Identity Columns with Databricks Delta

Ryan Chynoweth

2 min read · Apr 10, 2022

Introduction

Identity columns are used to generate unique values for different types of analytics and Lakehouse workloads. One of the most common reasons to use them is to build a relationship between fact and dimension tables for faster joins, lookups, and processing. Creating unique keys in Apache Spark has traditionally been more difficult than it should have been.

A common resource I like to share with customers is a live stream by Denny Lee and Douglas Moore from Databricks where they discuss generating unique keys and the different options when using Apache Spark. The key takeaway from the video is that there was a trade off between performance and reliability of the generated sequences. Meaning, I can have low spread of the numbers generated but I would suffer performance or I could have great performance but a large spread of the sequence.

New Feature in Databricks Delta

Now with Databricks Runtime 10.4 LTS we have finally made identity columns for Delta tables generally available (was available previously as a private preview feature). The values generated by this feature are not guaranteed to be consecutive but it is a best effort to keep the gap as small as possible while maintaining high performance. It does guarantee that the values will be unique so you can easily create surrogate keys for your workloads

Because there always seems to be FUD about these type of data warehouse features against Databricks. It is good to note that even Snowflake's identity column feature does not guarantee sequential keys without gaps, and but there are alternate options to doing so. This is an almost identical feature comparison between the two products. Although, the big drawback is that Snowflake requires you to store your data in a proprietary data format and access everything via SQL. While Databricks has open language and storage support.

Creating a Table with an Identity Column

Identity columns are created by using the Databricks Delta generated column feature. In this example we will do the following:

- Create a database with a location. Creating databases with a location will allow you to easily manage all your tables and save them to your own cloud storage account which is recommended.
- Change my default database
- Create my table and supply the generated column

- Copy data into my table using COPY INTO, notice I do not supply the id column.

```
CREATE DATABASE IF NOT EXISTS my_database
COMMENT 'my demo database'
LOCATION 's3a://demo/my_database' ;

USE my_database;

CREATE TABLE my_table (
id BIGINT GENERATED ALWAYS AS IDENTITY
value STRING,
amount double

) USING delta;

COPY INTO my_table
FROM (SELECT value, amount
       FROM 's3a://bronze/demo/table/*.parquet'
     )
FILEFORMAT = PARQUET;
```

For more information please refer to the Databricks Documentation, specifically the parameters section.

Conclusion

Identity columns using Delta Lake just got easier! Try it out and let me know what you think!

Disclaimer: these are my own thoughts and opinions and not a reflection of my employer

Databricks Delta Lake Identity Columns Databricks Sql

Written by Ryan Chynoweth

Edit profile

312 Followers

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

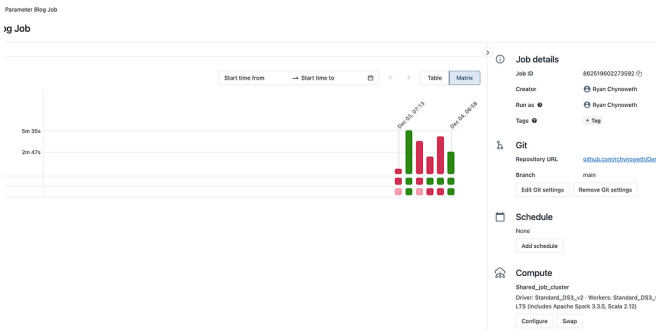
More from Ryan Chynoweth

Ryan Chynoweth

Delta Sharing: An Implementation Guide for Multi-Cloud Architecture

Introduction

8 min read · 3 days ago



Ryan Chynoweth

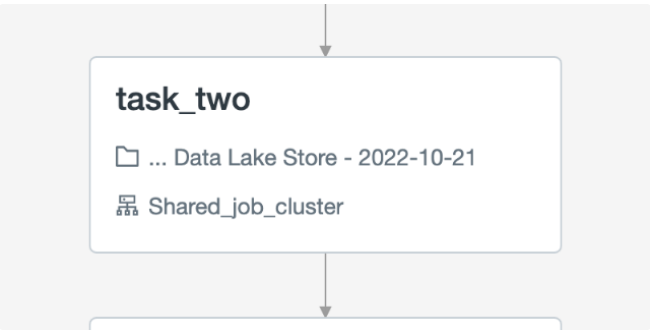
Task Parameters and Values in Databricks Workflows

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

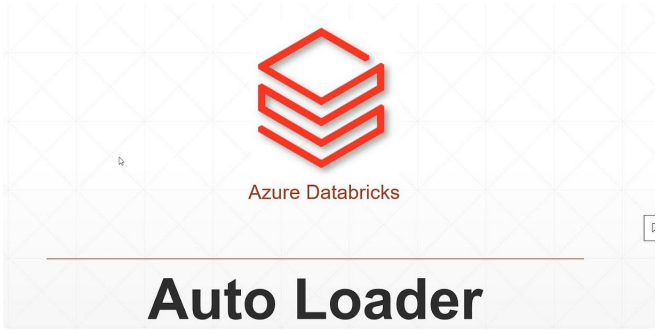
Introduction

3 min read · Apr 20, 2022

33

See all from Ryan Chynoweth

Recommended from Medium



SIRIGIRI HARI KRISHNA in Towards Dev

Auto Loader

Autoloader simplifies reading various data file types from popular cloud locations like...

5 min read · Dec 9, 2023

4 1

Daan Rademaker

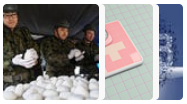
Do-it-yourself, building your own Databricks Docker Container

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

7 min read · Oct 16, 2023

26

Lists



Staff Picks

547 stories · 597 saves

Stories to Help You Level-Up at Work

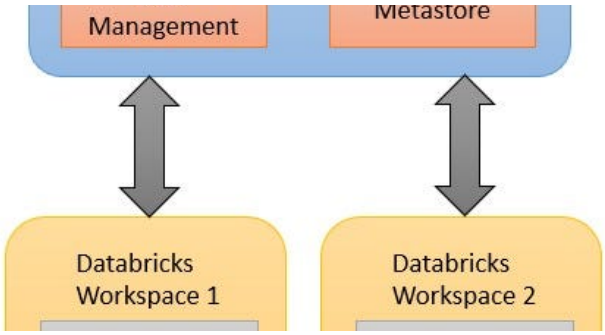
19 stories · 395 saves

Self-Improvement 101

20 stories · 1146 saves

Productivity 101

20 stories · 1047 saves



Oindrila Chakraborty

Introduction to “Unity Catalog” in Databricks

What is “Unity Catalog”?

10 min read · Aug 14, 2023

18 2

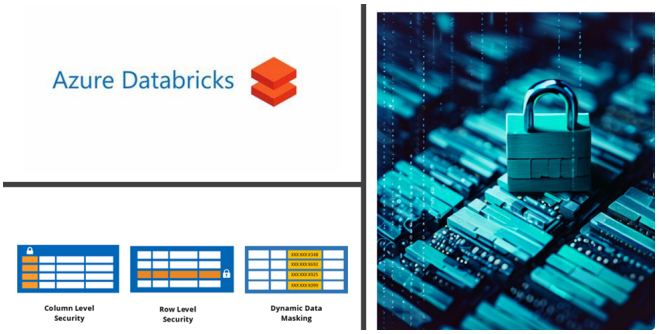
Kaleigh Spitzer

SCDs in Delta Live Tables

Slowly changing dimensions (SCDs) are dimensions that change over time. SCDs are ...

3 min read · Nov 20, 2023

11



Samarendra Panda

Dynamic Row Level Filtering and Column Level Masking in Azure...

Background

5 min read · Sep 23, 2023

29

Matthew Salminen

The Power Duo: Databricks Auto Loader and Delta Live Tables

In my last two posts, I explained the benefits of using autoloader for your data pipelines...

· 4 min read · Aug 20, 2023

16 1

See more recommendations