Overview:

This workshop will focus on hands-on labs and modules on ingestion, hydration, exploration, and consumption of data in a data lake in AWS.

It includes hands-on time with AWS analytics services, including AWS Data Migration service for batch data ingestion, AWS Glue for data catalog and running ETL on Data lake, Amazon Athena to query data lake, and Amazon Quicksight for visualization.

The workshops provided here will guide you through creating a modern data platform on AWS and will demonstrate the fundamental principles:

1. Scalable Data Lakes
2. Purpose Built Data Services
3. Seamless Data Movement
4. Easy Data Visualization
5. Performant and Cost-Effective

Architecture

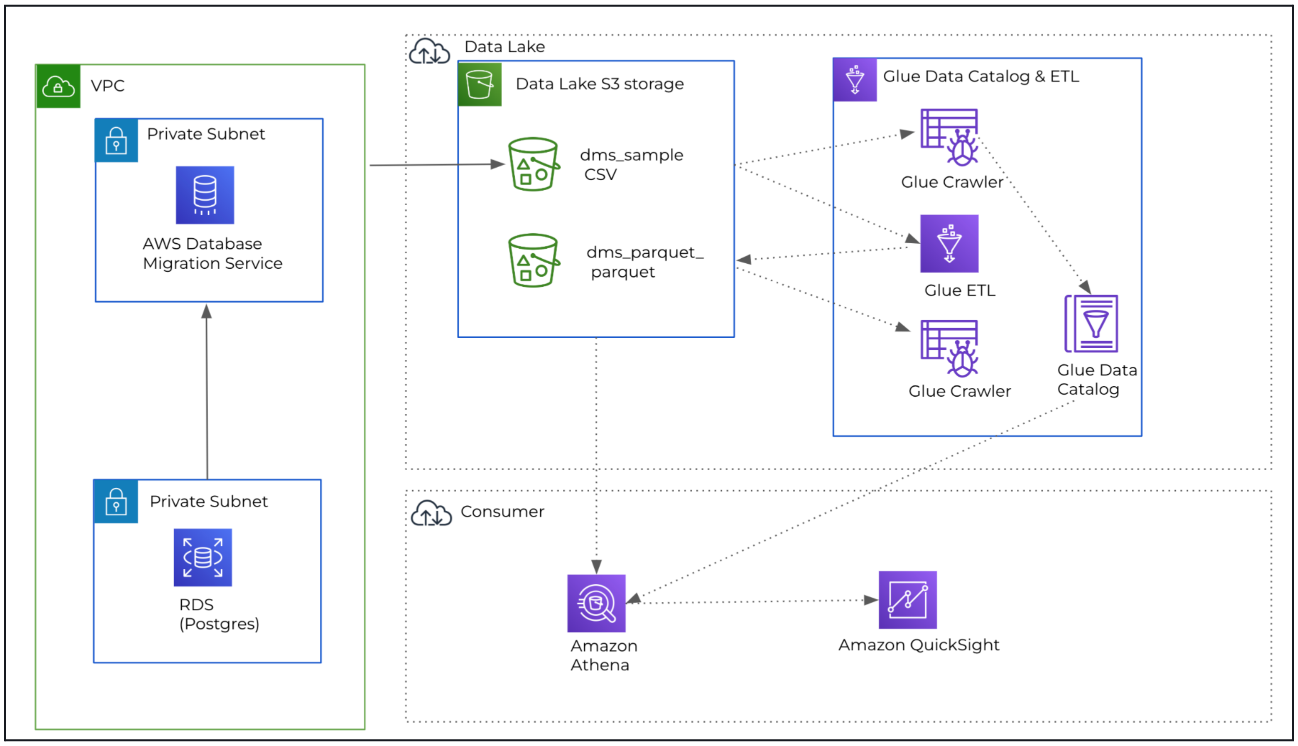

Fig: Workshop Architecture

Step 1 - Prerequisites:

- Deploy AWS IAM roles
    - In this task, you need to deploy an AWS Cloudformation template(CFT), and this template will deploy the following resource:
        - Amazon S3 bucket for storing data
        - AWS IAM roles for DMS tasks and Glue ETL
    - Link for CFT template

- Deploy RDS Postgres database as the data source
    - In this task, you need to deploy an AWS Cloudformation template(CFT), and this template will deploy the following resource:
        - Amazon VPC with 3 subnets, 1 Internet Gateway, 1 Route Table
        - Amazon RDS Instance with Postgres engine and following properties
            - DBName: sportstickets
            - UserName: adminuser
            - Password: admin123
            - Port: 5432
        - Amazon Ec2 as worker node to perform data loading into Amazon RDS Instance
    - Link for CFT template

Step 2 - Data Migration

In this part will give you an understanding of the AWS Database Migration Service (AWS DMS). We will migrate data from an existing Amazon Relational Database Service (Amazon RDS) Postgres database to an Amazon Simple Storage Service (Amazon S3) bucket that you create.

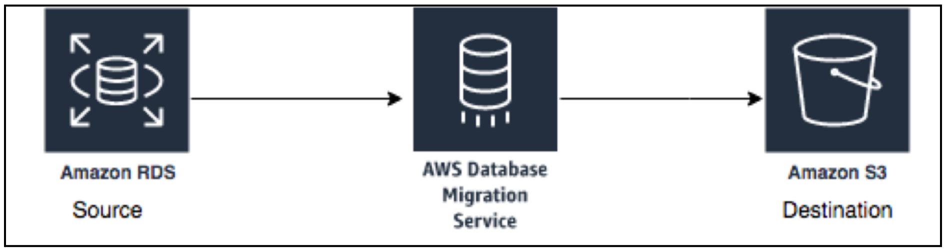

Fig: Data Migration Architecture

In this step we will complete the following tasks:

- **Create a subnet group within the DMS Lab VPC**

    a. On the DMS console , select Subnet Groups and Create subnet group.
    b. In the Identifier box, type a descriptive name that you will easily recognize (e.g., **dna-lab-subnet-grp**).
    c. In the Description box, type an easily recognizable description (e.g., **Replication instance for production data system**).
    d. For VPC, select the name of the VPC that you created earlier with AWS CloudFormation template. VPC name ending with **dna-lab**. The subnet list populates in the Available Subnets pane.
    e. Select as many subnets as you want and click Add. The selected subnets move to the Subnet Group pane.
    f. Click Create subnet group, the subnet group status displays Complete.

- **Create a DMS replication instance**

    a. On the DMS console, select Replication instances to create a new replication instance.
        i. For Name, type a name for the replication instance that you will easily recognize. (e.g., **DMS-Replication-Instance**).
        ii. For Description, type a description you will easily recognize. (e.g., **DMS Replication Instance**).
        iii. For Instance class, choose dms.t3.medium
        iv. Select the latest engine version
        v. For VPC, select the name of the VPC that you created earlier with AWS CloudFormation template or preprovisoned for you. VPC name ending with **dna-lab**.
        vi. For Multi AZ, from the dropdown select Dev or test workload (Single-AZ)
        vii. Click Advanced to expand the section.
        viii. Select the security group with **dna-lab** in the name.
    b. Leave other fields at their default values, then click Create.
    c. The DMS console displays creating for the instance status. When the replication instance is ready, the status changes to available. While replication instance is spinning up, you can proceed to next step for DMS endpoint creation.

Note: Please proceed to create your endpoints, without waiting for the step above.

- **Create a source endpoint**

  a. On the DMS console, select Endpoints to create one source Endpoint.
     i. select Source Endpoint type.
     ii. For Endpoint identifier, select your easily recognized name (e.g. **rds-source-endpoint**)
     iii. For Source engine, select PostgreSQL.
     iv. For "Access to Endpoint database", select "Provide access information manually"
     v. Enter the Server name, get the Database Endpoint from Environment Setup module on your event engine dashboard.
     vi. Fetch RDS endpoint from CFT output starting with **dnaworkshopinstance**.
     vii. For Port, enter **5432**.
     viii. For SSL mode, choose **none**.
     ix. For User name, type **adminuser**.
     x. For Password, type **admin123**.
     xi. For Database name, type **sportstickets**
  b. Accept other defaults and then click Create endpoint to create the endpoint. When available, the endpoint status changes to active.
  c. Check the replication instance created previously. Make sure the status is available.
  d. Select your newly created source endpoint, and choose Test connection on the Actions drop-down list.
  e. Click Run test. This step tests connectivity to the source database system. If successful, the message “Connection tested successfully” appears.

Note - Before start, make sure you have the following values handy from CFT output. ARN of DMSLabRole to access S3 & S3 Bucket name.

- **Create a target endpoint**

  a. On the DMS console, select Endpoint to create a target Endpoint .
     i. For Endpoint type, select Target endpoint.
     ii. For Endpoint identifier, type an easily recognized name such as **s3-target-endpoint**
     iii. For Target engine, choose Amazon S3.
     iv. For Service access role ARN, paste the **DMSLabRoleS3** ARN number noted earlier

    v. For Bucket name, paste the S3 Bucket Name noted earlier
    vi. For Bucket folder, type **tickets**.
    vii. Expand the Endpoint settings section.
    viii. Click on the Use endpoint connection attributes checkbox, type in **addColumnName=true** in the Extra connection attributes box”
    ix. Expand the Test endpoint connection (optional) section, and choose your “VPC name with **dna-lab**” on the VPC drop-down list.
    x. Click Run test. This step tests connectivity to the source database system. If successful, the message “Connection tested successfully” appears.

b. Click Create Endpoint. When available, the endpoint status changes to active.

- **Create a task to perform the initial migration of the data.**

a. On the DMS console, select Database Migration Tasks to create a task
    i. Type an easily recognized Task name e.g. **dms-full-dump-task**.
    ii. Select your Replication instance from drop down.
    iii. Select your Source endpoint from drop down.
    iv. Select your Target endpoint from drop down.
    v. For Migration type choose Migrate existing data.
    vi. Expand Task Settings.
    vii. Select the Enable CloudWatch logs check box
    viii. Go to Table Mappings.
    ix. Click on Add new selection rule and select “Enter a Schema” in Schema field.
    x. For Schema name, select **dms_sample** . Keep the settings for the remaining fields

b. Click Create task. Your task is created and starts automatically.
c. Once complete, the console displays 100% progress.
d. Select your task and explore the summary. Scroll down and you can observe all table information loaded in S3 from RDS by DMS
e. Open the S3 console and view the data that was copied by DMS.

   Path: BucketName/bucket_folder_name/schema_name/table_name/objects/

f. Navigate to one of the files and review it using S3 Select :
    i. Navigate in to the directory named player and select the check box next to the file name.
    ii. Click the Actions dropdown button and choose Query with S3 Select.

iii. In the Query with S3 Select page, leave the default value for *Input Settings* and *SQL Query* and click Run SQL query.
iv. It will execute the specified SQL query and return the first 5 lines from the CSV file.

Notice: You will notice that the file contains the column headers in the first row as requested by the "addColumnName=true" connection attribute we included when we created the s3 target endpoint. Note that column names are included in the file in the first row.

Step 3 - Data Transformation

In this part we will learn about AWS Glue, which is a serverless data integration service that makes it easier to discover, prepare, move, and integrate data from multiple sources for analytics, machine learning (ML), and application development.

We will use a crawler to populate the AWS Glue Data Catalog with tables. This is the primary method used by most AWS Glue users. A crawler can crawl multiple data stores in a single run. Upon completion, the crawler creates or updates one or more tables in your Data Catalog.

Extract, transform, and load (ETL) jobs that you define in AWS Glue use these Data Catalog tables as sources and targets. The ETL job reads from and writes to the data stores that are specified in the source and target Data Catalog tables.

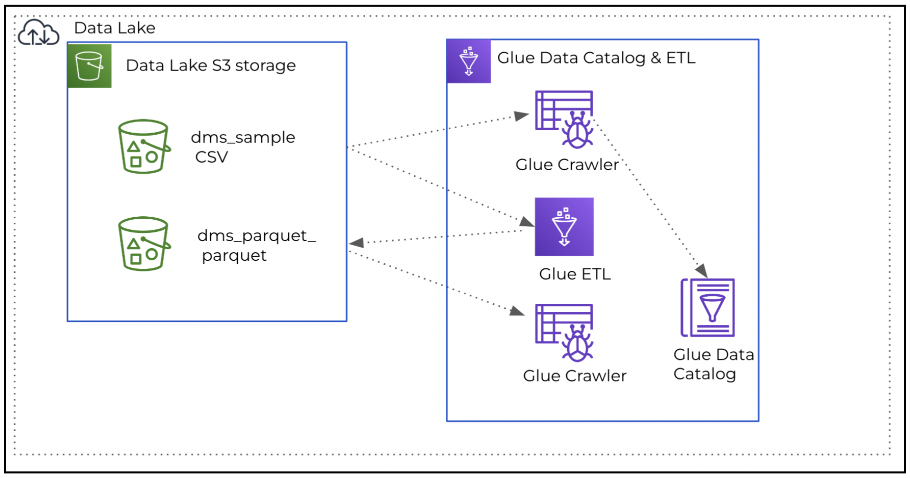

Fig: Data Transformation Architecture

In this steps we will be completing the following tasks.

- **Crawl Data using Glue crawler**

    a. Navigate to the AWS Glue Console
    b. On the AWS Glue menu, select Crawlers & Click Add crawler.
    c. Enter **glue-lab-crawler** as the crawler name for initial data load.
    d. Optionally, enter the description. This should also be descriptive and easily recognized and Click Next.
    e. Choose Data stores, Crawl all folders and Click Next
    f. On the Add a data store page, make the following selections:
        i. For Choose a data store, click the drop-down box and select S3.
        ii. For Crawl data in, select Specified path in my account.
        iii. For Include path, browse to the target folder stored CSV files, e.g., s3://xxx-s3bucket-xxx/tickets
    g. Click Next.
    h. On the Add another data store page, select No. and Click Next.
    i. On the Choose an IAM role page, make the following selections:
        i. Select Choose an existing IAM role.
        ii. For IAM role, select <stackname>-GlueLabRole-<RandomString> pre-created for you.
    j. Click Next.
    k. On the Create a schedule for this crawler page, for Frequency, select Run on demand and Click Next.
    l. On the Configure the crawler's output page, click Add database to create a new database for our Glue Catalogue.
    m. Enter ticketdata as your database name and click create
    n. For Prefix added to tables (optional), leave the field empty.
    o. For Configuration options (optional), select Add new columns only and keep the remaining default configuration options and Click Next.
    p. Review the summary page noting the Include path and Database output and Click Finish. The crawler is now ready to run.
    q. Select the checkbox next to the crawler name, click Run crawler button.
    r. Crawler will change status from starting to stopping, wait until crawler comes back to ready state (the process will take a few minutes), you can see that it has created 15 tables.
    s. In the AWS Glue navigation pane, click Databases → Tables. You can also click the **ticketdata** database to browse the tables.

- **Data Validation and ETL**

To store processed data in parquet format, we need a new folder location for each table, eg. the full path for sport_team table look like this s3://<s3_bucket_name>/tickets/dms_parquet/sport_team

Glue will create the new folder automatically, based on your input of the full file path, such as the example above. In the left navigation pane, under ETL, click AWS Glue Studio.

a. Choose “View jobs”
b. Leave the “Visual with a source and target” option selected, and click “Create”
c. Select the Data source - S3 bucket at the top of the graph.
d. In the panel on the right under “Data source properties - S3”, choose the ticketdata database from the drop down.
e. For Table, select the **sport_team** table.
f. Select the ApplyMapping node. In the Transform panel on the right and change the data type of id column to double in the dropdown.
g. Select the Data target - S3 bucket node at the bottom of the graph, and change the Format to Parquet in the dropdown. Under *Compression Type*, select Uncompressed from the dropdown.
h. Under “S3 Target Location”, select “Browse S3” browse to the “xxx-3bucket-xxx” bucket, select “**tickets**” item and press “Choose”.
i. In the textbox, append dms_parquet/sport_team/ to the S3 url. The path should look similar to s3://xxx-s3bucket-xxx/tickets/dms_parquet/sport_team/ - don’t forget the / at the end. The job will automatically create the folder.
j. Finally, select the Job details tab at the top. Enter **Glue-Lab-SportTeamParquet** under Name.
k. For IAM Role, select the role named similar to xxx-GlueLabRole-xxx.
l. Scroll down the page and under Job bookmark, select Disable in the drop down. You can try out the bookmark functionality later in this lab.
m. Press the Save button in the top right-hand corner to create the job.
n. Once you see the Successfully created job message in the banner, click the Run button to start the job.
o. Select Jobs from the navigation panel on the left-hand side to see a list of your jobs.
p. Select Monitoring from the navigation panel on the left-hand side to view your running jobs, success/failure rates and various other statistics.

q. Scroll down to the Job runs list to verify that the ETL job has completed successfully. This should take about 1 minute to complete.
r. We need to repeat this process for an additional 4 jobs, to transform the **sport_location, sporting_event, sporting_event_ticket and person tables**.

Note:

- During this process, we will need to modify different column data types. We can either repeat the process above for each table, or we can clone the first job and update the details.
- The steps below describe how to clone the job - if creating manually each time, follow the above steps but make sure you use the updated values from the tables below.
    1. Return to the Jobs menu, and select the **Glue-Lab-SportsTeamParquet** job by clicking the checkbox next to the name.
    2. Under the Actions dropdown, select Clone job. Update the job as per the following tables, then Save and Run.

1. Sport_Location:

Create a **Glue-Lab-SportLocationParquet** job with the following attributes:

Task / Action	Attribute	Values
"Data source - S3 bucket" node	Database	ticketdata
	Table	sport_location
"Transform - ApplyMapping" node	Schema transformations	None
"Data target - S3 bucket" node	Format	Parquet
	Compression Type	Uncompressed
	S3 target path	tickets/dms_parquet/sport_location/
"Job details tab"	Job Name Compression Type	Glue-Lab-SportLocationParquet
	IAM Role	xxx-GlueLabRole-xxx
	Job bookmark	Disable

2. Sporting_Event:

Create a **Glue-Lab-SportingEventParquet** job with the following attributes:

Task / Action	**Attribute**	**Values**
“Data source - S3 bucket” node	Database	ticketdata
	Table	sporting_event
“Transform - ApplyMapping” node	Schema tranformations	column “start_date_time” => TIMESTAMP
		column “start_date” => DATE
“Data target - S3 bucket” node	Format	Parquet
	Compression Type	Uncompressed
	S3 target path	tickets/dms_parquet/sporting_event/
“Job details tab”	Job Name	Glue-Lab-SportingEventParquet
	IAM Role	xxx-GlueLabRole-xxx
	Job bookmark	Disable

3. Sporting_Event_Ticket:

Create a **Glue-Lab-SportingEventTicketParquet** job with the following attributes:

Task / Action	Attribute	Values
“Data source - S3 bucket” node	Database	ticketdata
	Table	sporting_event_ticket
“Transform - ApplyMapping” node	Schema tranformations	column “id” => DOUBLE
		column “sporting_event_id” => DOUBLE
		column “ticketholder_id” => DOUBLE
“Data target - S3 bucket” node	Format	Parquet
	Compression Type	Uncompressed
	S3 target path	tickets/dms_parquet/sporting_event_ticket/
“Job details tab”	Job Name	Glue-Lab-SportingEventTicketParquet
	IAM Role	xxx-GlueLabRole-xxx
	Job bookmark	Disable

Troubleshoot: Glue-Lab-SportingEventTicketParquet job failing with error 'Unsupported case of DataType'

4. **Person**:

Create a **Glue-Lab-PersonParquet** job with the following attributes:

Task / Action	Attribute	Values
“Data source - S3 bucket” node	Database	ticketdata
	Table	person
“Transform - ApplyMapping” node	Schema tranformations	column “id” => DOUBLE
“Data target - S3 bucket” node	Format	Parquet
	Compression Type	Uncompressed
	S3 target path	tickets/dms_parquet/person/
“Job details tab”	Job Name	Glue-Lab-PersonParquet
	IAM Role	xxx-GlueLabRole-xxx
	Job bookmark	Disable

- **Create Glue Crawler for Parquet Files**

    a. In the Glue Studio navigation menu, select Crawlers to open the Glue Crawlers page in a new tab. Click Add crawler.
    b. For Crawler name, type glue-lab-parquet-crawler and Click Next.
    c. In next screen Specify crawler source type, select Data Stores as choice for Crawler source type and click Next.
    d. In Add a data store screen
        i. For Choose a data store, select “S3”.
        ii. For Crawl data in, select “Specified path in my account”.
        iii. For Include path, specify the S3 Path (Parent Parquet folder) that contains the nested parquet files e.g., s3://xxx-dmslabs3bucket-xxx/tickets/dms_parquet
        iv. Click Next.
    e. For Add another data store, select No and Click Next.
    f. On the Choose an IAM role page, select Choose an existing IAM role.
        i. For IAM role, select the existing role “xxx-GlueLabRole-xxx” and
        ii. Click Next.
    g. For Frequency, select “Run On Demand” and Click Next.

h. For the crawler's output database, choose your existing database which you created earlier e.g. ticketdata
i. For the Prefix added to tables (optional), type parquet_
j. Review the summary page and click Finish.
k. Click Run Crawler. Once your crawler has finished running, you should report that tables were added from 1 to 5, depending on how many parquet ETL conversions you set up in the previous section.

- **Confirm you can see the tables**

  a. In the left navigation pane, click Tables.
  b. Add the filter parquet to return the newly created tables.

Note: Glue Workflows

- In AWS Glue, you can use workflows to create and visualize complex extract, transform, and load (ETL) activities involving multiple crawlers, jobs, and triggers.
- Each workflow manages the execution and monitoring of all its components.
- As a workflow runs each component, it records execution progress and status, providing you with an overview of the larger task and the details of each step.

Step 4 - Perform Data Query

In this steps we will introduces you to Amazon Athena, and Amazon QuickSight. Amazon Athena provides the ability to run ad-hoc queries on your data in your data lake.

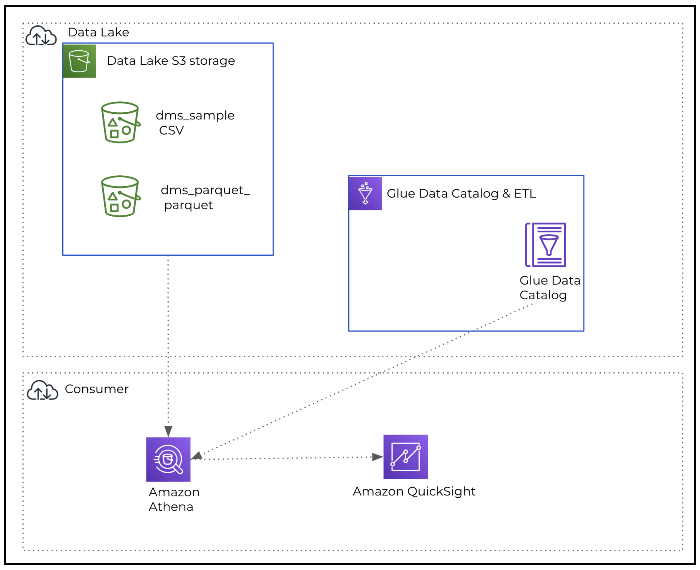

Fig: Data Query Architecture

In this steps we will be completing the following tasks.

- **Query data and create a view with Amazon Athena**

  a. In the AWS services console, search for Athena
  b. (optional) If it’s the first time you are using Athena in your AWS Account, then click set up a query result location in Amazon S3 at the top.
  c. In the Query Editor, select your newly created database e.g., "ticketdata”.
  d. Click the table named "parquet_sporting_event_ticket" to inspect the fields.

Note: The type for fields id, sporting_event_id and ticketholder_id should be (double). Next, we will query across tables parquet_sporting_event, parquet_sport_team, and parquet_sport location.

  e. Copy the following SQL syntax into the New Query tab and click Run Query.

     SELECT e.id AS event_id, e.sport_type_name AS sport, e.start_date_time AS event_date_time, h.name AS home_team, a.name AS away_team, l.name AS location, l.city FROM parquet_sporting_event e, parquet_sport_team h, parquet_sport_team a, parquet_sport_location l WHERE e.home_team_id = h.id AND e.away_team_id = a.id AND e.location_id = l.id;

  f. As shown above Click Create and then select Create view from query
  g. Name the view **sporting_event_info** and click Create
  h. Copy the following SQL syntax into the New Query tab.

     SELECT t.id AS ticket_id, e.event_id, e.sport, e.event_date_time, e.home_team, e.away_team, e.location, e.city, t.seat_level, t.seat_section, t.seat_row, t.seat, t.ticket_price, p.full_name AS ticketholder FROM sporting_event_info e, parquet_sporting_event_ticket t, parquet_person p WHERE t.sporting_event_id = e.event_id AND t.ticketholder_id = p.id

  i. Click on Save as button Give this query a name: **create_view_sporting_event_ticket_info** and some description and then, click on Save
  j. As shown above, click Create view from query.
  k. Name the view **sporting_event_ticket_info** and click Create.
  l. Copy the following SQL syntax into the New Query tab.

SELECT sport, count(distinct location) as locations, count(distinct event_id) as events, count(*) as tickets, avg(ticket_price) as avg_ticket_price FROM sporting_event_ticket_info GROUP BY 1 ORDER BY 1;

m. Click on Save as and give this query name: **analytics_sporting_event_ticket_info** and some description and then, click on Save.
n. Click on Run Query
o. You query returns two results in approximately five seconds. The query scans 25 MB of data, which prior to converting to parquet, would have been 1.59GB of CSV files.

Note:

- The purpose of saving the queries is to have clear distinction between the results of the queries running on one view.
- Otherwise, your query results will be saved under “Unsaved” folder within the S3 bucket location provided to Athena to store query results.
- Please navigate to S3 bucket to observe these changes

Step 5 - Build an Amazon QuickSight Dashboard

Amazon QuickSight provides visualization of the data you import. In this steps we will be completing the following tasks.

- Set up QuickSight

  a. In the AWS services console, search for QuickSight
  b. (optional) If this is the first time you have used QuickSight, you are prompted to create an account.
     - Click Sign up for QuickSight.
     - For account type, choose the default Standard/Enterprise Version.
     - Click Continue.
     - On the Create your QuickSight account page, for QuickSight account name give a unique name and email address.
     - Choose the appropriate AWS region based on where you are running this workshop on and the check boxes to enable auto discovery, Amazon Athena, and Amazon S3.
     - Select your DMS bucket (e.g., "xxx-s3bucket-xxx"), Click Finish

c. On the top right corner, click New analysis.
d. Click New Data Set.
e. On the Create a Dataset page, select Athena as the data source.
f. For Data source name, type ticketdata-qs , then click Validate connection.
g. Click Create data source
h. In the Database drop-down list, select the database ticketdata.
i. Choose the "sporting_event_ticket_info" table and click Select
j. To finish data set creation, choose the option Import to SPICE for quicker analytics and click Visualize. If your SPICE has 0 bytes available, choose the second choice Directly query your data

- Create QuickSight Charts

  a. In the Fields list, click the ticket_price column to populate the chart.
  b. Click the expand icon in corner of "ticket_price" field, and select Show as Currency to show the number in dollar value
  c. You can add visual by clicking Add button at top left corner of screen.
     - In the Visual types area, choose the Vertical bar chart icon.
     - This layout requires a value for the X-axis. In Fields list, select the event_date_time field and you should see the visualization update.
     - For Value Y-axis, select “ticket_price” from the Field list
  d. You can drag and move other visuals to adjust space in dashboard. In the Fields list, click and drag the seat_level field to the Group/Color box. You can also use the slider below the x axis to fit all of the data.
  e. In the Visual types area, choose the Clustered bar combo chart icon.
  f. In the Fields list, click and drag the ticketholder field to the Lines box.
  g. In the Lines box, click the dropdown box and choose Aggregate: Count Distinct for Aggregate. You can then see the y-axis update on the right-hand side.
  h. Click on insight icon on the left tabs section and explore insight information in simple English.

Note: Feel free to experiment with other chart types and different fields to get a sense of the data.

- Create QuickSight Parameters

  a. In the left navigation menu, select Parameters.
  b. Click Create one to create a new parameter with a Name.
  c. For Name, type EventFrom.

d. For Data type, choose Datetime.
e. For Time granularity, set Hourly.
f. For Default value, select the value from calendar as start date available in your graph for event_date_time. For example, 2022-01-01 00:00.
g. Click Create, and then close the Parameter Added dialog box
h. Create another parameter with the following attributes:
    - Name: EventTo
    - Data type: Datetime
    - For Time granularity, set Hourly.
    - For Default value, select the value from calendar as end date available in your graph for event_date_time. For example, 2022-01-01 00:00
    - Click Create
i. In next window, you can select any option to perform any operation with the parameter. Alternatively, you can click the drop-down menu or the EventFrom parameter and choose Add control.
j. For Display name, specify Event From and click Add.
k. Repeat the process to add a control for EventTo with display name Event To

- Create a QuickSight Filter

a. In the left navigation menu, choose Filter.
b. Click the plus icon (+) to add a filter for the field event_date_time.
c. Click this filter to edit the properties.
d. For Filter type, choose Date & Time range and Between.
e. Select option Use Parameter, click Yes to apply to all visual.
f. For Start date parameter, choose EventFrom.
g. For End date parameter, choose EventTo.
h. Click Apply.

- Add Calculated Fields

a. Click the Add button on the top left and select Add a calculated field.
b. Give it a name event_day_of_week
c. For Formula, type extract('WD',{event_date_time})

Note: extract() returns a specified portion of a date value. Requesting a time-related portion of a date that doesn't contain time information returns 0. WD: This returns the day of the week as an integer, with Sunday as 1.

d. Click Save.

e. Add another calculated field with the following attributes:
   - Calculated field name: event_hour_of_day
   - Formula: extract('HH',{event_date_time})
f. Click Add button on the top left and choose Add visual.
g. For field type, select the scatter plot.
h. In the Fields list, click the following attributes to set the graph attributes:
   - X-axis: "event_hour_of_day"
   - Y-axis: "event_day_of_week"
   - Size: "ticket_price"
i. Publish dashboard by clicking on the Share menu on the top right corner of screen.

Note: A dashboard is a read-only snapshot of an analysis that you can share with other Amazon QuickSight users for reporting purposes. In Dashboard other users can still play with visuals and data but that will not modify dataset.

Bonus Lab - AWS Glue Data Brew

Prerequisites

- Download the [dataset] from this link and upload it to the S3 bucket
- Download the [cloudformation template] from this link and Deploy it
- Once the Cloudformation stack is deployed successfully please capture the values for RoleName and S3Bucket details

Creating a project

- Navigate to the AWS Glue DataBrew service
- On the DataBrew console, select Projects
- Click Create project
- In the Project details section, enter covid-states-daily as the project name

Creating a dataset

- In the Select a dataset section, select New dataset and enter covid-states-daily-stats
- In the Connect to a new dataset section, select Amazon S3 under “Data lake/data store” and Enter the S3 path. Leave the default configuration values
- In the Permissions section, select the role DataBrew-DataBrewLabRole--xxxxx from the drop-down list
- Click Create project
- Glue DataBrew will create the project, this may take a few minutes.

Exploring the dataset

- Grid view - When the project has been created, you will be presented with the Grid view. This is the default view, where a sample of the data is shown in tabular format. The Grid view shows
    - Columns in the dataset
    - Data type of each column
    - Summary of the range of values that have been found
    - Statistical distribution for numerical columns

- Schema view - The Schema view shows the schema that has been inferred from the dataset. In schema view, you can see statistics about the data values in each column. In the Schema view, you can
    - Select the checkbox next to a column to view the summary of statistics for the column values
    - Show/Hide columns
    - Rename columns
    - Change the data type of columns

    - Rearrange the column order by dragging and dropping the columns

- Profile view - In the Profile view, you can run a data profile job to examine and collect statistical summaries about the data. A data profile is an assessment in terms of structure, content, relationships, and derivation.
    - Job Name
        - Click on Run data profile
        - In the job details and job run sample panels, leave the default values
    - Job Output Setting
        - In the Job output settings section, select the S3 bucket with the name DataBrew-DataBrewLabRole--xxxxx and a folder name (eg. data-profile)
        - In the Permissions section, select the IAM role with the name databrew-lab-DataBrewLabRole-xxxxx
        - Leave all other settings as the default values
    - Click Create and run job
    - When the profile job has successfully completed, click on View data profile under Jobs from the menu on the left hand side of the DataBrew console

- Click on the Column statistics tab to view a column-by-column breakdown of the data values.

Preparing the dataset

- In this section, we will apply the different transformations to the dataset.
- Rename columns
- Change the data type of columns
- Filled with the most frequent value
- Download the recipe from this link
- Select on Recipe from the menu on the left-hand side of the DataBrew console and click Upload Recipe
- Provide below details
    - Recipe Name
    - Upload Recipe json script downloaded under Step 1
    - Select Create and publish recipe
- Now let's apply this recipe, click the project we configured now and the right side, click Import recipe
- Under Import recipe, select the recipe we configured and click Next
- Select the Append option from the right side and click Next
- Now let's validate the recipe and wait for all validation to be successful
- Once the validation is successful, click Import
- Now we will be back to the project screen and with the recipe implied on the dataset

Creating a DataBrew job

- Click on Jobs from the menu on the left hand side of the DataBrew console
    - On the Recipe jobs tab, click on Create job
    - Enter covid-states-daily-prep for the job name
    - Select Create a recipe job
    - Select the covid-states-daily dataset
    - Select the ‘covid-states-daily-recipe’
    - In the Job output settings section, enter the S3 location s3://bucket-name/job-outputs/.
    - In the Permissions section, select the role DataBrew-DataBrewLabRole--xxxxx
    - Click Create and run job
- DataBrew job is created and the job status is Running

Viewing data lineage

- In DataBrew, navigate back to the covid-states-daily project
- Click on Lineage at the top right
- Select CloudTrail logs to view all the action on this dataset.