Problem Statement

Netflix needs to capture data(structured,semi and unstructured) data for

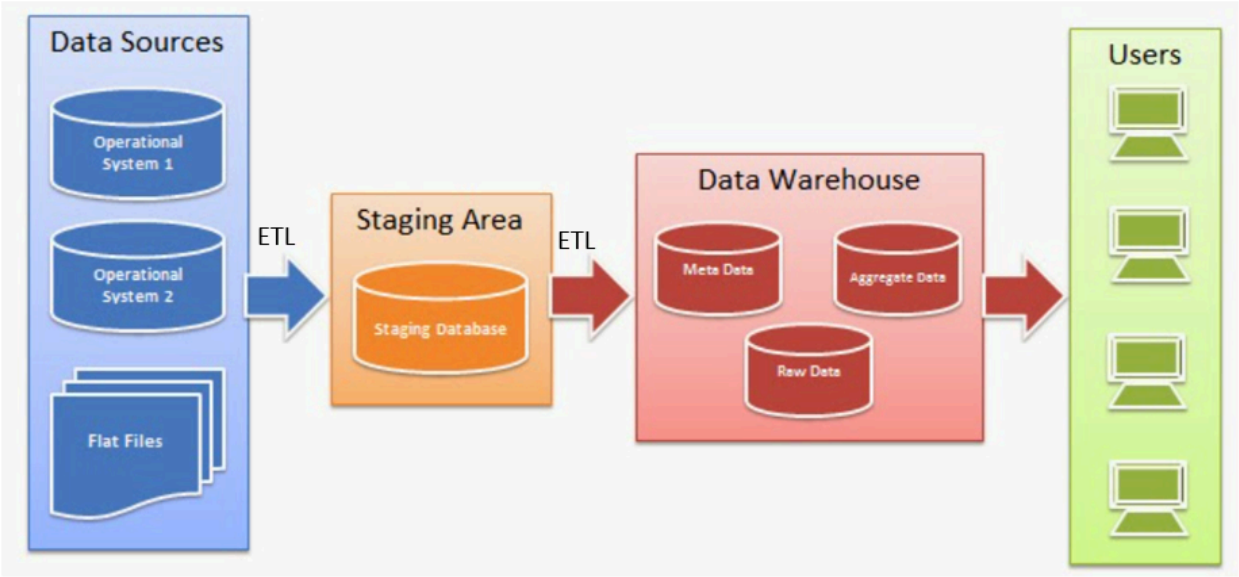
- **Personalization** → Netflix tailors content to make it suitable for the needs for each customer and for the same they need data points to make relevant decisions.
- **Performance analytics** → Netflix analytic team makes decisions around the company with useful metrics, derives insights, to do predictions, and analytic tools to benchmark its performance.
- **Experimentation** → Every product change Netflix considers goes through a rigorous testing process before becoming the default user experience

Existing Problem

As Netflix has famously grown to support more than one third of all internet traffic , the organization has needed to expand its data capabilities to suit.

Netflix started a Data warehouse with MySQL database to store their huge volume datasets which contains 10 years of customer data and billions of ratings, they started facing issues with existing solution
At 2016, Netflix had 60+ PB of data in its datahub and on top of it
> collects and ingests around 100+ TB of data daily
> ~3.5 PB of data process through ETL
> After ETL, writes back 500+ TB of data
Also removes or expires 400+ TB of data

Their existing Data warehousing solutions started giving issues like :



1. **To store all data -** Existing Data Warehouse systems create an ETL Bias, if collected data isn't used to answer specific business questions or in a defined report, it may be excluded from the warehouse as existing solution is for the business specific requirements and drop all the other data collected.

   Let's consider Netflix is collecting data on user activity to answer specific questions on how many users viewed complete video content or what genre has the most viewed content. If in future, they want to extend user experience by providing insights like most replayed watch time for each video content, they need history of data to analyze. But in the existing solution we might have dropped that data.

2. **Staging**– Staging is a storage zone used in ETL to store the intermediate result dataset during the data processing between source(data source) and destination(data warehouse). In the existing data warehouse solution, Staging area would often be in the data warehouse.

   This staging area increases the cost from duplication of data (storing multiple stages of datasets). This also comes with increased cost in processing and maintenance.

3. **No Early Access to Data-** We "Hop" the data through stages creating latency and fragility in processing. Users see the data at the end of the pipe and have usually not visibility thru the pipe, no early access to data and/or insights

4. **Data Variety :** Dealing with huge amounts of unstructured and semi structured data was never mainstream of Datawarehouse.

Netflix gathers every user's such as content rewinds, fast forwards or pauses, rating, search. These come in the form of semi or unstructured data.

5. **Scaling :** Scaling is extension of storage to existing systems. Existing DW solution needed hours and days of downtime.
   Netflix encountered scalability problems when the customer base growth increased along with the data collected from existing users watching more and more shows.

6. **Price**: Supporting and maintaining DWhs are very expensive.
   The immediate impact of enormous growth in data collected leads to the increase in data storage costs

Their Current pipelines need to be revamped to support the above discussed points.

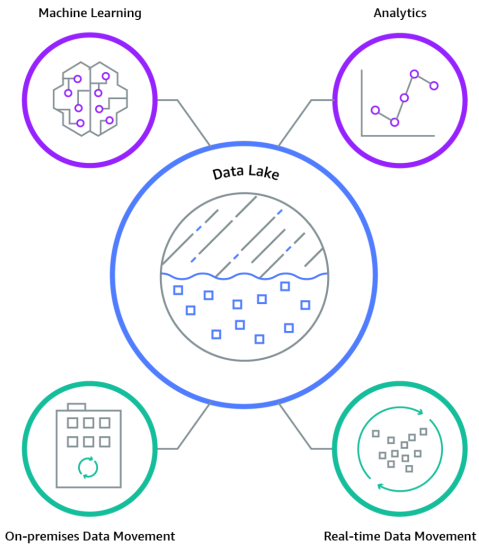
For this Netflix tried multiple options :

a. **SQL DB** → Supports low latency but scaling, storing real time data was challenge
b. **NoSQL DB** → Supports scaling and stream data but latency was high
c. **Hadoop(Hive)** → Supports scaling , low latency but only structure data support
d. **AWS Redshift** → Supports low latency but again only structure data and high cost

Now most of the combination has either

- limited data support
- scaling issues
- price is extremely high

Netflix realized that they can't rely on a single DB or DWh for their requirements.
And developed a **Data Lake to collect and store large volumes of data but also provide a high level view of the data to analyze and explore.**



So what is Data Lake ?

Data lakes are central data repositories used to store any data at any scale.

James Dixon, the person who coined the term "data lake", describes it as

*If you think of a data mart (**a simple form of data warehouse focused on a single area of business**)as a store of bottled water – cleansed and packaged and structured for easy consumption – the data lake is a large body of water in a more natural state. The contents of the data lake stream in from a source to fill the lake, and various users of the lake can come to examine, dive in, or take samples.*

Data lake allows to store any form of data including raw copies of source system data as it is without defining the schema.

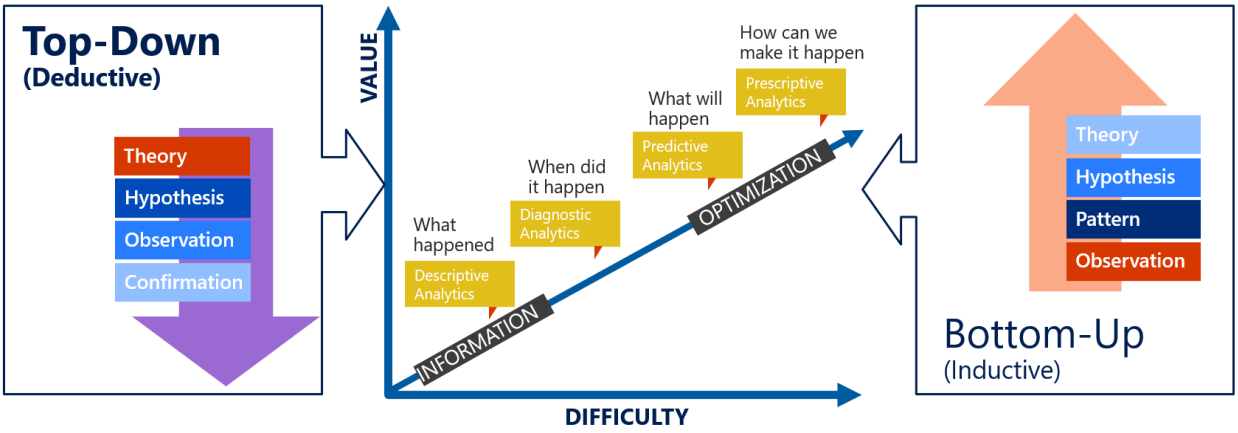
Features of data lake:

1) Inexpensively store unlimited data. Collect all data that “might” be required in future
2) Centralized place for multiple areas of business within organization
3) Easy integration of different format of data
4) No modeling Data storage - ‘Schema on Read’
5) Frees up expensive enterprise data warehouse(EDW) resources for queries instead of using EDW resources for transformation
6) Faster access to data for business users or data scientists (allowing for faster ROI)

7) Data exploration to identify whether data would be valuable before writing ETL and defining schema for relational database
8) It can ingest large files quickly and provide data redundancy
9) Cost savings and faster transformations: **separation of storage and compute resources** allowing multiple compute options like Stream, Batch, Distributed, Pay for Compute.
   It's common for data assets to grow exponentially year over year. However, compute needs might not grow at the same rate. Decoupling storage from compute allows you to manage cost of storage and compute separately, and implement different cost optimization features to minimize cost

How are they different from Data warehouses?

Two Approaches to getting value out of data: Top-Down and Bottoms-Up



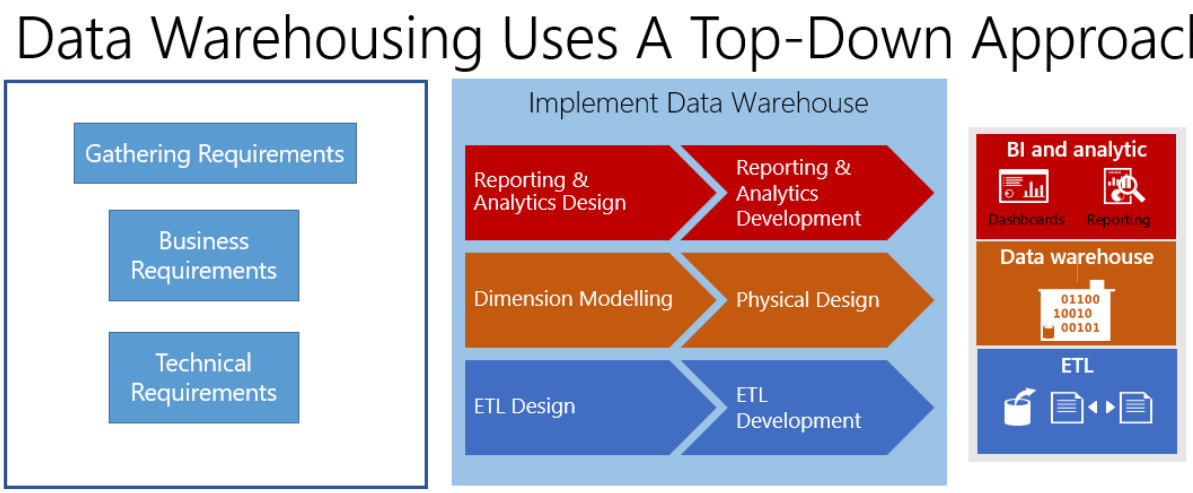
Top-down (deductive approach) -

- This is where analytics is done starting with a clear understanding of corporate strategy where theories and hypotheses are made up front.
- The required data model is designed and developed prior to any data collection.
- Top-down approach is meant for descriptive and diagnostic analytics.

Let's consider, In our case study if we have a clear business requirement like identifying top viewed content by Region and If we start the process from the requirement , it is top-down approach

The **Data Warehouses** leverages the top-down approach where there is a **well-architected information store and enterprise wide BI solution**.

1. Building a data warehouse starts with defining the organization's business first.
2. Next process is to gather business and technical requirements for the data warehouse.
3. This is followed by dimension modeling, ETL design and followed by the development of the warehouse.
4. Above steps are performed before collecting any data.
5. It utilizes a rigorous and formalized methodology because a true enterprise data warehouse supports many users/applications within an organization to make better decisions.



Bottom-up (inductive approach) -

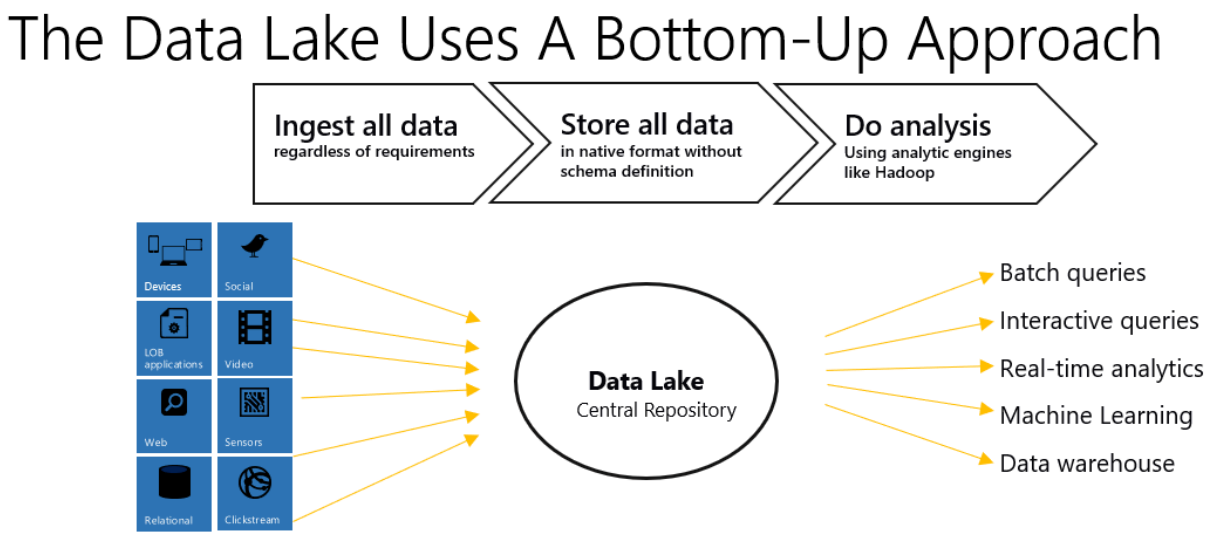
- This is the approach where data is collected up front before any theories and hypotheses are made.
- All the data is stored to derive the patterns and conclusions from the data itself.
- This dataset is likely for advanced analytics such as predictive or prescriptive analytics.

Let's consider, In our case study we need to collect all user activities or events related data and later we need to identify what can be done with the data or based on the data pattern identifying new insights is a bottom up approach.

The data lake supports a bottoms-up approach.

1. **A data lake is a central repository for every type of data collected and stored in a single place**.
2. Data of all types would be stored in the data lake before concluding any business requirements or schema for operational and exploratory analytics.

3. Advanced analytics can be done, or act as a lower cost data storage before moving cleansed data to a data warehouse.
4. In these cases, users load data into the data lake before defining any ETL logic.



To understand this we need to understand difference among them, they are listed below:

Features	Data Lakes	Data Warehouse
Data Variety Supported	Any kind of Data	Only Structured
Data Quality	Any data that may or may not be cleansed (raw data)	Highly cleansed data that serves as a central version of truth
Data Size	Usually in PBs	Usually in TBs
Schema Definition	Written at the time of analysis(schema on read)	Designed prior to DW implementation(schema on write)
Processing Type Supported	ELT	ETL
Cost	Designed for low cost storage. Pay only for the occupied space. This makes scalability easier	Expensive for large data volumes along with scaling
Users Type	Mostly Data scientists	Business Professionals like BA and

		DA
Use cases	Batch Processing, Data Cleaning, ETL Workloads, Backup/Archive Data, Sandbox for data exploration, One-time reports, Quick Access to data	Additional Security, Large support for reporting, on the need of complex joins
Example	Netflix decided to store, All **Operational data** to be fed into the **data lake**. Operational data is of business events such as user clicking on title, Pausing a movie.	Events **Processed Data** into **Data Warehouse.** For faster interactive analysis on subsets of processed data. Processed data is a cleansed form of raw data.

Why Netflix choose data lake

- **Cost** - As storing the data in a data lake was about 100 times **cheaper** than in traditional data warehouses along with faster data **scalability**
    - Data lake(s) are designed for a **low-cost commodity service**. A warehouse take a long time to build and comes with additional expenses
    - Effectively **eliminates the "Staging Area"** associated with Data Warehouse.
    - **Polyglot** - Allowing that not everything has to be in SQL as a cost reduction opportunity.
    - Separates Compute and Storage
- **Overcome the Traditional Analytic Solution Shortcoming**
    - Facilitate Agility and Opportunity - Experimentation and Exploration
    - More easily integrate larger datasets as then Industry Trends are moving away from relational only systems
- **Single source of truth** - In data warehousing, teams own multiple data warehouses based on requirements.
  Let's consider Netflix owns Sales Data warehouse to analyze business performance and Events data warehouse to analyze user activities.
  Here there is no single source of truth. Data lake acts as a central repository enabling **data unification and can connect all key datasets.** Stores data that can be shared across organizations.
- **Simplicity -** Eliminates the necessity of data modeling, allowing the data lake to store any form of data
- Since there is **no schema defined**, it allows netflix to **ingest data faster**.
- **Real time Decision Analytics** - Being able to find all data in one place involves a deep learning to provide more insights

As clearly highlighted, data warehouses can only manage only and only **structured** data but Netflix needs other types of data so DWh is not a good fit and data lake will take precedence here.

Some Success Stories of Data lakes across globe

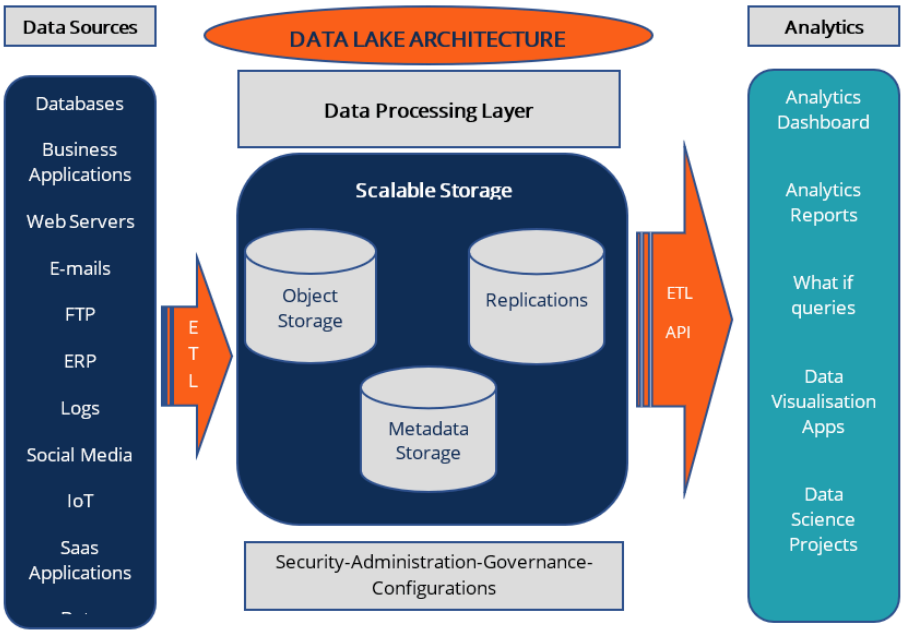
Yulu improved service efficiency by 30–35% using its prediction model and AWS data lake. Services used by yulu on AWS:

- RDS
- S3
- EMR

Broadcom modernized their data lake which increased the company's enterprise agility and translated to a reduction of 25% in monthly support calls. Services used are:

- Dataproc
- Cloud SQL
- Bigtable

What are the components involved in Data Lake Architecture



Let's explore these components in detail and see how this will fit in Netflix CL application

Data Sources :

- Data Sources are the providers real time business data to data lake
- Incoming data usually consists of **time series, messages, events**, db,FTP etc, highlight one data comes at high velocity

Data lake system:

- Provides a centralized storage for the internal data of an organization.
- The Data lake system comprises Datastore(Object Storage), Metadata store(Metadata Storage) and the Replication to support the High availability (HA) of data.

Object Storage

Manages data as objects. Unlike traditional file storage addressing hierarchy storage, object-based storage utilizes a ***flat file system*** that has no built-in limits. So, it can hold large volumes of unstructured data such as audio, video, emails, health records, and documents. Object storage does put data into files and folders.

Metadata Storage

Provides properties about the object and also describes how an object should be accessed

Replications

Data Replication is replicating objects across multiple locations of data lake for **disaster recovery** and considered as a backup of data

Data Processing layer

The retrieved ingested data is processed to produce information that is applicable to the business. This process can be machine learning too. Data may be converted to intermediate form retaining raw data.

Data lake analytics layer

- on-demand analytics on large volumes of data
- generates valuable insights from data, without pre-processing and organizing data in complex infrastructure

Organizing the Data Lake

- **Raw data (Bronze)** – Data ingested from the data sources in the raw data format, which is the immutable copy of the data. This can include all forms of data objects such as databases, JSON, CSV, XML, text files,backups, archives, or images.
  In Netflix, data collected directly from the users such as business events would be categorized as raw format

- **Curated (Gold)** –The transformed data can be further enriched by blending it with other data sets to provide additional insights.
  In Netflix, Processed data like aggregated sales performance over years are in curated zone

- **Dev or Work zone(Silver)** – This zone can be organized by user, by project, by subject, or in a variety of other ways. The data moves to the gold zone, when the development performed in the work zone gets productionized.

How to create a Successful Data Lake

Aligning with the company's business strategy and requirements is a must to create a successful data lake. Key prerequisites are classified below

1. Choosing the right platform

Setting up the infrastructure to form a data lake.

But based on the organization requirements the infrastructure may be changed. Below are some factors to be considered while choosing a vendor.

- Scalability
- Cost
- Security

- How tightly data lake can be coupled with existing data architecture of the organization.

Popular data lake technology providers include the following:

- **Amazon Data lakes** – Offers unlimited scalability
- **Apache** – Uses Hadoop open-source ecosystem
- **Google Cloud Platform (GCP)** – Google cloud storage
- **Oracle Big Data Cloud**
- **Microsoft** Azure Data Lake and Azure Data Analytics
- **Snowflake** – Process structured and semi-structured data

2. Understanding how the data stored in data lake will be used

While building a data lake, Identify and Define the Organization's Data Goal.

The practice of **collecting data without having a clear goal might make the existence of data irrelevant**.To avoid this, Start with what data the organization needs to collect and its business objective.

A well-organized data lake can get easily transformed into an unmanaged Data Lake that is either inaccessible to intended users or provides little value when companies don't set parameters about what kind of data they want to collect and why.

3. Understanding the Importance of Data Governance

Unlike a data warehouse, in a data lake you can store your data as-is without having to structure it first. This has resulted in **"dumping" lots of data into data lakes in an uncontrolled and thoughtless manner**.

This leads to an unmanaged Data Lake that is either inaccessible to intended users or provides little value

Data governance is a set of practices to **manage the data in high quality during its life cycle, from acquisition to use to disposal**.

Data quality data is a key to empower data-driven experiences

The absence of these protocols might lead to data getting dumped in one place with no clear idea on how long and why data is required.

How to ensure data governance in data lake

- **Data ownership and accountability**
    - Developing policies based on the organizational structure regarding **How the data will be used and who can access it.**

- **Ensuring Correct Metadata For Search**
    - Correct Metadata will enable **Easy data discovery** which helps data scientists, data analysts, and data engineers to quickly discover and reference relevant data and accelerate time to value.
- **Data lineage**
    - Data lineage to get end-to-end visibility into how data flows in the lake from source to consumption.

How to form a Data lake with AWS Components

Constructing a data lake and choosing supporting components and platforms are based on the organizational and business requirements.

Let's consider below given AWS components and build a data lake based on these.

1. DynamoDB - a NoSQL database offered by AWS. DynamoDB stores data in tables.
2. AWS Glue - data integration service that helps to discover, prepare, and combine data for analytics, machine learning, and application development.

    a. AWS Glue Data Catalog - metadata store that provides a uniform repository where disparate systems can store and find metadata to keep track of data. ETL jobs that will be defined in AWS Glue use Data Catalog tables as sources and targets.

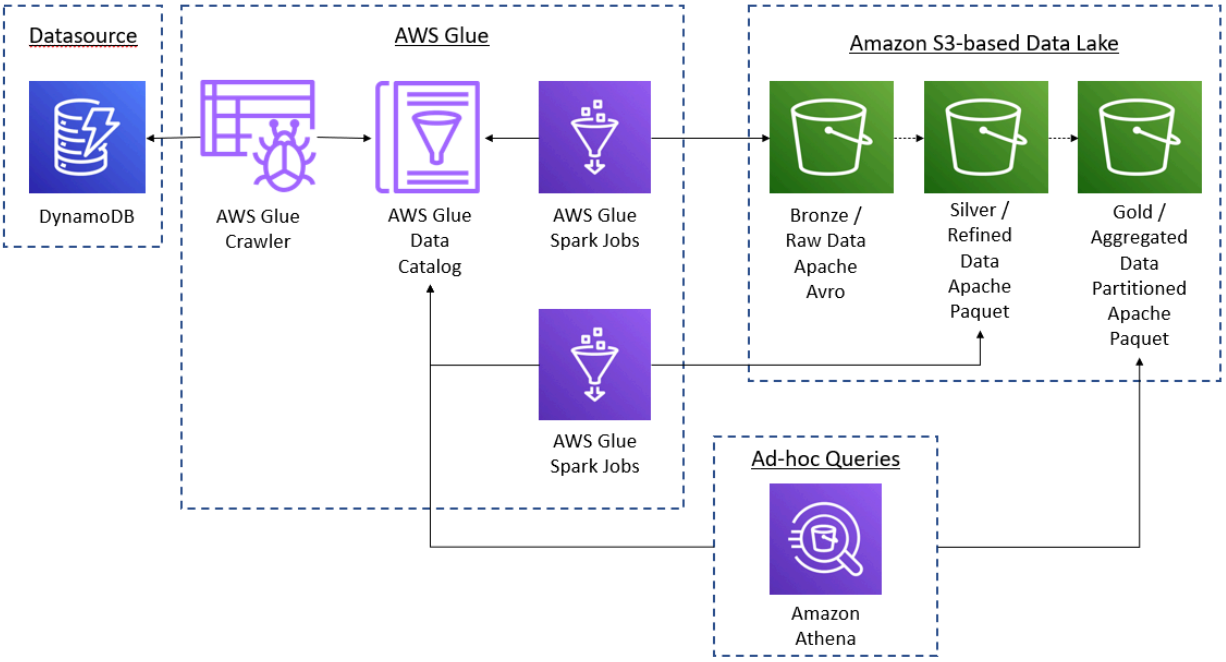
    b. AWS Glue Crawler - crawlers that can scan data in all kinds of repositories, classify it, extract schema information from it, and store the metadata automatically in the AWS Glue Data Catalog.

    c. AWS Glue Spark Job - A ETL Spark job runs in an Apache Spark environment managed by AWS Glue. It processes data in batches.

3. Amazon S3 - **Amazon Simple Storage Service(S3) is a cloud based object storage service architecture.**
    a. S3 allows the decoupling of storage and compute whereas **S3 is only a storage unit**
    b. S3 Features:

i. object storage service that offers industry-leading scalability
ii. provides 99.999999999% durability
iii. 99.99% availability with strong consistency
iv. unlimited data storage globally

4. Amazon Athena - Amazon Athena is a service that helps data analysts to perform interactive queries in the S3 on large-scale data sets.



Designing a data lake is not just a single solution. As expectation from data lakes is to store any sort of data, designing of Data lakes includes below components:

Setting up storage:

- To store data from source as is like that in file with different formats like avro,json, parquet and ORC formats for data storage and processing
- **Amazon S3 as a data storage area and stores below forms of data**
    - **Raw data** – Data is ingested and kept as close as possible to its original state.
    - **Transformed** – In this stage, data can be transformed into columnar data formats, such as Apache Parquet and Apache ORC, which can be used by Amazon Athena.
    - **Curated** – This layer typically contains S3 objects which are optimized for analytics, reporting using Amazon Athena, Amazon Redshift Spectrum, and loading into massively parallel processing data warehouses such as Amazon Redshift.

Moving data:

- Data ingestion is to move your existing data from data sources into a centralized data lake.
    - **Glue crawler crawls over one or multiple source to fetch and populate the AWS Glue Data Catalog with tables**
    - Glue Crawler also runs over data and infers schema. The schema then gets registered in the glue data catalog.

Preparing and cataloging data:

- One commonly seen challenge in the data lake architecture is the lack of knowledge on how the contents of raw data stored in the data lake would be used
- Organizations need governance, semantic consistency, and access controls to avoid the pitfalls of creating a data swamp with no curation.
    - **Glue Data Catalog - metastore of information** about the dataset like table names, column names, data types, schema
    - **Glue Spark jobs to prepare** the data and to perform **transformations** on the data

Configuring security policies

- Securing a data lake requires fine-grained controls to ensure authorized data access and use.
    - **By Default, all aws services provide security services**

Making data available for consumption

- Once data is prepared, it can be analyzed which ranges from simple reports of visualizations to advanced analytics and machine learning.
- One can use any purpose-built analytics services such as **Amazon Athena** to run ad hoc queries without complexity.

Implementation

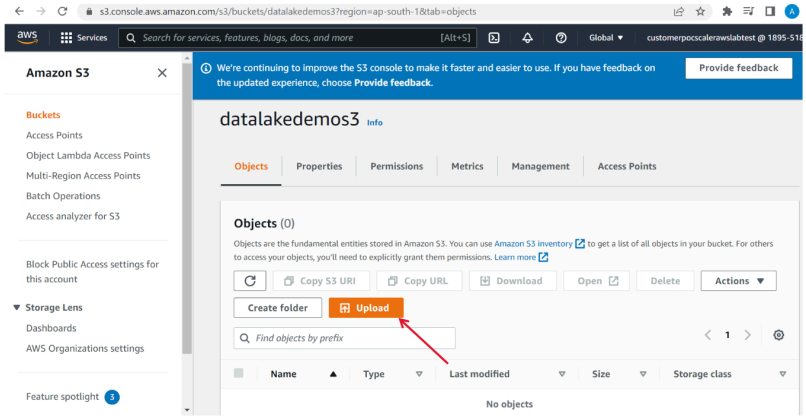
Let's implement the above given data lake environment for one of the datasets in Netflix.

Consider a dimensional dataset - Netflix_show which consists of information about Netflix shows like Show name, director, cast and below given demo briefs how to move data from data source to data lake environment.

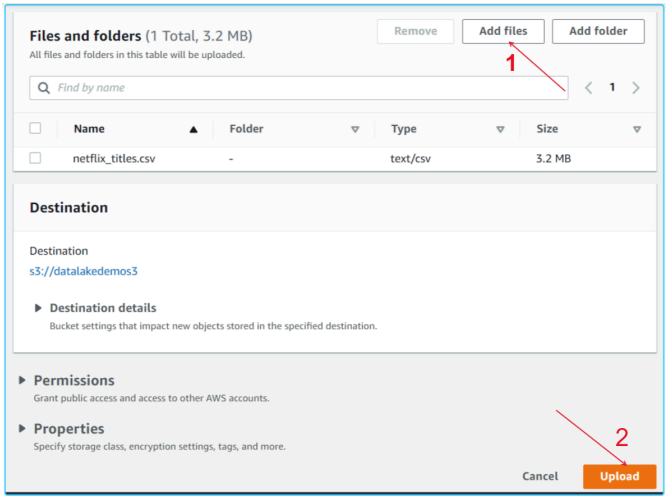
On the assumption that, DynamoDB is a data source environment which is directly connected to a web app and stores netflix show related data.

[For Instructors,By Default no table would be available in DynamoDB, To manually import bulk data from .csv file to DynamoDB, Follow below steps

1. Sample dataset has been uploaded in drive, Download and keep it in your local machine
2. Goto S3 console, click Create Bucket
3. Specify Bucket Name and click Next(Leave Other settings as it is)
4. Now you will be automatically redirected to the Home page and you can find a newly created bucket at the bottom of the page.
5. Click on your bucket
6. Under Objects, You can find Upload. Click on it

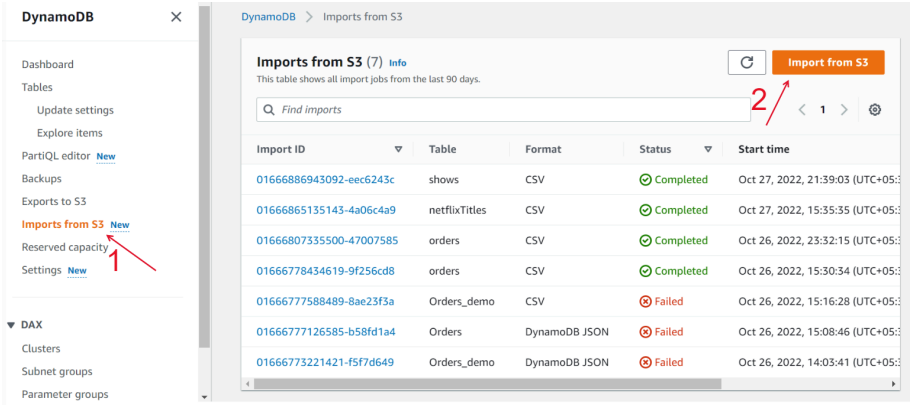


7. Select Add Files, and add the downloaded file from step1
8. Click Upload

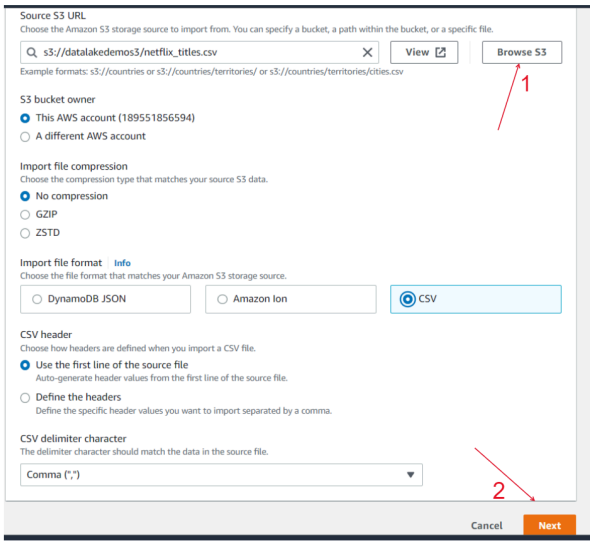


9. After Successful upload, Goto DynamoDB Console

10. From the left pane, select Import from S3
11. From the right pane, select Import from S3



12. Click Browse S3 and select the uploaded .csv file from step 8
13. Under Import File Format, select csv and leave other options as it is and click next



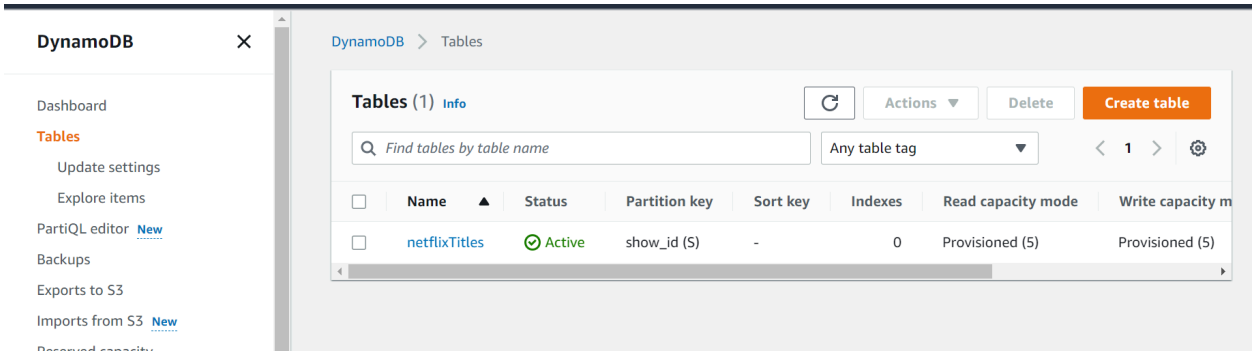
14. In the next tab, feed **netflixTitles** as Table Name and **show_id** as Partition Key and leave other settings as it is and click next.
15. Review all items and click **Import** and Verify whether import is successful.

]

DynamoDB

1. Go to Service DynamoDB in AWS Console
2. From the left pane, Click on the Tables

3. Validate netflixTitles table is available

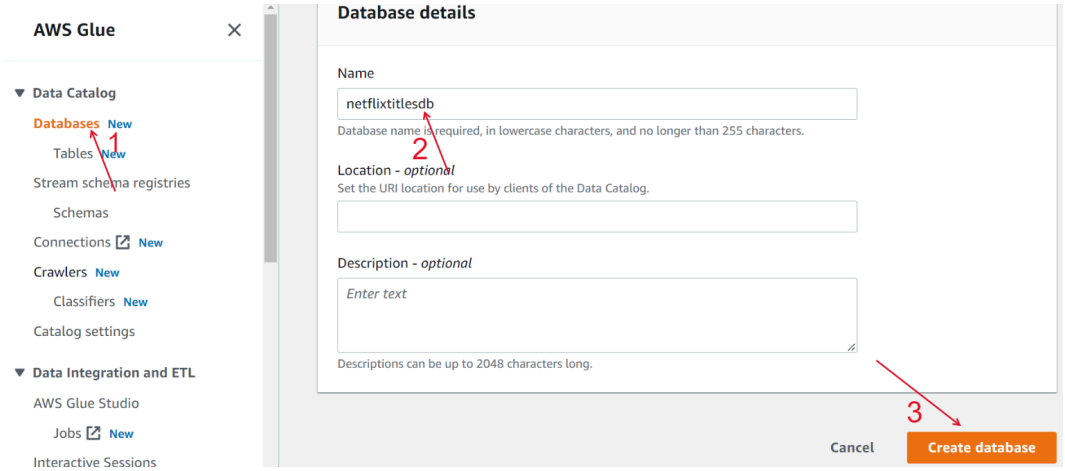


AWS Glue

1. Go to service [AWS Glue service]

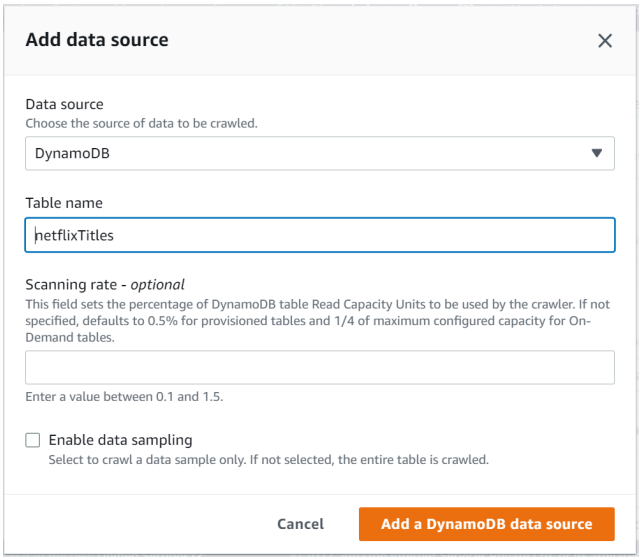
Data Catalog and crawlers

2. To create database, From the left pane, under Data Catalog click on **Databases**
   a. A database in the AWS Glue Data Catalog is a container that holds tables
   b. Select **create database** and provide **netflixtitlesdb** as name and click **Create Database**

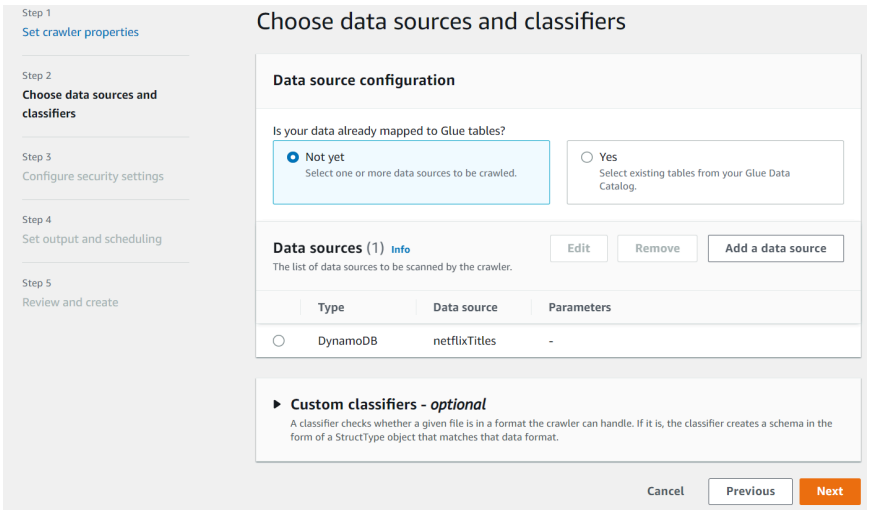


3. To create Crawlers, From the left pane, under Data Catalog click on **Crawlers**
   a. Select **Create crawler,** specify **netflixTitles** as crawler name and click next
   b. You will be navigated to next tab **Choose data sources and classifiers**, Don't change anything under Data source configuration
   c. Under Data Sources, select **Add a data source**
   d. Choose DynamoDB as a data source from drop down list

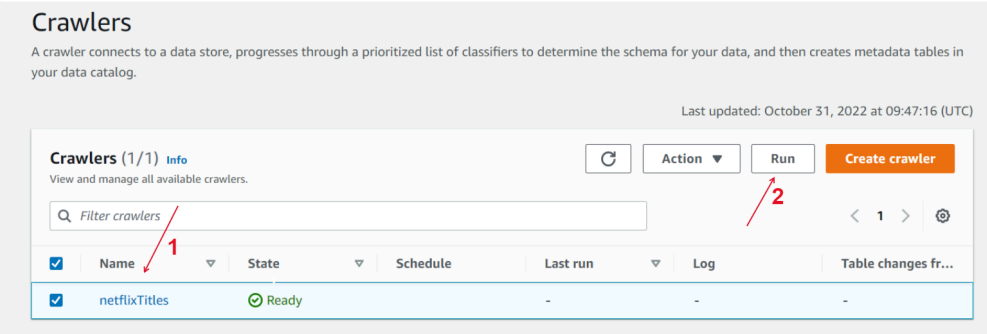
e. Feed Table Name as **netflixTitles**



f. Select **Add a DynamoDB data source** and click Next



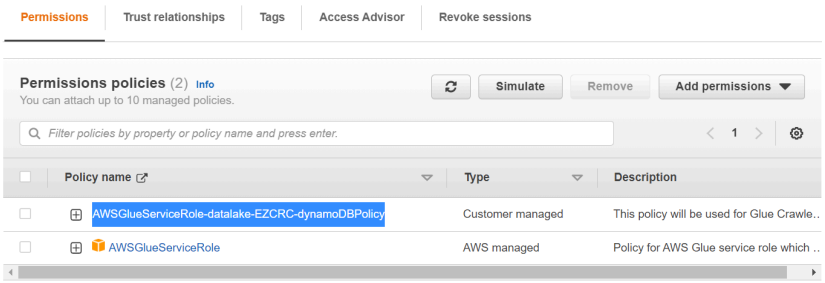
g. In the next tab Configure security settings, choose **Create new IAM role**
h. Specify IAM role name as **AWSGlueServiceRole-datalake** and click Next
i. In the next tab, Under Target Database choose the previously created **netflixTitlesdb**. Click Next
j. Review all the settings and select **Create Crawler**
k. To Verify crawler has been successfully created, select **netflixTitles** crawler and click run and validate whether run is successful.
l. After a successful run, data is imported into table



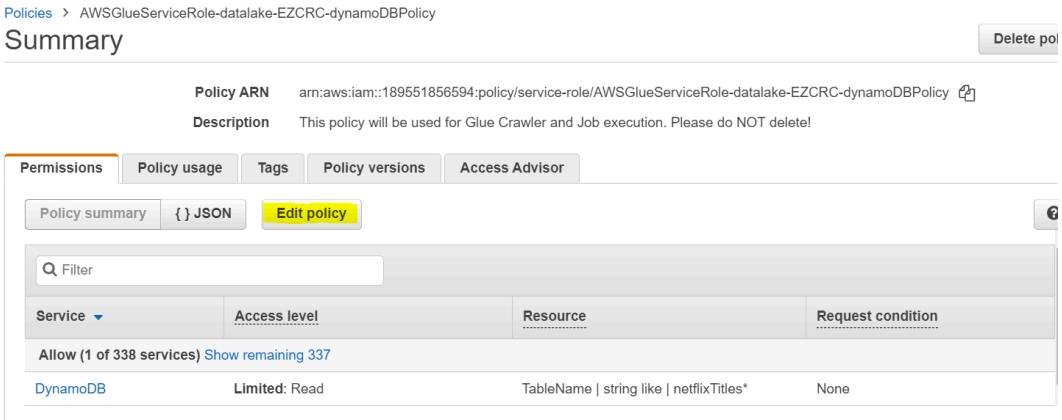
IAM

Without closing AWS glue, In another tab Open IAM dashboard

1. From the left pane, click **Roles**
2. Click on the previously created role **AWSGlueServiceRole-datalake**
3. Under Permissions Policy, click **AWSGlueServiceRole-datalake-EZCRC-dynamoDBPolicy**

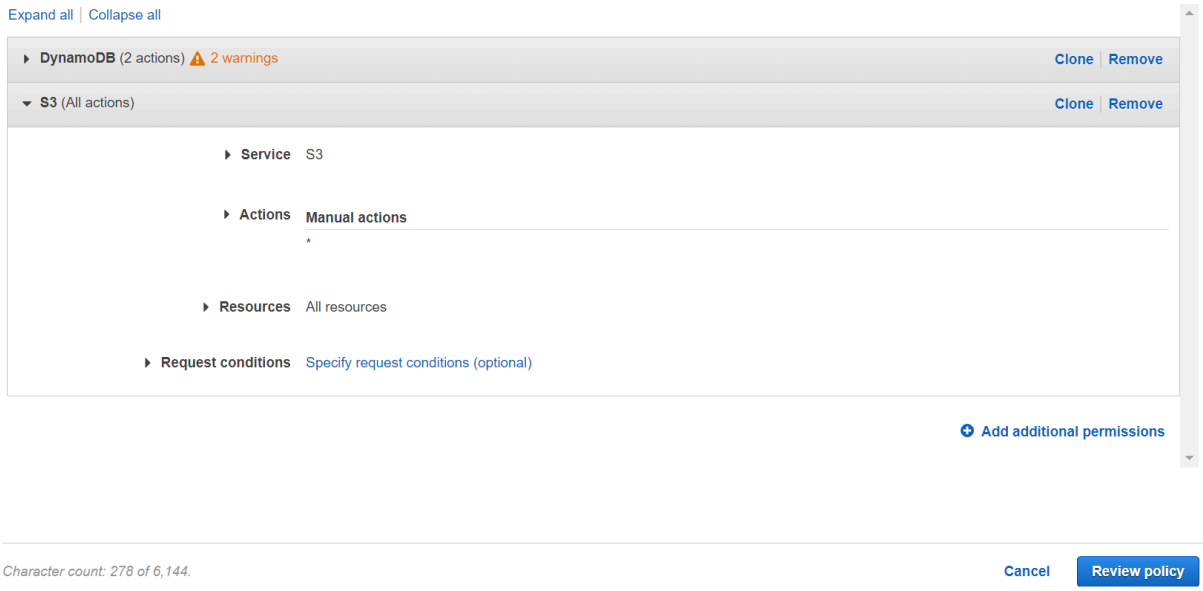


4. Click on **Edit Policy**

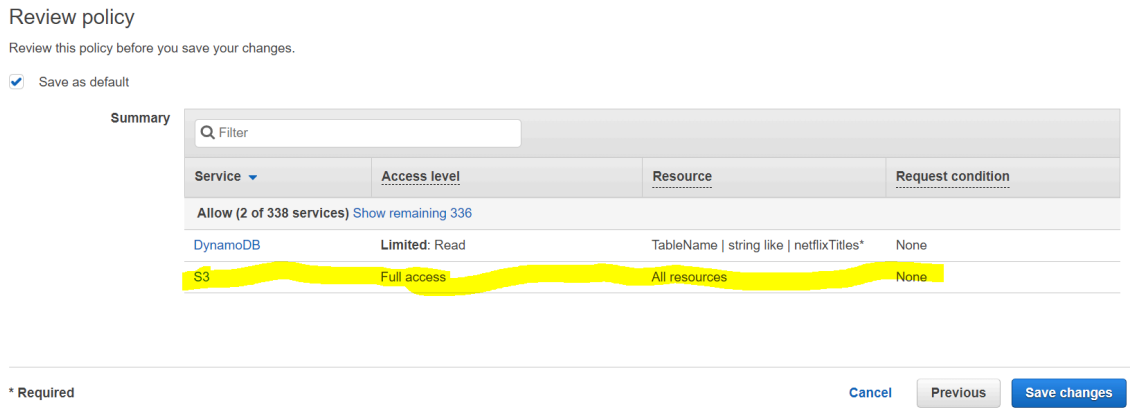


5. select **Add Additional Permissions**
6. Select S3 as **Service** from choose a service list
7. Under Actions select **All S3 actions (s3:*)**
8. Expand the resources and choose **All resources**

9. Click **Review Policy**



10. You should see the changes in the summary and click **Save Changes**



S3

To segregate dataset under raw, curated and dev category, we need to create 3 different buckets.(Note1: Buckets are containers for data stored in S3 Note2: Incase, if the business requirement does not demand for all these categories, we can omit and follow business or organization strategies)

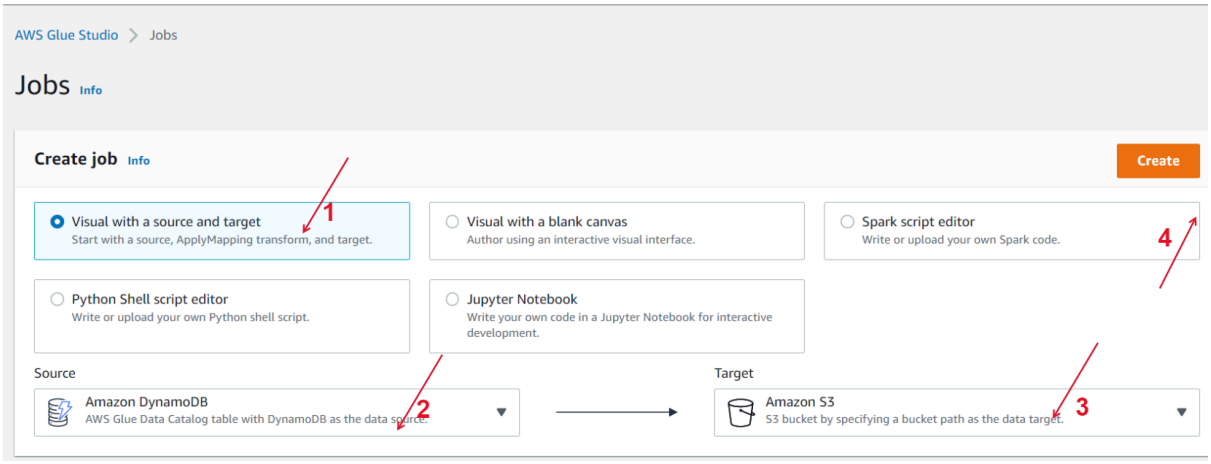
1. Open S3 Console
2. Click on **Create Bucket**
3. Specify **netflixraw_<userId>** as Bucket name (All users should have unique name)
4. Leave other settings as it is and select **create Bucket**

5. Repeat above steps from 2 and create two more buckets **netflixdev_<userId>** and **netflixcurated_<userId>**

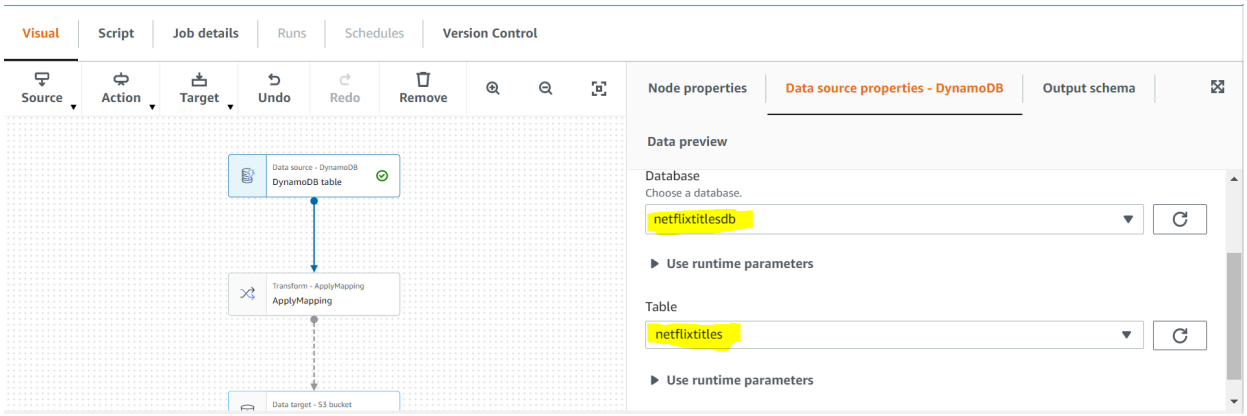
AWS Glue Studio

Go to AWS Glue console, select **AWS Glue studio** under Data Integration and ETL from the left pane

a. From the left pane, Select Jobs
b. In the Jobs dashboard, select **Visual with a source and target** under create job
c. Set DynamoDB as source and Amazon S3 as target
d. Select **create**

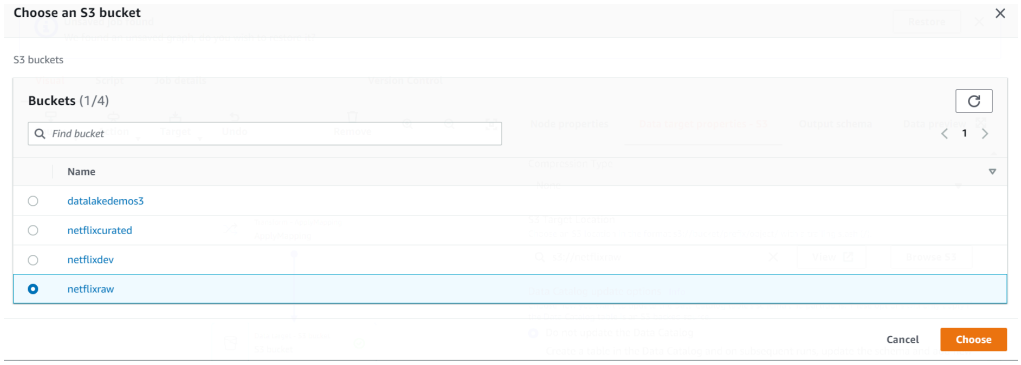


e. A visual representation of job is displayed
f. Click on the first box **DynamoDB Table**
    1. On the right pane, under **Data Source properties - DynamoDB**, select Database and choose **netflixtitlesdb** from the drop down list
    2. Select **netflixtitles** under Table
    3. You should find a green tick next to DynamoDB table in the designer to validate the changes

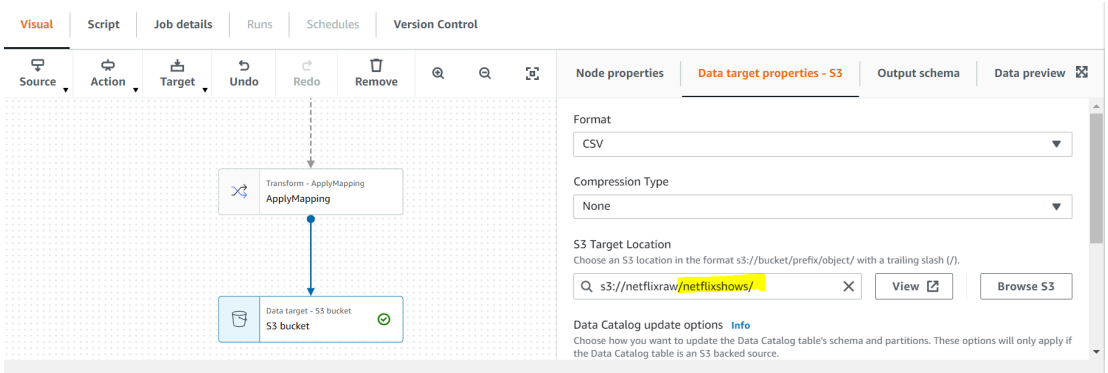


g. Click on the S3 Bucket box

1. On the right pane, Under **Data Target properties - S3**
2. Choose output format as CSV
3. Under S3 Target Location, click on **Browse S3** and click on round button - **netflixtitles** under **netflixraw** Bucket and select **choose**

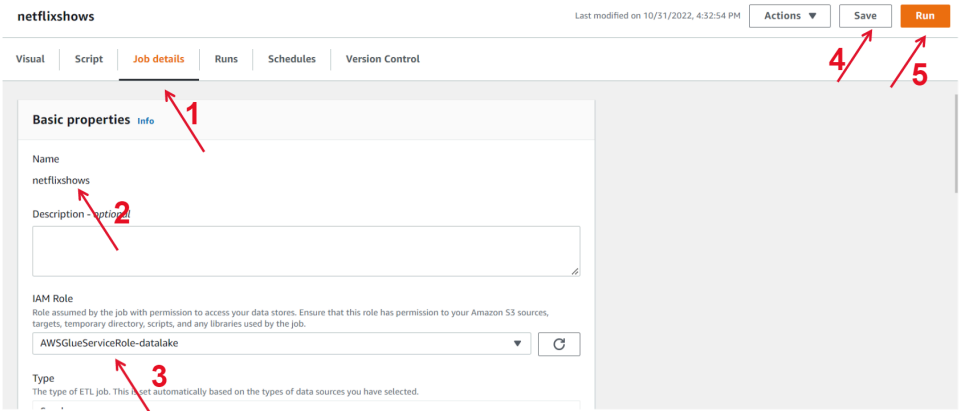


4. You can edit S3 Target Location to add folder name **netflixshows/** to create during run



h. Provide Job Details

1. Choose **Job Details** from the tabs
2. Specify the **name** for the job as **netflixshows_raw**
3. select **IAM role** from the drop down
4. Click on **Save** from top right pane
5. Now Run Option would have been enabled, Click **Run** and test the job

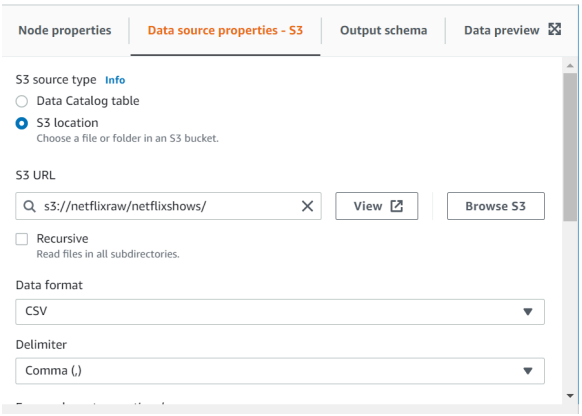


After a successful Job run, Validate the data in S3 location. You will find multiple objects under the S3 location /netflixraw/netflixshows/ created by Glue ETL spark job.

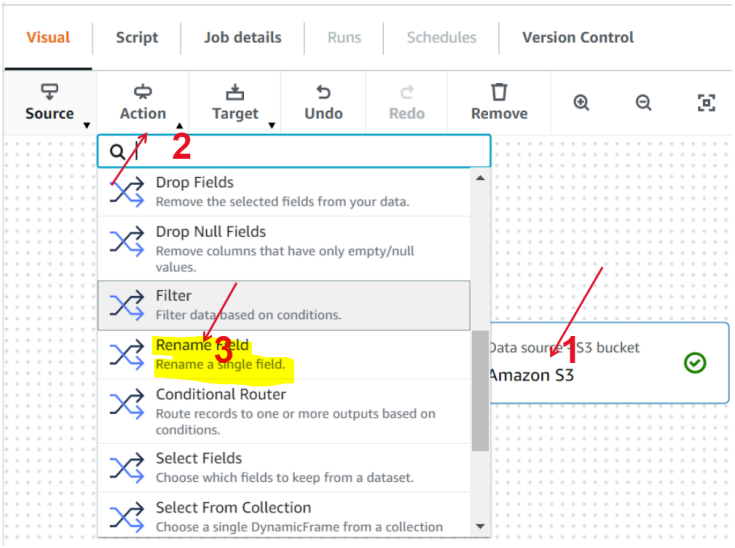
Let's create a new job **netflixshows_dev** to perform data movement from raw to dev S3 bucket.

Let's move data from raw to dev bucket with the first level of transformation of data. Let's rename a column **type** to **show_type** in netflixtitle dataset

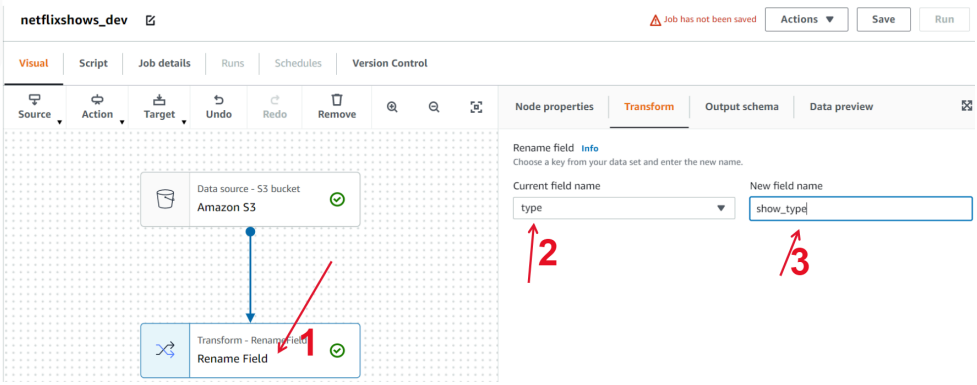
1. In the AWS Glue Studio, Create a new job and select **visual with blank canvas** and click **create**
2. Select S3 as Source from source tab in visual editor
3. On the Data Source properties, specify S3 source location as **netflixraw bucket** and add folder name **netflixshows/**(Previous job **netflixshows_raw** target S3 location is data source location here). All the objects under this location will be considered as source
4. Select csv as data format and leave other options as it is



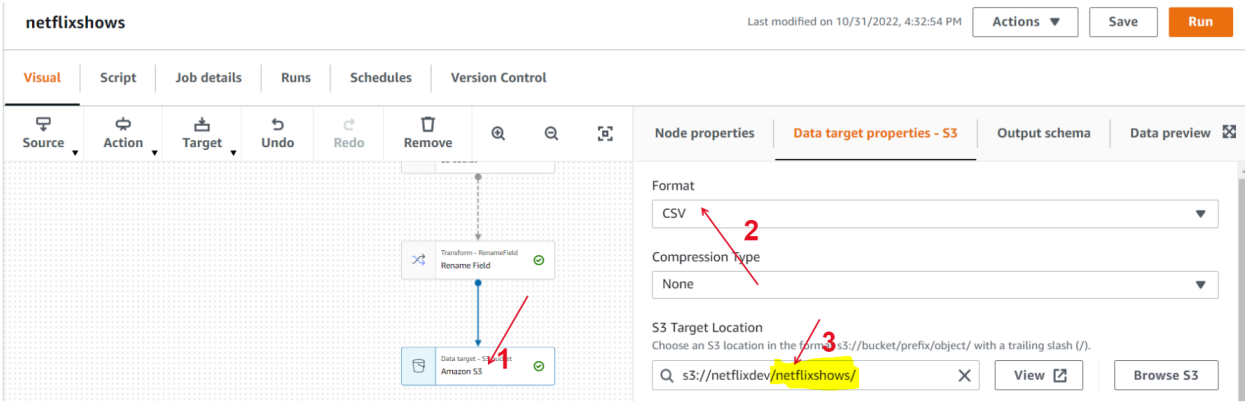
5. Click on the **Data Source node** and click Action and select **Rename Field** from the list



6. Now the new job will be displayed in the visual editor.
7. Click on the **Transform - RenameField node**
    a. Under Transform Tab, select type as current field name and show_type as new field name
    b. Now green tick should be shown next to the Transform node



8. Click on the **Transform - RenameField node** and click **Target** and Select **Amazon S3** from the drop down list
9. Select **Data Target - S3 Bucket** node
    a. As already demonstrated, add **netflixdev bucket** as target S3 target and add folder name **netflixshows/**



Now save the job and run it.

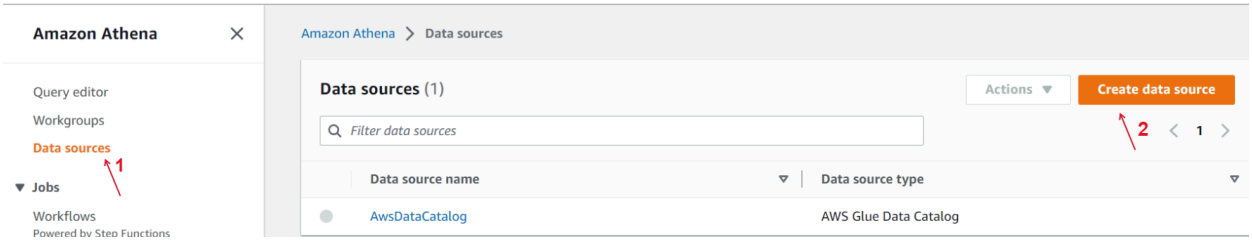
Task: Create a new job to move data from dev to curated bucket. Perform Filter transformation to filter only records where release_year > 2019.

[Hint: Change data type to int (action type - Change Schema) of release_year before filtering (action type - filter)]

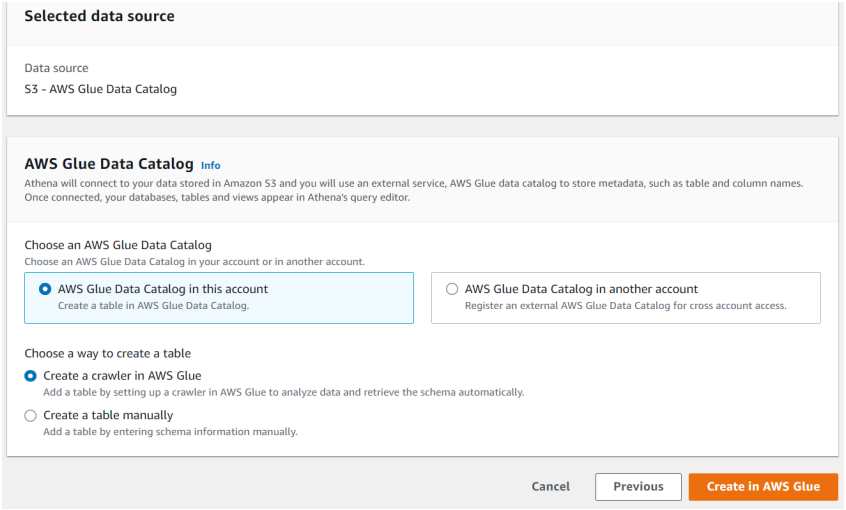
AWS Athena

Read and query S3 data from Athena

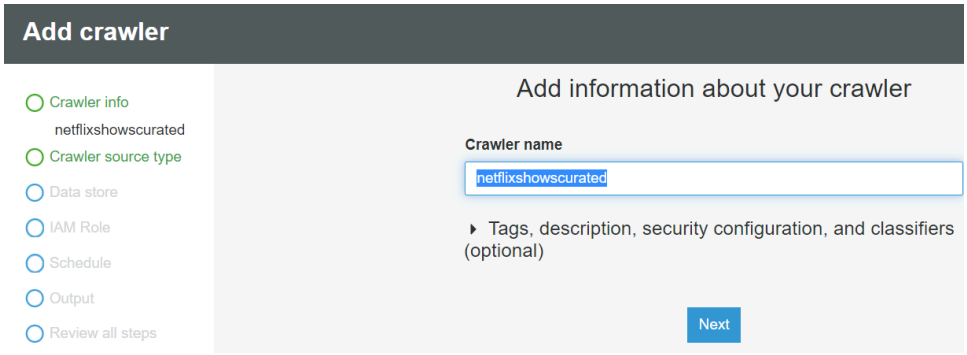
1. Open Athena console
2. On the left pane, select **Data Sources** and **Create Data Source**



3. Select **S3 - AWS Glue Data Catalog** and Click **Next**
4. Choose **AWS Glue Data Catalog in this Account** and click **Create in AWS Glue**

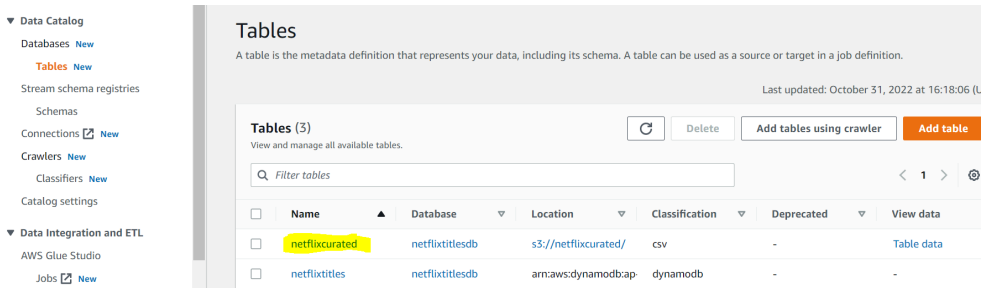


5. Provide **netflixshowscurated** as Crawler Name and **click Next**

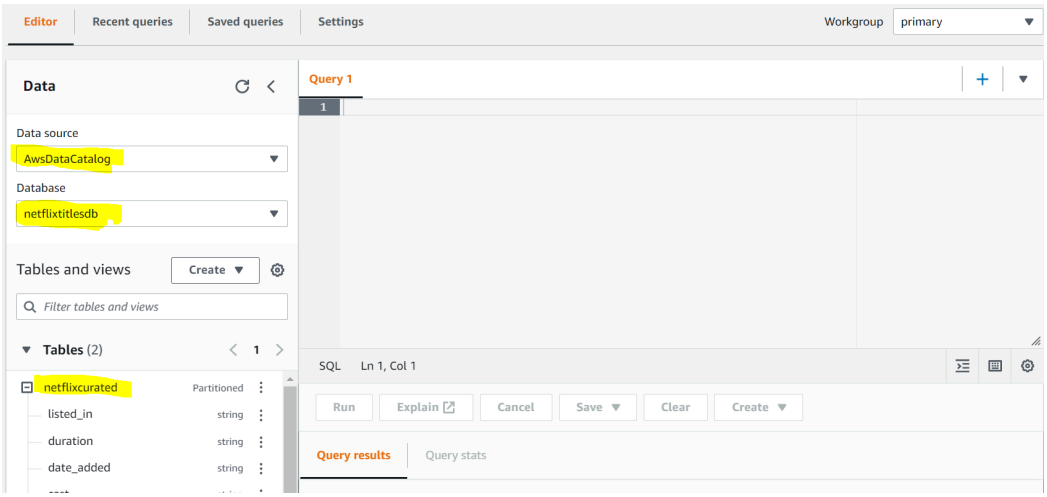


6. In the next tab **Specify crawler source type**, don't change anything and click **Next**
7. In the next tab, Under **Include Path,** Browse S3 and select **NetflixCurated** Bucket and click **Select** and click **Next**
8. Under **Add Another Data Source**, select **No** and click **Next**

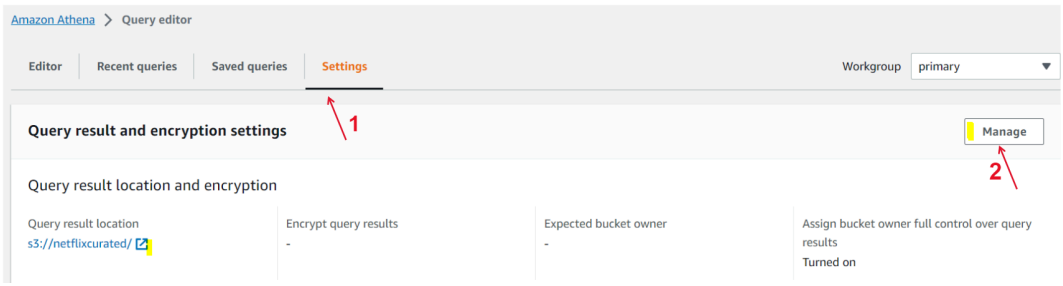
9. Under **Choose IAM Role,** select **Choose an Existing IAM Role** and select **AWSGlueServiceRole-datalake**
10. Under Frequency, choose **Run on Demand** and click **Next**
11. Under Database, select netflixtitlesdb and click **Next**
12. Review settings and create crawler
13. Run the crawler and after successful run you should see **netflixcurated** under tables in databases



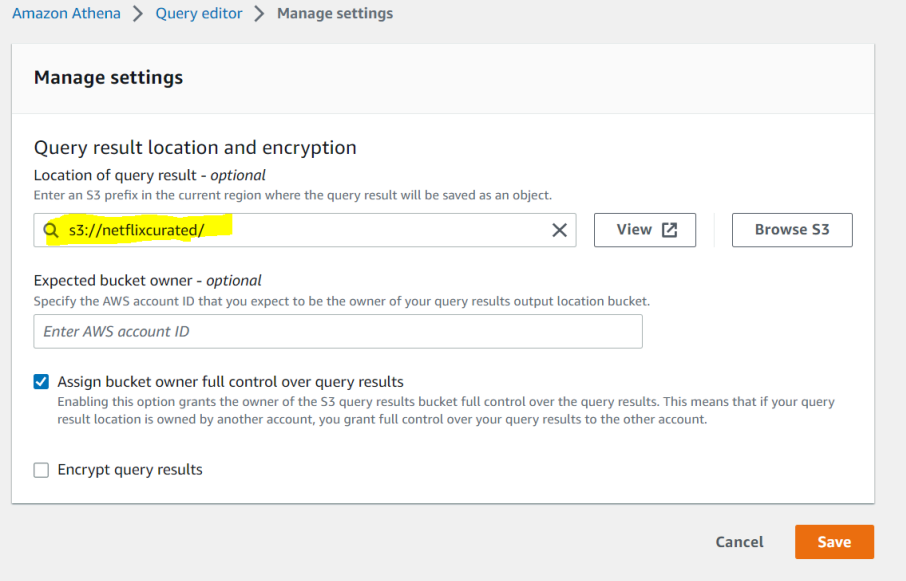
14. Goto Athena Dashboard
15. Under Editor, You can find newly created table through crawler



16. Goto Settings to change the location of Query results.

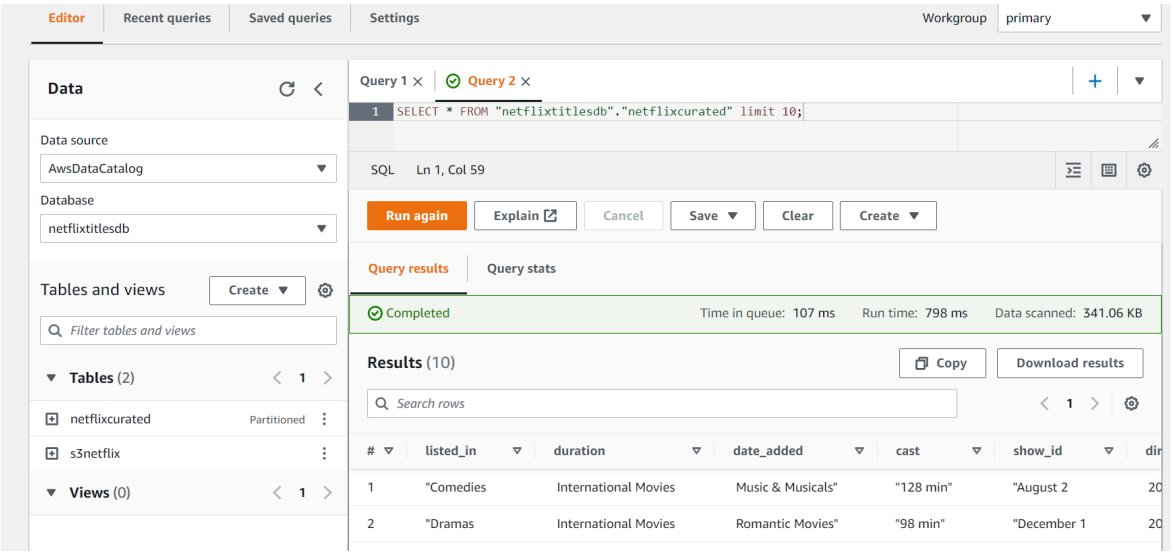


17. Under location of Query Result, Browse S3 and select **netflixcurated** bucket location and click **Save**



18. Go Back to Editor and run SQL queries

    Ex: SELECT * FROM "netflixtitlesdb"."netflixcurated" limit 10;



19. Explore yourself on editor

How Netflix formed a Data lake

Netflix implemented multiple Data lake ecosystems based on the requirements. Below given is one such use case

Event Data Lake environment

Netflix collects ~500 Billion events/day and it has a 5 min SLA to enter into datahub.

1. Business Events such as user searching for a movie, clicking on a title is collected through cloud apps.
2. Distributed event store and stream-processing platform kafka, ingests these data into the Amazon S3 central repository.
3. Event Ingestion Pipeline reads data directly from S3 in real time and writes back to the data hub.
4. Amazon S3 provides strong read after write, so it was faster for event ingestion pipeline to read after kafka write operation and it doesn't require to list the files in S3 continuously to identify new files

Benefits of data lake environment in Netflix

1. Increased Performance - read/write performance is faster. The average execution time of a query run on a data lake is 3s while the old data architecture takes around 5 mins to run a job.
2. Scalability - Since this data lake is based on cloud, scaling the components takes only minutes to spin off new nodes compared to procuring the physical resources in on prem data architecture which takes days
3. With a data lake environment, Previously data movement jobs took ~20 mins to move data from source to datahub but with data lake architecture, Netflix was able to achieve 5 mins. This is because, there is no predefined schema, read/write is designed to be faster in data lake and no preceding data transformation in pipeline.

What are the Maturity stages of Data Lakes?

Data Maturity is the drive towards the improvement and increased capability in using data.
Every organization including Netflix when implementing Data lakes , they have to start with some kind of maturity or after implementation they are assessed on their maturity levels.

Below are the Maturity Stages:

- **Stage 1: Data source identification**:
    - In the first stage, the data lake used to store the raw data regularly before making it ready for use.
    - **At this stage, a right set of secured management practices should be decided to classify and label the data to be stored.**
- **Stage 2: Testing and learning a data lake environment:**
    - At this stage, organizations and end users start to use the data lake more frequently.
    - This stage actively involves transforming and analyzing the collected data.
    - Experiments are conducted on raw data with the required tools decided by the company.
- **Stage 3: Integrate data lake with existing data warehouses:**
    - Companies use data warehouses and data lakes together for data analytics.
- **Stage 4: Data lake as a service:**
    - At this stage, the data lake becomes a central repository of the organization’s data infrastructure that provides data to the business users.