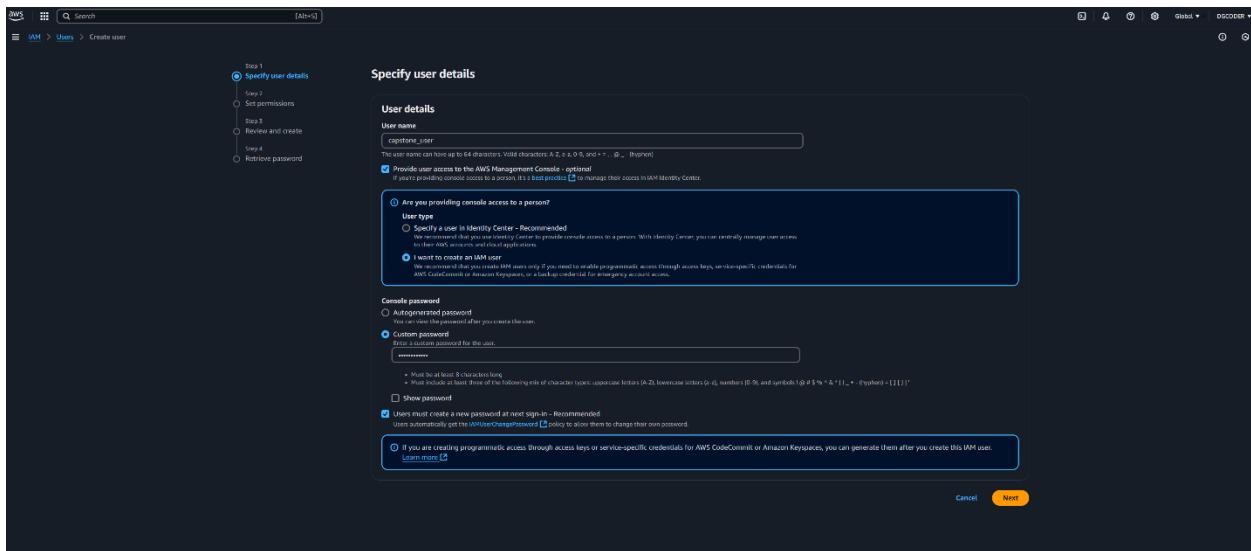
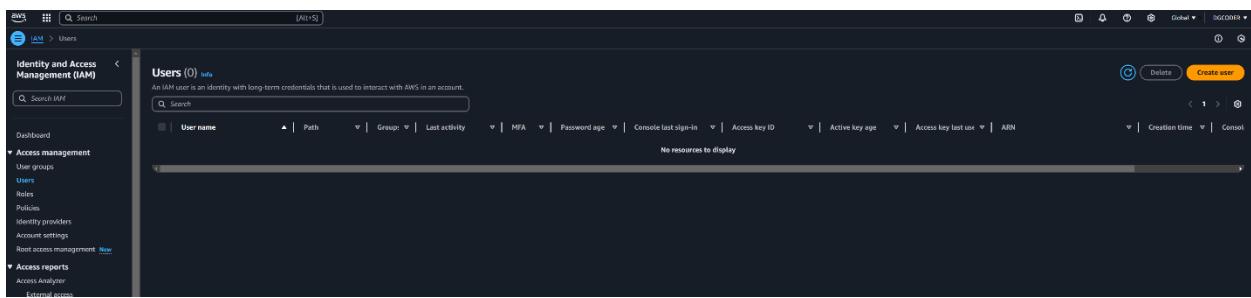


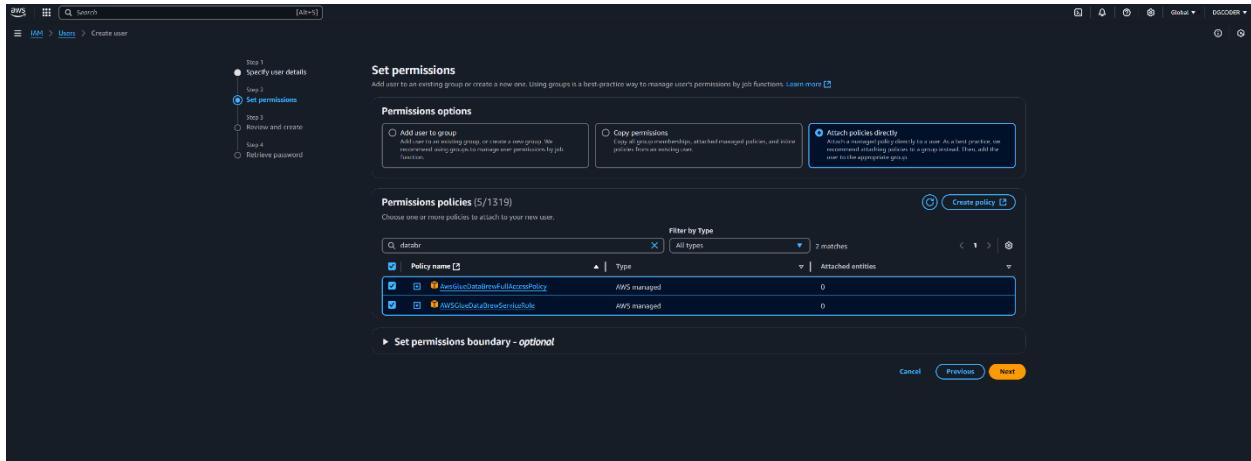
Capstone Project

1. Creating S3 Bucket and Uploading Data

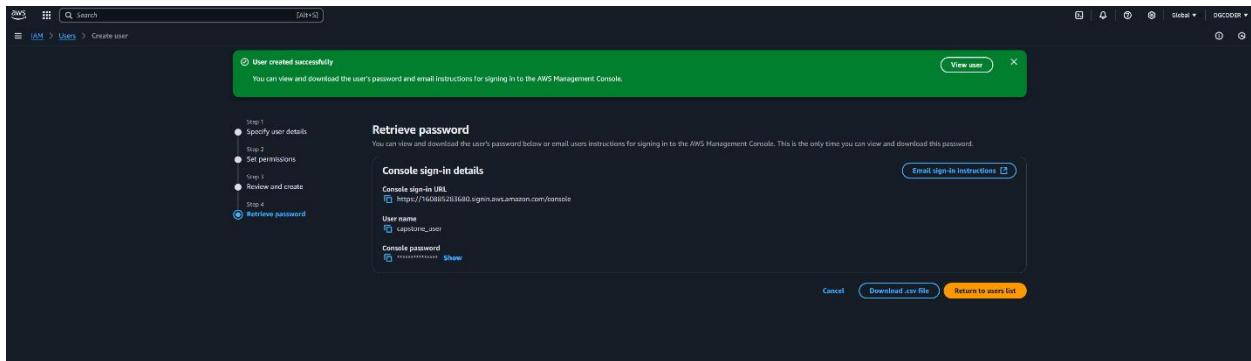
1.1 Creating a user from root user



Setting up the username (**capstone_user**) and password.

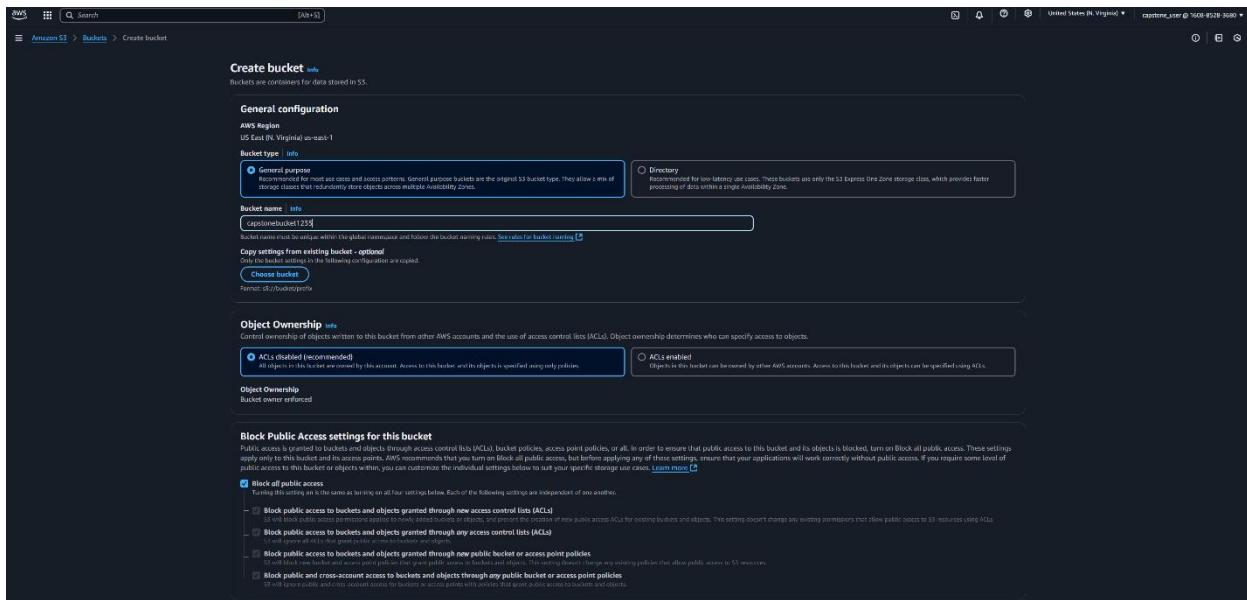


Attaching all the required policies.

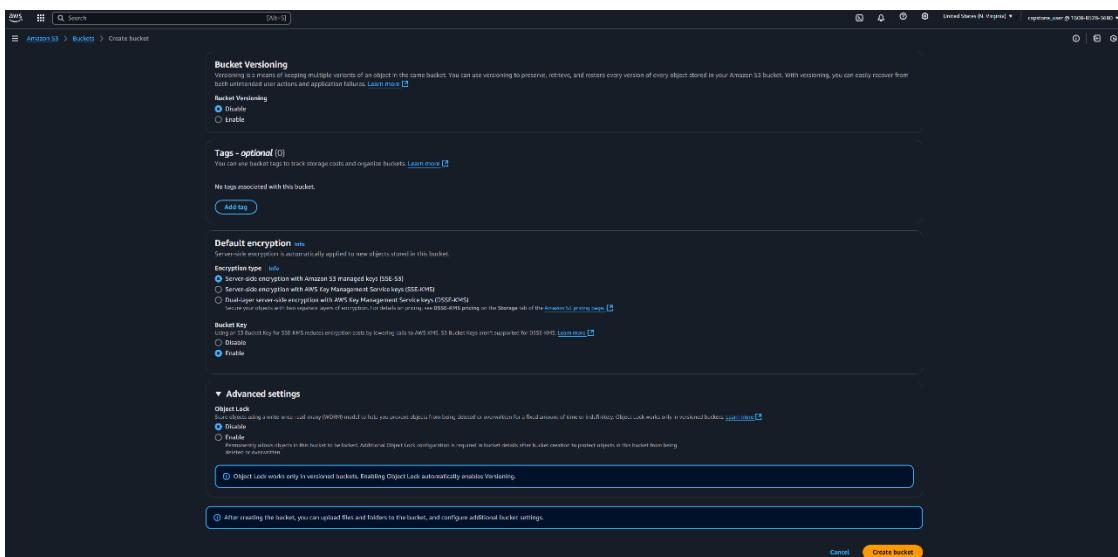


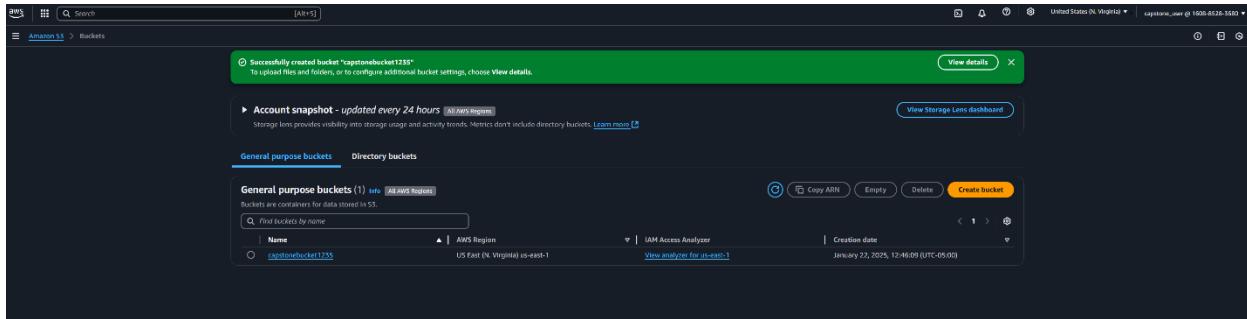
The user has been successfully created.

1.2 Creating an S3 Bucket



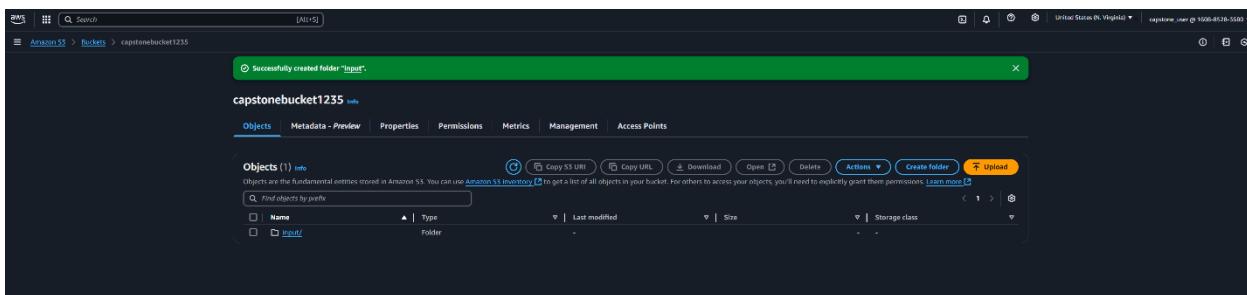
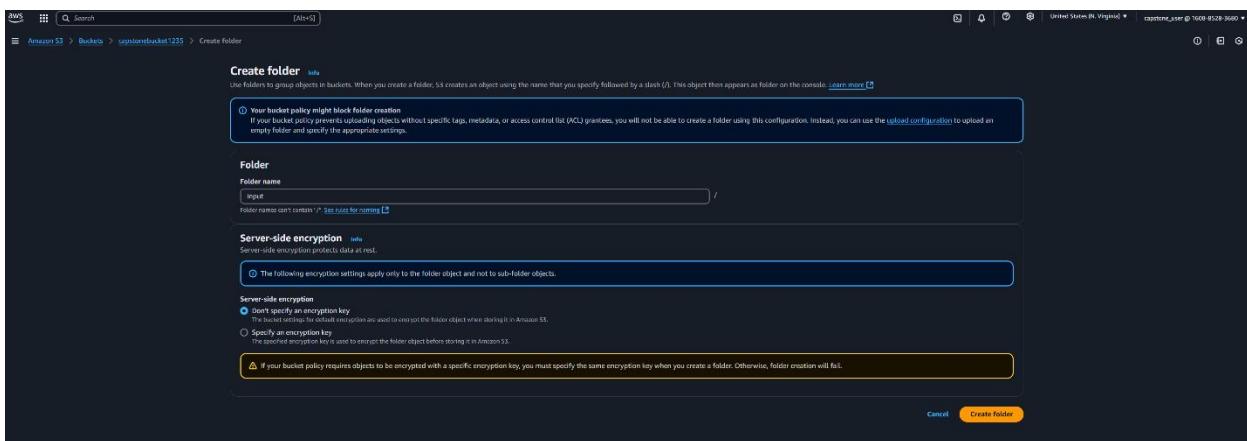
Creating an S3 bucket named (**capstonebucket1235**).





The S3 bucket(**capstonebucket1235**) has been successfully created.

1.3 Creating an Input Container



1.4 Uploading the heart disease sample csv file in input container

The screenshot shows the AWS S3 'Upload' interface. A single file, 'heart_disease_sample.csv', is selected for upload. The destination is set to 's3://capstonebucket1235/input'. The file is a CSV file (text/csv) of size 1.4 KB. The 'Upload' button is visible at the bottom right.

The screenshot shows the AWS S3 'Upload: status' page. It displays a green success message: 'Upload succeeded. For more information, see the Files and folders table.' The summary table shows one file uploaded: 'heart_disease_sample.csv' (Succeeded, 1.4 KB). The 'Close' button is visible at the top right.

1.5 Creating an Output Container

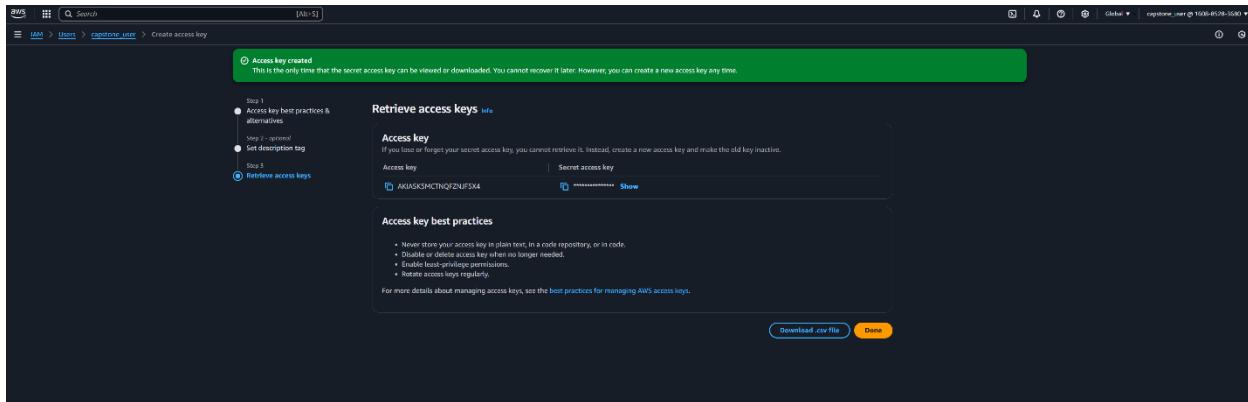
The screenshot shows the AWS S3 'Create folder' interface. A new folder named 'output' is being created. The 'Folder name' field contains 'output'. Under 'Server-side encryption', the 'Don't specify an encryption key' option is selected. A note at the bottom states: '⚠ If your bucket policy requires objects to be encrypted with a specific encryption key, you must specify the same encryption key when you create a folder. Otherwise, folder creation will fail.' The 'Create folder' button is located at the bottom right.

1.6 Creating the access key for mounting AWS with data bricks

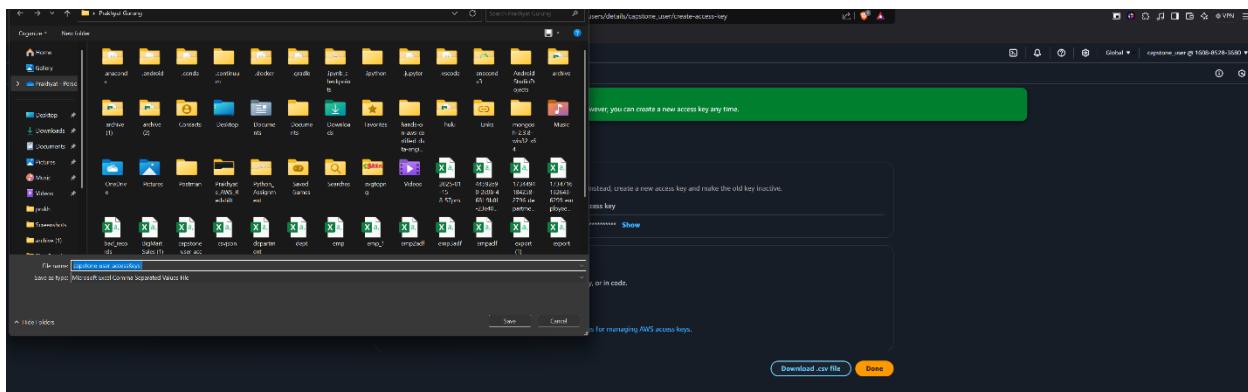
The screenshot shows the AWS IAM User Details page for a user named 'capstone_user'. The 'Security credentials' tab is active. In the 'Access keys (0)' section, there is a note: 'Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time.' Below this, a 'Create access key' button is highlighted.

The screenshot shows the 'Create access key' wizard, Step 1: 'Access key best practices & alternatives'. It lists several use cases: 'Command Line Interface (CLI)', 'Local code', 'Application running on an AWS compute service', 'Third-party service' (which is selected), and 'Application running outside AWS'. A note at the bottom states: 'AWS recommends using temporary security credentials (IAM roles) instead of creating long-term credentials like access keys, and don't create AWS account root user access keys.' A checkbox at the bottom indicates agreement with this recommendation.

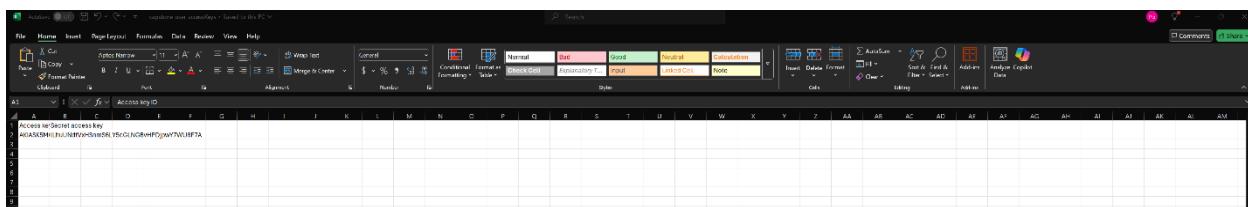
For creating access key for mounting AWS with data bricks we need to select Third-party service, so I selected the Third-party service.



The access key has been created.

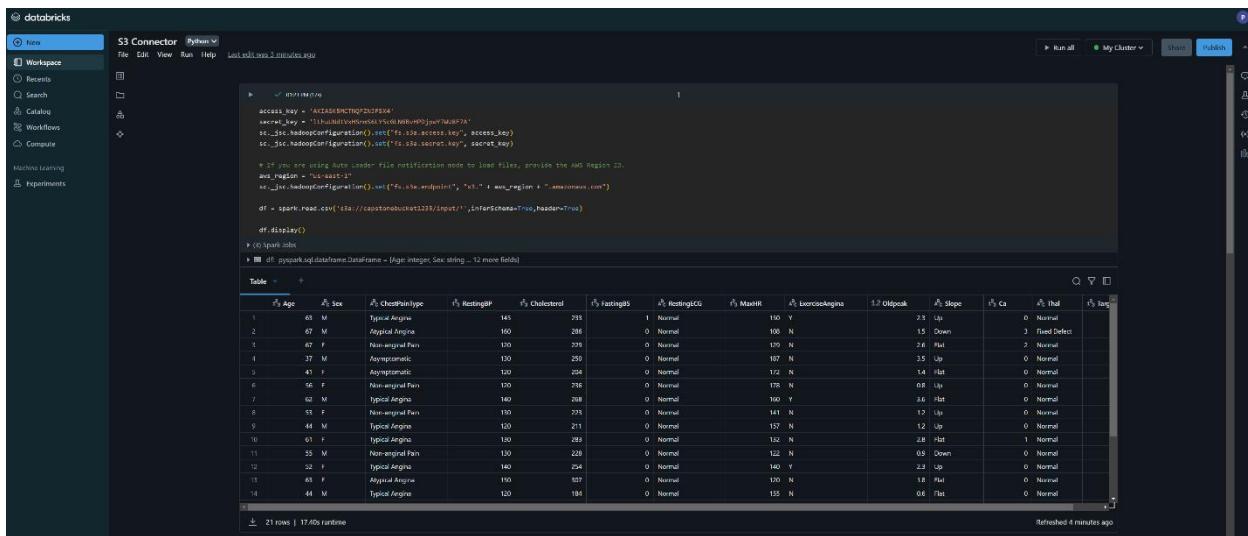


I downloaded the access key on my computer.



My access key and secret key is shown above on the picture.

1.7 Connecting the data bricks with the access key and the secret key



```
File Edit View Run Help Last edit was 2 minutes ago Run all My Cluster Share Publish
```

```
access_key = "AKIAZASHNHCNPQZC57P3KA"
secret_key = "11hduv4t0v0w6v6VnM6u0jjuw7dJ87K"
s3_sec_accesskey = fs.s3.access_key, access_key
s3_sec_secretkey = fs.s3.secret_key, secret_key

# If you are using Auto Loader file notification mode to load files, provide the AWS Region ID.
# and_region = "us-east-1"
# s3_sec_endpoint = "s3.us-east-1.amazonaws.com"

df = spark.read.csv("s3a://capstonebuckets1238/Input/*", inferSchema=True, header=True)

df.display()
```

| # | Age | Sex | chestPainType | HbA1c | Cholesterol | FastingB | FastingCG | MaxHR | ExerciseAngina | Oldpeak | Slope | Ca | thal | tag |
|----|-----|-----|------------------|-------|-------------|----------|-----------|-------|----------------|---------|-------|----|--------------|-----|
| 1 | 63 | M | Typical Angina | 145 | 295 | 0 | Normal | 150 | Y | 2.3 | Up | 0 | Normal | |
| 2 | 67 | M | Atypical Angina | 100 | 288 | 0 | Normal | 100 | N | 1.5 | Down | 3 | Fixed Defect | |
| 3 | 67 | F | Non-anginal Pain | 100 | 298 | 0 | Normal | 100 | N | 2.6 | Flat | 2 | Normal | |
| 4 | 37 | M | Asymptomatic | 100 | 250 | 0 | Normal | 107 | N | 3.5 | Up | 0 | Normal | |
| 5 | 41 | F | Atypical Angina | 100 | 256 | 0 | Normal | 122 | N | 3.4 | Flat | 0 | Normal | |
| 6 | 66 | F | Non-anginal Pain | 100 | 208 | 0 | Normal | 105 | N | 0.9 | Up | 0 | Normal | |
| 7 | 62 | M | Typical Angina | 100 | 288 | 0 | Normal | 100 | Y | 2.4 | Up | 0 | Normal | |
| 8 | 59 | F | Non-anginal Pain | 100 | 378 | 0 | Normal | 141 | N | 1.7 | Up | 0 | Normal | |
| 9 | 44 | M | Typical Angina | 100 | 213 | 0 | Normal | 157 | N | 1.2 | Up | 0 | Normal | |
| 10 | 61 | F | Typical Angina | 100 | 288 | 0 | Normal | 132 | N | 2.8 | Flat | 1 | Normal | |
| 11 | 55 | M | Non-anginal Pain | 100 | 228 | 0 | Normal | 122 | N | 0.5 | Down | 0 | Normal | |
| 12 | 52 | F | Typical Angina | 100 | 254 | 0 | Normal | 140 | Y | 2.3 | Up | 0 | Normal | |
| 13 | 68 | F | Atypical Angina | 100 | 107 | 0 | Normal | 120 | N | 1.8 | Flat | 0 | Normal | |
| 14 | 44 | M | Typical Angina | 100 | 184 | 0 | Normal | 135 | N | 0.6 | Flat | 0 | Normal | |

21 rows | 17.05s runtime Refreshed 4 minutes ago

2. Setting Up IAM Roles and Policies

2.1 Creating IAM Roles for AWS Glue

The screenshots illustrate the process of creating a new IAM role for AWS Glue:

- Screenshot 1: IAM Roles List**

This screenshot shows the IAM Roles list. It displays two existing roles: `AWSServiceRoleForSageMaker` and `AWSServiceRoleForTrustAdvisor`. The interface includes a search bar, a table header with columns for `Role name`, `Trusted entities`, and `Last activity`. A sidebar on the left provides navigation for Identity and Access Management.
- Screenshot 2: Create Role - Step 1: Select trusted entity**

This screenshot shows the first step of the "Create role" wizard. It asks to "Select trusted entity".
 - Trusted entity type:** Options include AWS service (selected), AWS account, Web identity, SAML 2.0 federation, and Custom trust policy.
 - Use case:** Options include Allow on AWS service like EC2, Lambda, or others to perform actions in this account (selected) and Service or use case (set to Glue).
 - Service or use case:** Set to Glue.Buttons for "Cancel" and "Next" are at the bottom.
- Screenshot 3: Create Role - Step 2: Add permissions**

This screenshot shows the second step of the "Create role" wizard. It asks to "Add permissions".
 - Permissions policies:** A search bar shows results for "glue". A list of AWS managed policies is displayed, with `AmazonGlueServiceRolePolicy` selected.
 - Set permissions boundary - optional:** A note states "Set a permissions boundary to limit the permissions granted by this role".
 - Buttons:** "Cancel", "Previous", and "Next" are at the bottom.

Screenshot of the AWS IAM 'Create role' wizard, Step 1: Select trusted entities.

Role details

- Role name:** AWSGlueSSAccessRole
- Description:** Allows glue to call AWS services on your behalf.

Step 1: Select trusted entities

Trust policy:

```
1 | { "Version": "2012-10-17", 2 | "Statement": [ 3 | { "Effect": "Allow", 4 | "Principal": "glue.amazonaws.com", 5 | "Action": "sts:AssumeRole" 6 | } 7 | ] 8 | } 9 | }
```

Step 2: Add permissions

Permissions policy summary:

| Policy name | Type | Attached as |
|--------------------|-------------|--------------------|
| AWSGlueSSAccess | AWS managed | Permissions policy |
| AWSGlueServiceRole | AWS managed | Permissions policy |

Step 3: Add tags

Add tags - optional

No tags associated with the resource.

Add new tag

You can add up to 50 tags.

Buttons: Cancel, Previous, Create role.

Screenshot of the AWS IAM Roles page.

Identity and Access Management (IAM)

Roles (3) info

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

| Role name | Trusted entities | Last activity |
|---------------------------------|------------------------------------|---------------|
| AWSGlueSSAccessRole | AWS Service: glue | - |
| AWSServiceRoleForSupport | AWS Service: support.Service-Lite | - |
| AWSServiceRoleForTrustedAdvisor | AWS Service: trustedadvisorService | - |

Roles Anywhere

Authenticates your non-AWS workloads and securely provide access to AWS services.

X.509 Standard

Use your own existing PKI infrastructure or use AWS Certificate Manager Private Certificate Authority to authenticate identities.

Temporary credentials

Use temporary credentials with ease and benefit from the enhanced security they provide.

Buttons: View role, Delete, Create role.

2.2 Creating IAM Roles for Databricks

Screenshot of the AWS IAM 'Create role' wizard, Step 1: Select trusted entity.

Role 1

- Select trusted entity:** AWS service
- Role 2**
- Add permissions**
- Role 3**
- Name, review, and create**

Select trusted entity

Trusted entity type:

- AWS service: Allow AWS services like EC2, Lambda, or others to perform actions in this account.
- AWS account: Allow entities in one AWS account belonging to you or a 3rd party to perform actions in this account.
- SAML 2.0 federation: Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
- Custom trust policy: Create a custom JSON policy to enable others to perform actions in this account.

Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Service or use case: EC2

Choose a use case for the specified service.

For EC2:

- EC2 Role for AWS Systems Manager
- EC2 Spot Test Role
- EC2 - Spot Fleet Auto Scaling
- EC2 - Spot Fleet Regions
- EC2 - Spot Instances
- EC2 - Scheduled Fleet
- EC2 - Scheduled Instances

Buttons: Cancel, Next.

Step 1: Add permissions

Choose one or more policies to attach to your new role.

| Policy name | Type | Description |
|--|-------------|--|
| AmazonS3FullAccess | AWS managed | Provides full access to all buckets via the ... |
| AmazonS3ReadOnlyAccess | AWS managed | Provides read only access to Amazon S3 ... |
| AmazonSQSReadOnlyAccess | AWS managed | Provides read only access to all buckets via ... |
| AmazonSShareWithAllAccess | AWS managed | Provides full access to all S3 table buckets. |
| AmazonSS3TableReadonlyAccess | AWS managed | Provides read only access to all S3 table ... |
| AmazonSS3BackupServiceRolePolicyForS3Backup | AWS managed | Policy containing permissions necessary f... |
| AmazonSS3BackupServiceRolePolicyForS3Restore | AWS managed | Policy containing permissions necessary f... |
| AmazonSQSSetS3DeploymentS3BucketRolePolicy | AWS managed | This policy grants permissions for listing ... |
| AmazonSSignInProviderStorageManagementRolePolicy | AWS managed | Policy used by QuickSight team to access ... |
| CloudWatchLogsFullAccess | AWS managed | Provides access required to unlock a S3 b... |

Step 2: Set permissions boundary - optional

Step 3: Name, review, and create

Role name: databricks-accessible

Description: Allows EC2 instances to call AWS services on your behalf.

Trust policy

```

1: {
2:   "Version": "2012-10-17",
3:   "Statement": [
4:     {
5:       "Effect": "Allow",
6:       "Action": "sts:AssumeRole",
7:       "Principal": [
8:         "arn:aws:iam::ec2.amazonaws.com"
9:       ]
10:      }
11:    ]
12:  }
13: }
14: }
15: }
16: }

```

Permissions policy summary

| Policy name | Type | Attached as |
|---------------------|-------------|--------------------|
| AmazonSS3FullAccess | AWS managed | Permissions policy |
| AmazonS3FullAccess | AWS managed | Permissions policy |

Step 4: Add tags

Add tags - optional

No tags are currently associated with this resource.

Create role

2.3 Editing the trust policy of data bricks

The screenshot shows the 'Edit trust policy' page for the 'DatabricksS3AccessRole'. The left panel displays the current trust policy in JSON format:

```
1 "Version": "2012-10-17",
2 "Statement": [
3     {
4         "Effect": "Allow",
5         "Principal": "*",
6         "Action": "sts:AssumeRole",
7         "ARN": "arn:aws:iam::160885283680:role/DatabricksS3AccessRole"
8     },
9     {
10        "Effect": "Allow",
11        "Action": "s3:ListBucket"
12    }
13 ]
```

The right panel contains a modal titled 'Edit statement' with the heading 'Select a statement'. It includes a button 'Add new statement' and a note 'Select an existing statement in the policy or add a new statement.' Below the modal are buttons for 'Check for new access', 'Preview external access', 'Cancel', and 'Update policy'.

The screenshot shows the 'Trust relationships' tab for the 'DatabricksS3AccessRole'. A green banner at the top indicates 'Trust policy updated.' The role summary includes the creation date (January 22, 2025, 14:47 (UTC-05:00)), ARN (arn:aws:iam::160885283680:role/DatabricksS3AccessRole), and a link to 'View roles in console'.

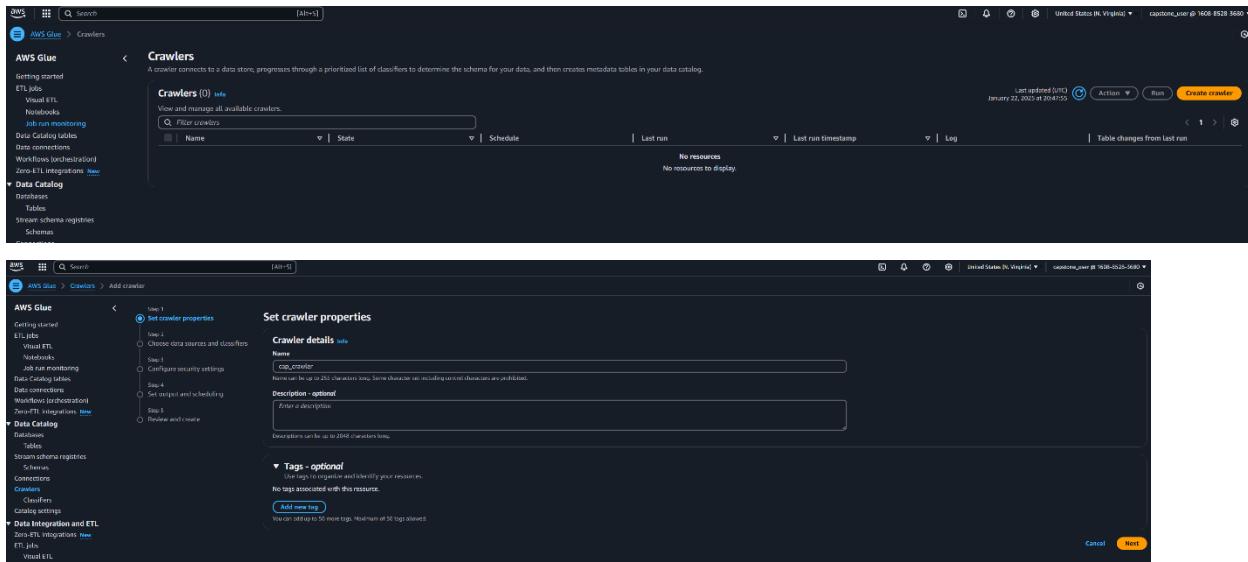
The 'Permissions' tab is selected, showing the trust policy document:

```
1 "Version": "2012-10-17",
2 "Statement": [
3     {
4         "Effect": "Allow",
5         "Principal": "*",
6         "Action": "sts:AssumeRole",
7         "ARN": "arn:aws:iam::160885283680:role/DatabricksS3AccessRole"
8     },
9     {
10        "Effect": "Allow",
11        "Action": "s3:ListBucket"
12    }
13 ]
```

Other tabs include 'Tags', 'Last Accessed', and 'Revoke sessions'. Buttons for 'Edit trust policy' and 'Delete' are visible.

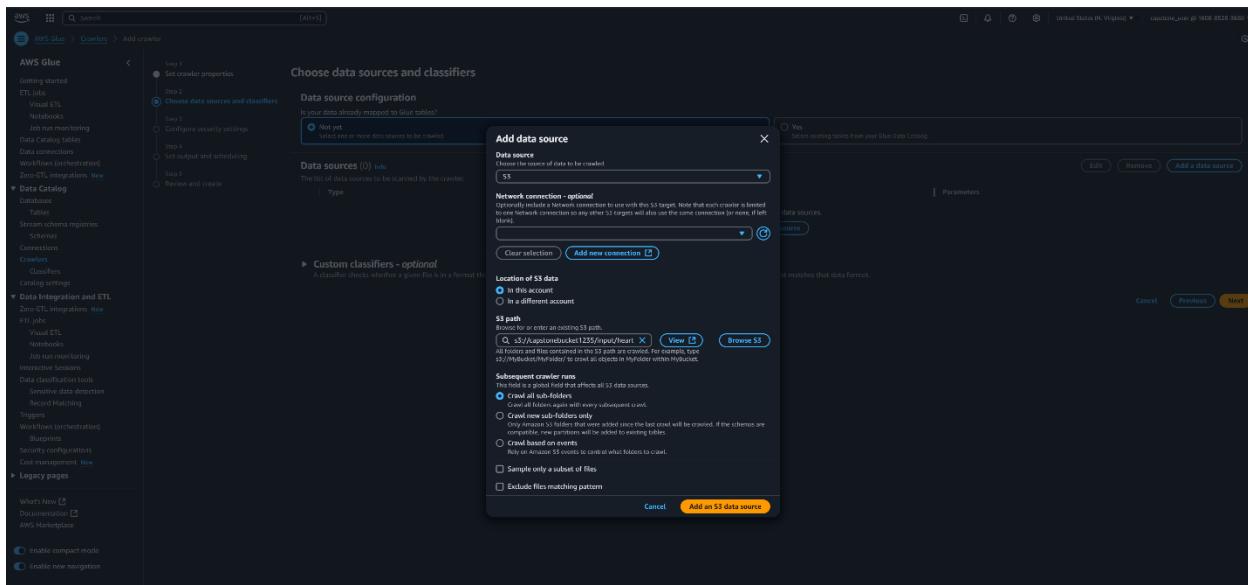
3. Setting Up AWS Glue ETL Job

3.1 Creating an AWS Glue Crawler



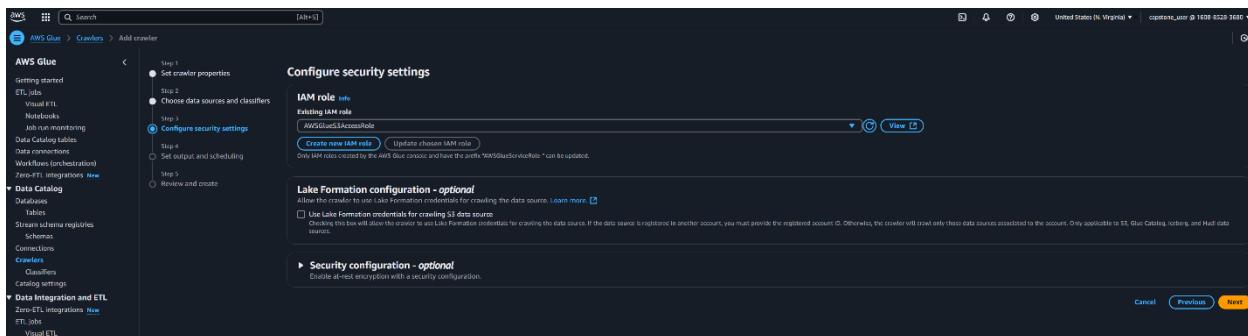
The screenshot shows the 'Add crawler' wizard in the AWS Glue console. Step 1: Set crawler properties. The crawler is named 'cap_crawler'. The 'Description (optional)' field contains 'Data to be crawled'. Under 'Tags - optional', there are no tags associated with this resource.

Creating an AWS Glue Crawler named (**cap_crawler**).

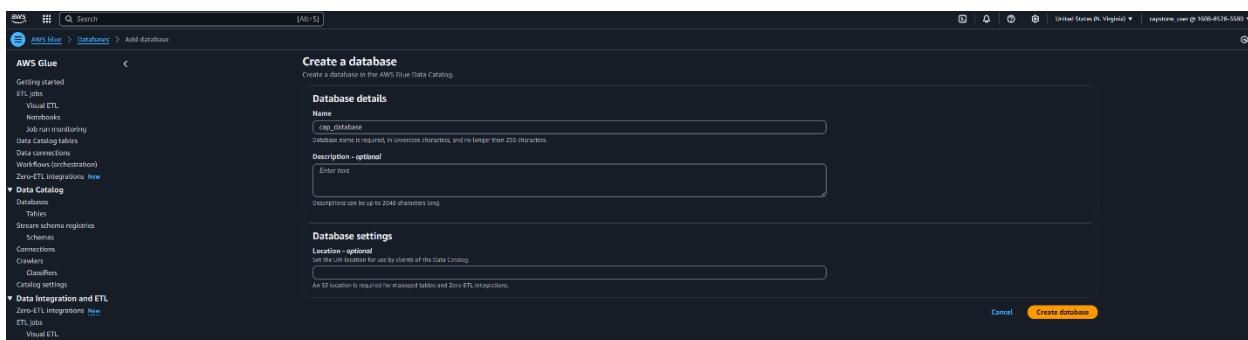


The screenshot shows the 'Choose data sources and classifiers' step of the wizard. The 'Data source configuration' section shows an S3 path: 's3://capstonebucket1235/input/heart_disease_sample.csv'. The 'Add data source' button is visible.

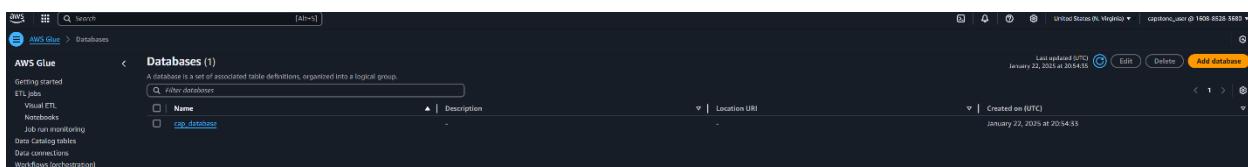
Setting up the S3 path (**s3://capstonebucket1235/input/heart_disease_sample.csv**).



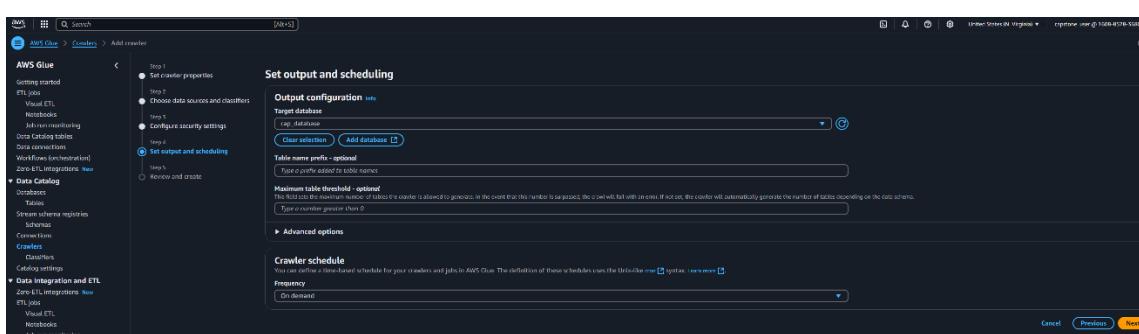
Selecting the created IAM role (**AWSGlueS3AccessRole**).



Creating the database named (**cap_database**).



Successfully created the database.



Selecting the database named (**cap_database**).

The screenshot shows the AWS Glue Crawler configuration page. The crawler is named 'cap_crawler'. It has an IAM role 'AWSGlueFullAccessRole' and is set to run in 'HEADLY' state. The crawler runs against a database 'cap_database' with a table prefix 'cap_'. There are no crawler runs listed.

Successfully created the AWS crawler.

The screenshot shows the AWS Glue Crawler configuration page. The crawler is now starting, indicated by the message 'Crawler successfully starting'. The status bar at the top shows 'Last updated 30s ago' and the date 'January 22, 2025 at 20:55:40'. The crawler runs section shows one run started on January 22, 2025, at 20:55:19, which is currently running.

Successfully started the AWS crawler.

3.2 Starting the process of ETL Job in AWS Console

The screenshot shows the AWS Glue Data Catalog interface. On the left, there's a navigation bar with tabs: Visual, Script, Job details, Runs, Data quality, Schedules, Version Control, and Upgrade analysis - preview. The 'Visual' tab is selected. In the center, there's a data source node labeled 'Data source - S3 bucket Amazon S3'. To the right, the 'Data source properties - S3' panel is open, showing the following configuration:

- Name:** Amazon S3
- S3 source type:** Info
- File location:** (View, Browse S3)
- Recurse:** Recurse into all subdirectories (checked)
- Data format:** CSV
- Delimiter:** Comma (,)
- Escape character - optional:** None (optional)
- Quote character:** Double quote (")
- First line of source file contains column headers:** (checked)
- Records in source files can span multiple lines:** (unchecked)
- Infer schema:** (checked)

At the bottom of the panel, there are 'Additional options' and a 'Save' button.

Below the data source, there's a 'Data preview' section showing a sample of 14 rows from the CSV file. The columns include age, sex, chestpainstype, restingbp, cholesterol, fastings, restingecg, and maxhr. The data preview table looks like this:

| age | sex | chestpainstype | restingbp | cholesterol | fastings | restingecg | maxhr |
|-----|-----|------------------|-----------|-------------|----------|------------|-------|
| 63 | M | Typical Angina | 145 | 233 | 1 | Normal | 150 |
| 67 | M | Atypical Angina | 160 | 266 | 0 | Normal | 108 |
| 67 | F | Non-anginal Pain | 120 | 229 | 0 | Normal | 129 |
| 57 | M | Asymptomatic | 130 | 250 | 0 | Normal | 187 |
| 41 | F | Asymptomatic | 120 | 204 | 0 | Normal | 172 |
| 56 | F | Non-anginal Pain | 120 | 236 | 0 | Normal | 178 |

The screenshot shows the AWS Glue Data Catalog interface with a different job configuration. The 'Visual' tab is selected. A data source node is connected to a transform node labeled 'Transform: DropDuplicates - Drop Duplicates'. To the right, the 'Transform' panel is open, showing the following configuration:

- Name:** Drop Duplicates
- Node parents:** Data source node provides input for this one
- Output:** Create new or reuse parent node (Amazon S3 - SV - Structure)
- Drop duplicates:**
 - Info: Detect pairs of repeating values from your data
 - Match entire rows (Default)**: Detect pairs of rows that contain the same values across all columns
 - Match specific keys**: Detect pairs of rows that contain the same values in specific columns

Below the transform, there's a 'Data preview' section showing a sample of 14 rows from the CSV file. The columns include age, sex, chestpainstype, restingbp, cholesterol, fastings, restingecg, and maxhr. The data preview table looks like this:

| age | sex | chestpainstype | restingbp | cholesterol | fastings | restingecg | maxhr |
|-----|-----|------------------|-----------|-------------|----------|------------|-------|
| 63 | M | Typical Angina | 145 | 233 | 1 | Normal | 150 |
| 57 | F | Typical Angina | 140 | 254 | 0 | Normal | 140 |
| 66 | M | Typical Angina | 170 | 184 | 0 | Normal | 155 |
| 69 | F | Non-anginal Pain | 130 | 240 | 0 | Normal | 158 |
| 67 | M | Atypical Angina | 160 | 286 | 0 | Normal | 168 |
| 57 | F | Non-anginal Pain | 120 | 229 | 0 | Normal | 129 |

Successfully updated job CapStone_ETL. To run the job choose the Run Job button.

Last modified on 1/22/2025, 4:46:55 PM Actions Save Run

CapStone_ETL

Visual Script Job details Runs Data quality Schedules Version Control Upgrade analysis - preview

Data preview (21) Info READY

Output schema

| Normalized_Age | Cholesterol | Na | Thal | Target | Cs | Slope | Oldpeak | ExerciseAngina | Na |
|--------------------|-------------|----|-------------|--------|----|-------|---------|----------------|----|
| 0.8666666666666667 | 0 | 0 | Normal | 1 | 0 | Up | 2.5 | Y | Y |
| 0.5 | 1 | 0 | Normal | 0 | 0 | Up | 2.5 | Y | Y |
| 0.2555555555555554 | 0 | 0 | Normal | 0 | 0 | Flat | 0.6 | N | N |
| 0.6 | 0 | 0 | Normal | 0 | 0 | Up | 2.1 | N | N |
| 1 | 1 | 5 | Pred Defect | 1 | 1 | Down | 1.5 | N | N |

Successfully updated job CapStone_ETL. To run the job choose the Run Job button.

Last modified on 1/22/2025, 4:46:55 PM Actions Save Run

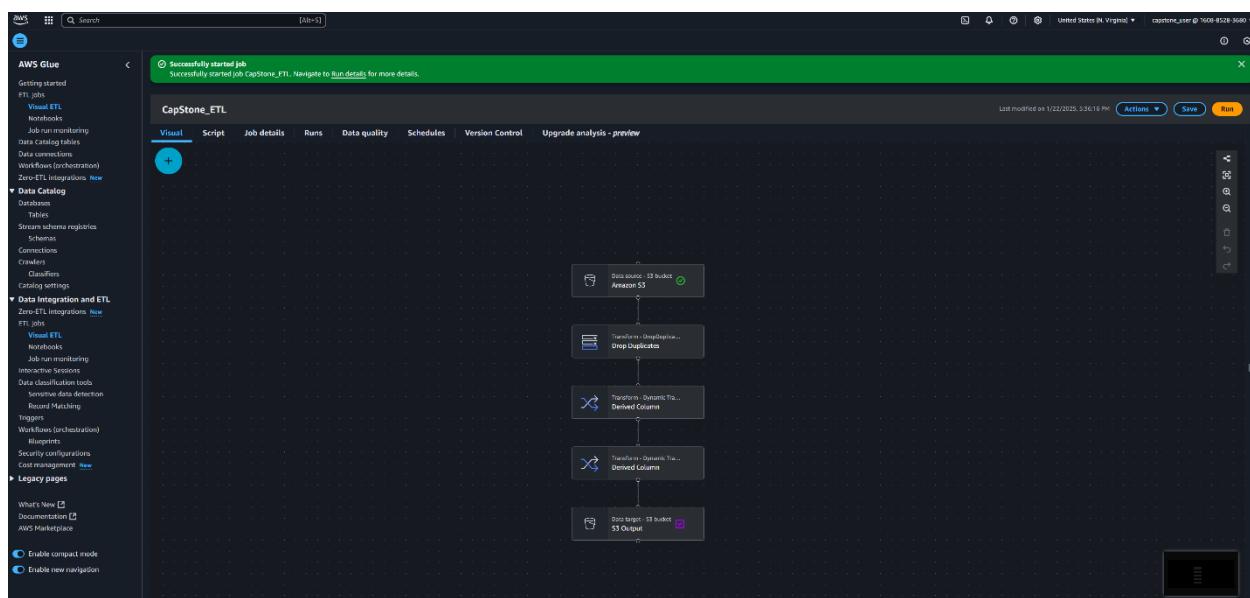
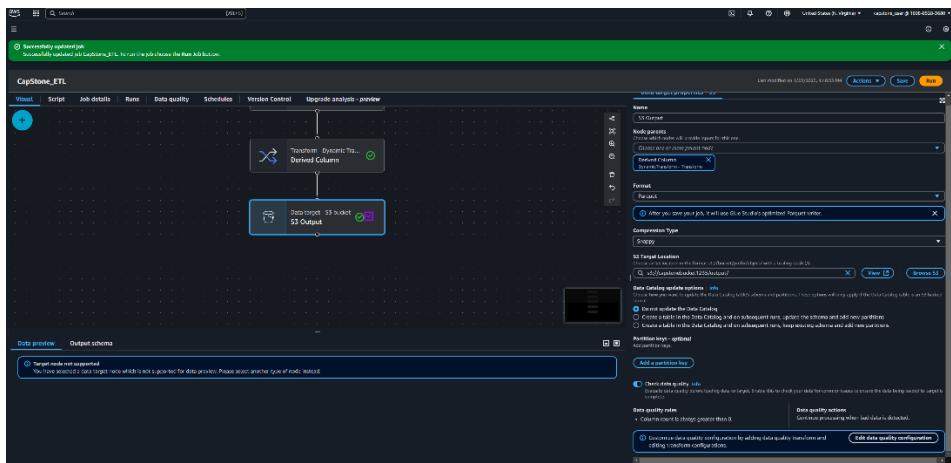
CapStone_ETL

Visual Script Job details Runs Data quality Schedules Version Control Upgrade analysis - preview

Data preview (21) Info READY

Output schema

| Normalized_Cholesterol | Na | Thal | Target | Normalized_Age | Cs | Slope | Oldpeak | ExerciseAngina | Na |
|------------------------|--------------------|------|--------|--------------------|----|-------|---------|----------------|----|
| -0.4605194040735978 | 0.8666666666666667 | 0 | 1 | 0.8666666666666667 | 0 | Up | 2.5 | Y | Y |
| 0.2097276463980104 | 0.5 | 0 | 1 | 0.5 | 0 | Up | 2.5 | Y | Y |
| -2.024958257860025 | 0.2555555555555554 | 0 | 0 | 0.2555555555555554 | 0 | Flat | 0.6 | N | N |
| -0.23715705862558018 | 0.4 | 0 | 0 | 0.4 | 0 | Up | 2.1 | N | N |
| 0.7670721604944644 | 1 | 1 | 1 | 1 | 1 | Down | 1.5 | N | N |



Successfully started job

Successfully started job CapStone_ETL. Navigate to Run details for more details.

CapStone_ETL

Script Job details Runs Data quality Schedules Version Control Upgrade analysis - preview

| Run status | Retries | Start time (Local) | End time (Local) | Duration | Capacity (DPU) | Worker type | Glue version |
|------------|---------|---------------------|---------------------|----------|----------------|-------------|--------------|
| Running | 0 | 01/22/2025 17:51:47 | - | 1 m 46 s | 10 DPU | G.1X | 5.0 |
| Succeeded | 0 | 01/22/2025 17:56:15 | 01/22/2025 17:58:08 | 1 m 47 s | 10 DPU | G.1X | 5.0 |
| Succeeded | 0 | 01/22/2025 17:56:28 | 01/22/2025 17:58:18 | 1 m 43 s | 10 DPU | G.1X | 5.0 |
| Succeeded | 0 | 01/22/2025 17:33:15 | 01/22/2025 17:35:15 | 1 m 50 s | 10 DPU | G.1X | 5.0 |

Run details

Last modified on 1/22/2025, 2:36:18 PM Actions Save Run

| Job name | Start time (Local) | End time (Local) | Duration | Capacity (DPU) | Worker type | Glue version |
|--|---------------------|---------------------|----------|----------------|-------------|--------------|
| CapStone_ETL | 01/22/2025 17:33:15 | 01/22/2025 17:35:15 | 1 m 50 s | 10 DPU | G.1X | 5.0 |
| p_48f910fe8e4923c6d21fb5a72ed79a549528c10337ebcaab4de4fb1b5af270 | 01/22/2025 17:41:08 | - | - | - | - | - |
| Run status | Start time | End time | Duration | Capacity | Worker type | Glue version |
| Succeeded | 7 seconds | - | - | - | - | - |
| Execution number | - | - | - | - | - | - |
| Initial run | - | - | - | - | - | - |
| Trigger name | - | - | - | - | - | - |

Continuous logs

Metrics

Troubleshooting analysis - preview

Spark UI

Last modified on 1/22/2025, 2:36:18 PM Actions Save Run

Screenshot of the AWS S3 Object Details page for object `run-1737587471519-part-r-00000`.

Object overview

- Owner: `djw1974@gmail.com`
- AWS Region: US East (N. Virginia) - us-east-1
- Last modified: January 22, 2025, 18:11:28 (UTC-05:00)
- Size: 7.2 KB
- Type: `text/plain`
- Key: `outputs/run-1737587471519/part-r-00000`

Object management overview

The following table provides an object management configuration for the file or object.

| Bucket properties | Object properties |
|-------------------|--|
| Bucket Versioning | Enabled: <code>run-1737587471519</code> has Bucket Versioning enabled. You must enable Bucket Versioning if you plan to automatically versioning or deleting objects. Learn more |

Management configurations

Replication status: `None` (No replication rules are applied to this object.)

View replication rules

Expiration rule: `None` (No expiration rule is applied to this object.)

View expiration rules

Encryption status: `None` (No encryption is applied to this object.)

View encryption rules

Screenshot of the AWS S3 Bucket Properties dialog for bucket `capstonebucket1235`.

Bucket properties

Bucket Versioning: Enabled

Management configurations

Replication status: `None` (No replication rules are applied to this object.)

View replication rules

Encryption status: `None` (No encryption is applied to this object.)

View encryption rules

Screenshot of the AWS S3 Object Details page for object `run-1737587471519-part-r-00000`.

Object overview

- Owner: `djw1974@gmail.com`
- AWS Region: US East (N. Virginia) - us-east-1
- Last modified: January 22, 2025, 18:11:28 (UTC-05:00)
- Size: 7.2 KB
- Type: `text/plain`
- Key: `outputs/run-1737587471519/part-r-00000`

Management configurations

Replication status: `None` (No replication rules are applied to this object.)

View replication rules

Screenshot of the AWS S3 Object Details page for object `run-1737587471519-part-r-00000`.

Object overview

- Owner: `djw1974@gmail.com`
- AWS Region: US East (N. Virginia) - us-east-1
- Last modified: January 22, 2025, 18:11:28 (UTC-05:00)
- Size: 7.2 KB
- Type: `text/plain`
- Key: `outputs/run-1737587471519/part-r-00000`

Management configurations

Replication status: `None` (No replication rules are applied to this object.)

View replication rules

3.3 Scheduling the job run

The screenshot shows the AWS Glue console interface for the 'CapStone_ETL' job. The left sidebar navigation includes 'AWS Glue', 'Getting started', 'ETL jobs', 'Visual ETL', 'Notebooks', 'Job run monitoring', 'Data Catalog tables', 'Data connections', 'Workflows (orchestration)', 'Zero-ETL integrations', 'Data Catalog', 'Databases', 'Tables', 'Stream schema registries', 'Schemas', 'Connections', 'Providers', 'Classifiers', 'Catalog settings', 'Data Integration and ETL', 'Zero-ETL integrations', and 'ETL jobs'. The 'Schedules' tab is selected. A search bar at the top says 'Filter schedules'. Below it, a message says 'No schedules' and 'No schedules available'. A 'Create schedule' button is visible.

The screenshot shows the 'Schedule job run' configuration dialog. The 'Schedule parameters' section includes a 'Name' field set to 'Job_Join', a 'Frequency' dropdown set to 'hourly', and a 'Minute of the hour' input field set to '59'. There is also a 'Description - optional' text area. A note below states: 'Description options can be up to 2548 characters long.' At the bottom are 'Cancel' and 'Create schedule' buttons.

The screenshot shows the 'Schedules' tab for the 'CapStone_ETL' job. A green success message at the top says 'Successfully updated job' and 'Successfully updated job CapStone_ETL. To run the job choose the Run job button.' The 'Schedules' list now includes a single entry: 'cap_run'. The details for 'cap_run' are: 'Description' (empty), 'Schedule' (At 59 minutes past the hour), 'Cron Expression' (59 0/1 * * *), and 'Status' (Active). The right side of the screen shows a message 'Last modified on 1/22/2025, 6:09:40 PM' and 'Actions' buttons.

3.4 Displaying the CloudWatch Logs

AWS Glue > Monitoring > Job run

Job Run - jr_254cd2ebda60aeab814c9859400d6fe14f79d6975ba56686645c496f3f66ceb3

Run details [Info](#)

jr_254cd2ebda60aeab814c9859400d6fe14f79d6975ba56686645c496f3f66ceb3

Job name: CapStone_ETL
Run ID: jr_254cd2ebda60aeab814c9859400d6fe14f79d6975ba56686645c496f3f66ceb3
Run status: Submitted
Glue version: 2.0

Retry attempt number: 0
Initial run: 01/22/2023 19:49:39
End time (UTC): 01/23/2023 00:51:26
Start time (UTC): 01/22/2023 19:51:26
Last modified on (UTC): 2023/01/23 00:51:26
Trigger name: -
Security configuration: -
Timeout: 480 minutes
Execution time: 1 minute 49 seconds
Last modified on (Local): 01/22/2023 19:51:26
Worker type: G1X
Execution class: Standard
Usage profile: -
Log group name: /aws-glue/obs
CloudWatch logs: [All logs](#) [Output logs](#) [Error logs](#) [View in CloudWatch Logs](#)

Continuous logs [Info](#)

Driver log

INFO 2023-01-23T00:50:26,684 [20245] org.apache.spark.SparkContext [Thread-2] Added file /tmp/glue/job-86540074202835570/extr... at spark://172.35.33.37:15157/file/a...
INFO 2023-01-23T00:50:26,684 [20245] org.apache.spark.util.Utils [Thread-2] Copying /tmp/glue/job-86540074202835570/extr... to /tmp/spark-3df04...-0009-/1788-400b-fc...
INFO 2023-01-23T00:50:26,684 [20245] org.apache.spark.util.Utils [Thread-2] Added file /tmp/glue/job-86540074202835570/extr... at spark://172.35.33.37:15157/file/b...
INFO 2023-01-23T00:50:26,684 [20245] org.apache.spark.SparkContext [Thread-2] Added JAR /tmp/glue/job-86540074202835570/jars/MWCN-AwsDataProcessor.jar at spark://172.35.33.57:15557/jar/s...
INFO 2023-01-23T00:50:26,684 [20245] org.apache.spark.SparkContext [Thread-2] Added JAR /tmp/glue/job-86540074202835570/jars/glib-glib-di-package-0.2.00.jar at spark://172.35.33.57:15557/jar/s...
INFO 2023-01-23T00:50:26,688 [20245] org.apache.spark.SparkContext [Thread-2] Registering OutputCommitterInfoSource for all sources are available.
INFO 2023-01-23T00:50:26,688 [20245] org.apache.spark.storage.memory.MemoryStore [Thread-2] MemoryStore started with capacity 3.0 GB
INFO 2023-01-23T00:50:26,688 [20245] org.apache.spark.storage.BlockManager [Thread-2] Created BlockManager at spark://172.35.33.37:15157/10d0d3...
INFO 2023-01-23T00:50:26,688 [20245] org.apache.spark.storage.BlockManager [Thread-2] Registering BlockManagerInfoSource
INFO 2023-01-23T00:50:26,688 [20245] org.apache.spark.storage.BlockManagerMasterEndpoint [Thread-2] BlockManagerMasterEndpoint
INFO 2023-01-23T00:50:26,688 [20245] org.apache.spark.storage.BlockManagerMasterEndpoint [Thread-2] Using org.apache.spark.storage.DefaultTopologyReplicator for getting topology information
INFO 2023-01-23T00:50:26,688 [20245] org.apache.spark.storage.BlockManagerMasterEndpoint [Thread-2] Starting BlockManagerMasterEndpoint
INFO 2023-01-23T00:50:26,688 [20245] org.apache.spark.SparkContext [Thread-2] BlockManagerMasterEndpoint
INFO 2023-01-23T00:50:26,688 [20245] org.apache.spark.SparkContext [Thread-2] Successfully started service 'sparkDriver' on port 3337.
INFO 2023-01-23T00:50:26,688 [20245] org.apache.spark.SecurityManager [Thread-2] SecurityManager: authentication enabled; ui access disabled; users with view permissions: hadoop0; groups: _
INFO 2023-01-23T00:50:26,688 [20245] org.apache.spark.SecurityManager [Thread-2] Checking if group hadoop0 ...

Driver and executor log streams [View](#)

Input arguments (11)

| Key | Value |
|-----------------|-------|
| -enable-metrics | true |

4. Analyzing Data Using Databricks (PySpark and SQL)

4.1 Starting the ETL process on Databricks

The screenshot shows a Databricks workspace with a Python notebook titled "CapStone Project". The notebook contains the following code:

```
access_key = "ARIAZASDCTQ21H5VA"
secret_key = "1lnuhiuytuvweds1v0iM9hnuP0pw7Xn0PfJN"
sc._jsc.hadoopConfiguration().set("fs.s3.access.key", access_key)
sc._jsc.hadoopConfiguration().set("fs.s3.secret.key", secret_key)

# If you are using Auto Loader file notification mode to load files, provide the AWS Region ID.
aws_region = "us-east-1"
sc._jsc.hadoopConfiguration().set("fs.s3.endpoint", "s3." + aws_region + ".amazonaws.com")

df = spark.read.csv("s3://capstonebucket1105/input/*", inferSchema=True, header=True)

df.display()
```

The output of the `display` command shows a table with 13 columns and 21 rows of data. The columns are:

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | |
|-----|-----|------------------|-----------|-------------|-----------|---|-----------|---|-----|------|----|------------|---|
| Age | Sex | ChestPainType | RestingBP | Cholesterol | FastingBS | <th>HeartRate</th> <thexerciseangina< th=""><tholdpeak< th=""><thslope< th=""><thca< th=""><ththal< th=""><thtarget< th=""></thtarget<></ththal<></thca<></thslope<></tholdpeak<></thexerciseangina<> | HeartRate | | | | | | |
| 53 | F | Non-anginal Pain | 130 | 232 | 0 | Normal | 141 | N | 1.2 | Up | 0 | Normal | 0 |
| 44 | M | Typical Angina | 120 | 211 | 0 | Normal | 157 | N | 1.2 | Up | 0 | Normal | 0 |
| 61 | F | Typical Angina | 130 | 203 | 0 | Normal | 132 | N | 2.8 | Flat | 1 | Normal | 0 |
| 55 | M | Non-anginal Pain | 130 | 238 | 0 | Normal | 127 | N | 0.5 | Down | 0 | Normal | 0 |
| 52 | F | Typical Angina | 140 | 254 | 0 | Normal | 140 | Y | 2.3 | Up | 0 | Normal | 0 |
| 68 | F | Atypical Angina | 150 | 207 | 0 | Normal | 120 | N | 1.8 | Flat | 0 | Normal | 0 |
| 44 | M | Typical Angina | 120 | 154 | 0 | Normal | 155 | N | 0.6 | Flat | 0 | Normal | 0 |
| 49 | F | Non-anginal Pain | 130 | 240 | 0 | Normal | 158 | N | 2.1 | Up | 0 | Normal | 0 |
| 64 | M | Typical Angina | 145 | 245 | 0 | Normal | 157 | Y | 2.5 | Up | 0 | Normal | 0 |
| 51 | M | Non-anginal Pain | 130 | 203 | 0 | Normal | 149 | N | 1.7 | Flat | 0 | Normal | 0 |
| 65 | F | Typical Angina | 120 | 229 | 0 | Normal | 134 | N | 3.1 | Up | 1 | Normal | 0 |
| 58 | M | Asymptomatic | 110 | 239 | 0 | Normal | 173 | N | 2.2 | Flat | 0 | Normal | 0 |
| 25 | X | Atypical Angina | 140 | 270 | 0 | Normal | 136 | N | 2.8 | Flat | 2 | Bad Defect | 0 |
| 57 | M | Typical Angina | 140 | 265 | 1 | Normal | 158 | Y | 1.5 | Down | 0 | Normal | 0 |

Below the table, it says "21 rows | 2.44s runtime" and "Refreshed 12 hours ago".

The screenshot shows a Databricks workspace with a Python notebook titled "CapStone Project". The notebook contains the following code:

```
root
|> age: integer (nullable = true)
|> sex: string (nullable = true)
|> ChestPainType: string (nullable = true)
|> RestingBP: integer (nullable = true)
|> Cholesterol: integer (nullable = true)
|> FastingBS: integer (nullable = true)
|> RestingECG: string (nullable = true)
|> HeartRate: integer (nullable = true)
|> ExerciseAngina: string (nullable = true)
|> Oldpeak: double (nullable = true)
|> Slope: string (nullable = true)
|> Ca: integer (nullable = true)
|> Thal: string (nullable = true)
|> target: integer (nullable = true)
```

```
df = df.withColumnRenamed("ca", "CoronaryArtery")
df = df.withColumnRenamed("thal", "ThalValue")
```

```
df.display()
```

The output of the `display` command shows a table with 13 columns and 14 rows of data. The columns are:

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | |
|-----|-----|------------------|-----------|-------------|-----------|---|-----------|---|-----|------|----|--------------|---|
| Age | Sex | ChestPainType | RestingBP | Cholesterol | FastingBS | <th>HeartRate</th> <thexerciseangina< th=""><tholdpeak< th=""><thslope< th=""><thca< th=""><ththal< th=""><thtarget< th=""></thtarget<></ththal<></thca<></thslope<></tholdpeak<></thexerciseangina<> | HeartRate | | | | | | |
| 82 | M | Typical Angina | 145 | 232 | 1 | Normal | 150 | Y | 1.5 | Down | 0 | Normal | 0 |
| 67 | M | Atypical Angina | 160 | 296 | 0 | Normal | 158 | N | 1.5 | Up | 3 | Fried Defect | 0 |
| 97 | F | Non-anginal Pain | 120 | 228 | 0 | Normal | 129 | N | 2.8 | Flat | 2 | Normal | 0 |
| 97 | M | Asymptomatic | 110 | 295 | 0 | Normal | 187 | N | 3.5 | Up | 0 | Normal | 0 |
| 41 | F | Atypical Angina | 120 | 204 | 0 | Normal | 172 | N | 1.4 | Flat | 0 | Normal | 0 |
| 56 | F | Non-anginal Pain | 120 | 238 | 0 | Normal | 178 | N | 0.8 | Up | 0 | Normal | 0 |
| 62 | M | Typical Angina | 140 | 268 | 0 | Normal | 160 | Y | 2.8 | Flat | 0 | Normal | 0 |
| 53 | F | Non-anginal Pain | 130 | 293 | 0 | Normal | 141 | N | 1.2 | Up | 0 | Normal | 0 |
| 44 | M | Typical Angina | 120 | 211 | 0 | Normal | 157 | N | 1.2 | Up | 0 | Normal | 0 |
| 61 | F | Typical Angina | 130 | 283 | 0 | Normal | 132 | N | 2.8 | Flat | 1 | Normal | 0 |
| 55 | M | Non-anginal Pain | 130 | 228 | 0 | Normal | 122 | N | 0.5 | Down | 0 | Normal | 0 |
| 52 | F | Typical Angina | 140 | 254 | 0 | Normal | 140 | Y | 2.8 | Up | 0 | Normal | 0 |
| 81 | F | Atypical Angina | 150 | 197 | 0 | Normal | 170 | N | 1.8 | Flat | 0 | Normal | 0 |
| 44 | M | Typical Angina | 120 | 184 | 0 | Normal | 155 | N | 0.6 | Flat | 0 | Normal | 0 |

Below the table, it says "14 rows | 12.00s runtime" and "Refreshed 12 hours ago".

CapStone Project Python

```

from pyspark.sql import functions as F

# Summary statistics using converted alias
df.select(
    F.mean("Age").alias("Avg Age"),
    F.stddev("Age").alias("Standard Deviation Age"),
    F.max("Age").alias("Max Age"),
    F.min("Age").alias("Min Age")
).show()

```

1 row | 1.20s runtime Refreshed 12 hours ago

Table

| | i ₀ Age | i ₀ Sex | i ₀ ChestPainType | i ₀ RestingBP | i ₀ Cholesterol | i ₀ FastingBS | i ₀ RestingECG | i ₀ MaxHR | i ₀ ExerciseAngina | i ₀ Oldpeak | i ₀ Slope | i ₀ CoronaryVessel | i ₀ Thal |
|---|--------------------|--------------------|------------------------------|--------------------------|----------------------------|--------------------------|---------------------------|----------------------|-------------------------------|------------------------|----------------------|-------------------------------|---------------------|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

No rows returned

1 row | 0.95s runtime Refreshed 12 hours ago

CapStone Project Python

```

# Handling missing values
df = df.dropna()
df.show()

```

1 row | 0.38s runtime Refreshed 12 hours ago

Table

| | i ₀ Age | i ₀ Sex | i ₀ ChestPainType | i ₀ RestingBP | i ₀ Cholesterol | i ₀ FastingBS | i ₀ RestingECG | i ₀ MaxHR | i ₀ ExerciseAngina | i ₀ Oldpeak | i ₀ Slope | i ₀ CoronaryVessel | i ₀ Thal |
|----|--------------------|--------------------|------------------------------|--------------------------|----------------------------|--------------------------|---------------------------|----------------------|-------------------------------|------------------------|----------------------|-------------------------------|---------------------|
| 1 | 61 M | Typical Angina | 145 | 233 | 1 | Normal | 150 Y | 210 | 2.8 Up | 0 | Normal | 0 | Normal |
| 2 | 67 M | Atypical Angina | 160 | 206 | 0 | Normal | 138 N | 215 Down | 3.6 Flat | 0 | Normal | 3 Fixed Defect | |
| 3 | 67 F | Non-anginal Pain | 130 | 298 | 0 | Normal | 129 N | 216 Up | 3.5 Up | 0 | Normal | 2 Normal | |
| 4 | 37 M | Asymptomatic | 130 | 256 | 0 | Normal | 137 N | 210 | 1.5 Up | 0 | Normal | 0 Normal | |
| 5 | 41 F | Asymptomatic | 120 | 204 | 0 | Normal | 172 N | 214 | 1.4 Flat | 0 | Normal | 0 Normal | |
| 6 | 56 F | Non-anginal Pain | 130 | 216 | 0 | Normal | 178 N | 208 Up | 0 | Normal | 0 Normal | 0 Normal | |
| 7 | 62 M | Typical Angina | 140 | 268 | 0 | Normal | 160 Y | 216 Up | 3.6 Flat | 0 | Normal | 0 Normal | |
| 8 | 51 F | Non-anginal Pain | 130 | 293 | 0 | Normal | 141 N | 217 Up | 0 | Normal | 0 Normal | 0 Normal | |
| 9 | 44 M | Typical Angina | 120 | 211 | 0 | Normal | 157 N | 212 Up | 1.2 Up | 0 | Normal | 0 Normal | |
| 10 | 61 F | Typical Angina | 130 | 283 | 0 | Normal | 132 N | 208 Flat | 0 | Normal | 1 Normal | 0 Normal | |
| 11 | 55 M | Non-anginal Pain | 130 | 228 | 0 | Normal | 122 N | 09 Down | 0 | Normal | 0 Normal | 0 Normal | |
| 12 | 52 F | Typical Angina | 140 | 254 | 0 | Normal | 140 Y | 212 Up | 0 | Normal | 0 Normal | 0 Normal | |
| 13 | 61 F | Atypical Angina | 150 | 207 | 0 | Normal | 150 N | 18 Flat | 0 | Normal | 0 Normal | 0 Normal | |
| 14 | 64 M | Typical Angina | 120 | 194 | 0 | Normal | 155 N | 06 Flat | 0 | Normal | 0 Normal | 0 Normal | |

21 rows | 0.38s runtime Refreshed 12 hours ago

```

df.describe().show()

```

(2) spark jobs

| [summary] | Age | Sex | ChestPainType | RestingBP | Cholesterol | FastingBS | RestingECG | MaxHR | ExerciseAngina | Oldpeak | Slope | CoronaryVessel | Thal | Target |
|-----------|--------------------|----------|-------------------|-------------------|--------------------|---------------------|---------------------|---------------------|----------------|----------------------|----------|--------------------|----------|----------|
| count | 21 | 21 | 21 | 21 | 21 | 21 | 21 | 21 | 21 | 21 | 21 | 21 | 21 | 21 |
| mean | 64.408571425717505 | 0.511111 | 71.03085714257147 | 147.428677449054 | 149.51889571425714 | 0.49521889571425714 | 0.49521889571425714 | 154.931589571425714 | 0.111111 | 0.09447610947610947 | 0.111111 | 0.111111 | 0.111111 | 0.111111 |
| stddev | 14.36378749088834 | 0.000000 | 14.94839785657131 | 31.32342683538462 | 31.30075243791213 | 0.39975243791213 | 0.39975243791213 | 12.82468275577443 | 0.111111 | 0.010353535353535353 | 0.111111 | 0.0791355635574595 | 0.111111 | 0.111111 |
| min | 37 | 1 | Atypical Angina | 110 | 158 | 0 | Normal | 106 | 0 | 0.6 Down | 0 | Fixed Defect | 0 | 0 |
| max | 61 | 0 | Typical Angina | 194 | 340 | 1 | Normal | 187 | Y | 3.6 Up | 3 | Normal | 3 | 1 |

CapStone Project Python

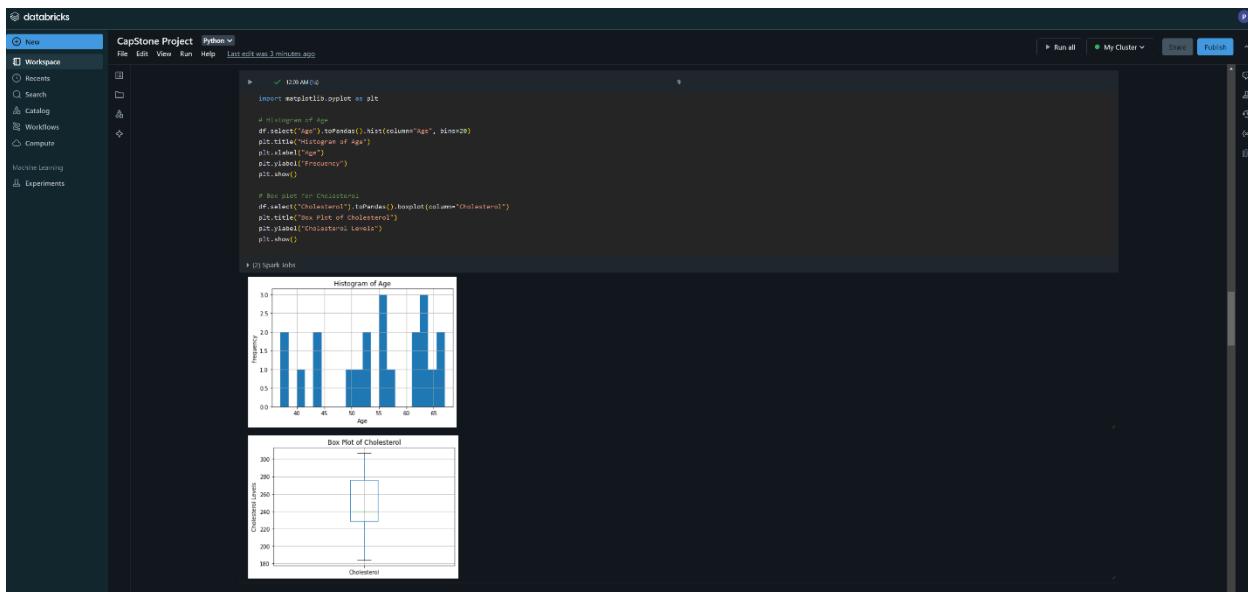
```

from pyspark.sql import functions as F

# Summary statistics using converted alias
df.select(
    F.mean("Age").alias("Avg Age"),
    F.stddev("Age").alias("Standard Deviation Age"),
    F.max("Age").alias("Max Age"),
    F.min("Age").alias("Min Age")
).show()

```

1 row | 0.38s runtime Refreshed 12 hours ago



The screenshot shows a Databricks workspace titled "CapStone Project" with a PySpark SQL notebook open. The notebook contains the following code:

```

query = """
SELECT Corr(Age, Cholesterol) AS Age_Cholesterol_Correlation
FROM heart_disease_data
"""

correlation = spark.sql(query)
correlation.show()

```

The output shows the correlation coefficient:

```

+-----+
| 0.4499341330894362 |
+-----+

```

Below this, another query is shown:

```

query2 = """
SELECT Sex, AVG(Age), AVG(Cholesterol), COUNT(Target) AS heart_Disease_Count
FROM heart_disease_data
GROUP BY Sex
"""

demographic = spark.sql(query2)
demographic.show()

```

The output shows demographic statistics grouped by sex:

| Sex | Avg_Age | Avg_Cholesterol | Heart_Disease_Count |
|-----|---------|-----------------|---------------------|
| M | 56.31 | 208.1 | 18 |
| F | 54.82 | 248.81 | 11 |

```

    # NLP (Part 1)
    # ...
    # NLP (Part 2)
    # ...

```

```

    CapStone Project Python
    # ...
    # ...

```

| Name | Last modified | Size | Type |
|--|--|---------|----------|
| commented_101020845861340014 | January 24, 2025, 00:05:54 (UTC-05:00) | 117.0 B | Standard |
| commented_101020845861340014 | January 24, 2025, 19:52:23 (UTC-05:00) | 244.0 B | Standard |
| commented_30161609528807095 | January 24, 2025, 00:05:49 (UTC-05:00) | 199.0 B | Standard |
| commented_30161609528807095 | January 24, 2025, 19:50:04 (UTC-05:00) | 250.0 B | Standard |
| commented_45117794109418998 | January 22, 2025, 22:19:57 (UTC-05:00) | 265.0 B | Standard |
| commented_47466070315525640 | January 24, 2025, 00:05:51 (UTC-05:00) | 112.0 B | Standard |
| commented_5752883623209183049 | January 24, 2025, 00:05:13 (UTC-05:00) | 197.0 B | Standard |
| commented_61102930762304505 | January 22, 2025, 20:07:22 (UTC-05:00) | 254.0 B | Standard |
| commented_757266478741216024 | January 22, 2025, 21:56:40 (UTC-05:00) | 199.0 B | Standard |
| commented_id_32233557057847958 | January 22, 2025, 19:50:33 (UTC-05:00) | 424.0 B | Standard |
| started_163525954534454542 | January 24, 2025, 00:05:53 (UTC-05:00) | 0 B | Standard |
| started_163525954534454542 | January 24, 2025, 00:05:48 (UTC-05:00) | 0 B | Standard |
| started_4724660708155545660 | January 24, 2025, 00:05:51 (UTC-05:00) | 0 B | Standard |
| started_5752883623209183049 | January 24, 2025, 00:00:13 (UTC-05:00) | 0 B | Standard |
| part-00000_id | January 24, 2025, 00:05:54 (UTC-05:00) | 219.0 B | Standard |
| 3045104895288090795-5333996-5115-4f11-9e01-e1a94d4c75-92-1-000.csv | January 24, 2025, 00:05:48 (UTC-05:00) | 47.0 B | Standard |
| part-00000_id | January 24, 2025, 00:05:51 (UTC-05:00) | 105.0 B | Standard |

The screenshot shows the AWS S3 Object Details page for a CSV file. The object key is `part-00000-tid-1835259954634445342-a1a99b2f-477d-4efc-af1f-50c19eb30702-98-1-c000.csv`. The object overview section displays details like Owner (dipsharma@umg79), AWS Region (US East (N. Virginia) us-east-1), Last modified (January 24, 2025, 00:05:54 (UTC-05:00)), Size (219.0 B), Type (CSV), and Key (`/output/part-00000-tid-1835259954634445342-a1a99b2f-477d-4efc-af1f-50c19eb30702-98-1-c000.csv`). The Storage class is listed as Standard. Management configurations include Bucket Versioning (disabled), Replication status (none), View replication rules, and Expiration date (the object will be permanently deleted on this date).

The screenshot shows the AWS Lambda console interface. A new function named '2-98-1-c000' has been created. It is triggered by an S3 event from the '2-98-1' bucket. The code is a ZIP file containing a single Python script named 'lambda_function.py'. The configuration includes a timeout of 10 seconds and a memory limit of 128 MB. The IAM role 'LambdaBasicExecutionRole' is assigned. The function is currently in a 'Pending' state.

A screenshot of Microsoft Excel showing a correlation matrix for age and cholesterol levels. The title bar indicates the file is named 'Age_Cholesterol_Correlation' and has a size of 6.4 KB. The ribbon menu shows tabs for File, Home, Insert, Page Layout, Formulas, Data, Review, View, and Help. The Home tab is selected, displaying various font and style tools. The main worksheet contains two rows of data: 'Age' and 'Cholesterol'. A formula bar at the top shows the formula =CORREL(A2:B2,B2:B2). The status bar at the bottom shows the cell range A2:B2.

Amazon S3

General purpose buckets

Amazon S3

Properties Permissions Versions

Object overview

Owner: dipesoharug797

AWS Region: US East (N. Virginia) us-east-1

Last modified: January 24, 2024, 00:05:51 (UTC-05:00)

Size: 103.0 B

Type: CSV

Key: output/part-00000-tid-4724660708155585660-59e377cb-0c2-42b1-9a53-c0fdb815fc09-94-1-c000.csv

Object management overview

The following bucket protection and object management configurations impact the behavior of this object.

Bucket properties

Bucket Versioning: This bucket contains multiple versions of an object can be stored in the bucket to easily recover from unintended user actions and application failures.

Enabled

Bucket "capstonebucket1235" doesn't have Bucket Versioning enabled: We recommend that you enable Bucket Versioning to help protect against unintentionally overwriting or deleting objects. [Learn more](#)

Object URL: <https://s3.amazonaws.com/capstonebucket1235/output/part-00000-tid-4724660708155585660-59e377cb-0c2-42b1-9a53-c0fdb815fc09-94-1-c000.csv>

Management configurations

Replication status: When a replication rule is applied to an object, the replication status indicates the progress of the operation.

Disabled

View replication rules

Expiration rule: You can use a lifecycle configuration to define expiration rules to schedule the removal of this object after a pre-defined time period.

Expiration date: The object will be permanently deleted on this date.

Amazon S3

General purpose buckets

Amazon S3

Objects

Preview

Organizer

New folder

Amazon S3

Bucket "capstonebucket1235" has 1 item

9-94-1-c000.csv

File type: Microsoft Excel Comma Separated Values (.csv)

Save as type: Microsoft Excel Comma Separated Values (.csv)

Save

Cancel

Object management overview

File Home Insert Page Layout Formulas Data Review View Help

Import matplotlib.pyplot as plt

Converting Spark DataFrame to Pandas DataFrame for visualization

age_data = df.select("Age").toPandas()

Plotting histogram

plt.figure(figsize=[10, 6], linewidth=1, color="skyblue", edgecolor="black")

plt.title("Age Distribution")

plt.xlabel("Age")

plt.ylabel("Frequency")

plt.show()

(1) Spark Jobs

Age Distribution

| Age | Frequency |
|-----|-----------|
| 40 | 2.0 |
| 45 | 2.5 |
| 50 | 1.5 |
| 55 | 2.0 |
| 60 | 2.5 |
| 65 | 2.0 |

Compute

Machine Learning

Experiments

16

Python

```
import matplotlib.pyplot as plt

# Converting Spark DataFrame to Pandas DataFrame for visualization
age_data = df.select("Age").toPandas()

# Plotting histogram
plt.figure(figsize=[10, 6], linewidth=1, color="skyblue", edgecolor="black")
plt.title("Age Distribution")
plt.xlabel("Age")
plt.ylabel("Frequency")
plt.show()

(1) Spark Jobs
```

Age Distribution

| Age | Frequency |
|-----|-----------|
| 40 | 2.0 |
| 45 | 2.5 |
| 50 | 1.5 |
| 55 | 2.0 |
| 60 | 2.5 |
| 65 | 2.0 |

CapStone Project Python ✓ Last edit was 22 minutes ago

```
# Importing libraries
import seaborn as sns
import pandas as pd

# Converting to Pandas for visualization
cholesterol_data = df.select("Cholesterol").toPandas()

# Plotting boxplot
sns.boxplot(data=cholesterol_data, x="Cholesterol", color="lightgreen")
plt.title("Cholesterol Levels")
plt.xlabel("Cholesterol")
plt.show()
```

(1) Spark jobs

CapStone Project Python ✓ Last edit was 24 minutes ago

```
# Importing libraries
import seaborn as sns
import pandas as pd

# Converting Spark DataFrame to Pandas DataFrame
data = df.toPandas()

# Computing correlation matrix
corr_matrix = data.corr()

# Plotting heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Feature Correlation Heatmap")
plt.show()
```

(1) Spark jobs

CapStone Project Python ✓ Last edit was 24 minutes ago

```
# Grouping data by gender and heart disease
gender_disease = df.groupby("Sex", "Target").count().toPandas()

# Plotting bar chart
sns.barplot(data=gender_disease, x="Sex", y="count", hue="Target", palette="viridis")
plt.title("Heart Disease Prevalence by Gender")
plt.xlabel("Gender")
plt.ylabel("Count")
plt.legend(title="Heart Disease")
plt.show()
```

(2) Spark jobs

