Cloudera Octopai Data Lineage 1.0.0

Use Cases and Best Practices

Date published: 2025-10-09
Date modified: 2025-10-20

CLOUDERA

https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2026. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Examples **4**
Use case: Applying data lineage for data quality 4
Use case: Applying data lineage for tracing a data propagation error 4

Migration guides **5**
Comprehensive guide to migrating legacy systems to Snowflake with Cloudera Data Lineage 6
Comprehensive guide to migrating Teradata to Google BigQuery with Cloudera Data Lineage 11

Change management **17**
Change management - Best practices for using Cloudera Data Lineage for CI/CD DataOps 17
Schema management - Best practices for handling changes and impact analysis 23

Examples

Data lineage improves data quality and traces the source of data propagation errors, as shown in column-level and cross-system examples.

Use case: Applying data lineage for data quality

Learn about the power of comprehensive data lineage with the ability to trace data at column level, within a system, and across systems.

With the help of comprehensive data lineage an organization can quickly identify and resolve data issues, improving overall data quality and trust for effective Data Change Management.

A Data Analyst named Alex in a large financial institution faces a data quality issue.

Alex is tasked with creating a Report to analyze customer transactions using a column named transaction_amount from a central data warehouse.

As he starts the planned change investigation, Alex notices some irregularities in the data. Negative values occur in the transaction_amount column, which is incorrect for this context. Alex begins investigating using the following different layers of data lineage:

1. **End-to-end column lineage** – Alex starts by investigating the end-to-end lineage of the transaction_amount column. He sees various transformations applied to this data point, and where the data point is used in downstream reports. He discovers that the column is derived from the transaction_type (credit or debit) and transaction_value columns in a source system. A transformation is applied to convert debit transaction values to negative.
2. **Inner-system lineage** – Alex then looks at the inner-system lineage within the source system. He notices that the transaction_type column is derived from several fields, including the transaction_code field. A particular transaction code identifies debit transactions, and an error in mapping this code might cause the issue.
3. **Cross-system lineage** – Finally, Alex uses the cross-system lineage to find all systems feeding into the transact ion_type column. He discovers an upstream system where the transaction_code field originates. A recent system update changed the transaction_code values for debit transactions.

Armed with this information, Alex collaborates with the data engineering team to correct the error in mapping the new transaction_codefield and ensures that the transformation logic applied to the transaction_amount column is accurate. As a result, the data quality issue is resolved, and Alex can confidently proceed with his report, trusting the data he is using.

Use case: Applying data lineage for tracing a data propagation error

Learn about how a small error can cause a large-scale impact in complex, interconnected systems. Comprehensive data lineage can act as both a detective and a roadmap for resolving such issues.

Scenario overview

Consider a hypothetical large healthcare organization.

The organization operates a core Electronic Health Records (EHR) system that stores critical patient information. This system feeds multiple downstream platforms, including billing, patient portals, insurance claims, and other operational tools.

During a routine upgrade, a coding error introduces a transformation issue in the EHR system. The patient_ID column, which uniquely identifies patients, is incorrectly mapped to the caregiver_ID column that identifies healthcare providers.

The domino effect

As the EHR system feeds data to downstream systems, the mapping error has the following impacts:

1. In the billing system, patients are billed for services provided to their healthcare providers.
2. The patient portals show the healthcare providers details instead of patient-specific information, leading to privacy breaches.
3. Insurance claims get denied due to incorrect patient information.
4. Patients, healthcare providers, insurance companies, and even the healthcare organization customer service and IT teams are heavily impacted.

Navigating the problem with data lineage

In this situation, data lineage plays a crucial role in both identifying and correcting the issue using the following different layers of data lineage:

- **Cross-System Lineage** – When the first issues that are incorrect billing and privacy breaches arise, the organization's data governance team uses cross-system lineage to trace the patient_ID column data across all systems. They identify that the error originates in the EHR system, which feeds most downstream systems.
- **Inner-System Lineage** – Using inner-system lineage within the EHR system, they realize that the patient_ID column is incorrectly mapped to the caregiver_ID column during a transformation process.
- **End-to-End Column Lineage** – To assess the full impact, the team looks at the end-to-end column lineage of the patient_ID column. They map out all the processes, systems, and reports that use this column. This information is vital for communicating with affected parties and directing corrective measures.

The IT team corrects the erroneous transformation in the EHR system and initiates a massive cleanup operation in all impacted downstream systems.

Prevention using an integrated platform

Having an integrated Cloudera Data Lineage (formerly Octopai) platform that encompasses data discovery, lineage, and data catalog can significantly enhance the ability to prevent such errors by gaining greater visibility into data flows and transformations. This visibility improves detection and prevention of potential issues, enhancing overall data quality and reliability.

Different roles use the integrated platform to achieve the improvements by performing the following actions:

- **Data Engineers** trace the data lineage during system development and maintenance using the catalog as a reference to check transformation logic and mapping to avoid incorrect data flow. They follow robust testing protocols before and after deploying any changes in data processing or transformation logic.
- **Data Stewards** perform metadata management by ensuring that the platform metadata is up-to-date and accurate. They oversee data quality and leverage the lineage tools to verify proper data flows. They establish and monitor data quality rules aligned with the metadata and lineage information.
- **Data Analysts** understand the origin and transformations of data they use for reporting and analytics. This help them spot potential issues and validate the data they use.
- **IT Security and Compliance** understand where sensitive data is stored, transformed, and used. This enables them to enforce access control and monitoring to prevent unauthorized changes, promptly detect and respond to any suspicious activities, and ensure adherence to regulatory requirements.

This proactive approach is governed by a rigorous change management process, during which any system change requires an impact analysis supported by data lineage. Automated data quality checks are also used to identify out-of-range values or unusual data distributions that can indicate issues.

Migration guides

Data lineage helps plan and validate migrations from Oracle, SQL Server, and Teradata to Snowflake, and from Teradata to Google BigQuery.

Comprehensive guide to migrating legacy systems to Snowflake with Cloudera Data Lineage

Learn about best practices for migrating data from Oracle, SQL Server, and Teradata to Snowflake with Cloudera Data Lineage (formerly Octopai).

Introduction

Data migration is a critical part of digital transformation and system upgrades, and it involves transferring data from one system to another. This guide focuses on the migration from Oracle, SQL Server, and Teradata, leading relational database management systems, to Snowflake, a cloud-based data warehousing platform designed for the cloud. It provides a detailed roadmap for a successful data assets migration, including key considerations, challenges, and best practices.

Conducting a comprehensive assessment of the legacy environment, choosing the right migration approach, optimizing data for Snowflake, using the native services of Snowflake, implementing data security and compliance, minimizing downtime and disruption, training the team, and monitoring and optimizing the Snowflake environment after migration are recommended best practices.

Additionally, data and IT teams will need to understand and prepare for differences in how each platform handles things like data types, partitioning, indexing, and cost management to ensure a smooth migration and efficient use of the new system.

Cloudera Data Lineage plays an integral role in enabling organizations to overcome technical challenges and maximize the value of their data migration from Oracle, SQL Server, and Teradata to Snowflake. Its capabilities are designed to ensure risk mitigation, cost reduction, and efficiency in man hours, thereby contributing to an overall smoother, more cost-effective, and risk-averse data migration process. Cloudera Data Lineage Data Discovery tool, for example, automates the process of scanning legacy systems to identify existing data assets, their location, format, and business relevance.

The value proposition of cloud migration

Transitioning from on-premise solutions to cloud-based platforms brings a host of benefits, including cost savings, scalability, increased collaboration, and access to advanced analytics and machine learning capabilities. The move also presents opportunities to adopt new methodologies like DevOps and Agile practices, promoting innovation and reducing time-to-market. However, this transition also has significant implications for Data and IT teams, as they need to manage the shift in technologies, adopt new skill sets, and adjust to different operational practices.

Why Snowflake?

Migration from On-Premise Systems

- Many organizations are moving from traditional on-premise data warehouses to cloud-based solutions like Snowflake. This is due to the scalability, flexibility, and cost-effectiveness of cloud solutions. Common on-premise systems that are often migrated include Oracle, Teradata, and IBM DB2.

Migration from Other Cloud Providers:

- Some organizations are migrating from other cloud data warehouses to Snowflake. This could be due to a variety of reasons, including better performance, cost, or features offered by Snowflake. Common cloud systems that are often migrated include Amazon Redshift, Google BigQuery, and Microsoft Azure SQL Data Warehouse.

Consolidation of Disparate Systems:

- Organizations with data spread across multiple systems (both on-premise and cloud) might choose to migrate to Snowflake to consolidate their data into a single, unified platform. This can simplify data management and improve the ability to derive insights from the data.

Unique aspects of Oracle, SQL Server, Teradata and Snowflake

When discussing the migration from Oracle, SQL Server, and Teradata to Snowflake, you must understand the unique aspects of both platforms that might affect the migration process. The systems have the following key characteristics:

- **Oracle, SQL Server, Teradata**
  - Mature and Robust

    These systems have been around for a long time, and they are known for their robustness and stability. They have a range of features built over time that can handle complex queries and large volumes of data.
  - Scripting

    These systems have their own scripting utilities that allow users to submit SQL commands in batch mode. They are highly versatile and can be used for importing and exporting data, as well as controlling database workflows.
  - Stored Procedures

    These systems support complex stored procedures. This feature might require significant effort to migrate, as the Snowflake scripting and stored procedure capabilities differ.
  - Data Loading and Extraction

    These systems have powerful utilities for data loading and extraction, which might have been used heavily in data and IT teams current setup.
- **Snowflake**
  - Serverless and Fully Managed

    Snowflake is a serverless, highly scalable, and cost-effective cloud data warehouse. It takes away the hassle of managing infrastructure, allowing organizations to focus more on data analysis.
  - Automatic Scaling

    Snowflake automatically scales up and down based on the workload, which means data and IT teams do not need to worry about capacity planning.
  - Real-Time Analysis

    Snowflake allows for real-time analytics on massive streaming data, which is not typically a strength of traditional databases.
  - Integrated with Cloud Platforms

    Snowflake is well-integrated with other cloud services, making it easy to connect with data storage, machine learning, and data processing tools.
  - Data Transfer Service (DTS)

    Snowflake offers DTS for automating data movement from multiple online and offline sources.
  - Security

    Snowflake provides robust security measures, including encryption at rest and in transit, identity and access management, and support for VPC Service Controls.

Migration considerations

Oracle Database

Oracle is a widely used traditional relational database management system. Many businesses are migrating from Oracle to Snowflake due to Snowflake's scalability, flexibility, and cost-effectiveness. The migration process typically involves data extraction from Oracle, data cleaning and transformation, and then loading the data into Snowflake.

Microsoft SQL Server

Another common source for migration is Microsoft SQL Server. This is due to the fact that while SQL Server is powerful and widely used, it can be expensive and lacks the flexibility and scalability that a cloud-native solution like Snowflake offers. The migration process is similar to Oracle, involving extraction, transformation, and loading (ETL).

Teradata

Teradata is a popular data warehouse solution, but many businesses are moving to Snowflake for its superior cloud capabilities, scalability, and cost-effectiveness. The migration process from Teradata to Snowflake involves a similar ETL process, but with additional considerations for the Teradata unique architecture and features.

Migrating from Teradata to Snowflake

Teradata, a popular legacy system, has its own unique challenges when migrating to Snowflake. The process involves understanding the differences between Teradata and Snowflake, such as the handling of NULLs, data types, and stored procedures.

- Data Types: Teradata and Snowflake have different data types. For example, the Teradata TIME data type does not have a direct equivalent in Snowflake. During migration, you must map Teradata data types to their Snowflake equivalents.
- Stored Procedures: The Teradata stored procedures use a different syntax than the Snowflake stored procedures. You must rewrite these procedures in JavaScript, which is the language Snowflake uses for stored procedures.
- NULL Handling: Teradata and Snowflake handle NULLs differently. In Teradata, a comparison with NULL is UNKNOWN, while in Snowflake, it is NULL. You must account for this difference during migration.

Migrating from Oracle and SQL Server to Snowflake

The following considerations apply for migrating from Oracle and SQL Server to Snowflake:

- Data Types: Both Oracle and SQL Server have unique data types that need to be mapped to Snowflake equivalents during migration.
- Stored Procedures: Oracle uses PL/SQL for stored procedures, while SQL Server uses T-SQL. These must be rewritten in JavaScript for Snowflake.
- Indexes: Unlike Oracle and SQL Server, Snowflake does not use indexes. You must rethink any optimization based on indexes for Snowflake.
- Sequences: Oracle and SQL Server handle sequences differently than Snowflake. If your legacy system uses sequences, you must recreate them in Snowflake.

Potential technical and business challenges

In general, migrating from an on-premises solution to a cloud-based one presents a variety of challenges that need to be carefully considered and managed. Migration from Oracle, SQL Server, and Teradata to Snowflake can pose both technical and business challenges, including schema migration, stored procedures conversion, standard and complex views adaptation, handling of scripts, performance considerations and complex risk management.

In some cases, it might be determined that certain tables from the legacy systems are no longer needed, even before the migration, or for example, that specific ETL processes loading data into these tables might be retired. After the migration, it is crucial to review the entire project, looking for opportunities to reduce maintenance, CPU, storage, people resources, and time.

With the right approach and the use of Cloudera Data Lineage platform that supports Cross Systems, Inner Systems, and End-to-End Column Lineage for both the legacy systems and Snowflake, data and IT teams can ensure a smooth and efficient migration, leading to better data management and utilization in the long run.

The following technical and business challenges can arise:

- **Technical challenges**

  Technical challenges can range from system compatibility issues, data conversion challenges, or issues with connectivity and access rights. They will arise due to differences in data types and SQL dialects, absence of traditional stored procedure support in Snowflake, differences in view handling, managing large files, and distinct performance characteristics of the legacy systems and Snowflake. Consider the following potential technical challenges:

- Schema Migration: The legacy systems and Snowflake have different data types and SQL dialects. This means that some data transformations might be necessary to convert legacy data types to Snowflake compatible ones. You might need to rewrite SQL queries due to these syntactical differences.
- Views: In the legacy systems, views are used extensively for data abstraction, simplifying queries, and controlling access. The concept of views exists in Snowflake as well, but the behavior might be different. For instance, the Snowflake views are logical views, not materialized. That means they compute the data when queried, which might affect the performance if not managed properly.
- Scripting and Stored Procedures: The legacy systems scripts and stored procedures might not be directly compatible with Snowflake, requiring rewriting and transformation. For example, you must consider how to transform the legacy scripts and stored procedures into Snowflake compatible SQL scripts or Cloud Functions, or how to replace the legacy utilities with Snowflake Data Transfer Service.
- Performance and Data Transfer Considerations: The performance characteristics of the legacy systems and Snowflake can be quite different due to the underlying architecture differences. Snowflake is a serverless, highly scalable, and cost-effective multi-cloud data warehouse designed for business agility, while the legacy systems are popular relational database management systems. Therefore, optimizing for performance might require different strategies in Snowflake as compared to the legacy systems. Moving large amounts of data from one platform to another can be time-consuming and risk-prone. There is a risk of data loss, corruption, or security breaches during the migration process.
- Utilizing Legacy Utilities: If you are used to the legacy utilities, you will need to figure out how to accomplish these tasks with the Snowflake tools.

- **Business challenges**

  Business challenges can include risk management, cost considerations, training and change management. For instance, there might be costs associated with potential downtime during the migration, training for staff to use the new system, and potential resistance from users who are comfortable with the legacy system. Consider the following potential business challenges:

  - Risk Management: Data migration always carries risk. These can include data loss, corruption, or breaches of security. You must have robust risk management strategies in place to mitigate these risks. This includes thorough testing, backup and recovery plans, and security measures.
  - Cost Considerations: While moving to a cloud-based solution like Snowflake can lead to cost savings in the long run, the migration process itself can be costly. This includes the cost of the migration tools and services, potential downtime during the migration, and the cost of training staff to use the new system.
  - Training and Change Management: Moving to a new system requires users to learn new tools and adapt to new workflows. This can lead to resistance, especially if users were comfortable with the legacy system. You must have a strong change management strategy in place to support users through this transition.
  - Performance Tuning: After migrating, data and IT teams might need to spend time optimizing query performance and cost in Snowflake. This can be a complex task and might require a deep understanding of how Snowflake processes queries.
  - Continuing to Monitor and Optimize: Once you migrated to Snowflake, you must continually monitor your system to ensure that it works effectively and adjust as needed.

Migration process overview and best practices

When migrating to Snowflake, you must follow a structured workflow that includes setting up and configuring data transfer, choosing the right extraction method, and identifying and customizing schema. These are key components of the migration workflow.

The migration process can be broken down into seven distinct phases: data discovery, dependencies and scoping for effort assessment, data cleansing and deprecation to migrate the right data, prioritization according to critical paths, selective migration, data conversion, documentation to capture knowledge, and post migration for onboarding and monitoring.

Assessment - data discovery and scoping

Understanding the existing data landscape is the first step - identifying existing data, its location, format, and business relevance. This phase forms the foundation for all subsequent steps, establishing a comprehensive inventory of data assets and their dependencies, which is critical for determining what data to migrate and how.

Cloudera Data Lineage Data Discovery will automatically scan your legacy system (Teradata, Oracle, or SQL Server), identify data entities, their relationships, and how they are used. By gathering insights about your data assets, data and IT teams gain a comprehensive view of what is in your system, guiding data teams to build a solid migration plan.

Data cleansing and deprecation

Once the data assets have been scoped, the next step is to cleanse the data in the legacy system. This involves removing redundant, irrelevant, or erroneous data to avoid migrating garbage. This step is also an opportune moment to deprecate unused or unnecessary reports fed by the legacy system. This prevents the migration of garbage, and presents an opportunity to deprecate duplicates, redundant or unused assets.

The Cloudera Data Lineage cross systems lineage and inner-system lineage tools play a crucial role in this phase. By visualizing the data lineage, Cloudera Data Lineage pinpoints data assets that are no longer being used and can be deprecated, streamlining the migration process and avoiding the migration of unnecessary data.

Prioritization

Selective migration is the best approach for prioritization of the most critical data assets to be migrated first. This can help minimize downtime and disruptions to key business operations. However, it can also introduce the challenge of double maintenance - managing both the legacy and new systems concurrently, which can put additional stress on IT and data teams.

For prioritizing and selecting data, the best approach is to migrate the most accessed tables or those critical to business operations first. The prioritization can be based on the data's business impact, data quality, or compliance needs.

The Cloudera Data Lineage Data Lineage and Catalog capabilities empower the data team to analyze table dependencies, allowing them to collaborate with the business stakeholders and tag critical accounting data assets within the Cloudera Data Lineage Data Catalog for prioritized migration.

Data conversion

In this phase, the selected data is converted and transformed to suit the schema, stored procedures, and views of the Snowflake system. Data conversion involves converting legacy-specific SQL code into Snowflake standard SQL. For example, the legacy system TIMESTAMP might need to be converted to the Snowflake TIMESTAMP, and similar conversions for other data types. You must consider stored procedures as well since legacy procedures might not work directly in Snowflake due to syntactical differences.

During the data conversion phase, Cloudera Data Lineage Live lineage functionality proves invaluable in simulating the effects of transformations, allowing for a comprehensive understanding of potential issues and their impact on data lineage. With Cloudera Data Lineage, the data team can simulate the conversion of legacy-specific SQL code into the Snowflake standard SQL, identifying any syntactical differences or required data type conversions. By testing and verifying the simulated data lineage, the team can confidently address errors, make necessary fixes, and ensure the accuracy and integrity of the data throughout the migration process.

Post-migration challenges and optimization

After the migration, you must review the entire project, looking for opportunities to reduce maintenance, CPU, storage, people resources, and time. Post-migration, regular monitoring is crucial. For example, data teams might set up a daily check of failed SQL jobs in Snowflake. If data and IT teams see a recurring failure, data and IT teams can then dig deeper to understand the issue that can be a data type mismatch, or syntax error, and address the root cause.

Additional considerations exist for adopting new practices and terminologies related to cloud technologies. You must consider the following key points:

- Cloud Terminology: Familiarize yourself with the terminology used in cloud computing and specifically in Snowflake. This includes terms like warehouses, databases, schemas, tables, and more. Understanding these terms will help you navigate and manage your cloud environment effectively.
- Cloud Data Management: Cloud-based data management introduces new practices and tools. You must understand how data is stored, processed, and managed in Snowflake. This includes concepts like data partitioning, clustering, and more. Learn about best practices for optimizing data storage, query performance, and cost management in Snowflake.
- Security and Compliance: Cloud environments have their unique security and compliance considerations. Familiarize yourself with the security features and options provided by Snowflake. Implement proper access controls, encryption, and data governance practices. Ensure compliance with relevant regulations and standards that apply to your data.
- Automation and Orchestration: Cloud platforms offer automation and orchestration capabilities that can streamline data pipelines and processes. Explore tools to automate data transformations, workflows, and data integration tasks. Leverage these tools to optimize your data pipelines in the cloud.
- Scalability and Elasticity: Cloud-based systems provide scalability and elasticity, allowing you to scale your resources up or down based on demand. Understand how to leverage the Snowflake auto-scaling capabilities to handle varying workloads efficiently. Design your data pipelines and infrastructure to take advantage of the cloud scalability benefits.
- Data Cataloguing and Documentation: Ensure that you create a comprehensive data catalog in Snowflake that accurately documents your data assets, including metadata, data lineage, and business descriptions. This will help users find and understand the data in the new system.
- Continuous Learning and Training: Cloud technologies and best practices evolve rapidly. Encourage continuous learning and training for your data and IT teams to stay updated with the latest advancements in Snowflake and cloud computing. Leverage training resources, online documentation, and community forums provided by Snowflake to enhance your knowledge and skills.

Efficient migration to Snowflake with Cloudera Data Lineage

Cloudera Data Lineage is a valuable solution that empowers businesses during the migration from legacy systems to Snowflake. It automates data discovery, streamlining the migration process and ensuring a smooth transition. With the Cloudera Data Lineage lineage and impact analysis capabilities, businesses can enhance data management, validate data integrity, and improve overall reliability. Cloudera Data Lineage also reduces the time and effort required for identifying and migrating relevant data assets. It optimizes resource allocation by identifying and deprecating unused data, while ensuring data security and compliance through robust lineage tracking. By minimizing downtime and disruptions, Cloudera Data Lineage facilitates uninterrupted business operations during the migration. Overall, Cloudera Data Lineage delivers significant benefits, making the migration to Snowflake efficient and successful.

Conclusion

Migrating from Oracle, SQL Server, and Teradata to Snowflake is a significant undertaking that requires careful planning, execution, and monitoring. However, with the right approach and tools, it can be a smooth and efficient process that results in significant benefits for your organization. By following the best practices outlined in this guide and leveraging the capabilities of Cloudera Data Lineage, you can ensure a successful migration that enhances your organization's data management capabilities and positions you for success in the era of cloud computing.

Comprehensive guide to migrating Teradata to Google BigQuery with Cloudera Data Lineage

Learn about migrating from Teradata to Google BigQuery, emphasizing best practices, challenges, and solutions.

Introduction

Data migration is a critical part of digital transformation and system upgrades, and it involves transferring data from one system to another. This guide focuses on the migration from Teradata, a leading relational database management

system, to Google BigQuery, a serverless, highly scalable, and cost-effective multi-cloud data warehouse. It provides a detailed roadmap for a successful data assets migration, including key considerations, challenges, and best practices.

Conducting a comprehensive assessment of the Teradata environment, choosing the right migration approach, optimizing data for BigQuery, using the BigQuery native services, implementing data security and compliance, minimizing downtime and disruption, training the team, and monitoring and optimizing the BigQuery environment after migration are recommended best practices.

Additionally, data and IT teams will need to understand and prepare for differences in how each platform handles things like data types, partitioning, indexing, and cost management to ensure a smooth migration and efficient use of the new system.

Cloudera Data Lineage (formerly Octopai) plays an integral role in enabling organizations to overcome technical challenges and maximize the value of their data migration from Teradata to Google BigQuery. Its capabilities are designed to ensure risk mitigation, cost reduction, and efficiency in man hours, thereby contributing to an overall smoother, more cost-effective, and risk-averse data migration process. Cloudera Data Lineage Data Discovery tool, for example, automates the process of scanning Teradata systems to identify existing data assets, their location, format, and business relevance.

The value proposition of cloud migration

Transitioning from on-premise solutions to cloud-based platforms brings a host of benefits, including cost savings, scalability, increased collaboration, and access to advanced analytics and machine learning capabilities. The move also presents opportunities to adopt new methodologies like DevOps and Agile practices, promoting innovation and reducing time-to-market. However, this transition also has significant implications for Data and IT teams, as they need to manage the shift in technologies, adopt new skill sets, and adjust to different operational practices.

Teradata compared to Google BigQuery with unique aspects

When discussing the migration from Teradata to Google BigQuery, it is essential to understand the unique aspects of both platforms that might affect the migration process. Teradata and Google BigQuery have the following key characteristics:

- **Teradata**
  - Mature and Robust: Teradata has been around for a long time, and it is known for its robustness and stability. It has a range of features built over time that can handle complex queries and large volumes of data.
  - BTEQ Scripting: BTEQ (Basic Teradata Query) is a utility in Teradata that allows users to submit SQL commands in batch mode. It is highly versatile and can be used for importing and exporting data, as well as controlling database workflows.
  - Stored Procedures: Teradata supports complex stored procedures. This feature might require significant effort to migrate, as BigQuery’s scripting and stored procedure capabilities differ.
  - TPump, FastLoad, and MultiLoad: Teradata has powerful utilities for data loading and extraction, which might have been used heavily in data and IT teams current setup.
- **Google BigQuery**
  - Serverless and Fully Managed: BigQuery is a serverless, highly scalable, and cost-effective cloud data warehouse. It takes away the hassle of managing infrastructure, allowing organizations to focus more on data analysis.
  - Automatic Scaling: BigQuery automatically scales up and down based on the workload, which means data and IT teams do not need to worry about capacity planning.
  - Real-Time Analysis: BigQuery allows for real-time analytics on massive streaming data, which is not typically a strength of traditional databases.
  - Integrated with Google Cloud: BigQuery is well-integrated with other Google Cloud services, making it easy to connect with data storage, machine learning, and data processing tools.
  - Data Transfer Service (DTS): BigQuery offers DTS for automating data movement from multiple online and offline sources.

- Security: BigQuery provides robust security measures, including encryption at rest and in transit, identity and access management, and support for VPC Service Controls.

Potential technical and business challenges

In general, migrating from an on-premises solution to a cloud-based one presents a variety of challenges that need to be carefully considered and managed.

Migration from Teradata to Google Big Query can pose both technical and business challenges, including schema migration, stored procedures conversion, standard and complex views adaptation, handling of BTEQ scripts (Batch Teradata Query), performance considerations and complex risk management.

In some cases, it might be determined that certain tables from Teradata are no longer needed, even before the migration, or for example, that specific ETL processes loading data into these tables might be retired.

After the migration, it is crucial to review the entire project, looking for opportunities to reduce maintenance, CPU, storage, people resources, and time.

With the right approach and the use of Cloudera Data Lineage platform that supports Cross Systems, Inner Systems, and End-to-End Column Lineage for both Teradata and Google BigQuery, data and IT teams can ensure a smooth and efficient migration, leading to better data management and utilization in the long run.

The following technical and business challenges can arise:

- **Technical challenges**

  Technical challenges can range from system compatibility issues, data conversion challenges, or issues with connectivity and access rights. It will arise due to differences in data types and SQL dialects, absence of traditional stored procedure support in BigQuery, differences in view handling, managing large files, and distinct performance characteristics of Teradata and BigQuery. Consider the following potential technical challenges:

  - Schema Migration: Teradata and Google BigQuery have different data types and SQL dialects. This means that some data transformations might be necessary to convert Teradata data types to BigQuery compatible ones. You might need to rewrite SQL queries due to these syntactical differences.
  - Microviews and Central Views: In Teradata, views (including Microviews and Central views) are used extensively for data abstraction, simplifying queries, and controlling access. The concept of views exists in BigQuery as well, but the behavior might be different. For instance, the BigQuery views are logical views, not materialized. That means they compute the data when queried, which might affect the performance if not managed properly.
  - BTEQ Scripting and Stored Procedures: The Teradata BTEQ scripts and stored procedures might not be directly compatible with BigQuery, requiring rewriting and transformation. For example, you must consider how to transform Teradata BTEQ scripts and stored procedures into BigQuery compatible SQL scripts or Cloud Functions, or how to replace the Teradata utilities with BigQuery Data Transfer Service.
  - Performance and Data Transfer Considerations: The performance characteristics of Teradata and BigQuery can be quite different due to the underlying architecture differences. BigQuery is a serverless, highly scalable, and cost-effective multi-cloud data warehouse designed for business agility, while Teradata is a popular relational database management system. Therefore, optimizing for performance might require different strategies in BigQuery as compared to Teradata. Moving large amounts of data from one platform to another can be time-consuming and risk-prone. There is a risk of data loss, corruption, or security breaches during the migration process.
  - Utilizing Teradata Utilities: If you are used to Teradata utilities, such as TPump, FastLoad, and MultiLoad, you must find a way to accomplish them with the BigQuery tools.
- **Business challenges**

  Business challenges can include risk management, cost considerations, training and change management. For instance, there might be costs associated with potential downtime during the migration, training for staff to use the new system, and potential resistance from users who are comfortable with the legacy system. Consider the following potential business challenges:

  - Risk Management: Data migration always carries risk, including data loss, corruption, or breaches of security. Robust backup and recovery strategies must be in place.

- Cost: The cost of migration can be high, including the cost of potential downtime, the manpower costs of performing the migration, and the costs of any necessary new software.
- Training and Change Management: Staff might need to be retrained to use BigQuery, which can require both time and expense. Change management is also a significant concern, as employees need to adapt to the new system.

Cloudera Data Lineage is instrumental in addressing data migration challenges, by effectively mitigating risks, cutting costs, and reducing man hours, including wasted time, in transitioning from Teradata to Google BigQuery. Its platform enhances risk management through thorough data mapping, ensuring optimal security while minimizing data loss or corruption. The Cloudera Data Lineage automation facilities contribute to substantial savings in both costs and time, streamlining the migration process and making staff training more efficient. As such, Cloudera Data Lineage is fundamental in facilitating a more cost-efficient, safer, and time-saving data migration process.

Migration process overview and best practices

Make sure you are following the migration workflow by setting up and configuring data transfer, choosing the right extraction method, and identifying and customizing schema are key components of the migration workflow.

The suggested migration process consists of the following steps:

- Download the Migration Agent: The process starts with downloading the migration agent which facilitates the data transfer.
- Configure a Transfer in the BigQuery Data Transfer Service: Set up the data transfer based on your requirements.
- Run the Transfer Job: Copy the table schema and data from your data warehouse to BigQuery.
- Monitor Transfer Jobs: You can optionally monitor the transfer jobs using the Google Cloud Console.

During the migration, the BigQuery Data Transfer Service provides two methods for transferring data: the Teradata Parallel Transporter (TPT) tbuild utility and extraction using a JDBC driver with a FastExport connection. The TPT method is faster but if storage space is a constraint, the JDBC driver with FastExport connection is recommended.

The service also provides automatic schema detection and data type mapping during the data transfer. You can alternatively specify a custom schema file, which can be created manually or generated by the migration agent.

The migration process can be broken down into seven distinct phases: data discovery, dependencies and scoping for effort assessment, data cleansing and deprecation to migrate the right data, prioritization according to critical paths, selective migration, data conversion, documentation to capture knowledge, and post migration for onboarding and monitoring.

Assessment - data discovery and scoping

Understanding the existing data landscape is the first step - identifying existing data, its location, format, and business relevance. This phase forms the foundation for all subsequent steps, establishing a comprehensive inventory of data assets and their dependencies, which is critical for determining what data to migrate and how.

The data discovery and scoping phase is fundamental. For instance, in a large company, you might have hundreds of Teradata databases, each with hundreds or thousands of tables. By using Cloudera Data Lineage Discovery and integrated Data Catalog, Data teams can search, scope and document data entities (tables, views), their owners, data volumes, and how often they are accessed. This information helps to build a solid understanding of what resides in your current databases, guiding your migration plan.

Cloudera Data Lineage Data Discovery will automatically scan your Teradata system, identify data entities, their relationships, and how they are used. By gathering insights about your data assets, data and IT teams gain a comprehensive view of what is in your system, guiding data teams to build a solid migration plan.

Data cleansing and deprecation

Once the data assets have been scoped, the next step is to cleanse the data in the Teradata system. This involves removing redundant, irrelevant, or erroneous data to avoid migrating garbage. This step is also an opportune moment

to deprecate unused or unnecessary reports fed by Teradata. This prevents the migration of garbage and presents an opportunity to deprecate duplicates, redundant or unused assets.

Cloudera Data Lineage cross-system lineage and inner-system lineage tools play a crucial role in this phase. By visualizing the data lineage, Cloudera Data Lineage pinpoints data assets that are no longer being used and can be deprecated, streamlining the migration process and avoiding the migration of unnecessary data.

Data teams will operate with Cloudera Data Lineage Teradata lineage and see which tables and ETL processes are no longer in use. For example, you might have ETL processes or tables created for specific projects that have now ended, and these data assets are no longer used or updated. Identifying and removing these from your migration scope will save considerable time and resources.

Prioritization

Selective migration is the best approach for prioritization of the most critical data assets to be migrated first. This can help minimize downtime and disruptions to key business operations. However, it can also introduce the challenge of double maintenance - managing both the legacy and new systems concurrently, which can put additional stress on IT and data teams. With selective migration, you could start by migrating a small subset of data, perhaps one business area or a specific application. This would allow data and IT teams to identify any issues on a smaller scale before moving on to the rest of the data. It is important to note that this likely requires your team to manage both the old and new systems concurrently, which could increase workload in the short term.

For prioritizing and selecting data, the best approach is to migrate the most accessed tables or those critical to business operations first. Suppose you have a database heavily used by the finance department for monthly reporting. In that case, this might be prioritized to ensure continued business operation. The prioritization can be based on the data's business impact, data quality, or compliance needs.

After cleansing, data assets are prioritized for migration based on factors like business value, user needs, or other defined criteria. The final selection of data to be migrated is then made by the migration team.

The Cloudera Data Lineage Data Lineage and Catalog capabilities empower the data team to analyze table dependencies, allowing them to collaborate with the business stakeholders and tag critical accounting data assets within the Cloudera Data Lineage Data Catalog for prioritized migration. You can ensure a seamless transition, focusing on migrating the essential data assets first and minimizing any potential disruptions to their financial operations during the migration to BigQuery.

Data conversion

In this phase, the selected data is converted and transformed to suit the schema, stored procedures, and views of the BigQuery system.

The migration process involves downloading the migration agent, configuring a transfer in the BigQuery Data Transfer Service, running the transfer job, and optionally monitoring transfer jobs using the Google Cloud console.

Data conversion involves converting Teradata-specific SQL code into BigQuery standard SQL.

For example, the Teradata TIMESTAMP might need to be converted to the BigQuery TIMESTAMP, and similar conversions for other data types. You will need to consider stored procedures as well since Teradata procedures might not work directly in BigQuery due to syntactical differences.

During the data conversion phase, Cloudera Data Lineage Live lineage functionality proves invaluable in simulating the effects of transformations, allowing for a comprehensive understanding of potential issues and their impact on data lineage. With Cloudera Data Lineage, the data team can simulate the conversion of Teradata-specific SQL code into the BigQuery standard SQL, identifying any syntactical differences or required data type conversions. By testing and verifying the simulated data lineage, the team can confidently address errors, make necessary fixes, and ensure the accuracy and integrity of the data throughout the migration process.

The Cloudera Data Lineage Data Lineage Tracking ensures that data is accurately translated during migration. It serves as a validation method, comparing the lineage of data before and after migration.

Special considerations for stored procedures migration

Despite differences in stored procedures between Teradata and Google BigQuery, migration can be achieved by identifying and documenting Teradata stored procedures, learning about BigQuery stored procedures, manually rewriting procedures for BigQuery, testing the new procedures, and iterating on the converted procedures.

When it comes to BTEQ scripts, an important Teradata utility, no direct equivalent exists in BigQuery. This presents a challenge since these scripts often contain business logic. One way to tackle this issue is to convert BTEQ scripts into SQL scripts, then refactor them to comply with BigQuery SQL syntax.

Post-migration challenges

Adopting New Tools and Infrastructure: BigQuery is a part of Google Cloud Platform, and it works well with other Google services. Your team might need to familiarize themselves with a whole new set of tools.

- New Data Management Practices: BigQuery uses a different model of computing resources, and it automatically scales based on the workload. This could change how data and IT teams manage data and resources, including considerations around cost control.
- SQL Syntax Differences: BigQuery uses a slightly different SQL syntax than Teradata. If your team is used to writing SQL for Teradata, they might need to learn some new syntax and conventions.
- Change Management: Adapting to new systems can be difficult. Your team will need to change their workflow and habits, which can lead to resistance.
- Performance Tuning: After migrating, data and IT teams might need to spend time optimizing query performance and cost in BigQuery. This can be a complex task and might require a deep understanding of how BigQuery processes queries.
- Training: Staff might need to be retrained to use BigQuery, which can require both time and expense.
- Continuing to Monitor and Optimize: Once you migrated to BigQuery, you must continually monitor your system to ensure that it works effectively and adjust as needed.

Once the data is successfully migrated, a review and optimization phase is crucial. This phase involves running tests to verify data integrity, analyzing performance, and fine-tuning the system as required. Additionally, feedback from users is collected to further improve and customize the system to better fit their needs. Post-migration, regular monitoring is crucial. For example, data teams might set up a daily check of failed SQL jobs in BigQuery. If data and IT teams see a recurring failure, data and IT teams can then dig deeper to understand the issue - perhaps a data type mismatch, or syntax error, and address the root cause.

Additional considerations exist for adopting new practices and terminologies related to cloud technologies. You must consider the following key points:

- Cloud Terminology: Familiarize yourself with the terminology used in cloud computing and specifically in Google Cloud Platform (GCP) and BigQuery. This includes terms like projects, datasets, tables, buckets, regions, and zones. Understanding these terms will help you navigate and manage your cloud environment effectively.
- Cloud Data Management: Cloud-based data management introduces new practices and tools. You must understand how data is stored, processed, and managed in BigQuery. This includes concepts like data partitioning, clustering, and streaming inserts. Learn about best practices for optimizing data storage, query performance, and cost management in BigQuery.
- Security and Compliance: Cloud environments have their unique security and compliance considerations. Familiarize yourself with the security features and options provided by GCP and BigQuery. Implement proper access controls, encryption, and data governance practices. Ensure compliance with relevant regulations and standards that apply to your data.
- Automation and Orchestration: Cloud platforms offer automation and orchestration capabilities that can streamline data pipelines and processes. Explore tools to automate data transformations, workflows, and data integration tasks. Leverage these tools to optimize your data pipelines in the cloud.
- Scalability and Elasticity: Cloud-based systems provide scalability and elasticity, allowing you to scale your resources up or down based on demand. Understand how to leverage the BigQuery auto-scaling capabilities to handle varying workloads efficiently. Design your data pipelines and infrastructure to take advantage of the cloud scalability benefits.

- Data Cataloging and Documentation: Ensure that you create a comprehensive data catalog in BigQuery that accurately documents your data assets, including metadata, data lineage, and business descriptions. This will help users find and understand the data in the new system.
- Continuous Learning and Training: Cloud technologies and best practices evolve rapidly. Encourage continuous learning and training for your data and IT teams to stay updated with the latest advancements in BigQuery and cloud computing. Leverage training resources, online documentation, and community forums provided by Google Cloud to enhance your knowledge and skills.

Efficient Teradata to BigQuery migration with Cloudera Data Lineage

Cloudera Data Lineage is a valuable solution that empowers businesses during the migration from Teradata to Google BigQuery. It automates data discovery, streamlining the migration process and ensuring a smooth transition. With the Cloudera Data Lineage lineage and impact analysis capabilities, businesses can enhance data management, validate data integrity, and improve overall reliability. Cloudera Data Lineage also reduces the time and effort required for identifying and migrating relevant data assets. It optimizes resource allocation by identifying and deprecating unused data, while ensuring data security and compliance through robust lineage tracking. By minimizing downtime and disruptions, Cloudera Data Lineage facilitates uninterrupted business operations during the migration. Overall, Cloudera Data Lineage delivers significant benefits, making the Teradata to BigQuery migration efficient and successful.

Change management

Data lineage supports change management for data flows in CI/CD DataOps and impact analysis when you handle schema changes.

Change management - Best practices for using Cloudera Data Lineage for CI/CD DataOps

As data developers and data governance teams, learn about how to effectively use the Cloudera Data Lineage (formerly Octopai) automated data lineage solution in managing changes to data flows within a CI/CD DataOps process through best practices for environment setup, impact analysis, automated refreshes, pre-production testing, and troubleshooting to ensure data integrity and governance.

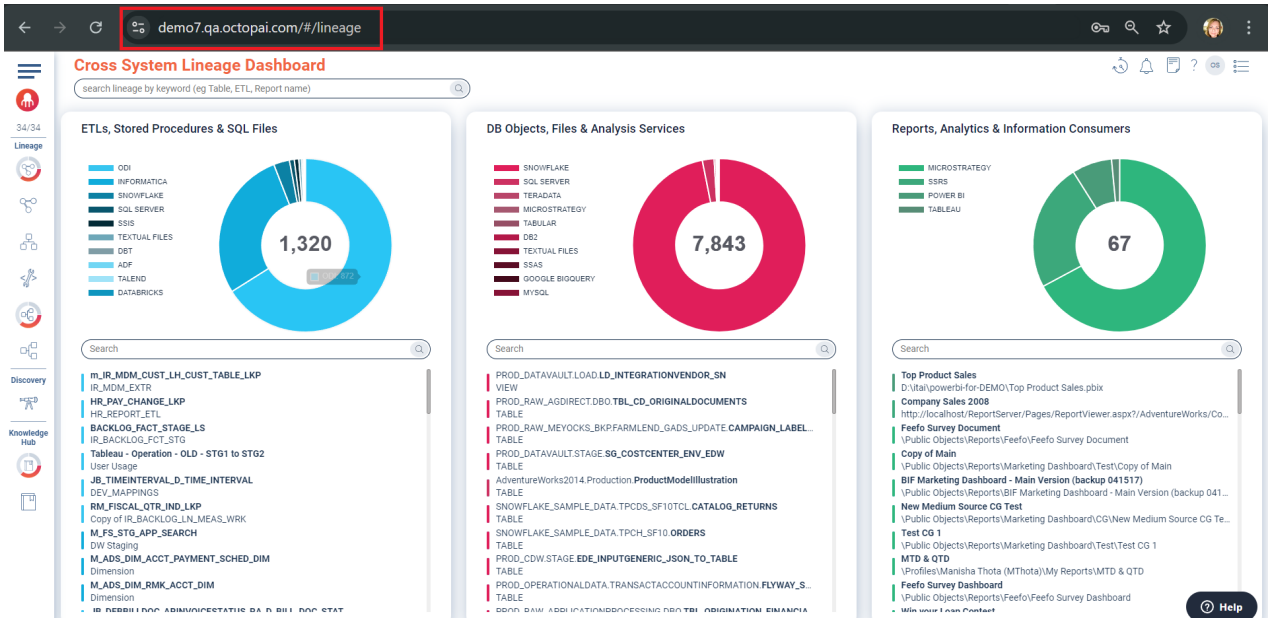
Your environments

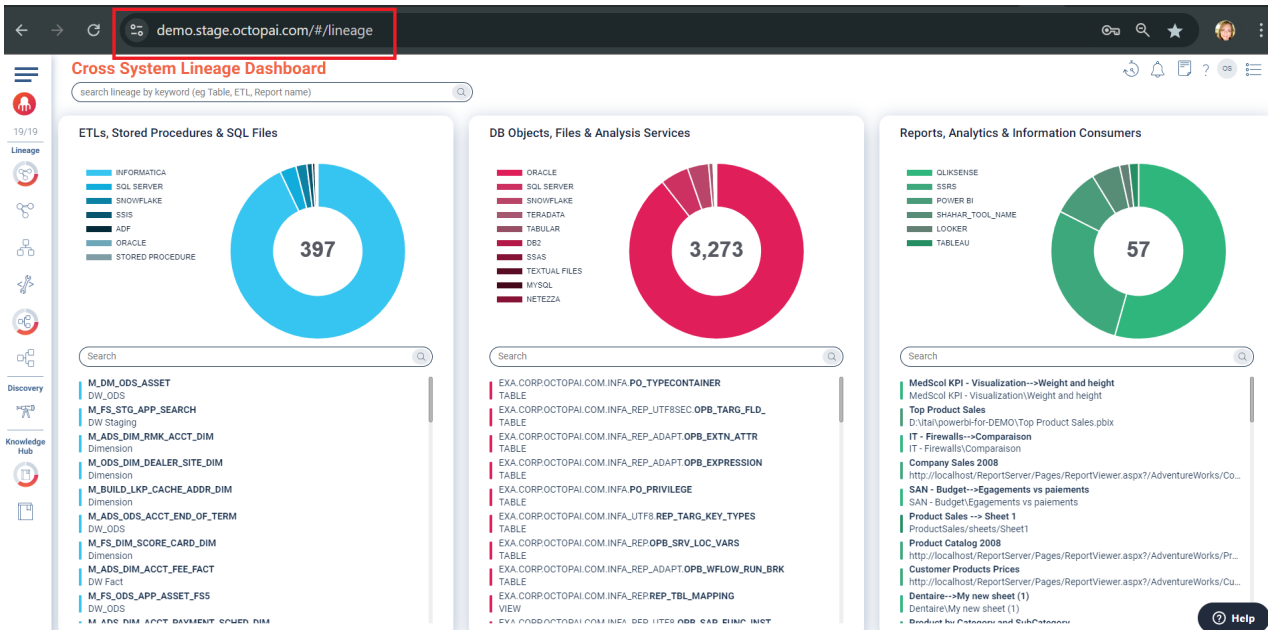
For data developers, establishing multiple environments is crucial to manage changes efficiently. This setup allows you to rigorously test changes before they reach production, while also aligning with data governance protocols. The best practice is to effectively use the following environments within Cloudera Data Lineage:

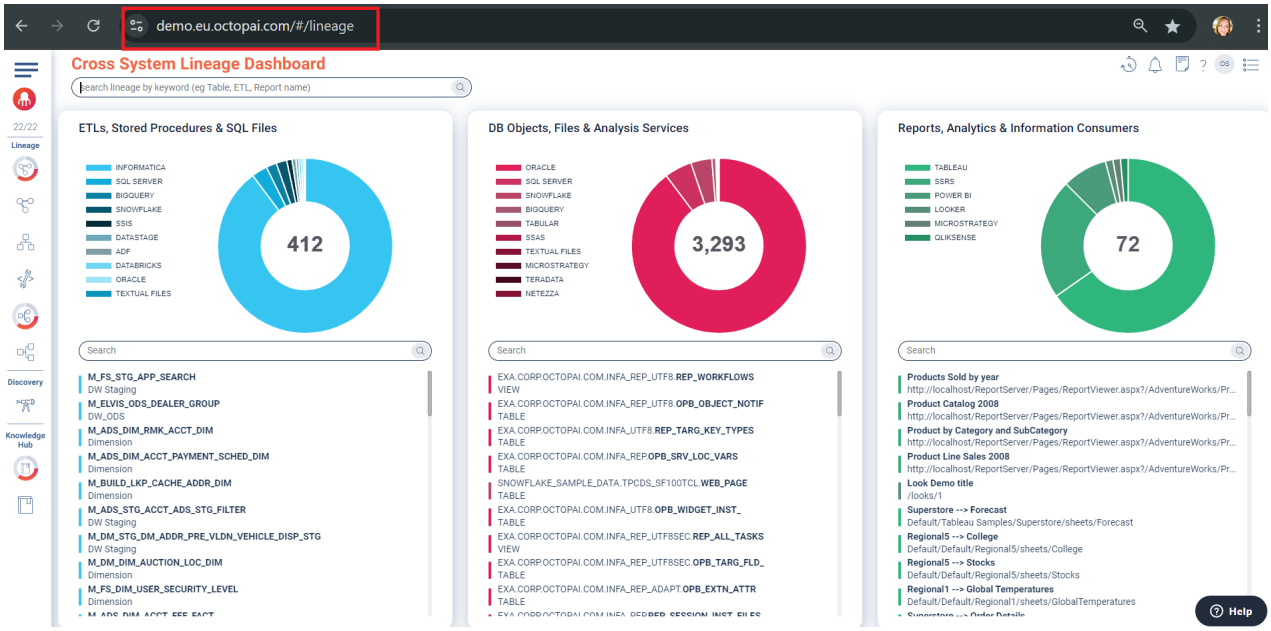
- **Development Environment** – In this environment, data developers introduce and iterate on changes, ensuring that the changes meet initial requirements.
- **QA Environment** – In this environment, data governance teams can validate changes against governance policies and standards by executing test plans to ensure compliance and data integrity.
- **Staging Environment** – In this environment, data developers and governance teams can mimic production and perform final validation.
- **Production Environment** – In this live environment, end-users and business processes actively use data with strict governance oversight.



Important: You must ensure that your organization's contract includes licensing for multiple environments. Coordinated this setup with the Cloudera Data Lineage Support team as part of your licensing agreement. Having this in place allows you to fully leverage Cloudera Data Lineage capabilities across all necessary environments, ensuring a smooth and compliant change management process.


demo7.qa.octopai.com/#/lineage
Cross System Lineage Dashboard
search lineage by keyword (eg Table, ETL, Report name)
ETLs, Stored Procedures & SQL Files
ODI
INFORMATICA
SNOWFLAKE
SQL SERVER
SSIS
TEXTUAL FILES
DBT
ADF
TALEND
DATABRICKS
1,320
DB Objects, Files & Analysis Services
SNOWFLAKE
SQL SERVER
TERADATA
MICROSTRATEGY
TABULAR
DB2
TEXTUAL FILES
SSAS
GOOGLE BIGQUERY
MYSQL
7,843
Reports, Analytics & Information Consumers
MICROSTRATEGY
SSRS
POWER BI
TABLEAU
67
Search
Help


demo.stage.octopai.com/#/lineage
Cross System Lineage Dashboard
search lineage by keyword (eg Table, ETL, Report name)
ETLs, Stored Procedures & SQL Files
INFORMATICA
SQL SERVER
SNOWFLAKE
SSIS
ADF
ORACLE
STORED PROCEDURE
397
DB Objects, Files & Analysis Services
ORACLE
SQL SERVER
SNOWFLAKE
TERADATA
TABULAR
DB2
SSAS
TEXTUAL FILES
MYSQL
NETEZZA
3,273
Reports, Analytics & Information Consumers
QLIKSENSE
SSRS
POWER BI
SHAHAR_TOOL_NAME
LOOKER
TABLEAU
57
Search
Help



Applying impact analysis and risk assessment

Understanding the potential impact of changes on data flows is critical for both data developers and governance teams by performing the following actions:

1. Trigger impact analysis.
   - Use Cloudera Data Lineage to identify upstream and downstream dependencies that might be affected by the change.
   - Ensure that the impact analysis aligns with governance policies, documenting any risks or compliance issues.
2. Conduct risk assessment.
   - Evaluate the technical risks associated with the change, such as potential disruptions to dependent systems.
   - Assess the risks from a compliance perspective, ensuring that all regulatory requirements are met.

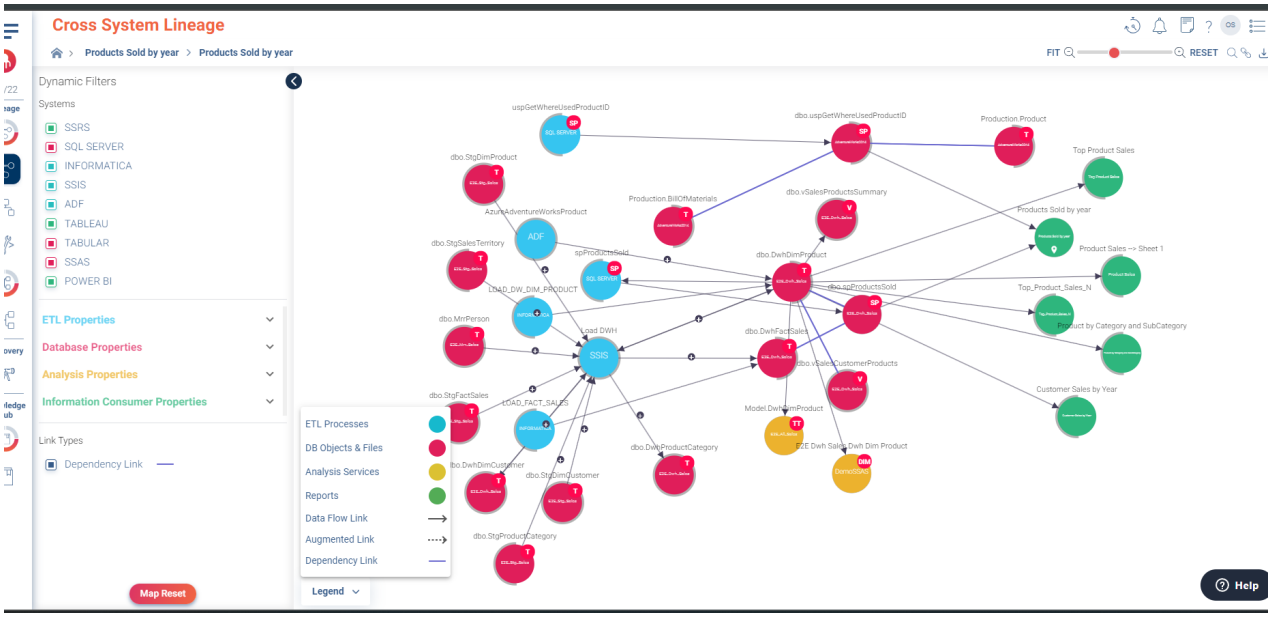


Figure 1: Upstream impact analysis

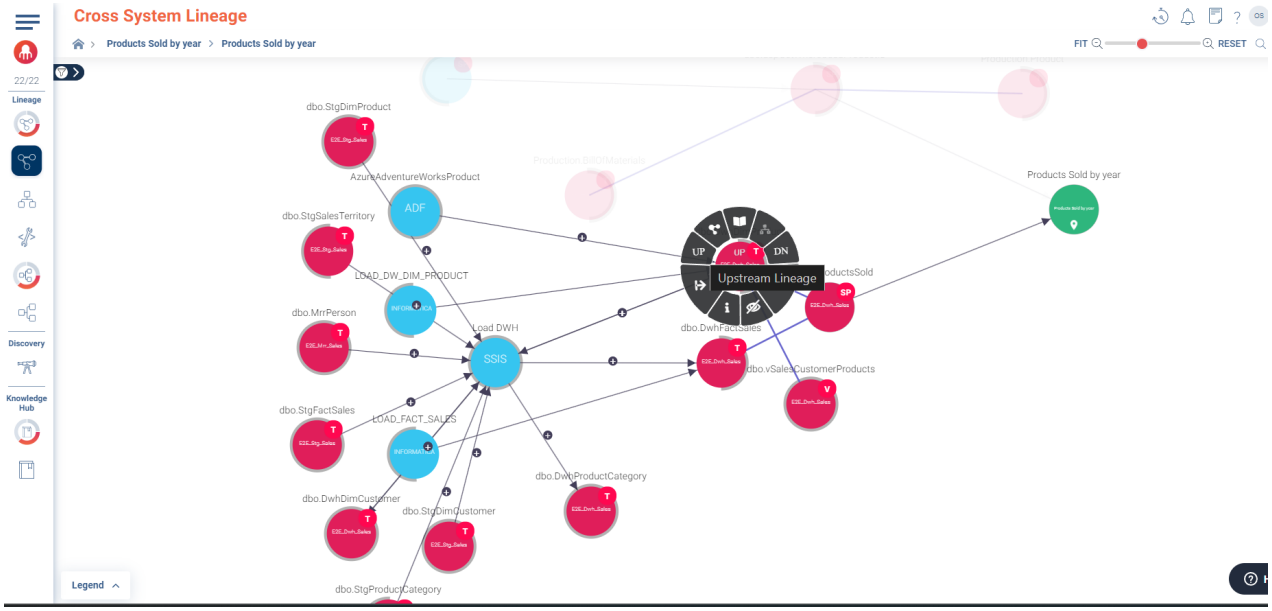
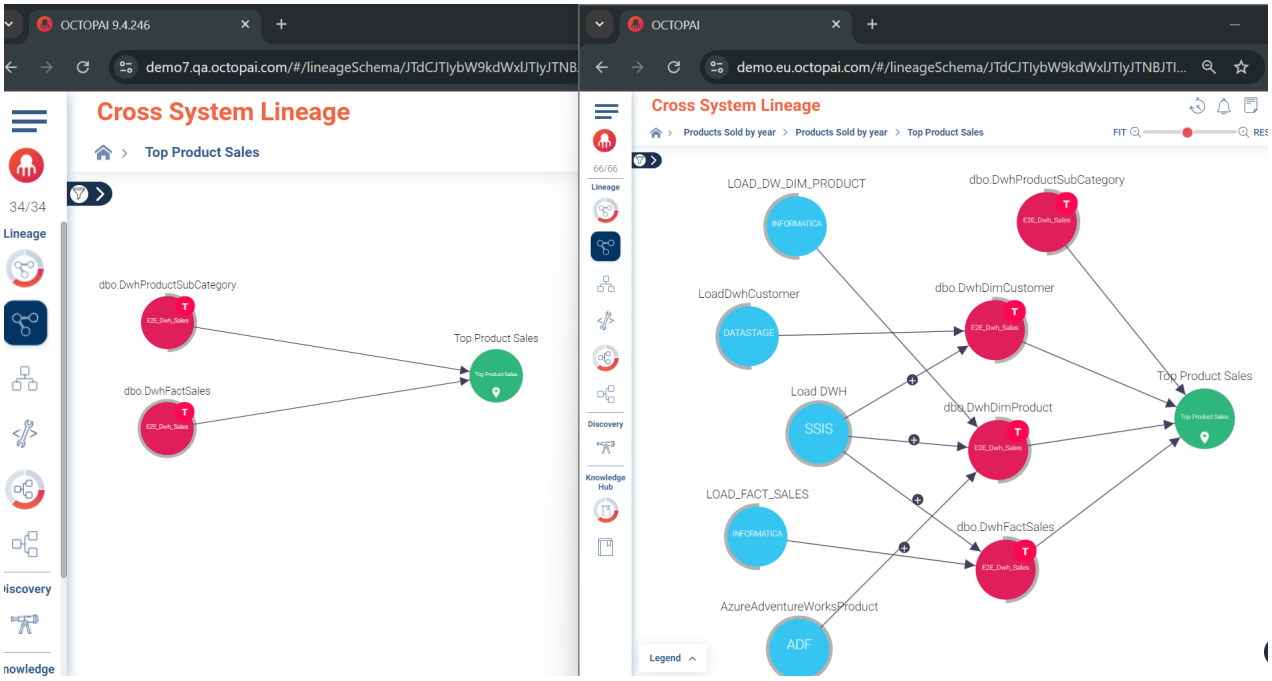


Figure 2: Comparison upstream impact analysis between QA and production



Automating data refresh for testing

Automating the data refresh process in Cloudera Data Lineage ensures that all environments reflect the most recent changes, which is essential for both effective development and governance. To automate data refresh, consider the following recommendations:

- Use Jenkins. For teams using Jenkins in their CI/CD pipeline, Cloudera Data Lineage can be integrated to automate the data refresh process. This allows data developers to work with the latest data lineage and governance teams to ensure continuous compliance.
- Use Octopai Client built-in feature for scheduling automatic data refreshes. This ensures that the lineage is always up-to-date, facilitating both development and governance efforts.
- Apply Jenkins in environments with established CI/CD pipelines where rigorous testing and governance are required. If not, the Cloudera Data Lineage scheduling features can serve as a simpler alternative.

- Apply the Cloudera Data Lineage automatic refresh solution on a connection level that is metadata source level.

For more information, see https://docs.cloudera.com/octopai/latest/administration/topics/oct-admin-user-octopai-client.html.

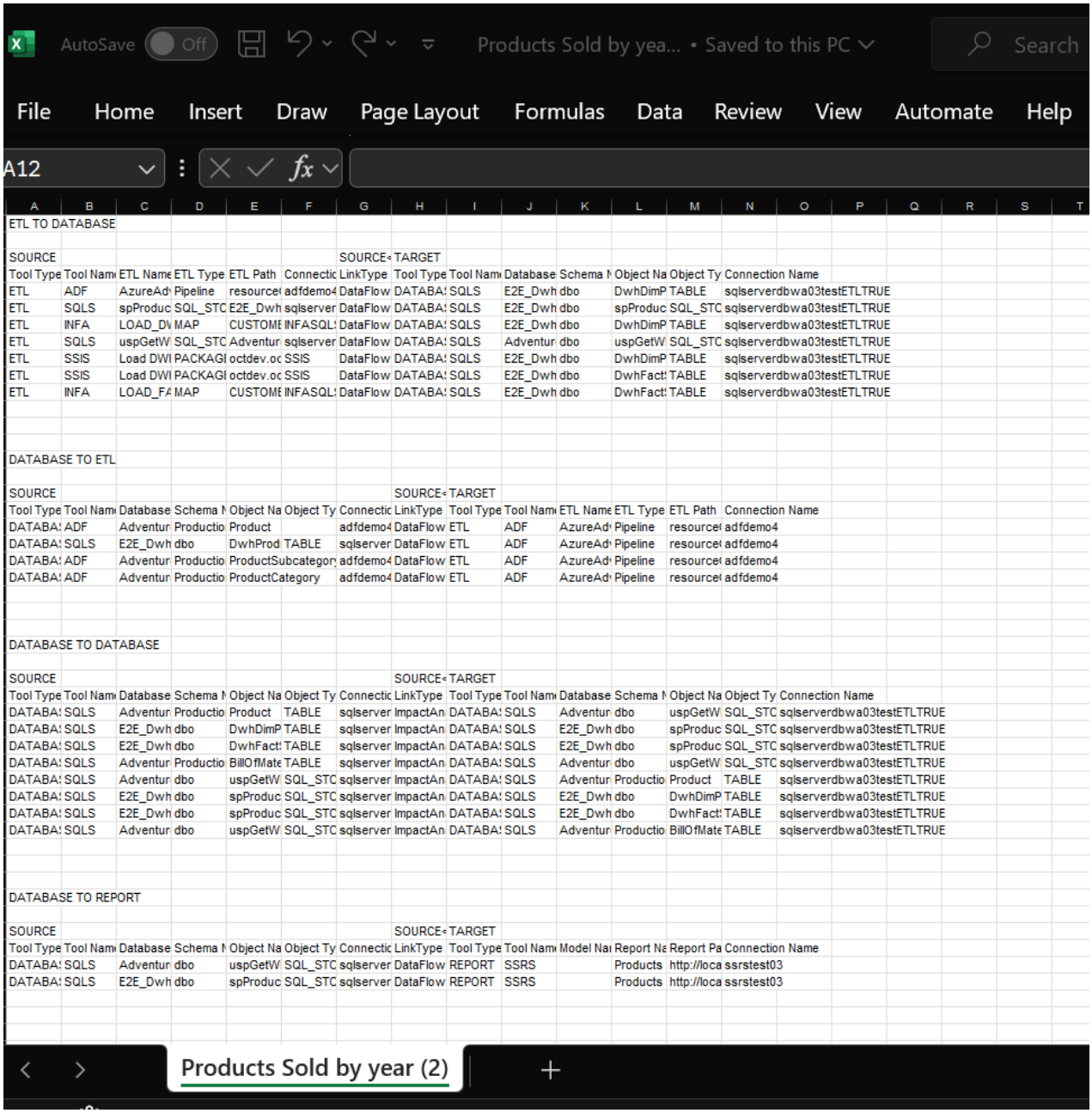
Simulating changes in pre-production

Simulating changes in a pre-production environment is essential to avoid unintended consequences in production, particularly when governance standards must be met. To simulate changes, perform the following actions:

- Simulate impact.
  - Use Cloudera Data Lineage to simulate the change and understand its technical implications across environments.
  - Ensure that the simulated impact is analysed for compliance risks, validating that the change adheres to governance policies.
- Create a regression test plan.
  - Develop a test plan that covers all critical data flows impacted by the change.
  - Validate that the test plan includes checks for compliance and data integrity.

Use the Export of Cloudera Data Lineage E2E Column Lineage capability as a foundation for your test plan.

Figure 3: Export of Cloudera Data Lineage E2E Column Lineage



ETL TO DATABASE

SOURCE						SOURCE<	TARGET										
Tool Type	Tool Nam	ETL Name	ETL Type	ETL Path	Connectic	LinkType	Tool Type	Tool Nam	Database	Schema N	Object Na	Object Ty	Connection Name				
ETL	ADF	AzureAd	Pipeline	resource	adfdemo4	DataFlow	DATABA	SQLS	E2E_Dwh	dbo	DwhDimP	TABLE	sqlserverdbwa03testETLTRUE				
ETL	SQLS	spProduc	SQL_STC	E2E_Dwh	sqlserver	DataFlow	DATABA	SQLS	E2E_Dwh	dbo	spProduc	SQL_STC	sqlserverdbwa03testETLTRUE				
ETL	INFA	LOAD_DV	MAP	CUSTOME	INFASQL	DataFlow	DATABA	SQLS	E2E_Dwh	dbo	DwhDimP	TABLE	sqlserverdbwa03testETLTRUE				
ETL	SQLS	uspGetW	SQL_STC	Adventur	sqlserver	DataFlow	DATABA	SQLS	Adventur	dbo	uspGetW	SQL_STC	sqlserverdbwa03testETLTRUE				
ETL	SSIS	Load DWI	PACKAGI	octdev.oc	SSIS	DataFlow	DATABA	SQLS	E2E_Dwh	dbo	DwhDimP	TABLE	sqlserverdbwa03testETLTRUE				
ETL	SSIS	Load DWI	PACKAGI	octdev.oc	SSIS	DataFlow	DATABA	SQLS	E2E_Dwh	dbo	DwhFact	TABLE	sqlserverdbwa03testETLTRUE				
ETL	INFA	LOAD_FA	MAP	CUSTOME	INFASQL	DataFlow	DATABA	SQLS	E2E_Dwh	dbo	DwhFact	TABLE	sqlserverdbwa03testETLTRUE				

DATABASE TO ETL

SOURCE							SOURCE<	TARGET					
Tool Type	Tool Nam	Database	Schema N	Object Na	Object Ty	Connectic	LinkType	Tool Type	Tool Nam	ETL Name	ETL Type	ETL Path	Connection Name
DATABA	ADF	Adventur	Productio	Product		adfdemo4	DataFlow	ETL	ADF	AzureAd	Pipeline	resource	adfdemo4
DATABA	SQLS	E2E_Dwh	dbo	DwhProd	TABLE	sqlserver	DataFlow	ETL	ADF	AzureAd	Pipeline	resource	adfdemo4
DATABA	ADF	Adventur	Productio	ProductSubcategor		adfdemo4	DataFlow	ETL	ADF	AzureAd	Pipeline	resource	adfdemo4
DATABA	ADF	Adventur	Productio	ProductCategory		adfdemo4	DataFlow	ETL	ADF	AzureAd	Pipeline	resource	adfdemo4

DATABASE TO DATABASE

SOURCE								SOURCE<	TARGET						
Tool Type	Tool Nam	Database	Schema N	Object Na	Object Ty	Connectic	LinkType	Tool Type	Tool Nam	Database	Schema N	Object Na	Object Ty	Connection Name	
DATABA	SQLS	Adventur	Productio	Product	TABLE	sqlserver	ImpactAn	DATABA	SQLS	Adventur	dbo	uspGetW	SQL_STC	sqlserverdbwa03testETLTRUE	
DATABA	SQLS	E2E_Dwh	dbo	DwhDimP	TABLE	sqlserver	ImpactAn	DATABA	SQLS	E2E_Dwh	dbo	spProduc	SQL_STC	sqlserverdbwa03testETLTRUE	
DATABA	SQLS	E2E_Dwh	dbo	DwhFact	TABLE	sqlserver	ImpactAn	DATABA	SQLS	E2E_Dwh	dbo	spProduc	SQL_STC	sqlserverdbwa03testETLTRUE	
DATABA	SQLS	Adventur	Productio	BillOfMate	TABLE	sqlserver	ImpactAn	DATABA	SQLS	Adventur	dbo	uspGetW	SQL_STC	sqlserverdbwa03testETLTRUE	
DATABA	SQLS	Adventur	dbo	uspGetW	SQL_STC	sqlserver	ImpactAn	DATABA	SQLS	Adventur	Productio	Product	TABLE	sqlserverdbwa03testETLTRUE	
DATABA	SQLS	E2E_Dwh	dbo	spProduc	SQL_STC	sqlserver	ImpactAn	DATABA	SQLS	E2E_Dwh	dbo	DwhDimP	TABLE	sqlserverdbwa03testETLTRUE	
DATABA	SQLS	E2E_Dwh	dbo	spProduc	SQL_STC	sqlserver	ImpactAn	DATABA	SQLS	E2E_Dwh	dbo	DwhFact	TABLE	sqlserverdbwa03testETLTRUE	
DATABA	SQLS	Adventur	dbo	uspGetW	SQL_STC	sqlserver	ImpactAn	DATABA	SQLS	Adventur	Productio	BillOfMate	TABLE	sqlserverdbwa03testETLTRUE	

DATABASE TO REPORT

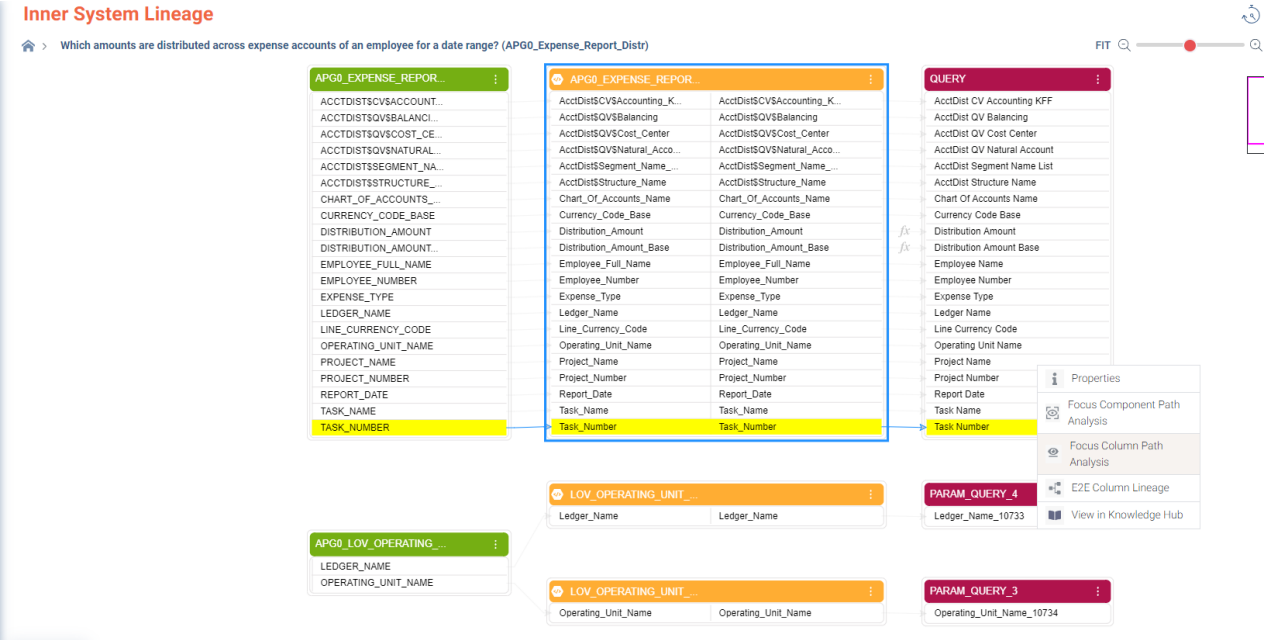
SOURCE							SOURCE<	TARGET					
Tool Type	Tool Nam	Database	Schema N	Object Na	Object Ty	Connectic	LinkType	Tool Type	Tool Nam	Model Na	Report Na	Report Pa	Connection Name
DATABA	SQLS	Adventur	dbo	uspGetW	SQL_STC	sqlserver	DataFlow	REPORT	SSRS		Products	http://loca	ssrstest03
DATABA	SQLS	E2E_Dwh	dbo	spProduc	SQL_STC	sqlserver	DataFlow	REPORT	SSRS		Products	http://loca	ssrstest03

Troubleshooting in production

When a change leads to an issue in production, Cloudera Data Lineage helps both data developers and governance teams quickly identify and resolve the problem.

Use lineage information for troubleshooting by performing the following actions:

- Trace the issue back to its source within the data flow, identifying the root cause.
- Ensure that the resolution process aligns with governance standards, updating documentation as needed.



Document the solution by performing the following actions:

- Document the technical steps taken to resolve the issue.
- Update governance documentation to reflect the resolution and any changes to compliance processes.

Schema management - Best practices for handling changes and impact analysis

Learn about best practices for managing schema changes and conducting impact analysis using Cloudera Data Lineage (formerly Octopai).

When comparing schemas across different environments, several types of changes are commonly encountered. These changes can have varying degrees of impact on your data flows and applications. By focusing on the strengths of Cloudera Data Lineage, such as its Dynamic Filters, inner system maps, Discovery Space, and detailed impact analysis, you can effectively manage and compare schema changes across environments, ensuring consistency, compliance, and operational efficiency.

You might need to address the following typical changes:

- **Table structure modifications**
  - Addition or removal of columns – A common change when new columns are added to or existing columns are removed from a table. This can affect how data is transformed and used downstream.
  - Data type changes – Modifying the data type of a column, for example changing an INT to a VARCHAR. Such changes can lead to data mismatches or errors if not handled consistently across environments.
  - Column renaming – Renaming a column can break dependencies in data flows if the new name is not updated throughout the pipeline.
- **Creating or dropping indexes** – Indexes are often added or removed to optimize performance. However, these changes can impact query performance differently across environments, potentially leading to inconsistent results.
- Adding or dropping constraints – Constraints like primary keys, foreign keys, and unique constraints ensure data integrity. Changes to these can lead to different behaviour in data validation and integrity checks.
- **Stored procedures and triggers**
  - Modifications to business logic – Changes to stored procedures or triggers, which encapsulate business logic, can have cascading effects on data operations and need to be carefully managed and tested across environments.

- **Partitioning and clustering changes**
  - Adjusting partition schemes – Modifying how tables are partitioned can impact query performance and storage efficiency, requiring careful comparison to ensure consistency in data processing across environments.
- **New or altered views** – Views are often used to simplify complex queries or present data differently. Any changes to views should be examined for their impact on dependent reports or applications.

Recommended best practices

Conduct impact analysis with Cloudera Data Lineage

With the Cloudera Data Lineage robust data lineage capabilities, you can perform detailed impact analysis before implementing schema changes across environments. This helps you foresee how modifications will affect data flows and downstream systems in different environments, ensuring smooth transitions and preventing disruptions.

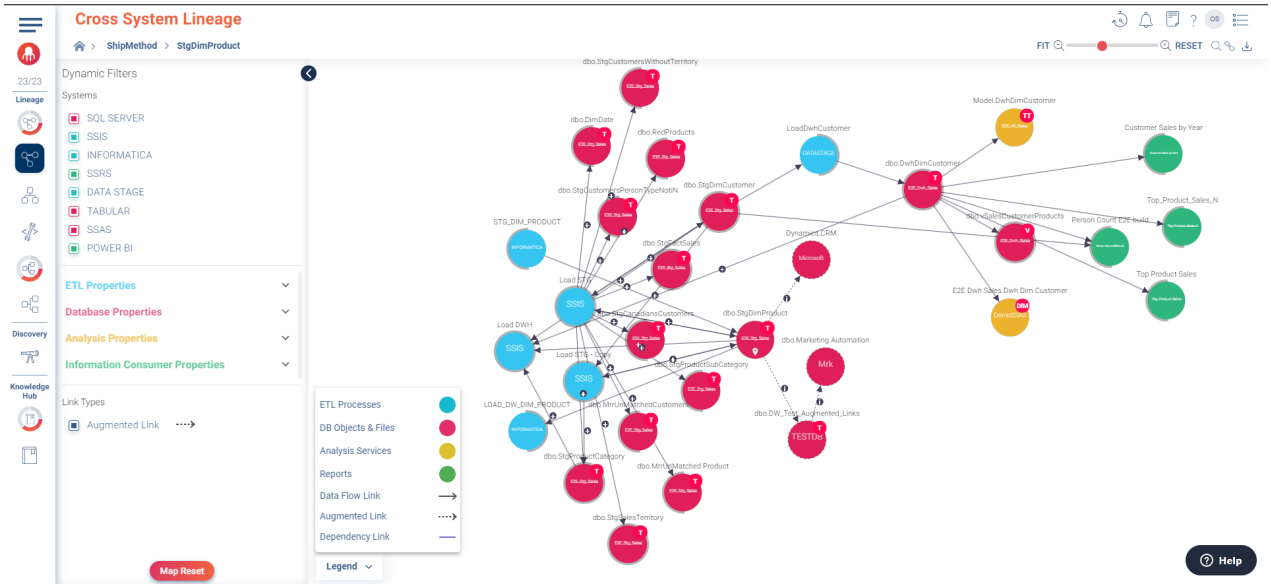
Cross-environment schema management

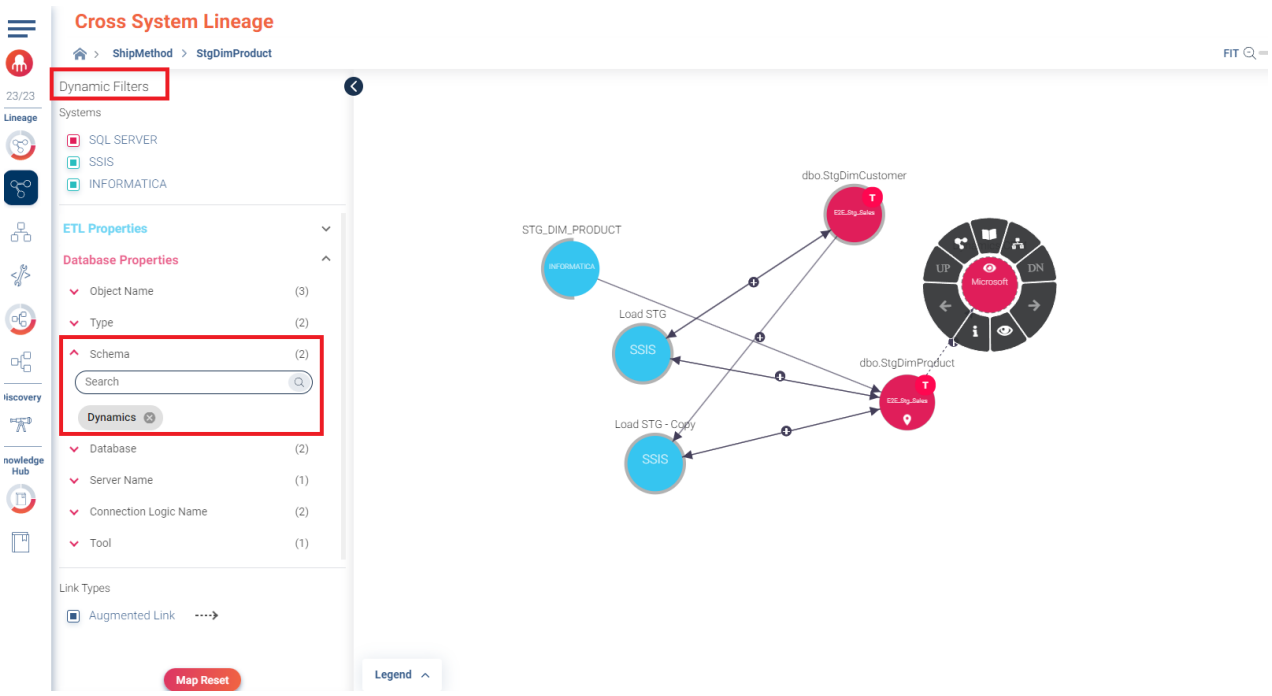
Use the Cloudera Data Lineage metadata management features to maintain consistent schemas across different environments, such as development, QA, and production. By carefully monitoring and comparing metadata, you can ensure that all changes are implemented consistently, reducing the risk of discrepancies and data integrity issues across environments.

Leverage dynamic filters for focused analysis

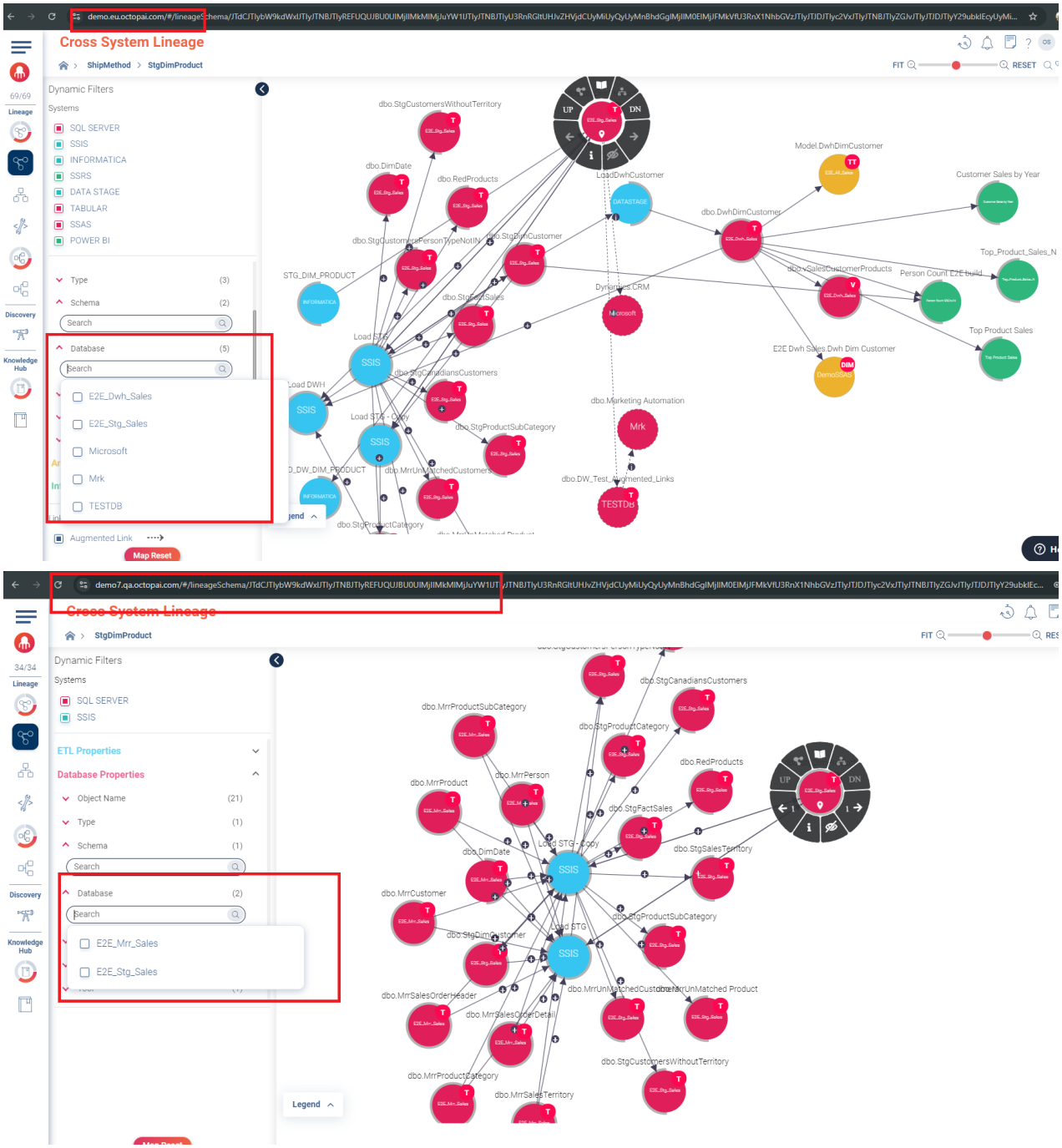
Cloudera Data Lineage Dynamic Filters feature is a powerful tool for managing and analysing schema changes across different environments. Dynamic Filters allow you to narrow down the scope of your analysis by focusing on specific environments, systems, or metadata properties. You can utilize operational metadata properties of an asset to concentrate on specific object characteristics, such as schema, type, and database. This capability helps ensure that you are analysing the most relevant aspects of your data lineage, making it easier to compare environments and identify discrepancies.

Figure 4: Dynamic filters




Cross System Lineage
ShipMethod > StgDimProduct
FIT
Dynamic Filters
Systems
SQL SERVER
SSIS
INFORMATICA
ETL Properties
Database Properties
Object Name (3)
Type (2)
Schema (2)
Search
Dynamics
Database (2)
Server Name (1)
Connection Logic Name (2)
Tool (1)
Link Types
Augmented Link
Map Reset
Legend
dbo.StgDimCustomer
STG_DIM_PRODUCT
Load STG
Load STG - Copy
dbo.StgDimProduct
SSIS
INFORMATICA
Microsoft

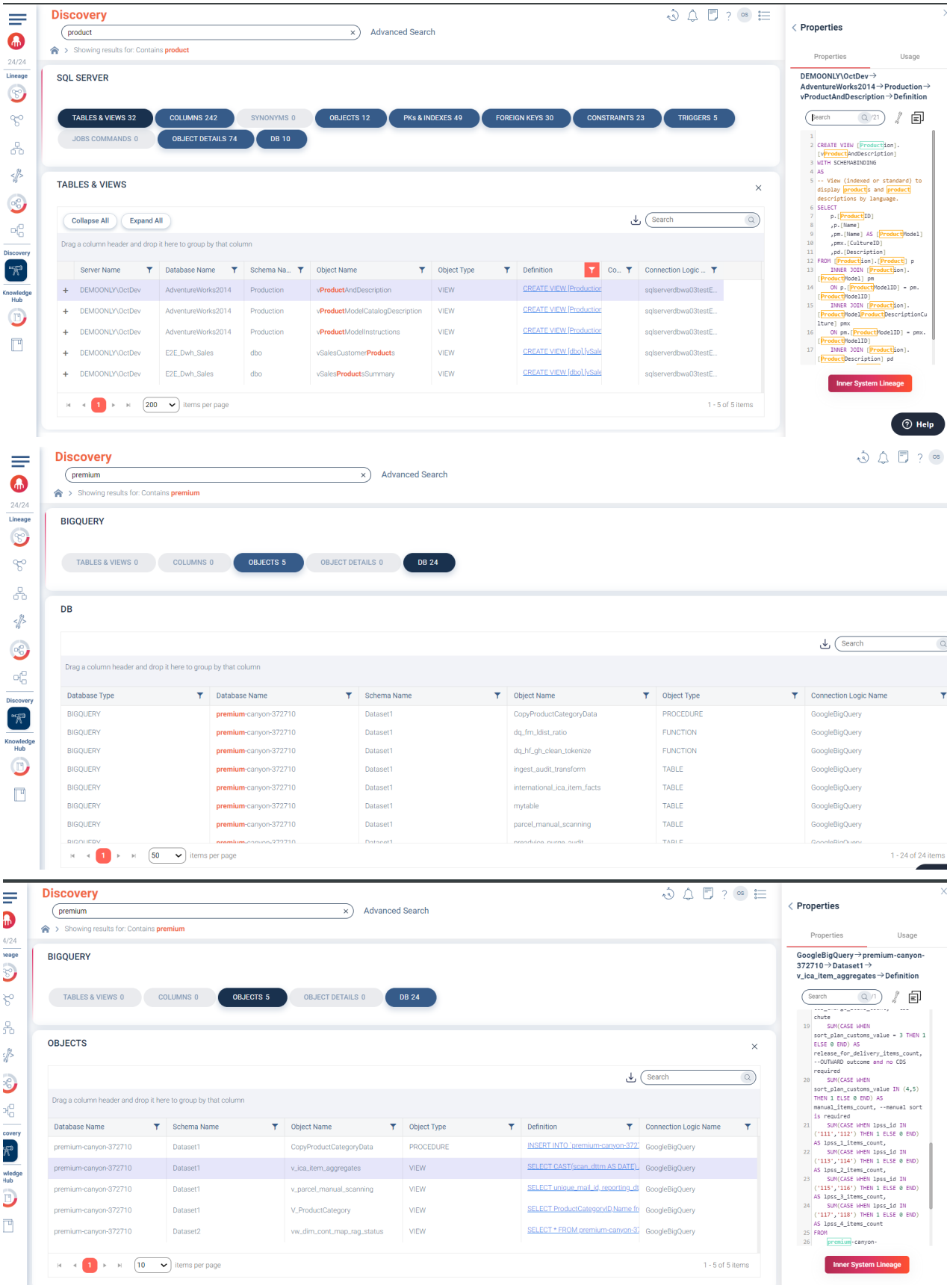

Cross System Lineage
ShipMethod > StgDimProduct
Dynamic Filters
Systems
SQL SERVER
SSIS
INFORMATICA
ETL Properties
Database Properties
Object Name (3)
Type (2)
Search
APPLICATION
TABLE
Schema (2)
Search
dbo
Dynamics
Database (2)
Search
E2E_Stg_Sales
Server Name (1)
Connection Logic Name (2)
Tool (1)
Link Types
Augmented Link
Legend
dbo.StgDimCustomer
STG_DIM_PRODUCT
Load STG
Load STG - Copy
dbo.StgDimProduct
Dynamics CRM
SSIS
INFORMATICA



Utilize the discovery space for in-depth environment comparison

The Discovery Space in Cloudera Data Lineage is an essential feature for comparing environments and gaining deeper insights into your data landscape. It allows you to visually explore and compare data assets, their relationships, and dependencies across different systems and environments. This space is particularly useful for identifying potential issues, understanding the broader context of specific data elements, and conducting in-depth analysis of how different environments align or diverge. You can drill down into specific assets within the Discovery Space, explore their lineage, and uncover hidden connections, making it a powerful tool for comprehensive environment comparison.

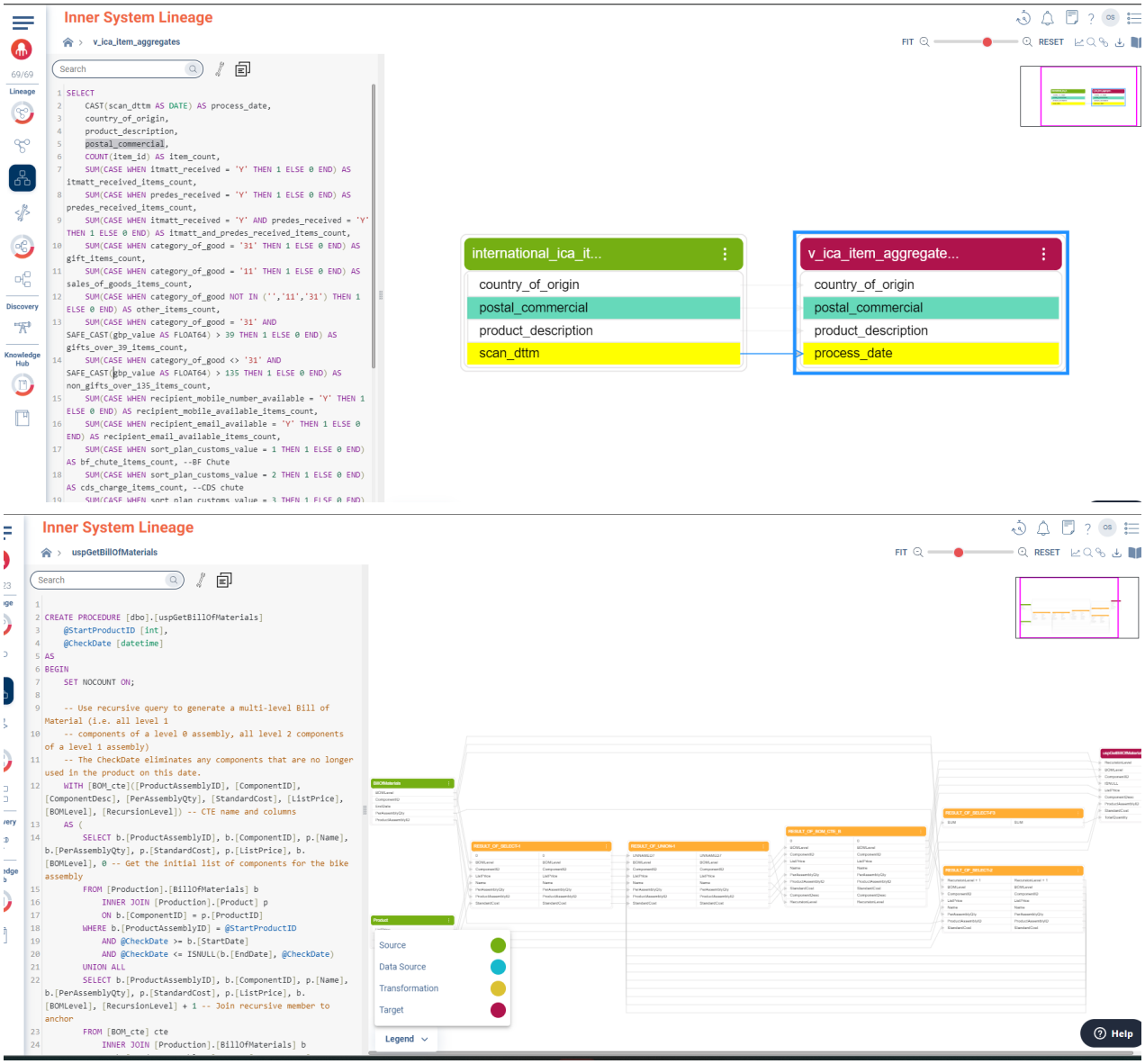
Figure 5: Schema discovery

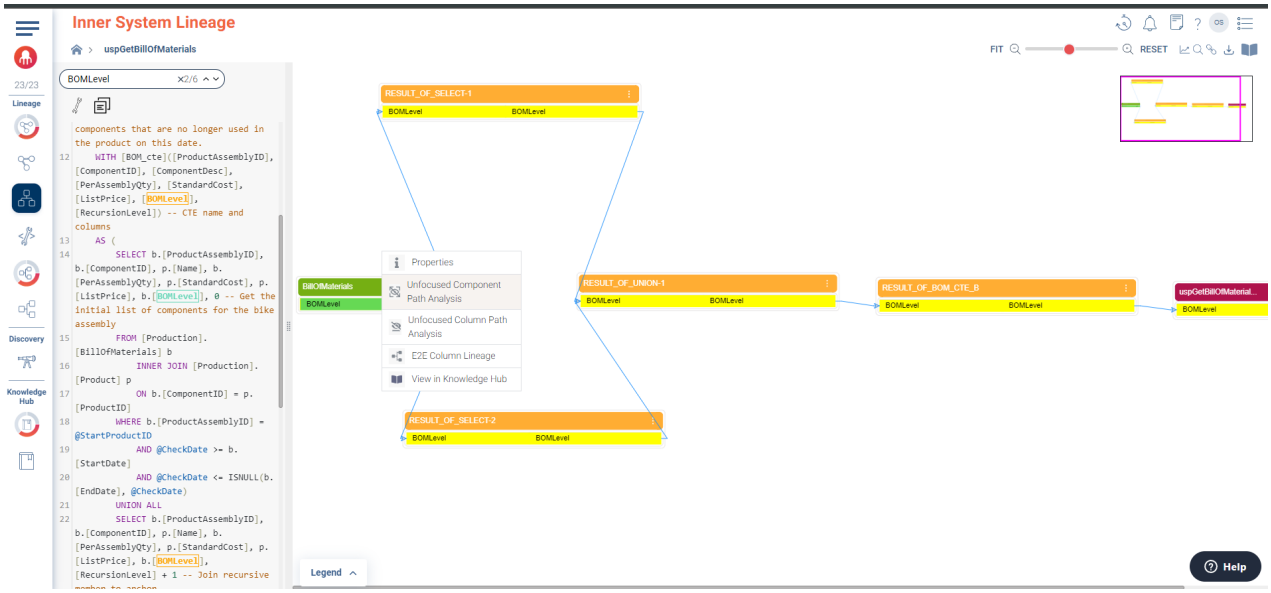


Use inner system maps for detailed comparison

Within Cloudera Data Lineage, you can also utilize inner system maps to perform focused component and asset analysis. This feature enables you to compare specific elements between different environments with precision. By honing in on specific components, you can identify and address differences that might impact data consistency or performance across your environments, ensuring that all environments are aligned and functioning optimally.

Figure 6: Inner system lineage





Documentation and compliance

Cloudera Data Lineage excels in creating comprehensive documentation of your data environments. This documentation is essential for compliance, as it provides a clear trail of how data moves and changes across your systems. Regularly updated lineage documentation supports audits and helps maintain a clear understanding of your data landscape across all environments.

Post-deployment verification

After deploying changes, the Cloudera Data Lineage tools can be used to verify that your environments remain consistent and functional. This step is crucial for ensuring that your production environment continues to operate smoothly and efficiently.