# ADVANCED DATA ENGINEERING IN SNOWFLAKE

Parag Jain | Snowflake AFE

# Safe Harbor and Disclaimers

Other than statements of historical fact, all information contained in these materials and any accompanying oral commentary (collectively, the “Materials”), including statements regarding (i) Snowflake’s business strategy, plans or priorities, (ii) Snowflake’s new or enhanced products, services, and technology offerings, including those that are under development or not generally available, (iii) market growth, trends, and competitive considerations, (iv) our vision for Snowpark, the Data Cloud, and industry-specific Data Clouds, including the expected benefits and network effects of the Data Cloud; and (v) the integration, interoperability, and availability of Snowflake’s products, services, or technology offerings with or on third-party platforms or products, are forward-looking statements. These forward-looking statements are subject to a number of risks, uncertainties and assumptions, including those described under the heading “Risk Factors” and elsewhere in the Annual Reports on Form 10-K and the Quarterly Reports on Form 10-Q that Snowflake files with the Securities and Exchange Commission. In light of these risks, uncertainties, and assumptions, the future events and trends discussed in the Materials may not occur, and actual results could differ materially and adversely from those anticipated or implied in the forward-looking statements. As a result, you should not rely on any forwarding-looking statements as predictions of future events. Forward-looking statements speak only as of the date the statements are first made and are based on information available to us at the time those statements are made and/or management's good faith belief as of that time. Except as required by law, we undertake no obligation, and do not intend, to update the forward-looking statements in these Materials.

Any future product or roadmap information (collectively, the “Roadmap”) is intended to outline general product direction. The Roadmap is not a commitment, promise, or legal obligation for Snowflake to deliver any future products, features, or functionality; and is not intended to be, and shall not be deemed to be, incorporated into any contract. The actual timing of any product, feature, or functionality that is ultimately made available may be different from what is presented in the Roadmap. The Roadmap information should not be used when making a purchasing decision. In case of conflict between the information contained in the Materials and official Snowflake documentation, official Snowflake documentation should take precedence over these Materials. Further, note that Snowflake has made no determination as to whether separate fees will be charged for any future products, features, and/or functionality which may ultimately be made available. Snowflake may, in its own discretion, choose to charge separate fees for the delivery of any future products, features, and/or functionality which are ultimately made available.

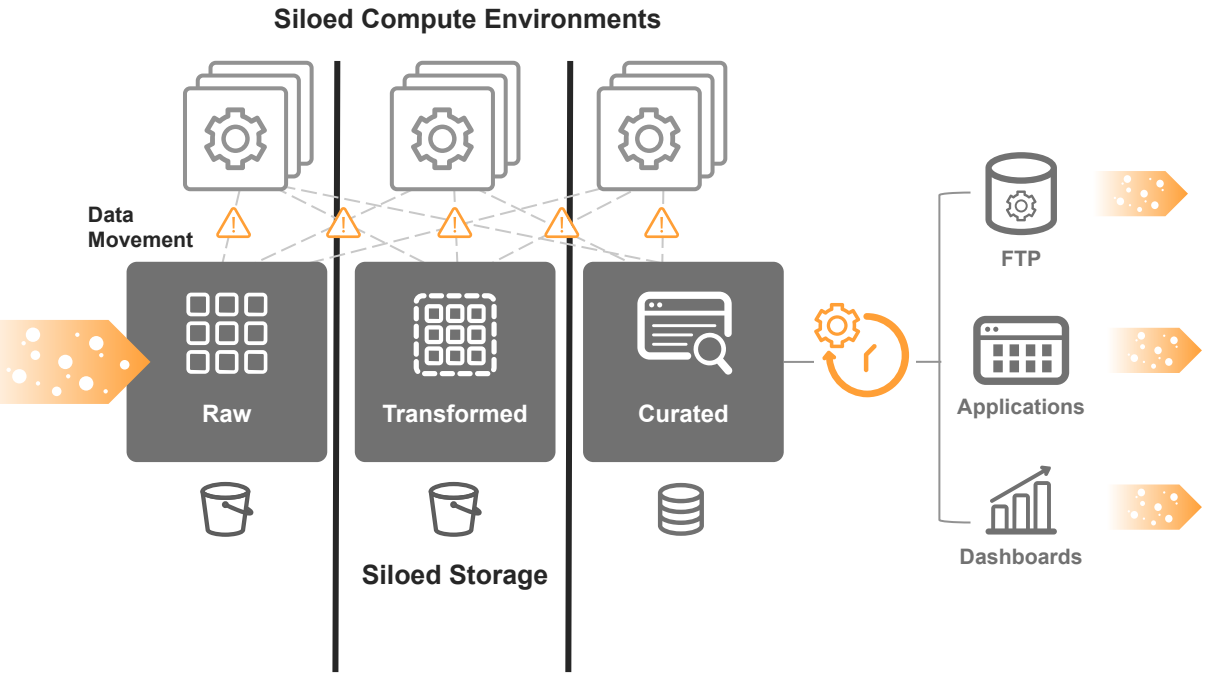
The Materials may contain information provided by third-parties. Snowflake has not independently verified this information, and usage of this information does not mean or imply that Snowflake has adopted this information as its own or independently verified its accuracy.

# Forward Looking Statement

This content is being provided under NDA. These materials are for your organization’s internal use only. You may not disseminate the materials or any of the information contained herein, in whole or in part, to any person outside of your organization. Metrics contained within this document are frequently updated, and are not guaranteed to be current.

BEFORE

# Challenges with Traditional Data Engineering

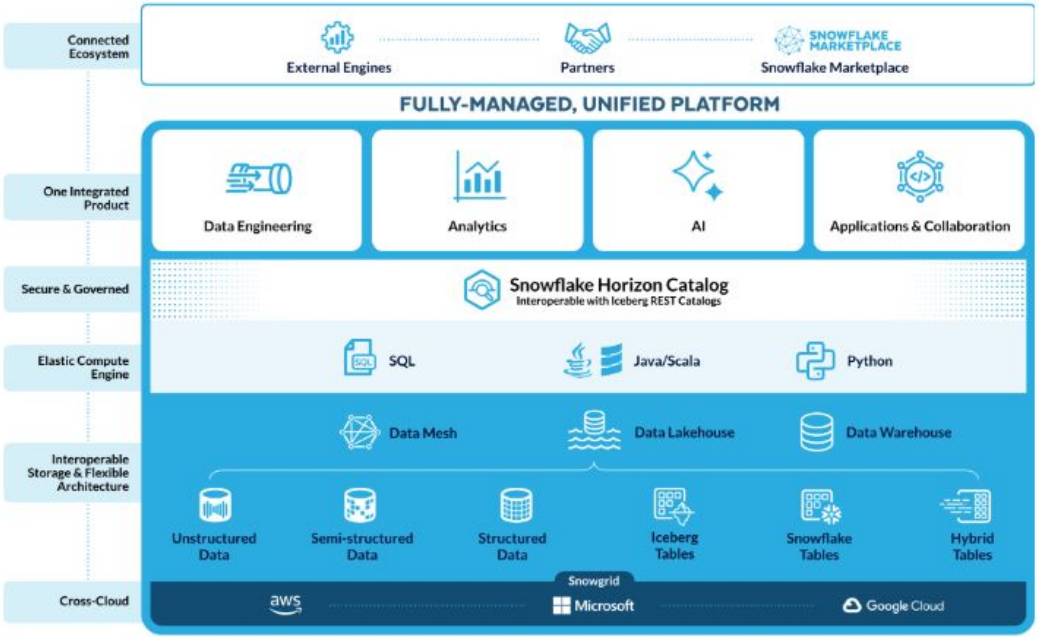



## CHALLENGES

- Customers often run separate processing clusters for different languages
- Complex capacity management, resource sizing and configurations
- Vendor locked-in with proprietary format & catalog
- Data can be siloed and difficult to share
- Disjointed governance and security

AFTER

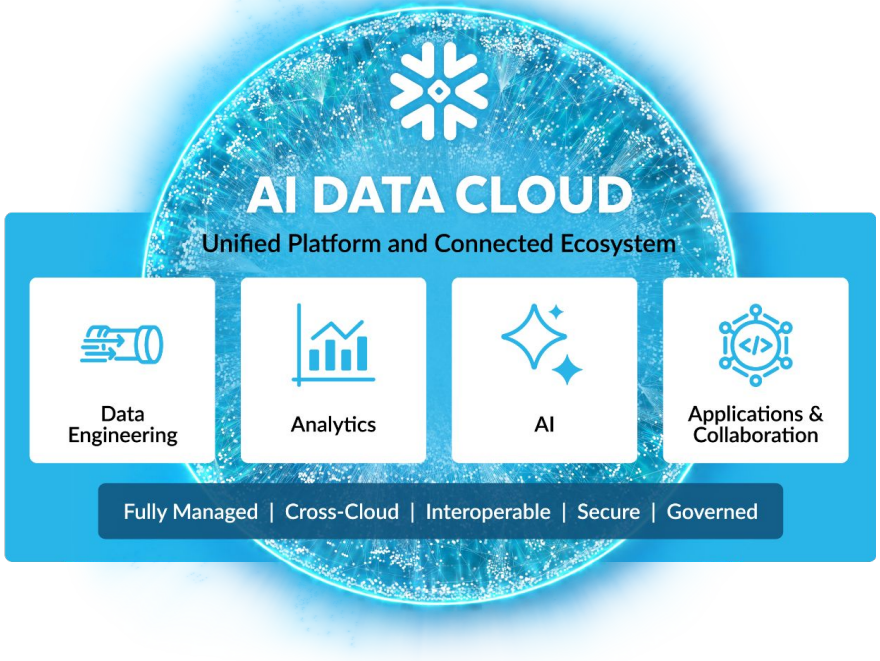
# Streamlined Architecture with Snowflake




## TECHNICAL REQUIREMENTS

- Single platform with native support for different languages
- Automatic capacity management and resource sizing
- Interoperability with open formats and flexible architecture
- Workload isolation and high concurrency
- Consistent governance and security policies

# Why Build Data Pipelines With Snowflake?




- Simple Options for Any Pipeline Use Case
- Data Sharing → Fewer Pipelines
- Python and SQL in One Compute Engine

# Build a Connected & Governed Lakehouse

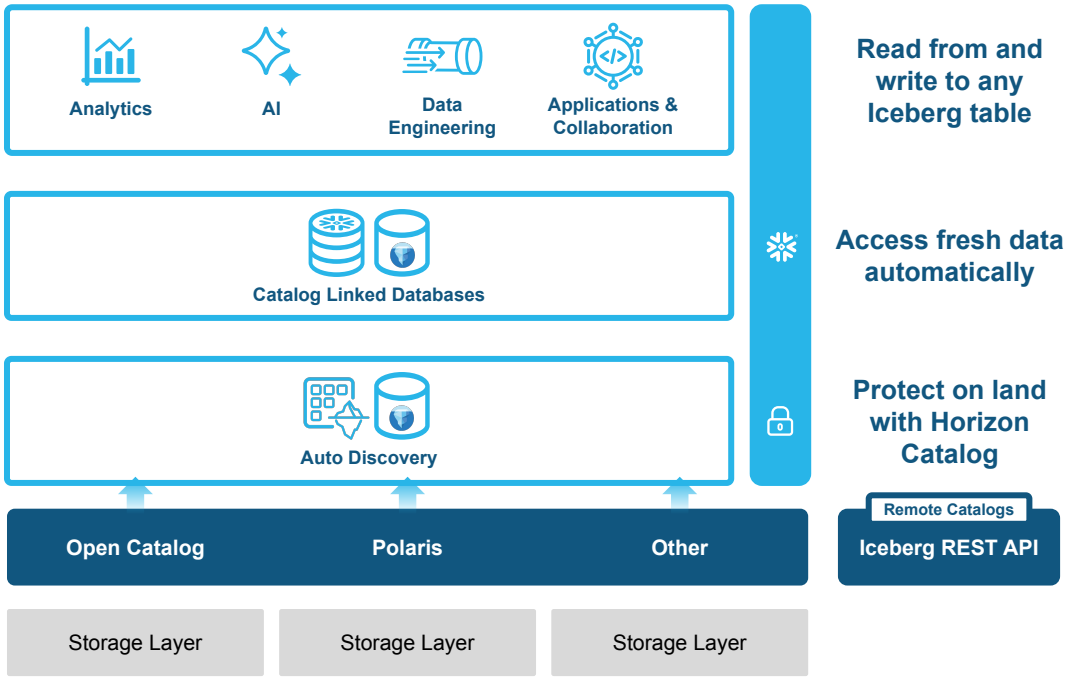
## Centralize & activate data from nearly anywhere

Fully **extend Snowflake** to any Iceberg table for **read and write** operations

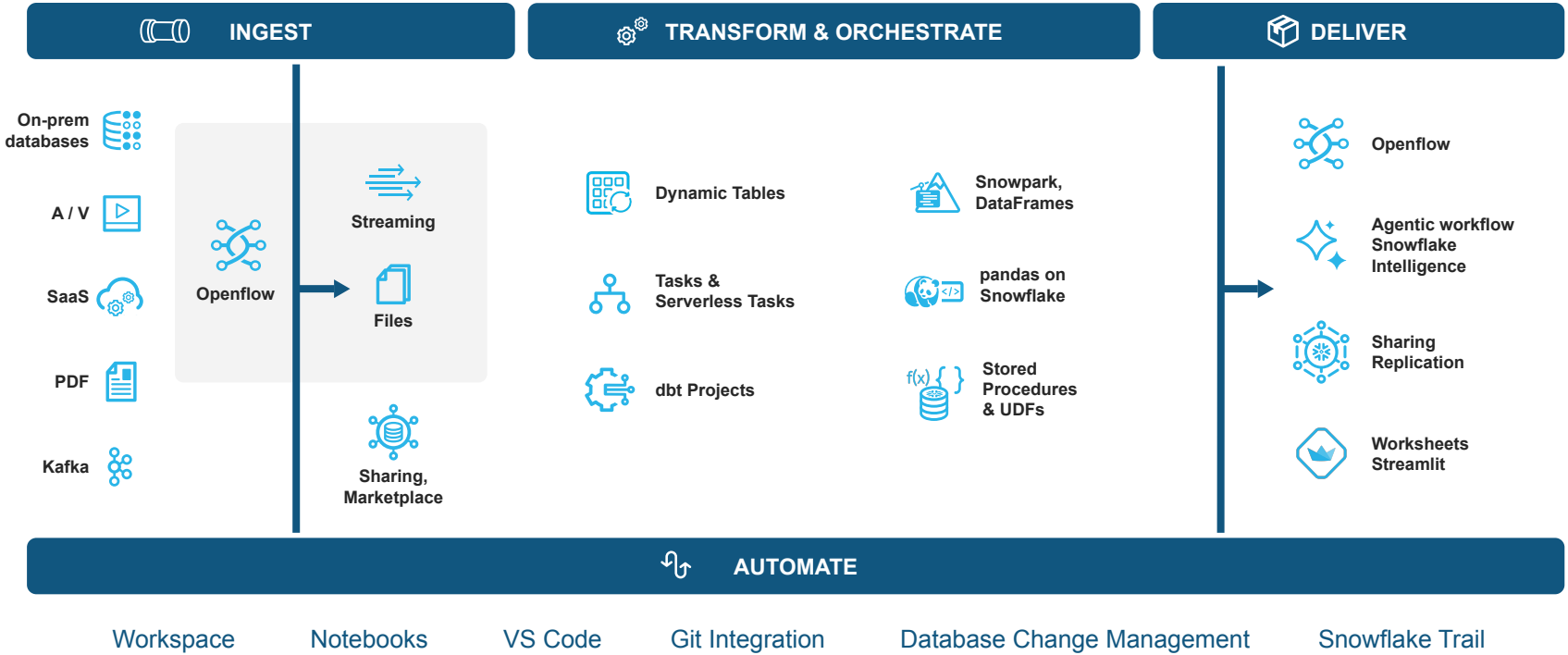
**Automatically detect & update changes** to tables in a given database and your storage layer

Make external Iceberg tables accessible within Snowflake with **Auto Discovery**

**Link Open Catalog** or any remote Iceberg REST Catalog to Snowflake

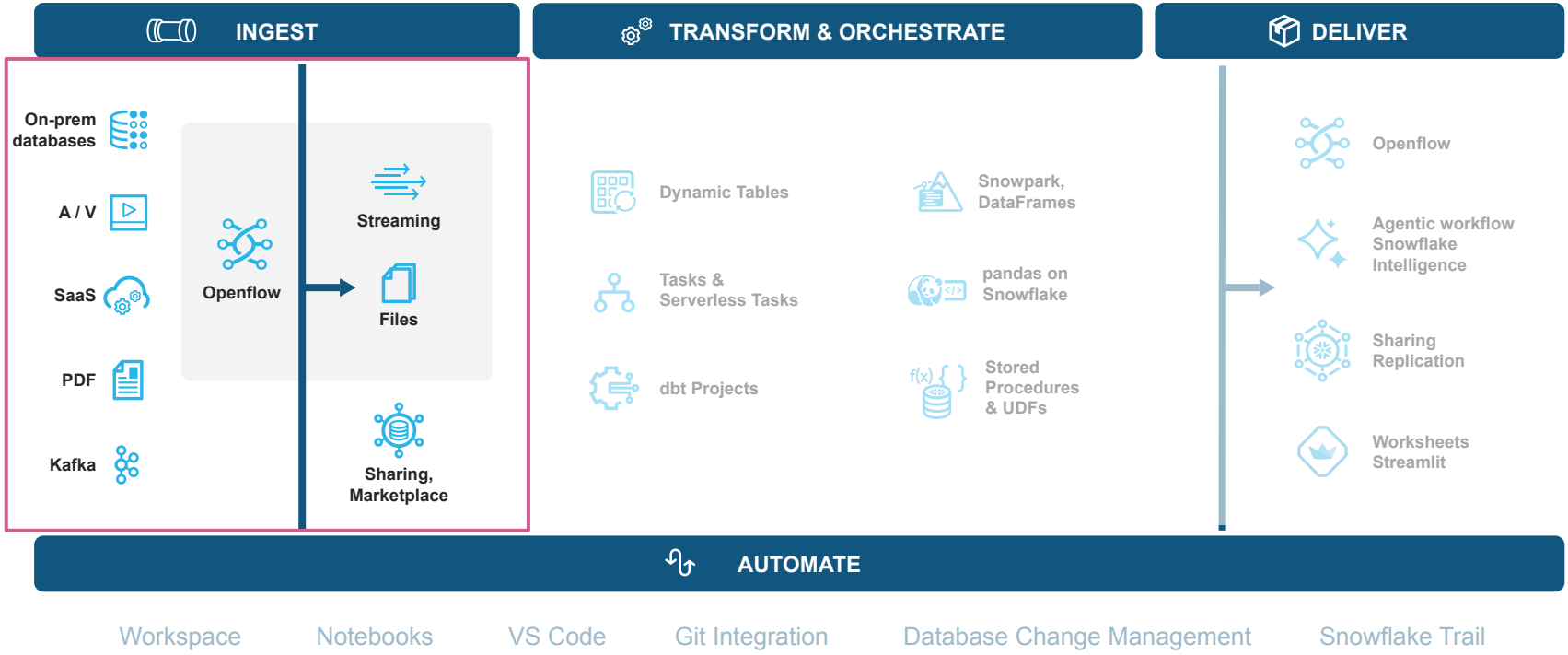


# Snowflake for Data Engineering



# Deep Dive

# Snowflake for Data Engineering



In Dev | Private** | Public | GA*

# Snowflake Openflow

## WHAT IS IT

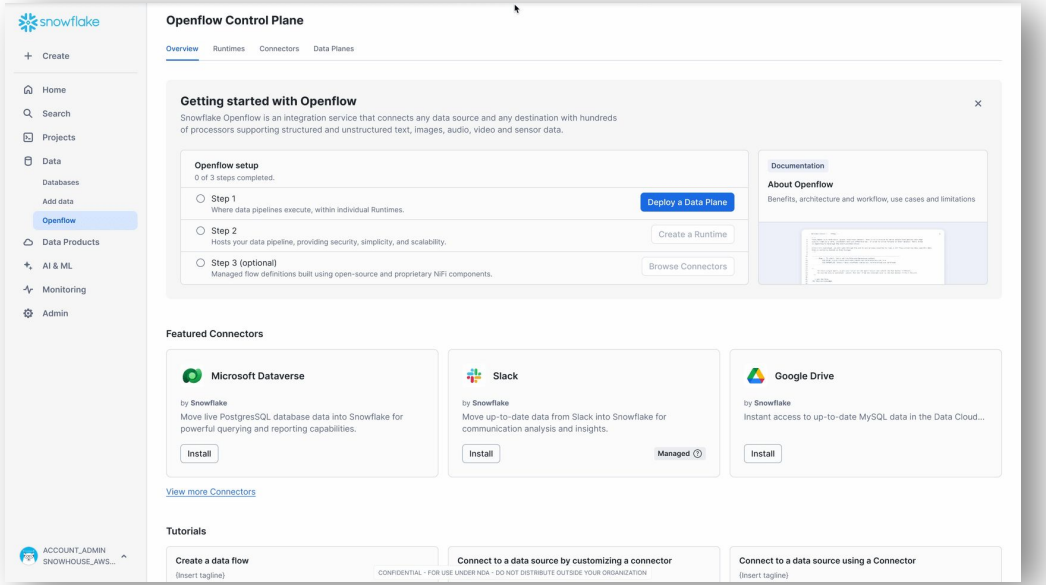
An open, extensible, managed, multi-modal data integration service that makes data movement effortless between data sources and destinations, supporting all data types including structured and unstructured, batch and streaming.

## WHY USE IT

Move data effortlessly and scale with confidence for all your integration needs, in one single platform.

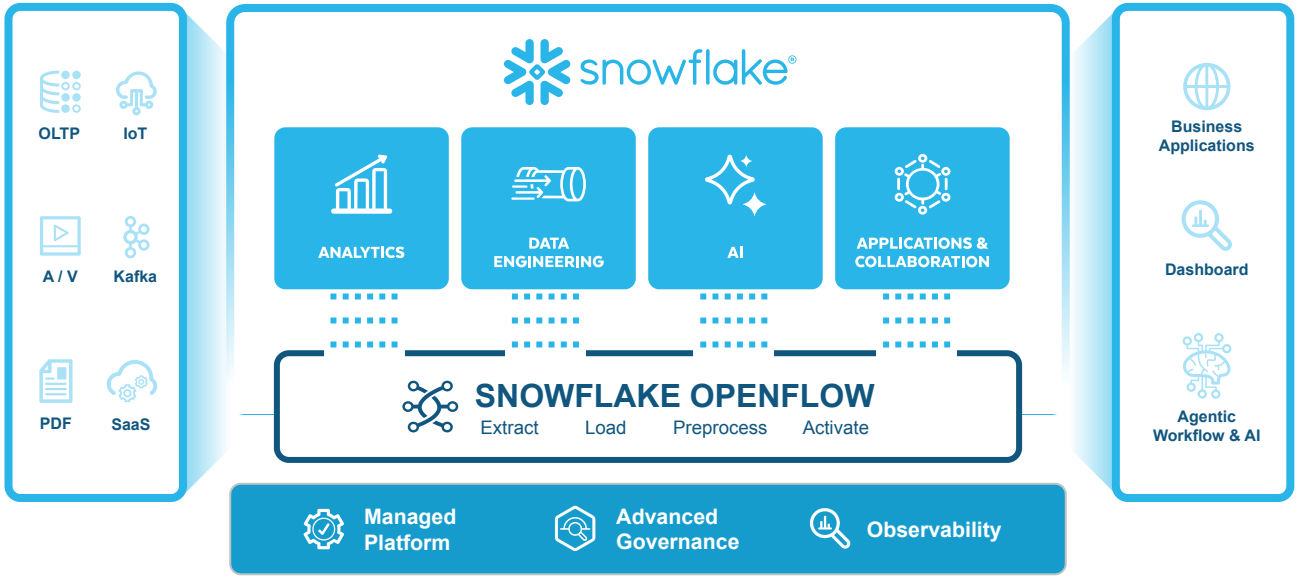
## HOW TO USE IT

Access Openflow via Snowsight, then launch Openflow runtime. Customers can choose where to deploy their runtimes via BYOC or SPCS, and build, manage, and observe data integration through the unified UI.




GA* - BYOC AWS
PrPr** - SPCS AWS & Azure

# Snowflake Openflow



# Key Capabilities with Openflow

## Low Code Visual Interface

Intuitive visual interface to build connectivity and integration effortlessly

## Turnkey Connectivity & Interoperability

Over 30 turnkey connectors and 200+ processors, ready to build for any connectivity and architecture

## Streaming Data Flows

High throughput, low latency streaming data in near-real time, powered by Snowpipe Streaming

## Deployment Flexibility

Flexible deployment options to meet where you are, no need to sacrifice simplicity

## Easy To Govern & Secure

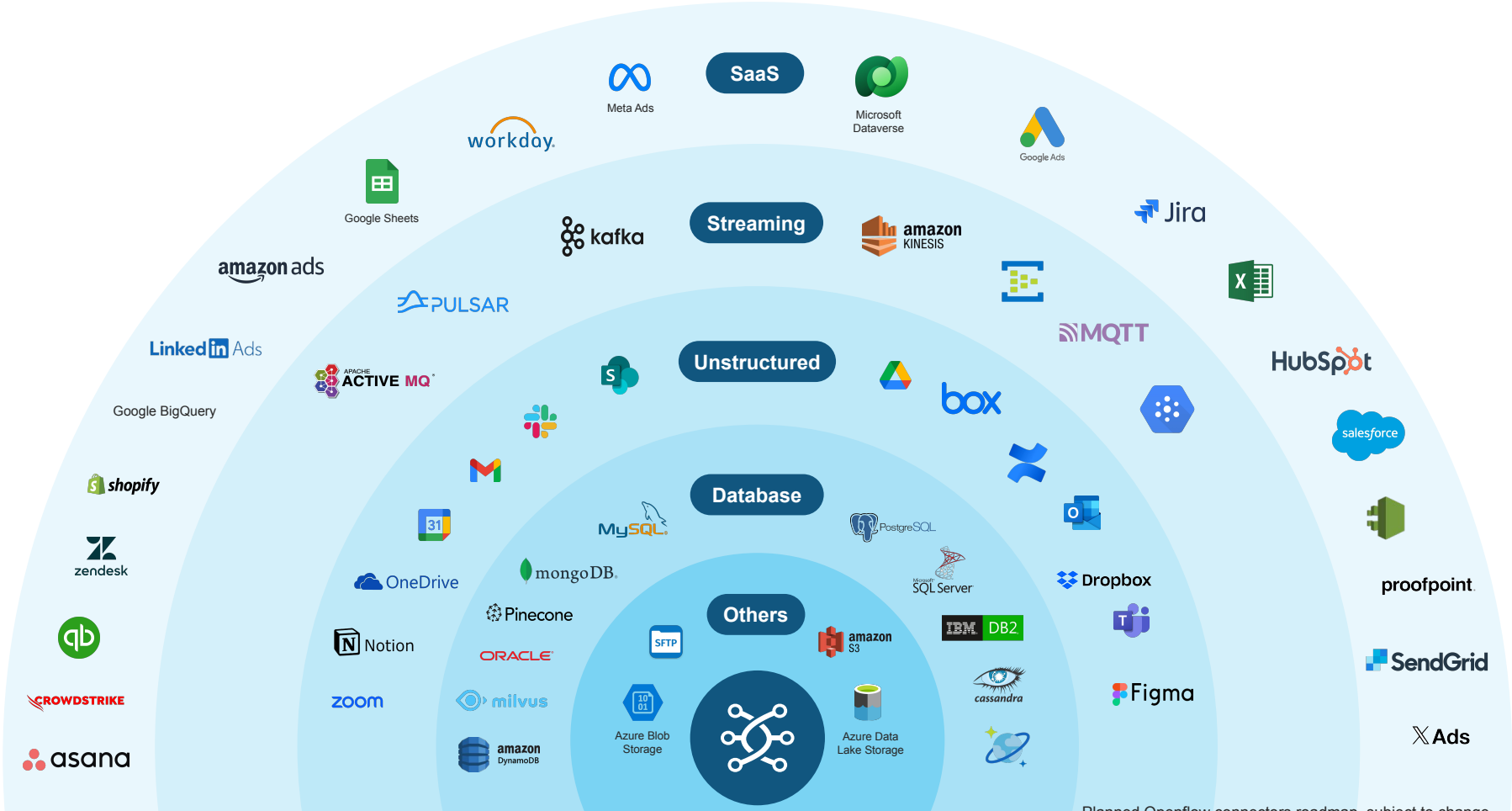
Proven security with advanced authentication, authorization, and data in-transit. Bring Snowflake governance to integration pipelines.

## Real-Time Monitoring & Alerts

Deeply observable pipelines with DAG visualization, status monitoring, refresh history information

# Openflow: Connectors


Openflow Connectors in GA and PuPr as of June 3rd, 2025




Planned Openflow connectors roadmap, subject to change

# Snowpipe Streaming

**WHAT IS IT**

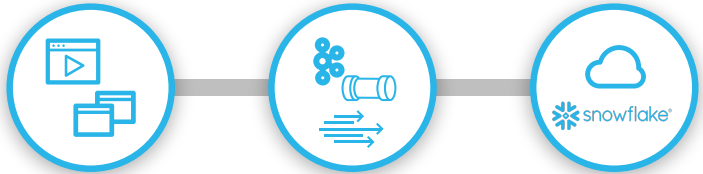
Snowpipe Streaming provides a serverless near-real time ingestion service into Snowflake. The new architecture enables up to **10GB/s per table with 5-10 seconds** ingest-to-query latency.

**WHY USE IT**

- High-throughput ingestion with higher throughput and lower latency, powered by a new Rust-based SDK and ingestion architecture.
- New REST interface for lightweight direct streaming into Snowflake
- Transparent throughput-based pricing model

**HOW TO USE IT**

- Streaming integrations through Snowflake Openflow, Snowflake connector for Kafka, and other partners
- Java SDK or REST based ingestion




<10 second E2E
10 GB/s+ per Table

REST API

Java

# Snowpipe

Simpler, More Predictable pricing for Snowflake’s ingestion services

## WHAT IS IT

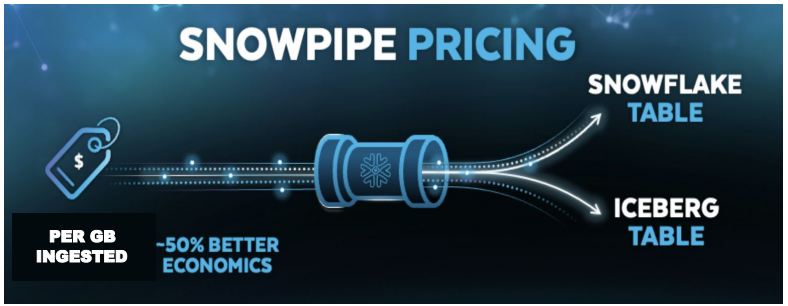
Instead of a per-second/per-core compute charge and a per-1,000-files fee, you are now charged a fixed credit amount per gigabyte (0.0037 credits per GB[1]) of data ingested with Snowpipe. This pricing makes it easier for you to estimate your Snowpipe costs.

## WHY USE IT

Benefit from a simpler, more predictable Snowpipe pricing model that significantly lowers your Snowflake ingestion costs for most types of workloads - a direct result of significant optimizations we've made to our ingestion infrastructure. This also allows you to eliminate complex pre-processing or workarounds like file batching if done for cost considerations, so your engineers can focus on driving business value.

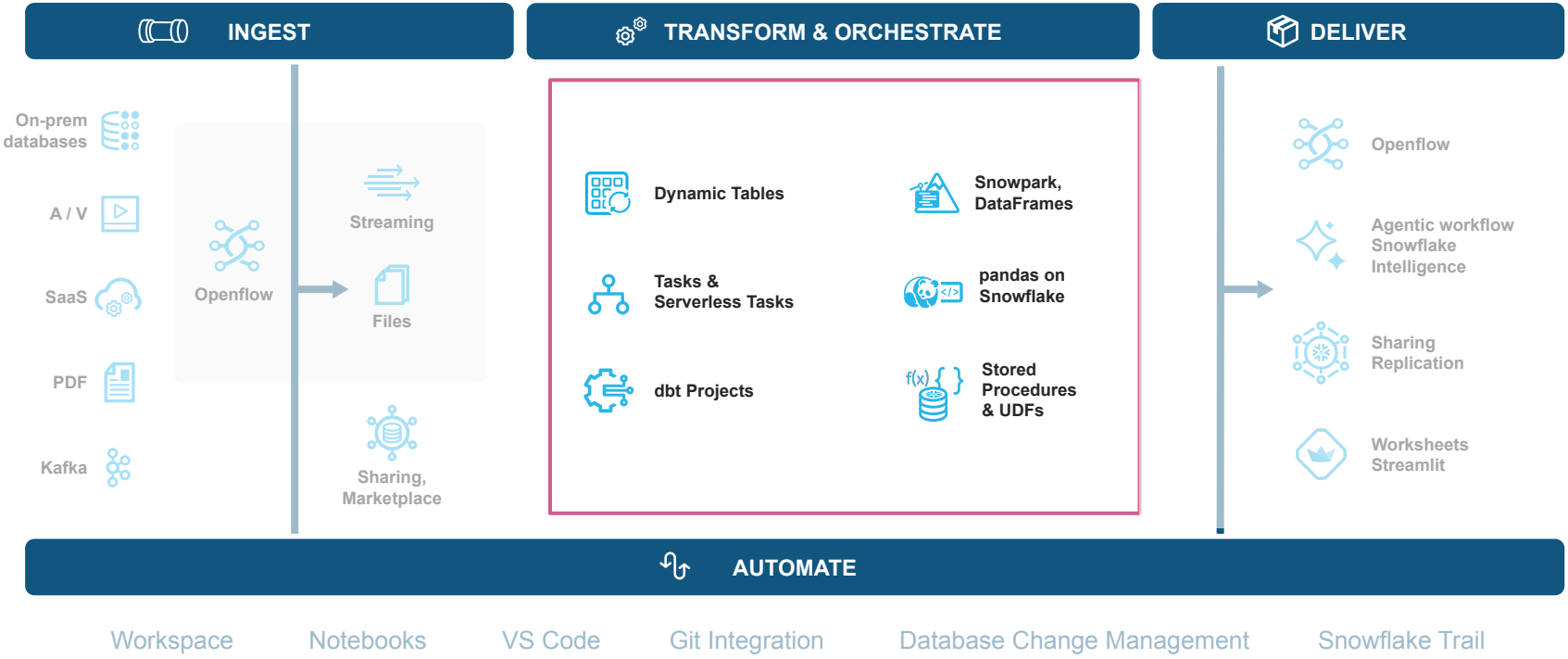
## HOW TO USE IT

For BCE/VPS customer, this update has been applied automatically to your account on Aug 1, 2025. For customers on Enterprise and Standard editions, this update was rolled out on Dec 8th, 2025.




[1] For text files (like CSV, JSON, XML), you will be charged based on their uncompressed size. For binary files (like Parquet, Avro, ORC), you will be charged based on their observed size regardless of compression.

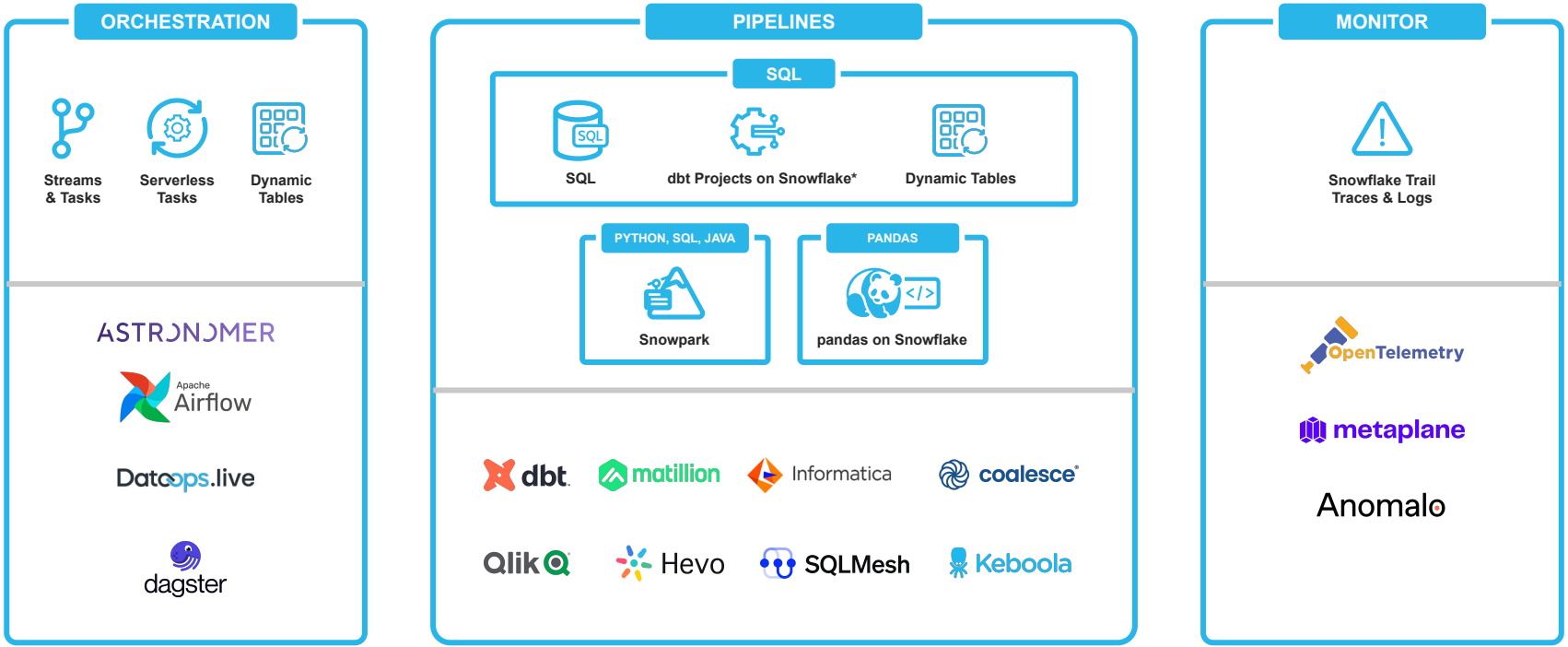
# Snowflake for Data Engineering



In Dev | Private | Public* SOON | GA

# Build flexible Pipelines

## Combining Snowflake's Power with Partner and Open Source Tools



In Dev | Private | **Public** | GA

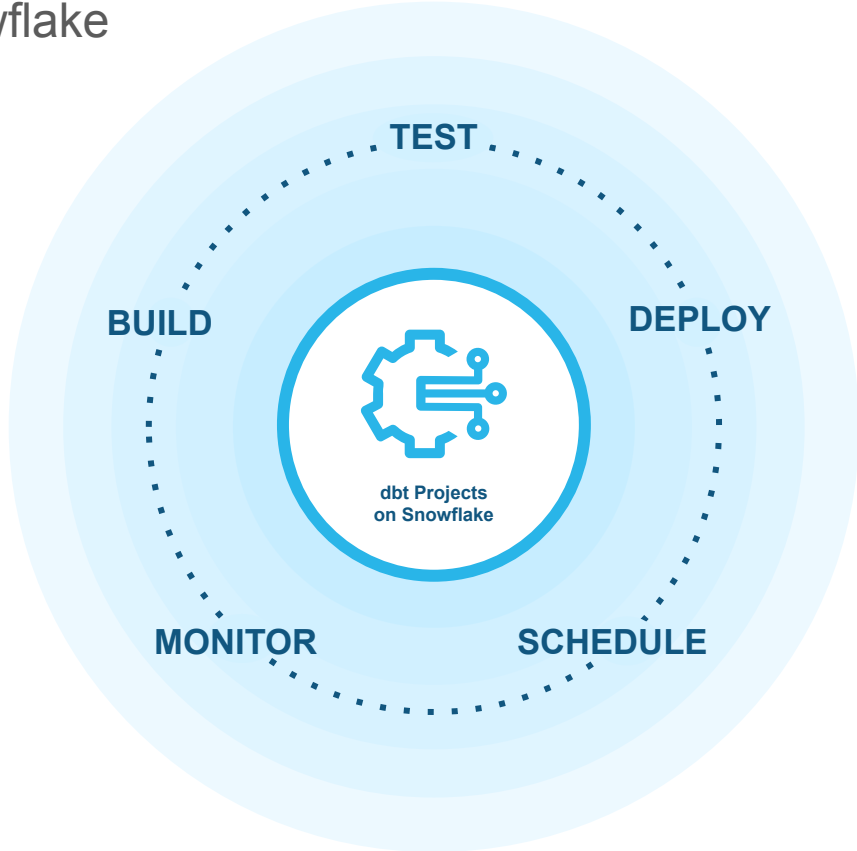
SOON

# dbt Projects on Snowflake

## Run and govern dbt natively on Snowflake

**Consolidate systems to reduce administration and improve debugging**

**Enable new teams to build and deploy pipelines**



# Self-Service Transformations within Snowflake

## Run dbt natively on Snowflake

**Intuitive**

**Fast**

**Reliable**

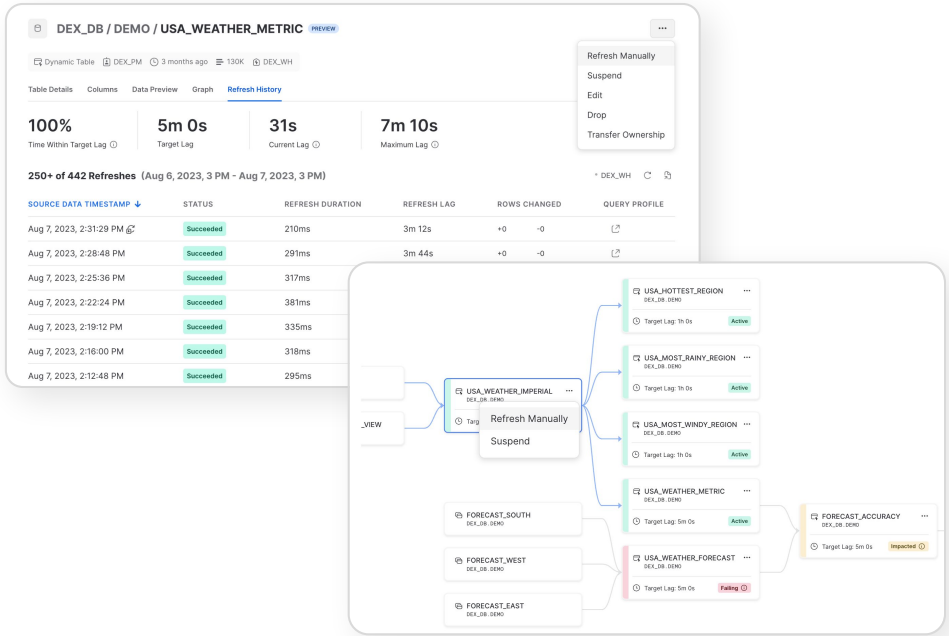
> Everything I'm seeing here is massive leaps ahead. I'm really liking what I'm seeing … You've got the **availability of the whole pipeline running**. You've got the availability of different tasks. And, from the looks of it, you've got the ability to run dbt tests … I think **this is awesome.**

**Chris Hastie**
InterWorks – Data Superhero

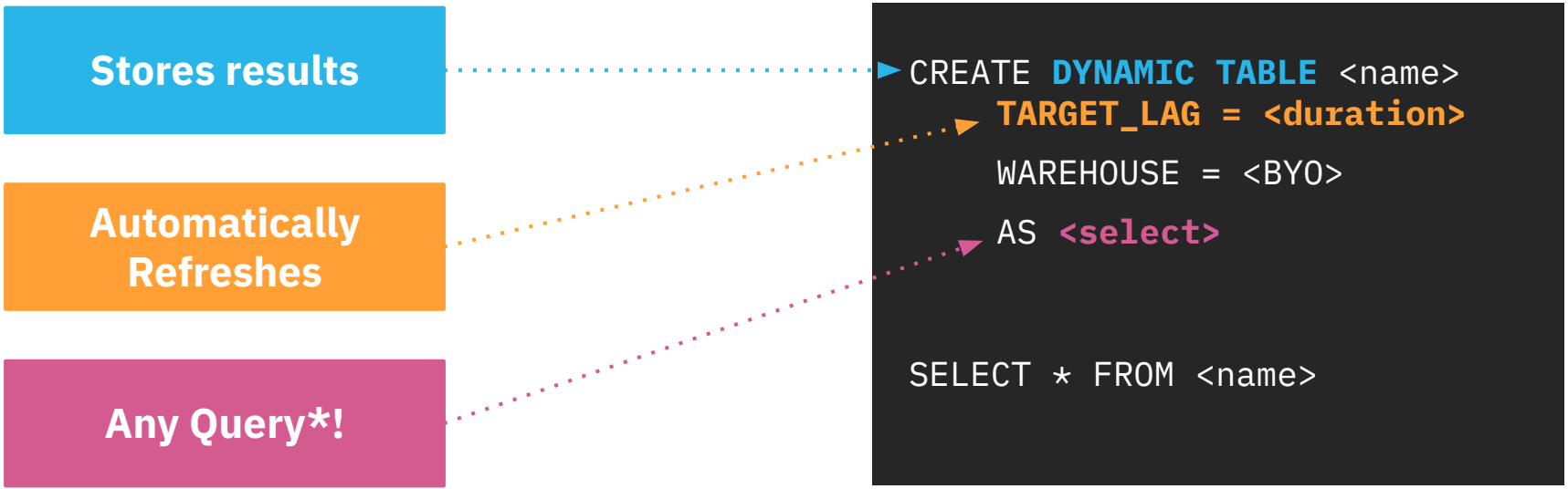
# Dynamic Tables

## Declarative ETL framework for batch and streaming pipelines

- Simple declarative ETL
- Automatic orchestration with configurable freshness
- Continuous Incremental processing
- Fully observable with DAG visualization, real time alerts, and data quality



# Dynamic Tables: Simple declarative interface






*See limitations documentation

# Dynamic Tables: Query like a table

**Immediate results**

**Freshness within LAG**

**Snapshot isolation**

```
CREATE DYNAMIC TABLE <name>
    TARGET_LAG = <duration>
    WAREHOUSE = <BYO>
    AS <select>

SELECT * FROM <name>
```

Consistently fast to query!

# Dynamic Tables: Key Highlights

## DECLARATIVE DATA PIPELINES

Continuous data pipelines as easy as SELECT. Complex pipelines with hundreds of branches. Dynamic Tables manage the scheduling and orchestration

## LOW LATENCY FRESHNESS

Controlled by a target lag for each table or full pipeline. Data freshness as low as 1 minute, single-digit seconds in the near future


## SQL SUPPORT (PYTHON SOON)

Use any core SQL syntax to define transformations or snowpark data frames and UDFs. Python and REST API coming soon

## AUTOMATIC EFFICIENT REFRESHES

Automatic query analysis with refresh mode recommendations. Detailed performance guide for tinkerers.

## PIPELINE CONTROLS

Ability to suspend refreshes for quiet periods, manually refresh a single DT or full DAG, enforce refresh mode and more


## OBSERVABILITY

Deeply observable pipelines with DAG visualization, status monitoring, refresh history information

# Data Processing: Challenges at Scale

Data is often stored and processed in separate environments

## HIGHER COSTS, LOWER PERFORMANCE

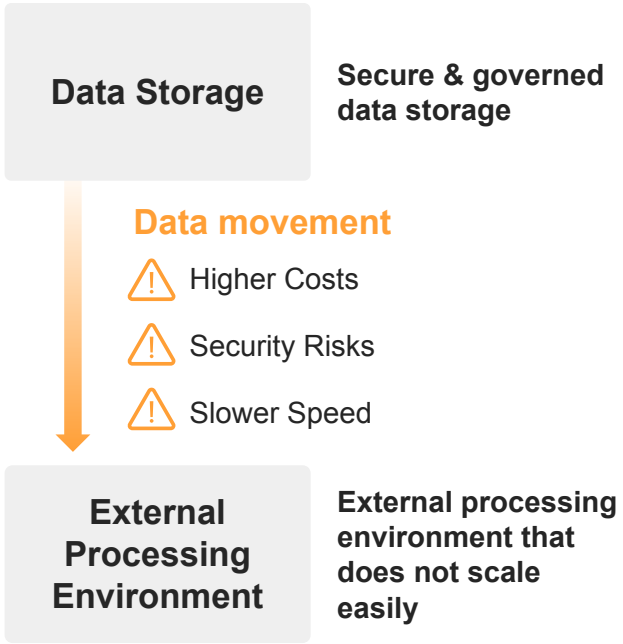
Data movement from where it's stored to a separate processing environment adds data transfer costs and latency

## MANUAL & COMPLEX TUNING

Legacy processing environments often require manual management and cluster tuning by experts, which is costly to maintain and difficult to scale

## SECURITY & GOVERNANCE

There are security and governance risks as data is moved across environments



1. **Jumpstart development**
2. **Transform data in Python or other languages**
3. **Process with security & scalability**

**Work in your favorite environment**

Pre-configured and hosted Snowflake Notebooks

Seamless integration with popular third-party IDEs

**Snowpark DataFrames**
DataFrame-style programming for querying and transforming data

```
df.filter(df.state == 'WA')
```

**pandas on Snowflake**
Familiar pandas API for flexible data transformation and analytics

```
df[df['Date'].dt.year>=2025]
```

**User-Defined Functions & Stored Procedures**
Write and operationalize custom code

```
@udf def detect_fraud()
```

**Code Execution**

**Elastic Compute Runtime with Snowflake Virtual Warehouse**

**Install & Manage Packages* from PyPI**

python Package Index

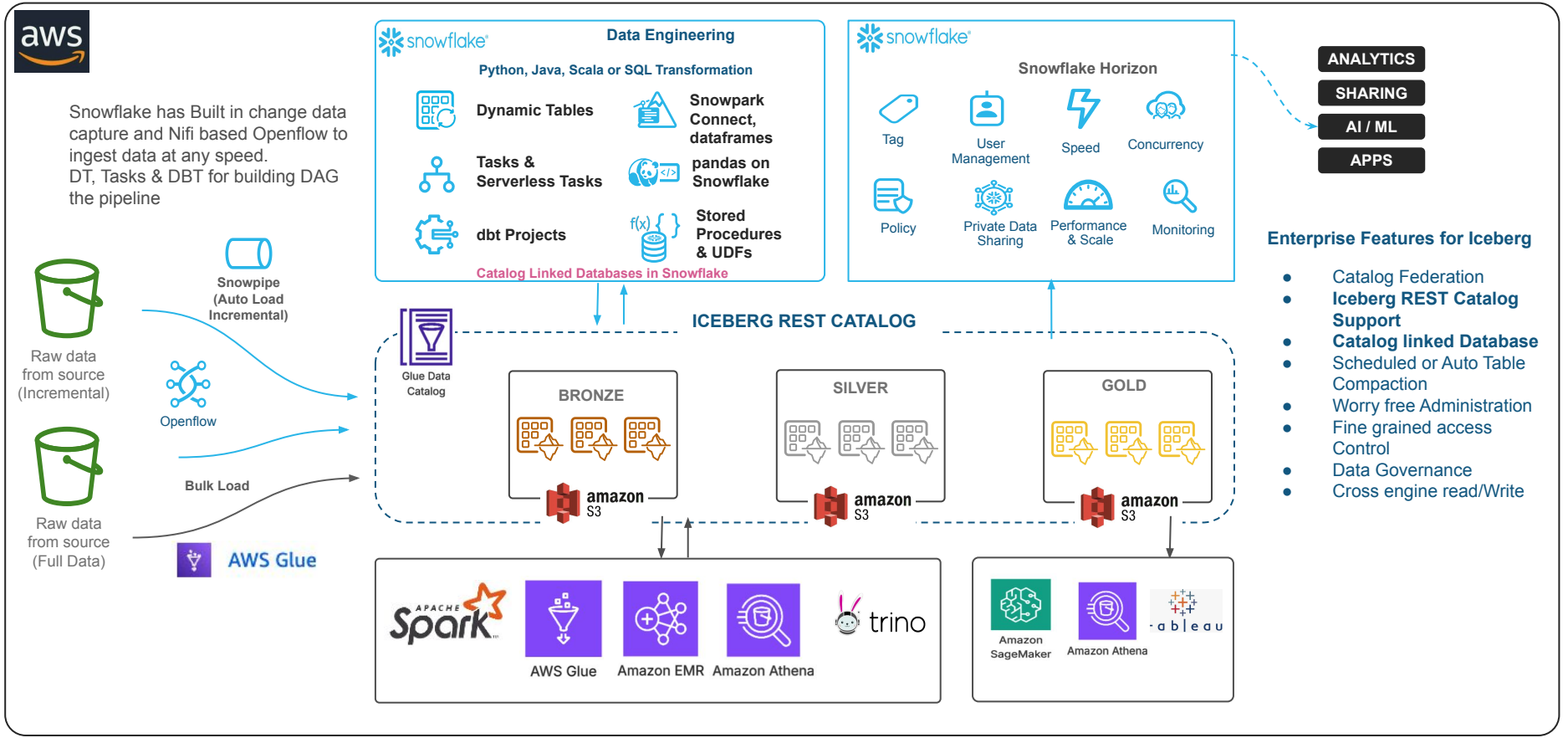
**Built-in Anaconda Python packages** and thousands more

scikit-learn pandas Keras

**Sandbox**

- Python
- Java
- Scala

# Build a Connected & Governed Lakehouse



# Ease of Use Meets Powerful Vectorized Engine

## Customers on average are seeing

**5.6x**

**Faster Average Performance Over Managed Spark[1]**

**41%**

**Cost Savings with Snowpark Over Managed Spark[1]**

1. Based on customer production use cases and proof-of-concept exercises comparing the speed and cost for Snowpark versus managed Spark services between November 2022 and May 2025. All findings summarize actual customer outcomes with real data and do not represent fabricated datasets used for benchmarks.



CTC
CHICAGO TRADING COMPANY

> Now with fewer ephemeral failures and higher visibility in Snowflake, we have a platform that's much easier and cost-effective to operate than managed Spark.

**David Trumbell**
Head of Data Engineering and Principal Engineer, CTC

**54%**

Cost savings – amounting to millions of dollars annually – by moving from managed Spark to Snowflake

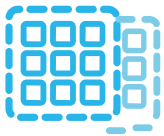
**$800K**

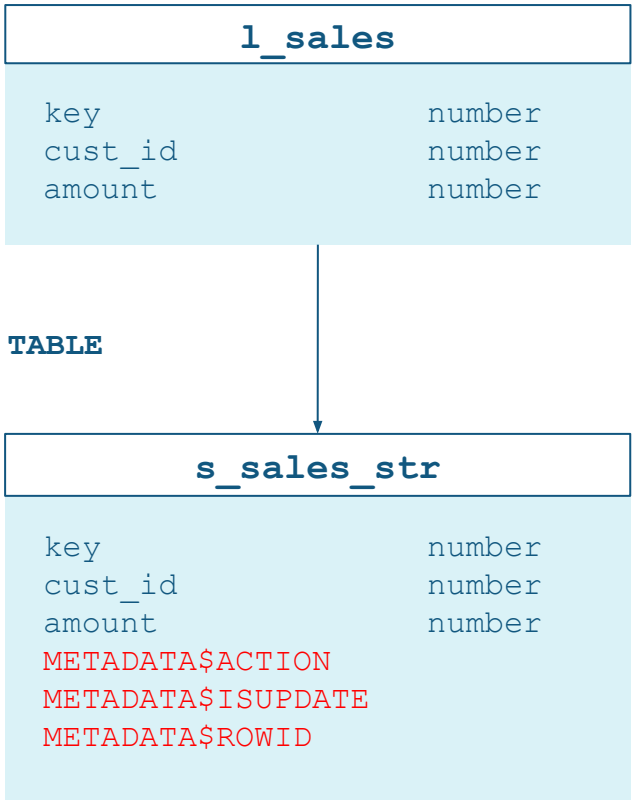
Saved annually by eliminating data movement out of Snowflake and back

# ETL Pipelines & Orchestration

# Streams

```
CREATE STREAM
s_sales_str ON TABLE
l_sales;
```






Adding a stream to a table appends three metadata columns that can be queried

These columns track the CDC records and their type: appends, deletes, or both (updates = inserts + deletes)

Little additional storage is required, as the stream is a logical pointer to the table's existing Time Travel micro-partitions

# Tasks

Build, Plan and Orchestrate

```
create or replace task t_sales_task
    warehouse = xsmall_vwh
    schedule = '1 minute' as
    insert into f_sales (
    select key, cust_id, amount
    from s_sales_str
    where metadata$action = 'INSERT');
```

# Actions

## Stream:

Identifies Inserts, Updates, Deletes

## Task:

Executes on a

- **Schedule** (CRON)
- **Interval** (Minutes/seconds)
- **Predecessor** – child task runs after a parent task completes; can daisy chain

Single SQL statement or a stored procedure call

Can process a stream

Check for one or more streams to be populated before executing

In Dev | Private** | Public* | GA

# Supporting Capabilities

## Ingestion

- Snowpipe Streaming & Kafka for rowset streaming
- File Auto-Ingest
- Snowflake native connectors
- In-place data sharing across regions
- Schema Detection / Evolution

## Transformation

- Snowpark DataFrame & pandas APIs
- Dynamic Tables with incremental refreshes
- Stored Procedures

## Orchestration

- Streams and Tasks
- Serverless Tasks
- Serverless Tasks Flex**
- Triggered Tasks
- Tasks Backfill
- Open-source

## Developer Experiences & DevOps

Notebook | VS Code | Python API | Snowflake CLI | Git Integration | Database Change Management* | Snowflake Trail

## Interoperability with Snowflake Tables and Iceberg Tables

# Snowflake Workspaces

## Your one-stop shop for code editing and collaboration

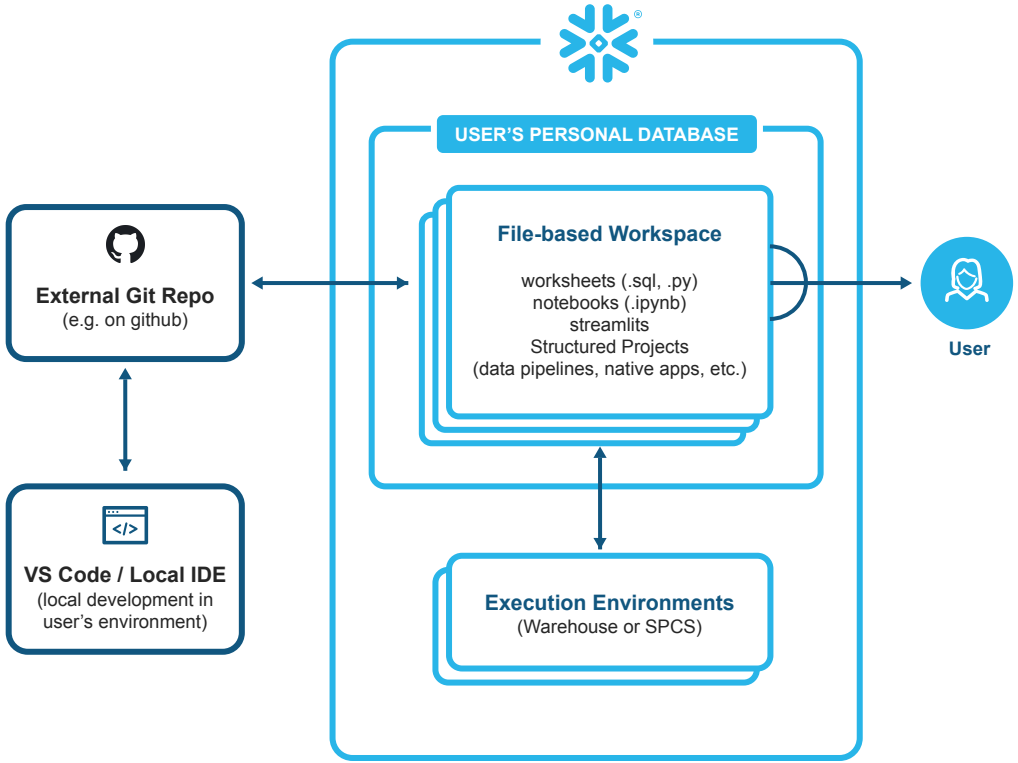
**One editor for all**
SQL, notebooks, python, pipeline projects all in one editor

**File based development**
All your code organized in files and folders

**Private by default**
Leveraging personal databases for private development

**Git integration**
Work directly with your git repos

**Collaborate with others**
Collaborate safely in shared workspaces



# Snowflake Notebooks

## WHAT IS IT

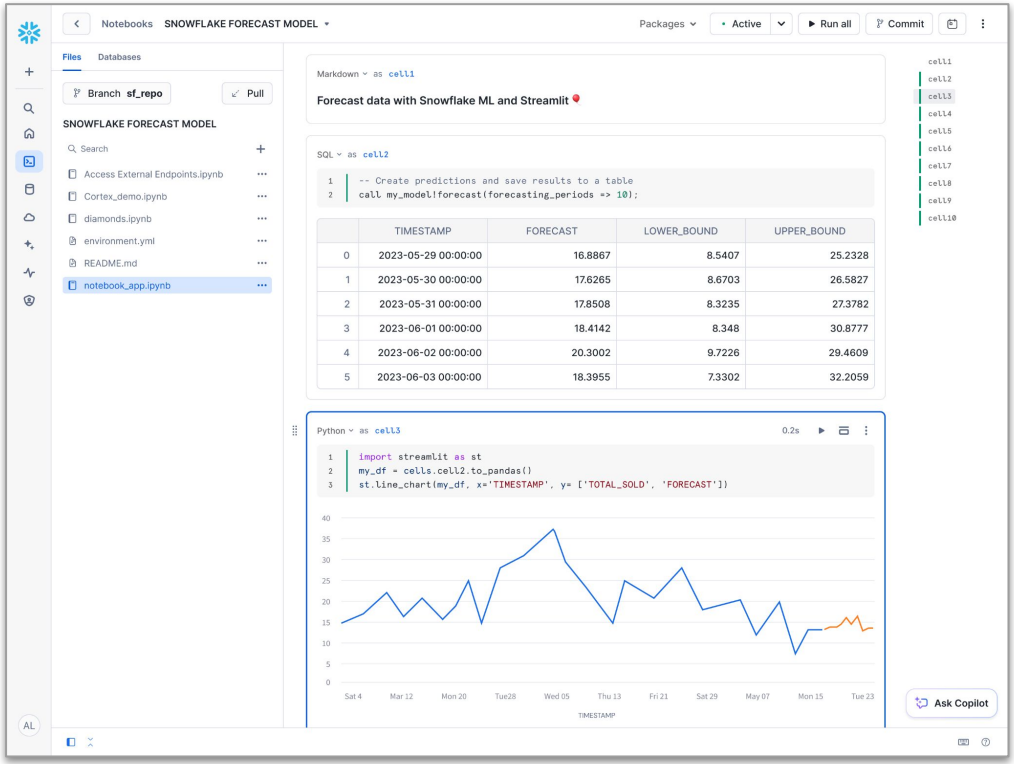
A familiar, easy-to-use notebook interface for your data work that seamlessly blends Python, SQL, and Markdown as well as integrations with key Snowflake offerings like Snowpark ML, Streamlit, Cortex and Iceberg

## WHY USE IT

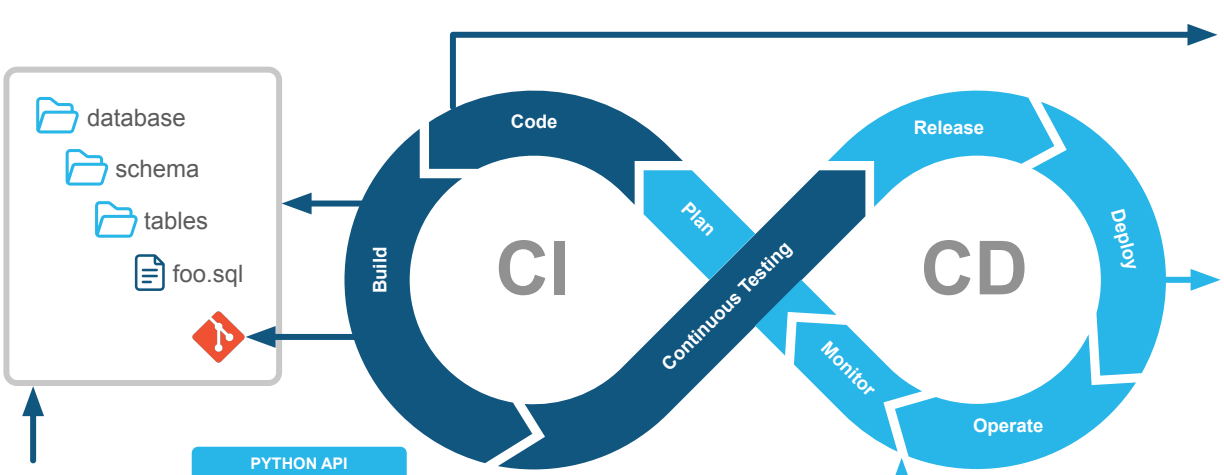
Simplify the process of connecting to your data and to streamline your data engineering, analytics, and machine learning workflows

## HOW TO USE IT

- Create a notebook directly in Snowsight or from an existing project
- Sync with your remote Git repo for version control and collaboration
- Schedule notebooks to run automated pipelines



# Automated & Declarative CI/CD Pipelines




**Monitor & Deploy**
I can automatically or manually trigger a CI/CD pipeline whenever changes are pushed to my remote Git repository.

**Deploy & Release**
I can automate Snowflake deployments by integrating my remote Git repo with the **CLI** to trigger a CD pipeline.

**Build & Code**
As a data engineer, I need to package up my code and define my project configuration.

PYTHON API
GA soon

**Monitor & Operate**
I can expect that **Database Change Management** will automatically update my database objects.

DCM: CREATE OR ALTER
PuPr

DCM: EXECUTE IMMEDIATE FROM
PuPr

In Dev Private Public GA

# Git Integration & Snowflake CLI

## WHAT IS IT

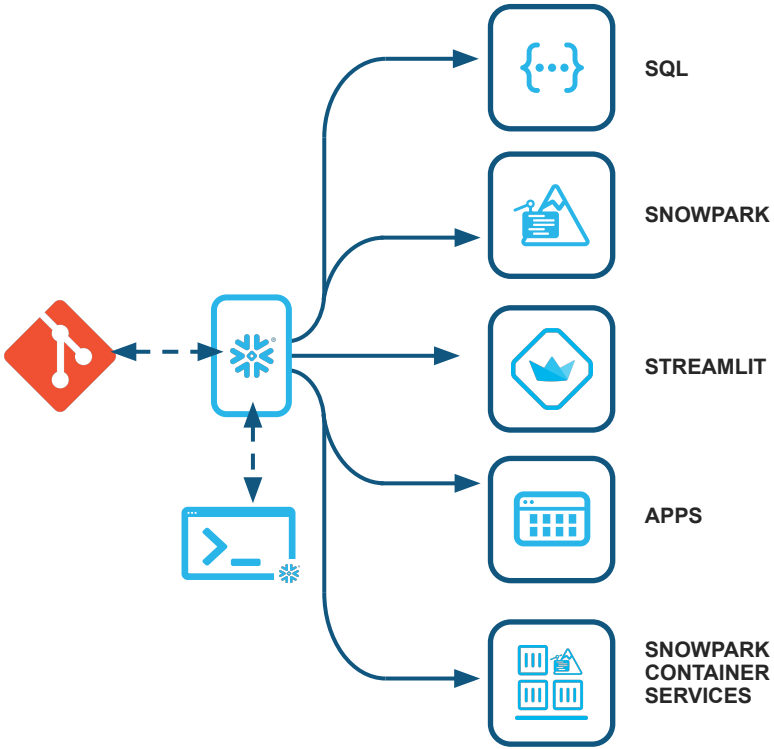
View, run, edit, and collaborate within Snowflake experiences with assets that exist in a git repo.
Enables simple commands to create, manage, update, and view apps running on Snowflake across app-centric workloads.

## WHY USE IT

Build collaboratively and deploy faster with seamless CI/CD.

## HOW TO USE IT

Create, manage, update, and view apps running on Snowflake with simple commands. Integrate with source control to allow for collaboration, source code integration, and maintaining a single source of truth for team artifacts.



# Observability with Snowflake Trail

## WHAT IS IT

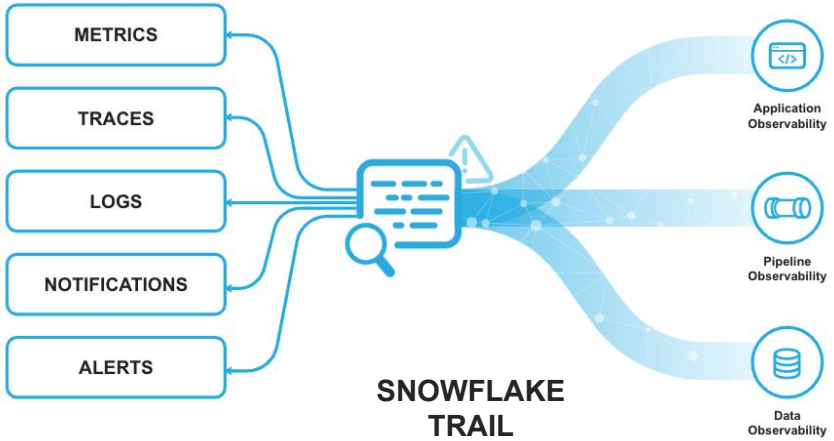
A set of Snowflake capabilities for developers to better monitor, troubleshoot, debug, and take actions on pipelines, apps, user code and compute utilizations

## WHY USE IT

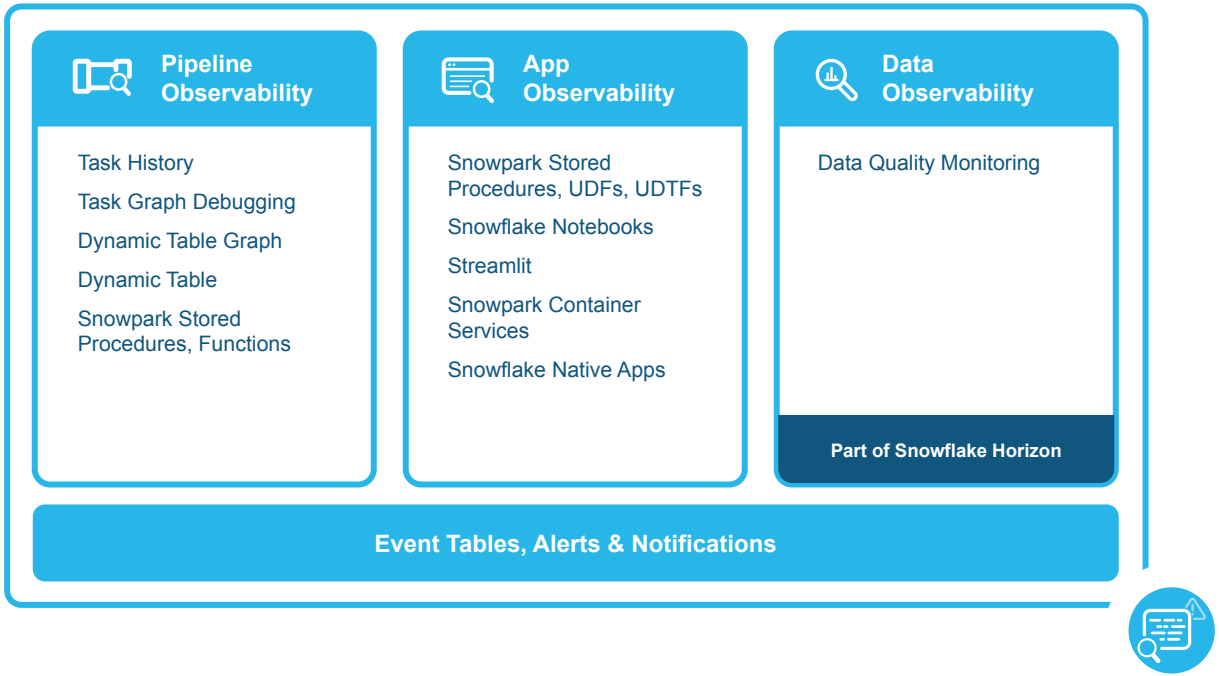
Developers can visualize and monitor their code better, be able to diagnose and troubleshoot knowing where exactly the error lays, and take actions faster.

## HOW TO USE IT

Use Snowsight directly, or just Bring Your Own Tool as Snowflake Trail is built with OpenTelemetry standard that offers easy integration with the observability ecosystem.



# Snowflake Trail for Pipelines, Apps, and Data



In Dev | Private** | Public* | GA

# Supporting Capabilities

## Ingestion



Snowpipe Streaming & Kafka for rowset streaming



File Auto-Ingest



Snowflake native connectors



In-place data sharing across regions



Schema Detection / Evolution

## Transformation



Snowpark DataFrame & pandas APIs



Dynamic Tables with incremental refreshes




Stored Procedures

## Orchestration



Streams and Tasks
Serverless Tasks
Serverless Tasks Flex**
Triggered Tasks
Tasks Backfill



Open-source

## Developer Experiences & DevOps

Notebook | VS Code | Python API | Snowflake CLI | Git Integration | Database Change Management* | Snowflake Trail

## Interoperability with Snowflake Tables and Iceberg Tables

In Dev Private Public GA

# Iceberg Tables

Table type that brings the choice of Apache Iceberg to the Snowflake platform

## WHAT IS IT

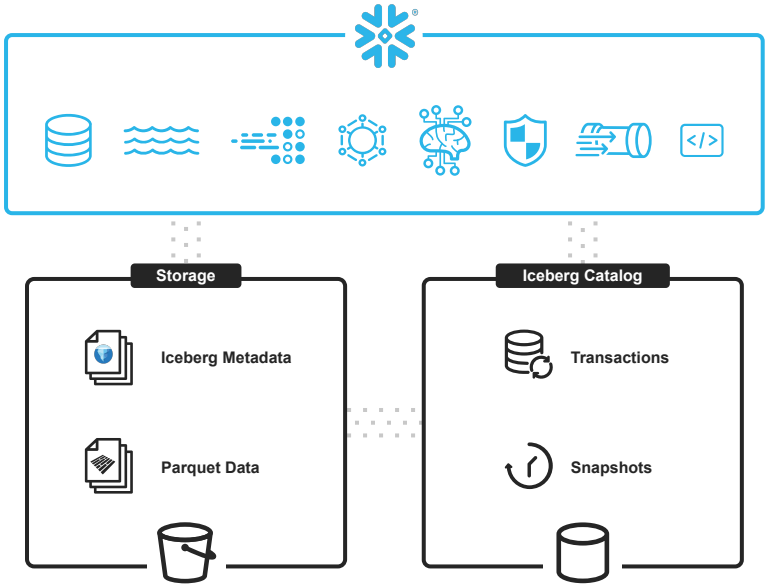
Table type that stores data externally in an open format.

## WHY USE IT

- Extends the Data Cloud as the single pane of glass for using and managing data, even if stored externally
- Support data architectures requiring interoperability while retaining the easy management and great performance of Snowflake

## HOW TO USE IT

- Set up integration with bucket, maybe catalog too
- Create Iceberg Table from Snowflake
- Query and/or modify tables via SQL, Snowpark, connectors, etc.



In Dev Private Public GA

SOON

# Catalog Linked Databases

## Centralize and activate data from nearly anywhere

**WHAT IS IT**

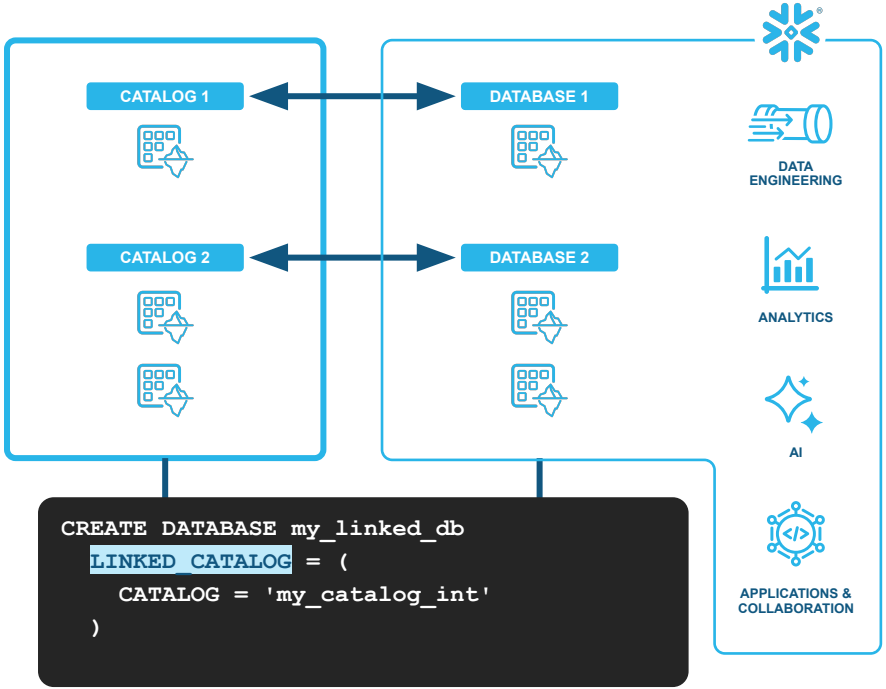
A Snowflake database with bi-directional connection to your Iceberg tables via Open Catalog or a remote Iceberg REST catalog.

**WHY USE IT**

Discover and activate data from nearly anywhere. Extend Snowflake's performance, ease of use, and reliability to your Iceberg ecosystem for simplified data management across your lakehouse.

**HOW TO USE IT**

Create a new type of database in Snowflake from any Iceberg REST API catalog integration.




Supports External Volume or Vended Credentials

In Dev | Private | **Public** | GA

SOON

# Write to Externally Managed Iceberg Tables

## Extend Snowflake to any Iceberg REST API Catalog

**WHAT IS IT**

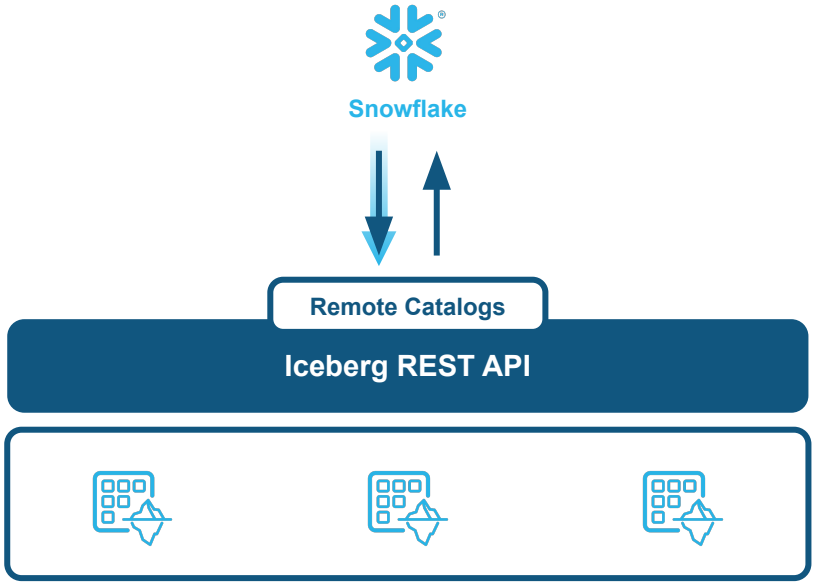
Full DDL and DML capabilities on Externally Managed Iceberg tables across any REST API Catalog

**WHY USE IT**

Get more value from your Iceberg tables. Effortlessly connect, transform, and optimize Iceberg tables with Snowflake's reliable performance. Built in security and governance, powered by Horizon Catalog, keep your data secure.

**HOW TO USE IT**

Integrate a Iceberg REST Catalog to Snowflake, browse namespaces and tables, and activate your data.



# Thank you!