Databricks System Tables — An Introduction

Ryan Chynoweth

5 min read · Aug 30, 2023

System tables in Databricks serve as an analytical repository for operational data related to your account. They offer historical observability and can be highly useful for tracking various aspects of your Databricks environment. Currently, Databricks provides system tables for audit logs, billable usage logs, as well as table and column lineage. The audit logs can be accessed at `system.access.audit`, billable usage logs at `system.billing.usage` with associated list pricing available at `system.billing.list_prices`, and the table and column lineage information is available in tables under `system.access`. These system tables provide valuable insights into the activities, resource utilization, and data lineage within your Databricks account and can be used for historical KPI tracking, monitoring and alerting, and forecasting expected usage for an intelligent Lakehouse.

How It Works

Let's dive into how these system tables function and how you can leverage them to gain deeper insights into your Databricks environment.

To access and utilize system tables, your account must have at least one Unity Catalog-enabled workspace. Unity Catalog acts as the governing framework for these tables, enabling their functionality. System tables are populated with data from **all** workspaces in your account, but you can only access them from a Unity Catalog-enabled workspace.

System tables must be enabled by an account admin. This activation can be accomplished using the Unity Catalog REST API. Initially, no users have access to these tables. To grant access, metastore administrators need to provide `USE` and `SELECT` permissions on the relevant system schemas.

System table data is made available by Databricks and is processed through Extract, Transform, Load (ETL) processes in the control plane. Importantly, this data is readily accessible through the regular use of the Databricks platform, requiring only the activation of the system tables feature.

The system tables are provided to users through Delta Sharing, an open-source data sharing protocol. This protocol ensures efficient and secure sharing of data, enabling you to seamlessly access and analyze the valuable insights stored within system tables. The audit log and lineage tables cover operational data from all workspaces within a region, while the billing system table (`system.billing.usage`) covers data from all workspaces across regions.

Even though system tables can only be accessed through a Unity Catalog-enabled workspace, they encompass operational data not only for Unity

Catalog workspaces but also for non-Unity Catalog workspaces in your account.

Where To Find Them

The system tables are organized within a catalog named `system`, which is a fundamental component of every Unity Catalog metastore. Inside this catalog, you'll find schemas: `access` and `billing` that house the system tables. These tables offer a comprehensive view of your Databricks environment, enabling you to make informed decisions and optimize resource allocation.

Sample Use Cases

Key Performance Indicators (KPIs) for Historical Reporting

In this KPI section, we'll explore vital metrics to gauge the effectiveness of your Databricks usage over time:

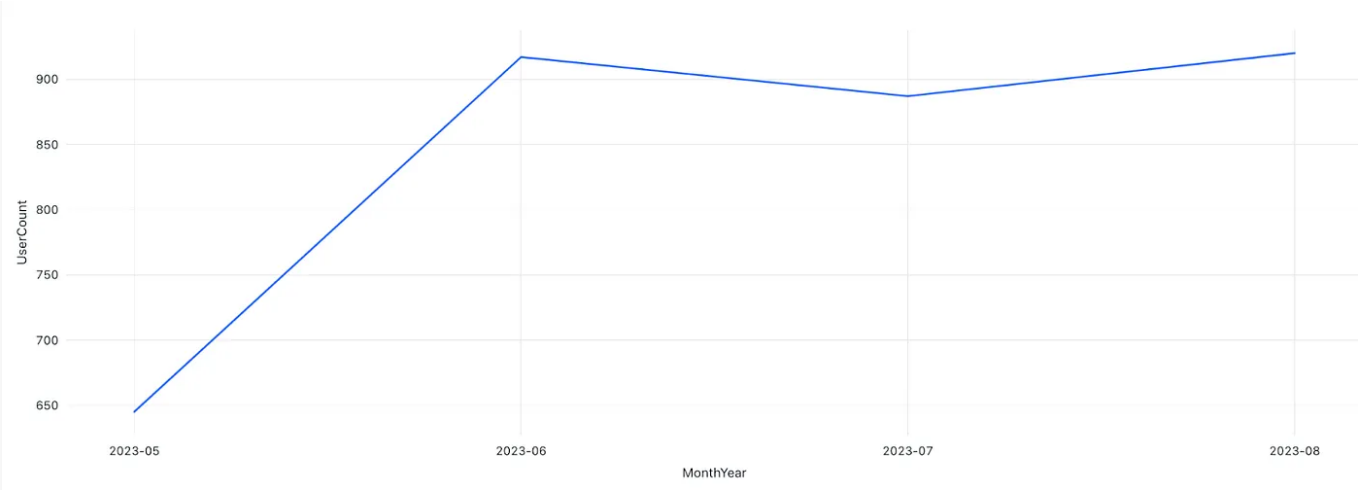
1. **Monthly Active Users**: Measure engagement and user adoption by tracking the number of users active on the platform each month.
2. **Jobs with Specific Runtimes**: Identify efficiency outliers by analyzing job runtimes, optimizing resource allocation, and refining query performance.
3. **Number of Workspaces**: Scale your operations effectively by monitoring the count of deployed workspaces, informing collaboration and resource planning.
4. **Weekly Number of Jobs**: Gain insights into activity patterns with weekly job execution numbers, aiding in resource allocation and task scheduling.

These KPIs, accompanied by targeted queries, offer actionable insights into your Databricks utilization, facilitating informed decisions and enhanced operational performance.

Monthly Active Users

Monitoring and visualizing user growth over time is a KPI that shows the impact of Databricks in an organization.

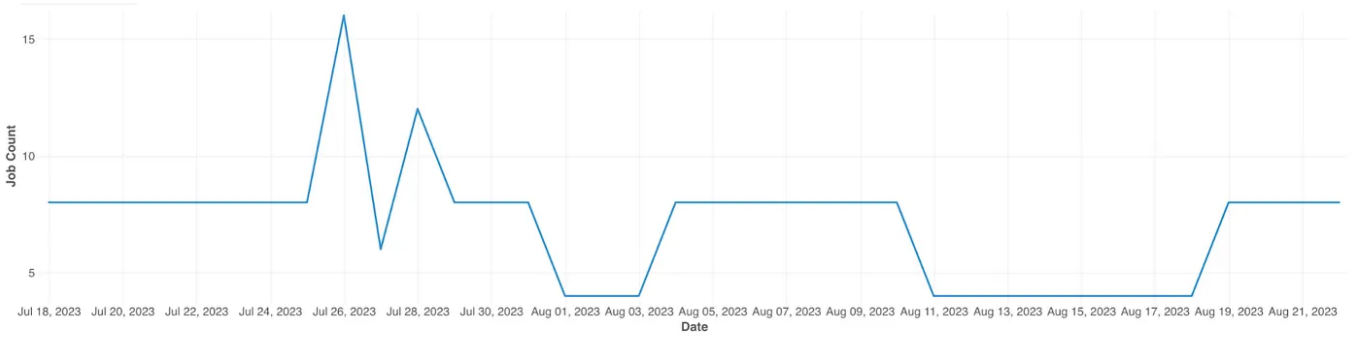
```
select
  date_format(event_date, 'yyyy-MM') as MonthYear
  , count(distinct user_identity.email) as UserCount
from system.access.audit
group by 1
```



Monitoring Databricks Runtimes

Let's see the number of daily jobs running on Databricks Runtime 7.3 LTS to ensure that we are phasing out older runtimes.

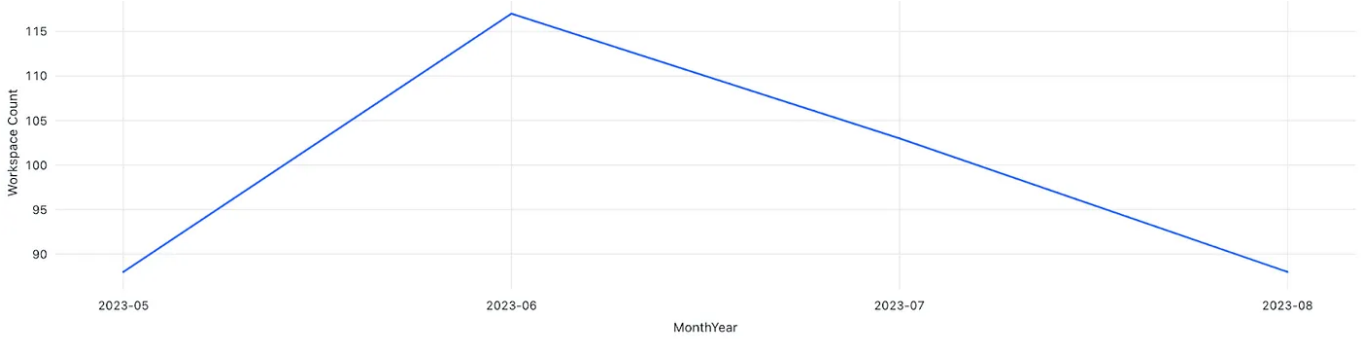
```
select
  workspace_id
  , event_date
  , request_params.cluster_name
  , request_params.custom_tags
  , get_json_object(request_params.custom_tags,'$.JobId' ) as jobId
  , get_json_object(request_params.azure_attributes, '$.availability')
  , request_params.spark_version
  , request_params
from system.access.audit
where service_name = 'clusters' and action_name = 'create'
  and contains(request_params.spark_version, 'dlt') = False
  and contains(request_params.spark_version, '7.3')
  and get_json_object(request_params.custom_tags,'$.JobId' ) is not null
```



Workspace Count

Customers will create workspaces with different strategies. Some will simply want to have three consolidated workspaces for development, staging, and production, while others will separate workspaces by business groups.

```
SELECT
  date_format(event_date, 'yyyy-MM') as MonthYear
  , count(distinct workspace_id) as WorkspaceCount
from system.access.audit
group by 1
```



Monthly Jobs

Monitoring the number of jobs allows for a greater understanding of platform usage. Jobs often result in value realized as it represents production workloads that produce a return on investment from the historical development spend.

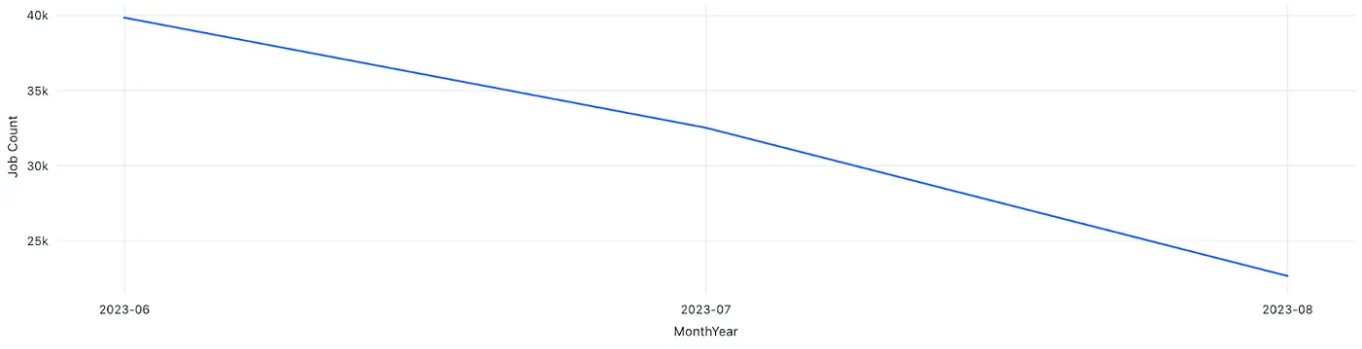
```
select
date_format(usage_start_time, 'yyyy-MM') as MonthYear,

count(distinct concat(usage_metadata.job_id, usage_metadata.cluster_id)) as JobC

from system.billing.usage

where usage_metadata.job_id is not null

group by 1
```



Leveraging the List Pricing Table

Customers are also provided a lookup table that allows users to join billable usage data (DBUs) to a list price table, `system.billing.list_prices`, to allow for estimating Databricks costs. Please see the query below for an example that shows the cost for a particular job and cluster associated to that job.

```
select

  usage.workspace_id,
  usage.cloud,
  usage.sku_name,
  usage.usage_metadata.job_id,
  usage.usage_metadata.cluster_id,
  list_prices.pricing.default as list_price,
  -- list_prices.price_end_time ,
  -- list_prices.price_start_time,
  min(usage.usage_start_time) as starting_hour, -- this is not the exact start t
  max(usage.usage_end_time) as ending_hour, -- this is not the exact end time of
  sum(usage.usage_quantity) as dbus,
  sum(usage.usage_quantity)*list_prices.pricing.default as list_cost

from
  system.billing.usage usage
  inner join system.billing.list_prices list_prices on
    usage.cloud = list_prices.cloud and
    usage.sku_name = list_prices.sku_name and
    usage.usage_start_time >= list_prices.price_start_time and
    (usage.usage_end_time <= list_prices.price_end_time or list_prices.price_end

where usage.workspace_id = '<workspace_id>' and usage.usage_metadata.job_id = '<

group by all
```

Forecasting Costs

Cost management in data analytics is a critical task, and Databricks system tables offer a powerful tool for tackling it. By tapping into the insights stored within these tables, businesses can predict future costs with precision,

optimizing resource allocation and budget planning. In turn, this allows for dynamic proactive alerting of spend when unexpected behavior occurs. Let's explore how Databricks system tables can revolutionize cost forecasting and enhance financial efficiency. Check out the Databricks forecasting demo available here!

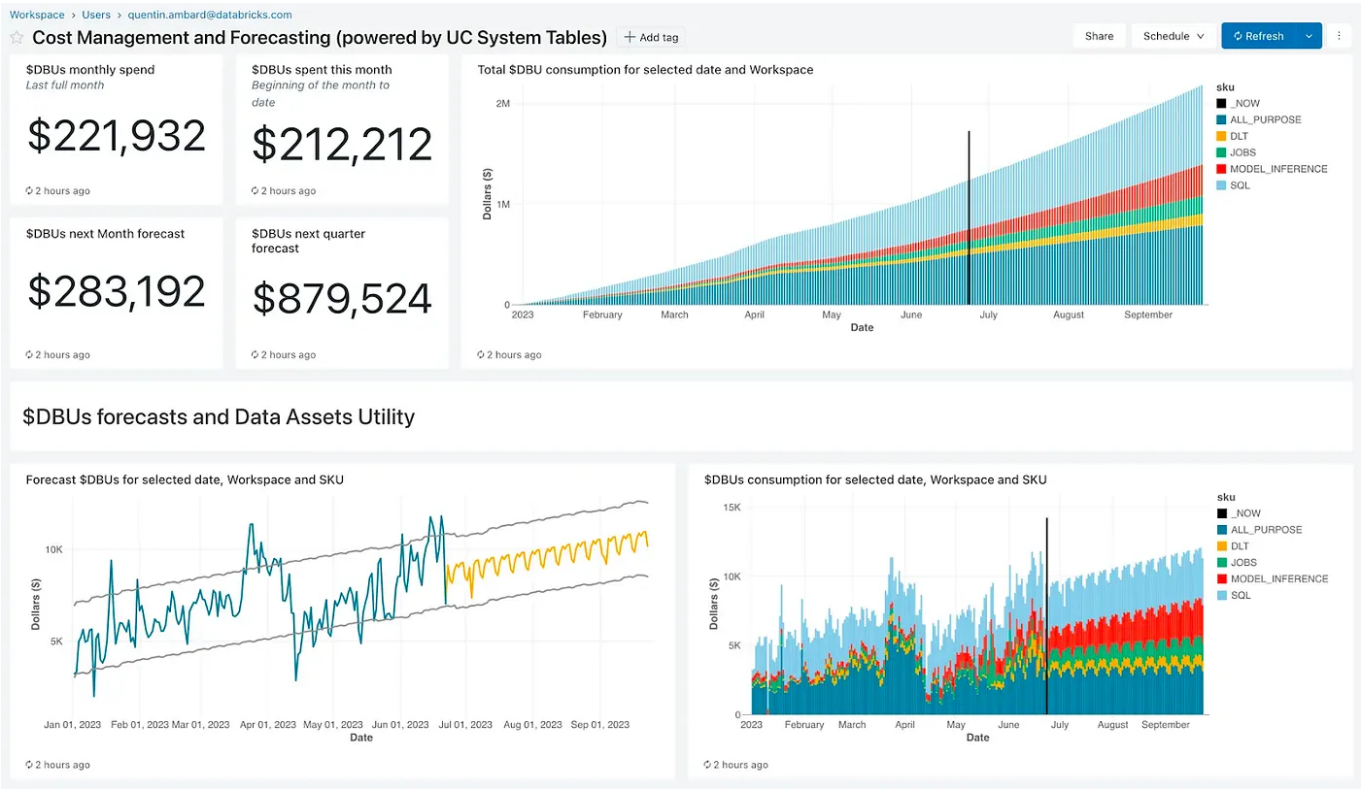


Image Sourced from Databricks Demo Page

Conclusion

Databricks system tables provide an avenue for historical observability and data-driven decision-making. By integrating seamlessly with Unity Catalog, these tables grant you access to a wealth of operational insights that can help you better understand and optimize your Databricks account's performance.

With this understanding of how system tables work and their capabilities, you're equipped to explore and harness the power of these tables to drive actionable insights and enhance your Databricks experience.

Databricks Systemtables

Written by Ryan Chynoweth

Edit profile

312 Followers

Senior Solutions Architect Databricks — anything shared is my own thoughts and opinions

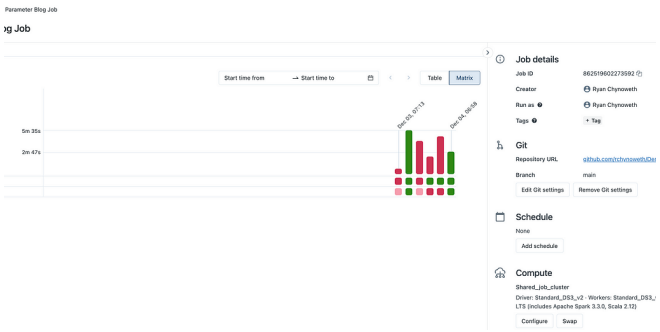
More from Ryan Chynoweth

Ryan Chynoweth

Delta Sharing: An Implementation Guide for Multi-Cloud Architecture

Introduction

8 min read · 3 days ago



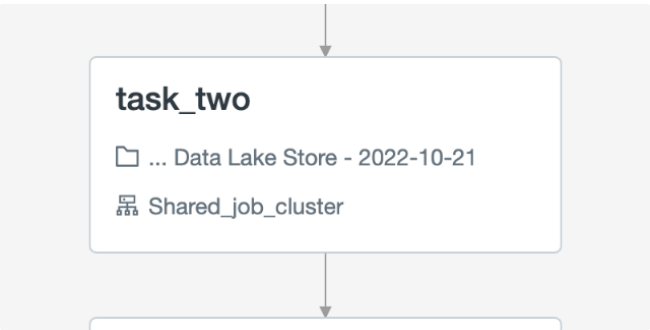
Ryan Chynoweth

Task Parameters and Values in Databricks Workflows

Databricks provides a set of powerful and dynamic orchestration capabilities that are...

11 min read · Dec 7, 2022

3 2 50 3



Ryan Chynoweth

Converting Stored Procedures to Databricks

Special thanks to co-author Kyle Hale, Sr. Specialist Solutions Architect at Databricks.

14 min read · Dec 29, 2022

116 5

Ryan Chynoweth

Recursive CTE on Databricks

Introduction

3 min read · Apr 20, 2022

33

See all from Ryan Chynoweth

Recommended from Medium

Daan Rademaker

Do-it-yourself, building your own Databricks Docker Container

In my previous LinkedIn article, I aimed to persuade you of the numerous advantages o...

7 min read · Oct 16, 2023

26

M Manasreddy

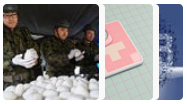
How to pass: Databricks Data Engineer Professional Certification

Conquering the Databricks Data Engineer Professional Exam: A Definitive Guide

3 min read · Aug 24, 2023

8

Lists



Staff Picks

547 stories · 597 saves

Stories to Help You Level-Up at Work

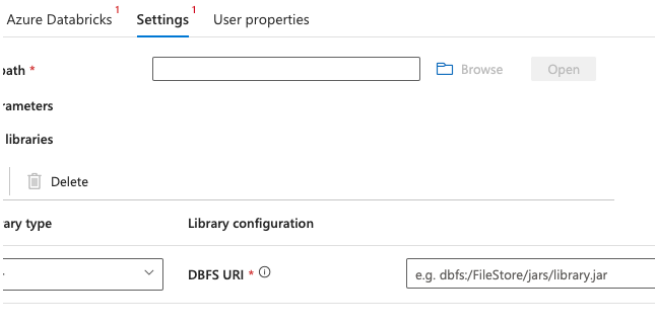
19 stories · 395 saves

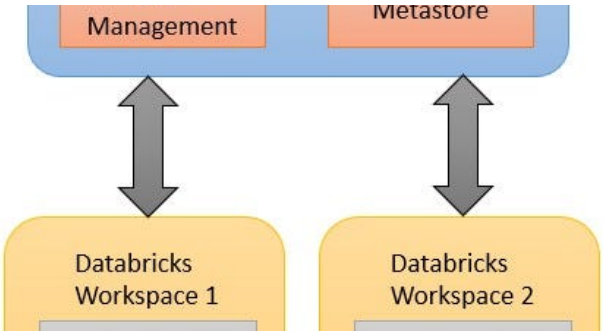
Self-Improvement 101

20 stories · 1146 saves

Productivity 101

20 stories · 1047 saves





Matt Bradley

Azure Data Factory tips for running Databricks jobs

Following on from an earlier blog around using spot instances with Azure Data...

4 min read · Nov 2, 2023

14

Oindrila Chakraborty

Introduction to “Unity Catalog” in Databricks

What is “Unity Catalog”?

10 min read · Aug 14, 2023

18 2

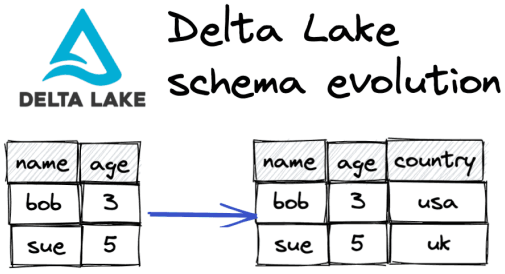
SIRIGIRI HARI KRISHNA

1.0 Unity Catalog

Unity Catalog by Databricks provides a unified governance solution for managing...

2 min read · Nov 28, 2023

1



Matthew Salminen

Databricks Autoloader and Medallion Architecture... Pt 2.

In my last post, Ingest Data with Databricks Autoloader, I introduced autoloader as a way...

4 min read · Aug 13, 2023

11

See more recommendations