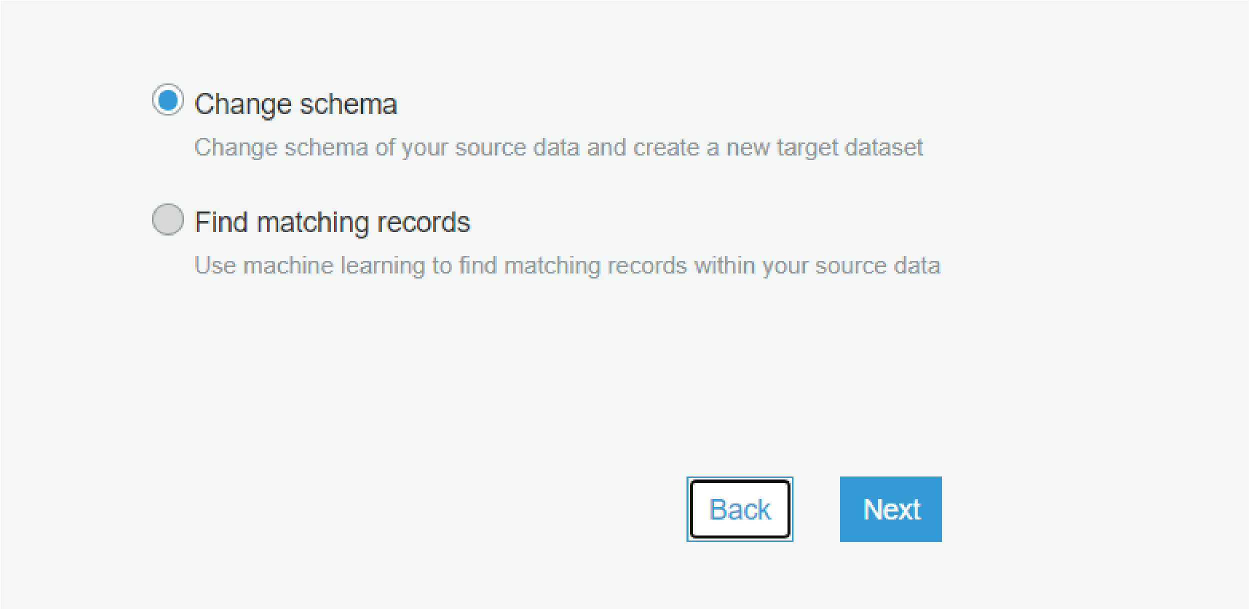
☑ Continuous logging

Log filtering

● Standard filter ○ No filter

□ Spark UI

13. Select Change schema as shown below:Click Next.



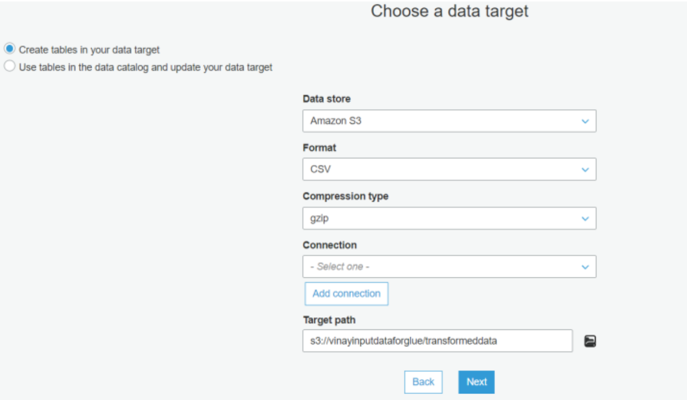
● Change schema

Change schema of your source data and create a new target dataset

○ Find matching records

Use machine learning to find matching records within your source data

14. Choose Create tables in your data target and select options as shown in below snapshot. Ensure that *transformeddata* folder is created in S3 bucket. Click Next



Choose a data target

● Create tables in your data target
○ Use tables in the data catalog and update your data target

Data store: Amazon S3

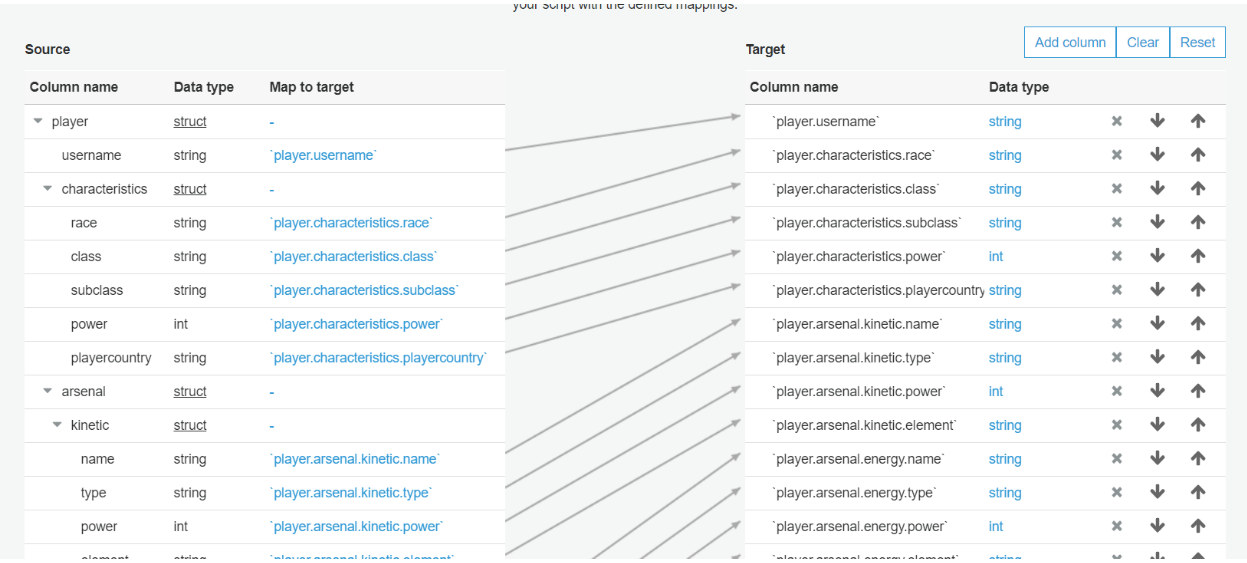
Format: CSV

Compression type: gzip

Connection: - Select one -

Target path: s3://vinayinputdataforglue/transformeddata

15. Output Schema definition will be created as shown below:



your script with the defined mappings.

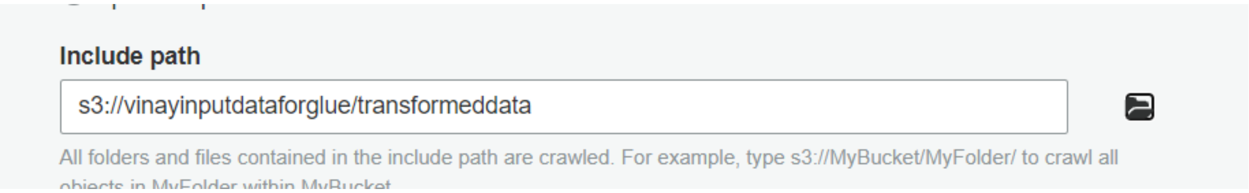
Source

Column name	Data type	Map to target
player	struct	-
username	string	`player.username`
characteristics	struct	-
race	string	`player.characteristics.race`
class	string	`player.characteristics.class`
subclass	string	`player.characteristics.subclass`
power	int	`player.characteristics.power`
playercountry	string	`player.characteristics.playercountry`
arsenal	struct	-
kinetic	struct	-
name	string	`player.arsenal.kinetic.name`
type	string	`player.arsenal.kinetic.type`
power	int	`player.arsenal.kinetic.power`
element	string	`player.arsenal.kinetic.element`

Target

Column name	Data type
`player.username`	string
`player.characteristics.race`	string
`player.characteristics.class`	string
`player.characteristics.subclass`	string
`player.characteristics.power`	int
`player.characteristics.playercountry	string
`player.arsenal.kinetic.name`	string
`player.arsenal.kinetic.type`	string
`player.arsenal.kinetic.power`	int
`player.arsenal.kinetic.element`	string
`player.arsenal.energy.name`	string
`player.arsenal.energy.type`	string
`player.arsenal.energy.power`	int
`player.arsenal.energy.element`	string

16. Click on Save Job and Edit Script.
17. Save Job and Run Job.
18. Go to transformeddata folder in S3 Bucket and download the zip file. Extract the zip and view the file data.
19. In order to query this data using Athena, create a crawler to construct Data Catalog from the data in compressed form available in *transformeddata* folder.
20. Click on Add crawler and give the name as csv-crawler. Select transformeddata folder on S3 as include path.



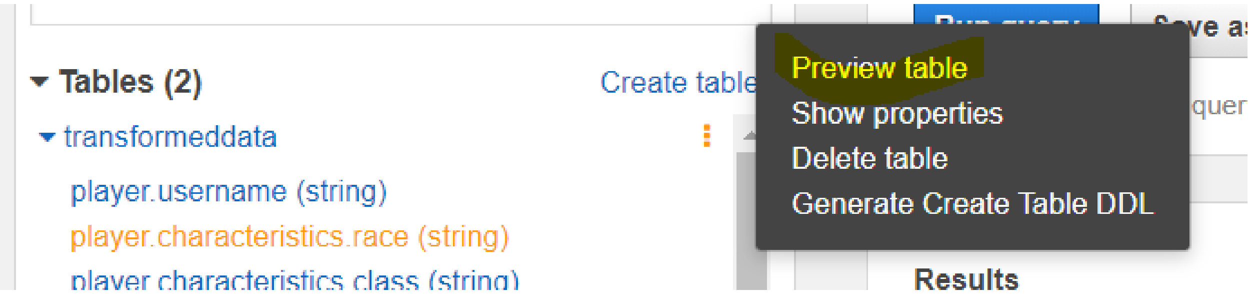
Include path

s3://vinayinputdataforglue/transformeddata

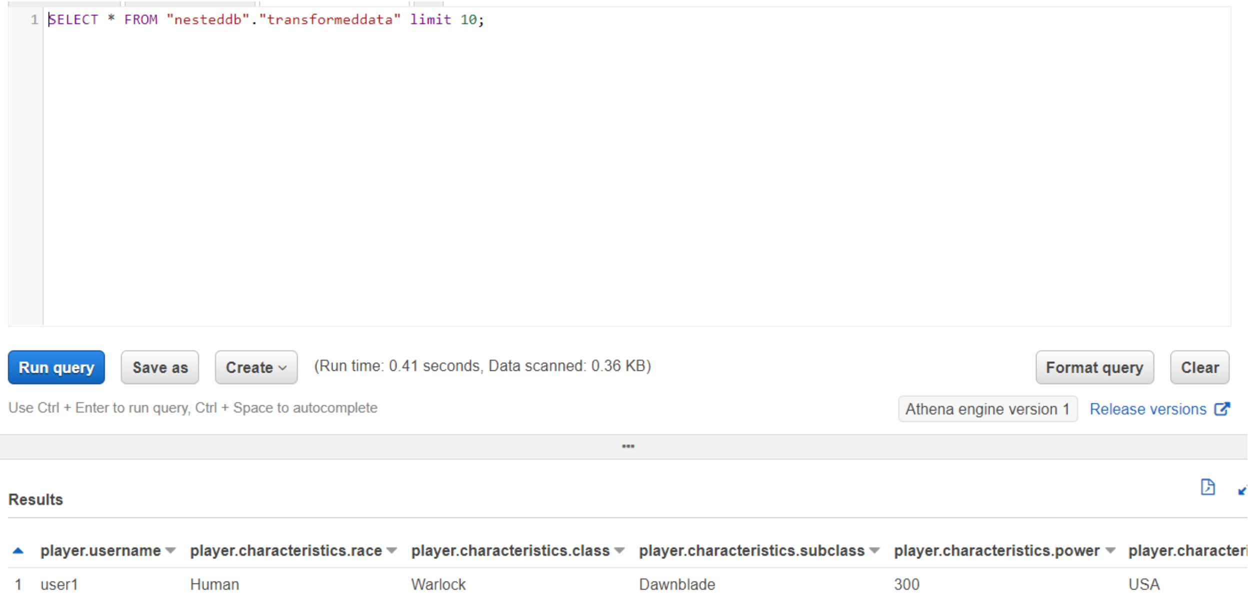
All folders and files contained in the include path are crawled. For example, type s3://MyBucket/MyFolder/ to crawl all objects in MyFolder within MyBucket

21. Choose the same database as created in earlier step i.e. nesteddb
22. Run the crawler.
23. Go to nesteddb and view tables. Click on transformeddata table and view schema.
24. Go to Athena.
25. Expand transformeddata table.

26. Click on Preview Table as shown below.



27. You can now see the result as shown below :



```
SELECT * FROM "nesteddb"."transformeddata" limit 10;
```

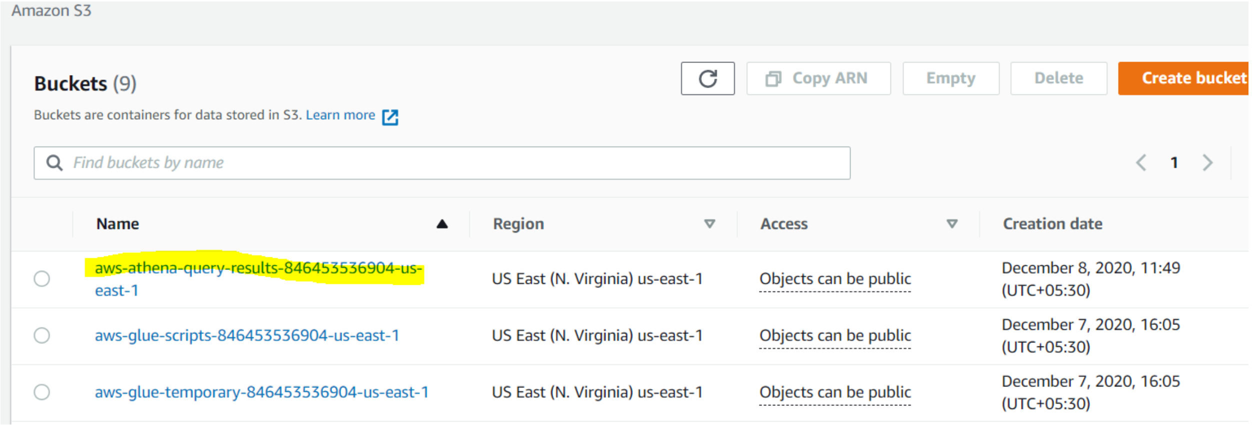
(Run time: 0.41 seconds, Data scanned: 0.36 KB)

Use Ctrl + Enter to run query, Ctrl + Space to autocomplete

Results

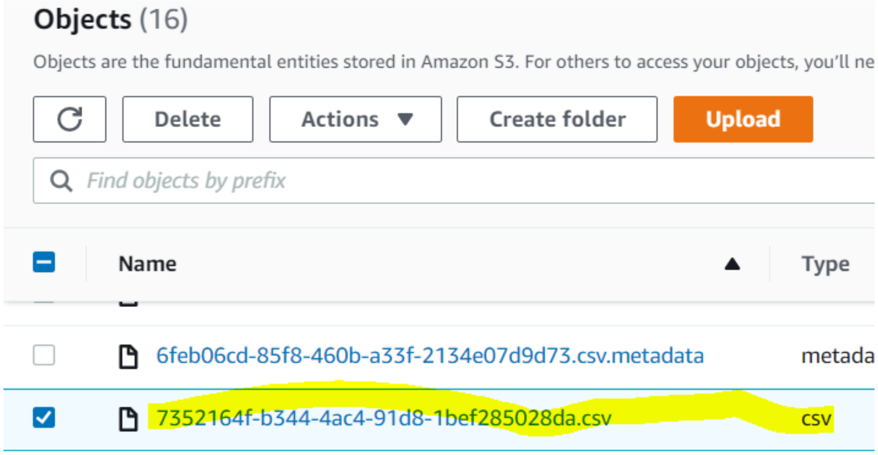
	player.username	player.characteristics.race	player.characteristics.class	player.characteristics.subclass	player.characteristics.power	player.characteri
1	user1	Human	Warlock	Dawnblade	300	USA

28. Navigate to S3 and all query results are written to S3. Observe the folder structure. Data will be available in csv format. Download the csv file and have a look at query result.



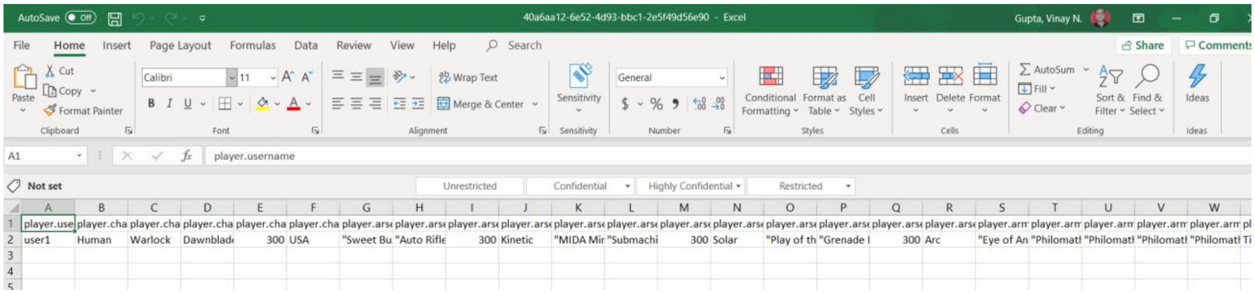
Buckets are containers for data stored in S3. Learn more

	Name ▲	Region	Access	Creation date
○	aws-athena-query-results-846453536904-us-east-1	US East (N. Virginia) us-east-1	Objects can be public	December 8, 2020, 11:49 (UTC+05:30)
○	aws-glue-scripts-846453536904-us-east-1	US East (N. Virginia) us-east-1	Objects can be public	December 7, 2020, 16:05 (UTC+05:30)
○	aws-glue-temporary-846453536904-us-east-1	US East (N. Virginia) us-east-1	Objects can be public	December 7, 2020, 16:05 (UTC+05:30)



Objects are the fundamental entities stored in Amazon S3. For others to access your objects, you'll ne

	Name ▲	Type
□	6feb06cd-85f8-460b-a33f-2134e07d9d73.csv.metadata	metada
☑	7352164f-b344-4ac4-91d8-1bef285028da.csv	csv



A1 | player.username

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	player.use	player.cha	player.cha	player.cha	player.cha	player.cha	player.ars	player.ars	player.ars	player.ars	player.ars	player.ars	player.ars	player.ars	player.ars	player.ars	player.ars	player.ars	player.arm	player.arm	player.arm	player.arm	player.arm
2	user1	Human	Warlock	Dawnblad	300	USA	"Sweet Bu	"Auto Rifle	300	Kinetic	"MIDA Mir	"Submachi	300	Solar	"Play of th	"Grenade I	300	Arc	"Eye of An	"Philomatl	"Philomatl	"Philomatl	"Philomatl
3																							
4																							