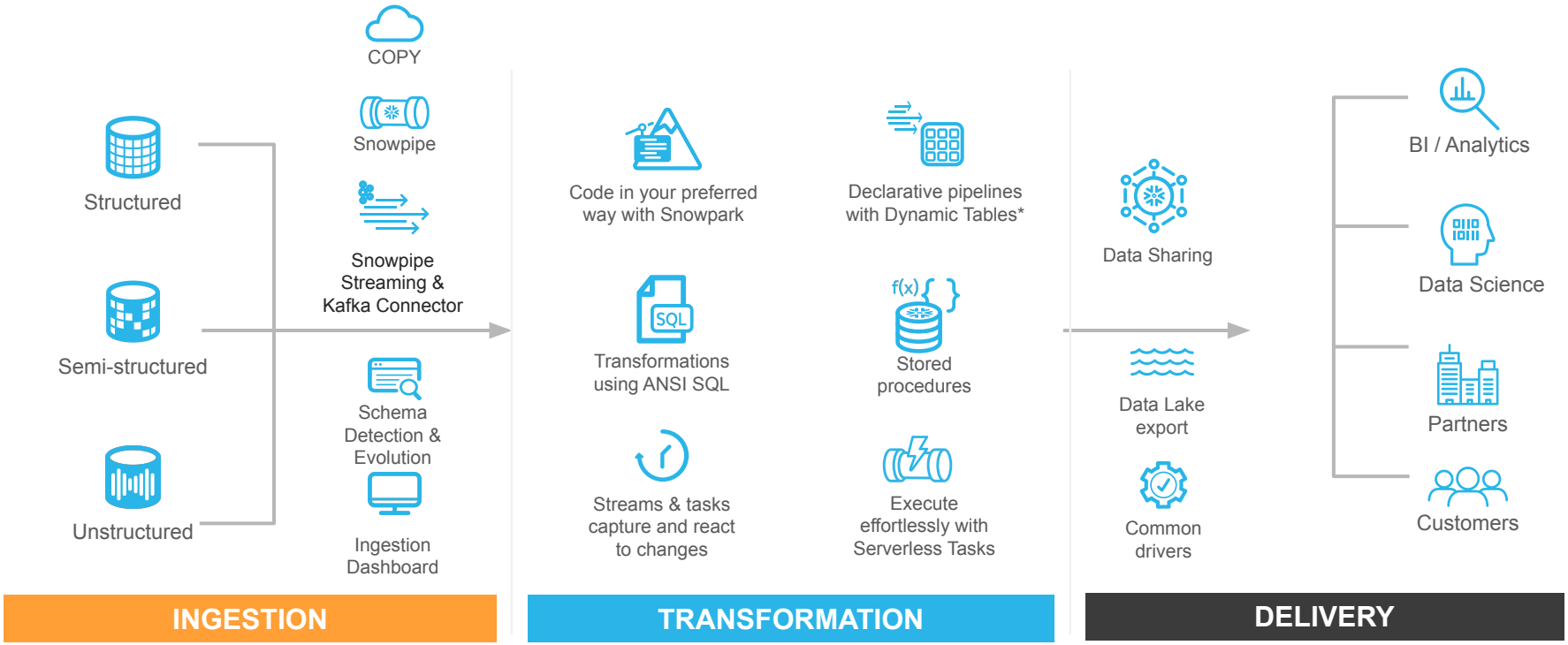
SNOWFLAKE DATA INGESTION BEST PRACTICES

Antti Sirkka - Data Cloud Architect

DATA ENGINEERING WITH SNOWFLAKE



SNOWFLAKE DATA INGESTION

No More Boundaries between Streaming & Batch Pipelines

Simplified Architecture

No more separation of streaming and batch data pipelines

Work with Streaming Data Easily

Serverless ingestion and automated transformation with easy joins

Native to Snowflake's Data Cloud

Expanding ecosystem in the Data Cloud with consistent and strong security & governance

Ingestion Options

Copy

1. Efficient bulk loading of files
2. Control your own warehouse
3. Deterministic latency

Snowpipe

1. Continuous ingestion of files
2. Serverless
3. Median latency ~30s

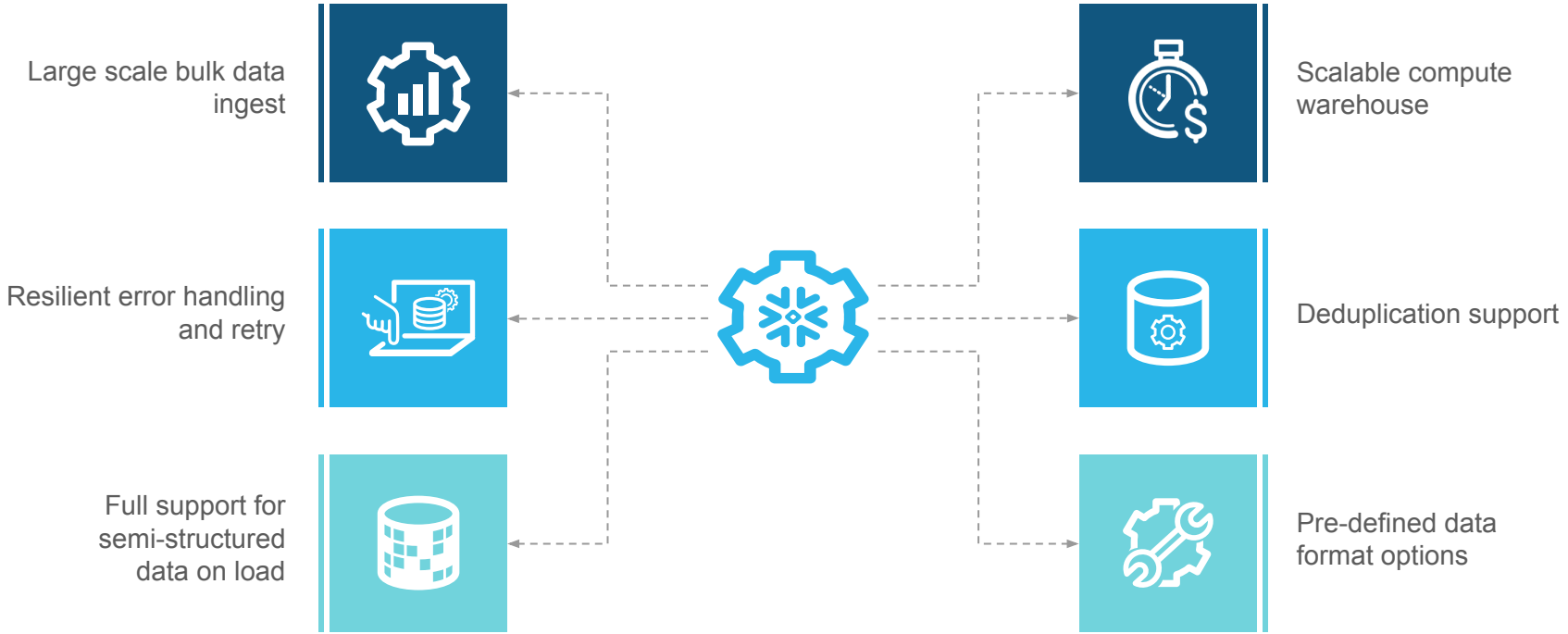
Snowpipe Streaming

1. Near real-time ingestion of rowsets
2. Client application needed
3. < 5s median latency

COPY INTO <TABLE>

SCALABLE RESILIENT BULK FILE LOADING

BENEFITS OF COPY



Copy Best Practices

Do

Leverage object path partitioning for efficient listing and loading

Use the FILES or PATTERN option to specify the exact files when loading ad-hoc

Utilize schema detection for simple table creation and schema evolution for continuous data loading

Take advantage of file format and copy options such as ON_ERROR

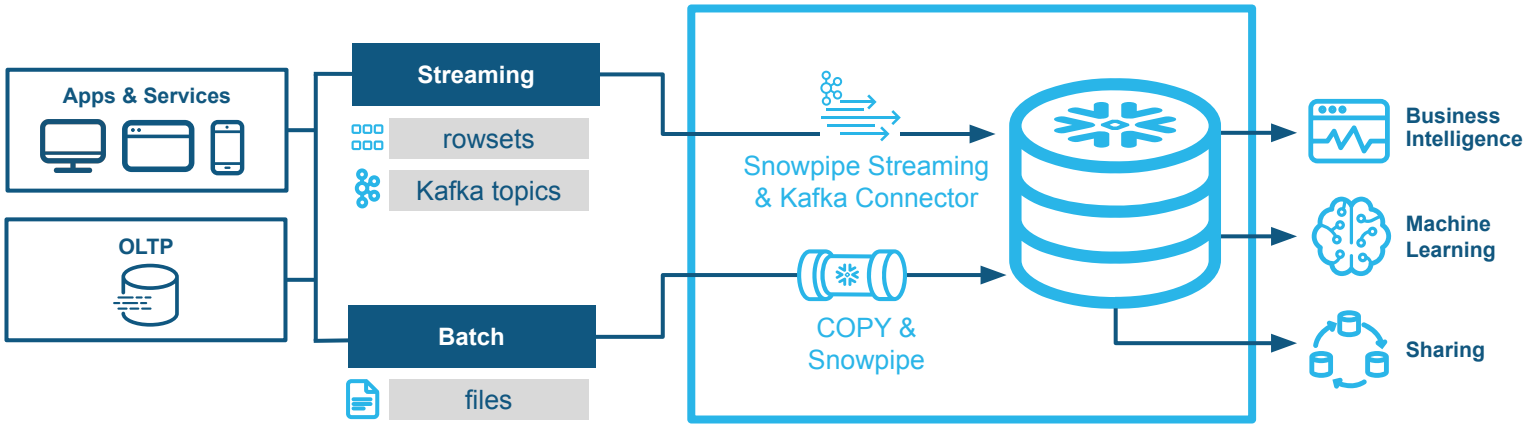
Don’t

Upsized warehouses won’t load single files faster

Don’t over-complicate your large data loads; let Snowflake handle the job management

Don’t split or merge files unless your data source supports it natively; not worth building a separate tool or process

SNOWPIPE: FILES & STREAMING



Snowpipe (file-based)

Designed for batched rowsets as files
Auto-scaled ingestion (10M files/10TB per hr)
Deduplication with file tracking
Concurrent, unordered file ingestion for throughput
Optimized for large-ish files (100MB+ recommended)
Files committed in chunks
~30s median latency for first chunk
Simplest continuous file ingestion with auto-ingest

Snowpipe Streaming (streaming-rows)

For rowsets with variable arrival frequency: insertRows()
Focus on lower latency & cost
Ordered ingestion within a channel
GB/sec throughput to a single table
No files, API in SDK so some source client required
Direct commit of rows to Snowflake tables
<5s typical latency

SNOWPIPE

CONTINUOUS SERVERLESS FILE LOADING

SNOWPIPE

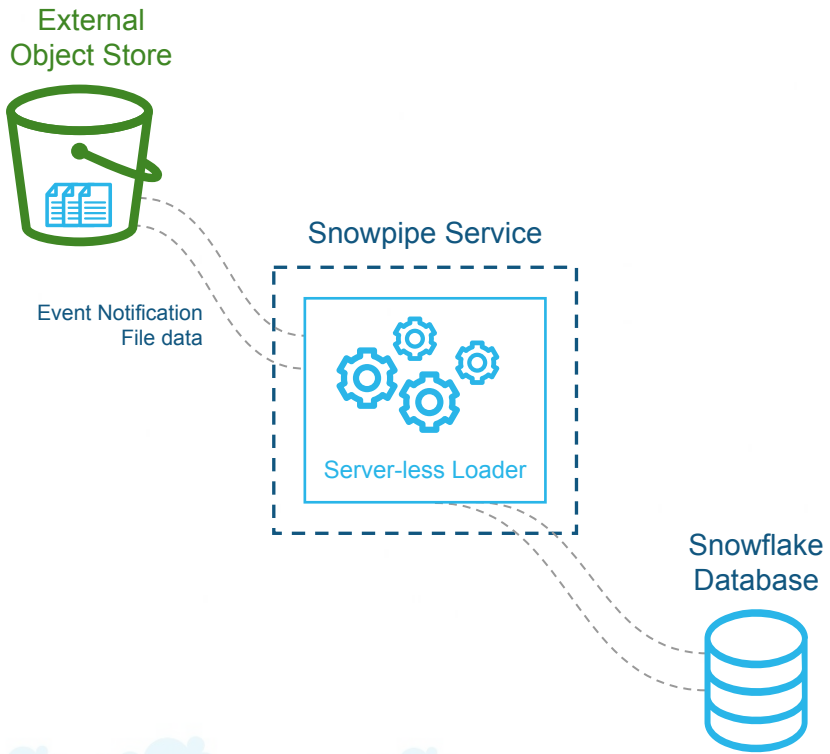
Account

Billing & Usage | Reader Accounts

[redacted] - Billing & Usage

Warehouses	Credits Used	Average Storage Used
8	29.47	47.927 GB

Warehouse Name	Credits Used
LOAD_WH	17.58
PLAYWH	10.20
BI_MEDIUM_WH	1.64
XSMALL	0.03
AUTOMATIC_CLUSTERING	0.01
SNOWPIPE	0.01
CLOUD_SERVICES_ONLY	0.00
DEMO_WH	0.00

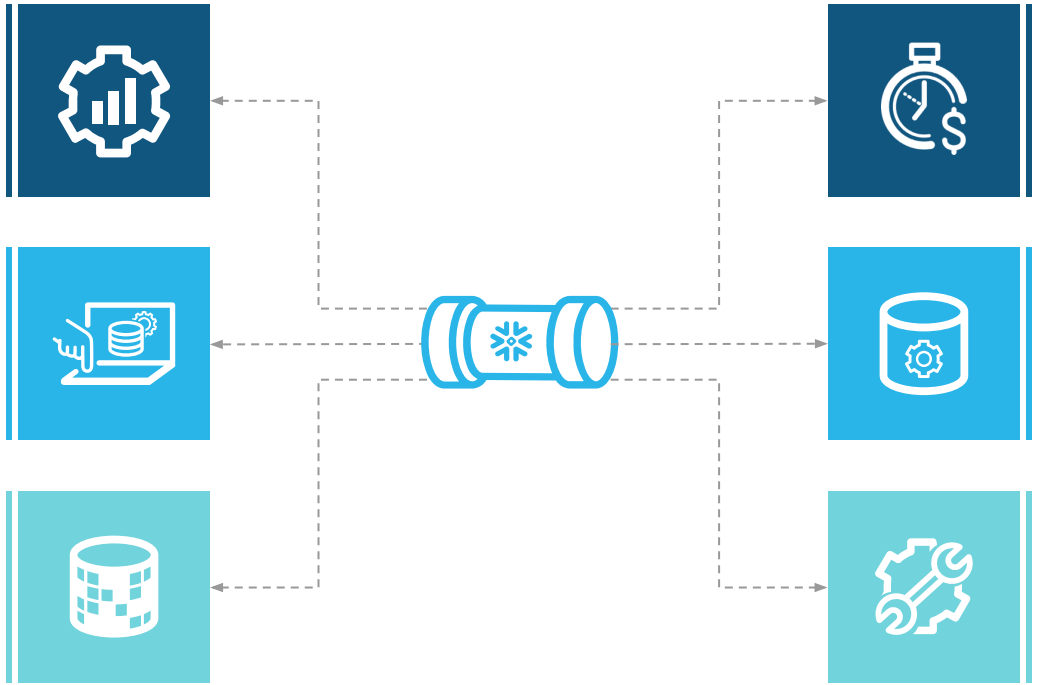


BENEFITS OF SNOWPIPE

Continuously generated data is available for analysis in <1 minute

Avoid repeated manual COPY commands

Error notifications for failure alerting



You only pay for the compute time you use to load data

Zero management. No indexing, tuning, partitioning or vacuuming on load

Serverless: No servers to manage or concurrency to worry about.

Snowpipe Best Practices

Do

Utilize native cloud provider event filtering

Ensure optimal file sizing (100 - 250 MB)

Utilize the same notification integration for multiple Azure/GCP pipes

Utilize error notifications for failure alerting

Check pipe status for debugging and health

Don’t

Ingest tiny files (1KB); use Snowpipe Streaming where possible to bypass files

Build your own frankenstein auto-ingest solution or custom replication

Drop stages, tables, integrations

on running pipes

Ignore COPY best practices

SNOWPIPE STREAMING

LOW-LATENCY STREAMING INGESTION

Streaming in Snowflake

PAIN POINTS: BEFORE

COMPLEXITY

Managing dependencies, scheduling, and orchestration

INEFFICIENCY

Rebuilding tables completely, no incremental materialization

MANAGEABILITY

Brittle pipelines unable to react to changes upstream

BENEFITS: AFTER

SIMPLIFIED PIPELINES

Native support for streaming and continuous batch data pipelines. Easy, declarative semantics and no orchestration required, no infrastructure management

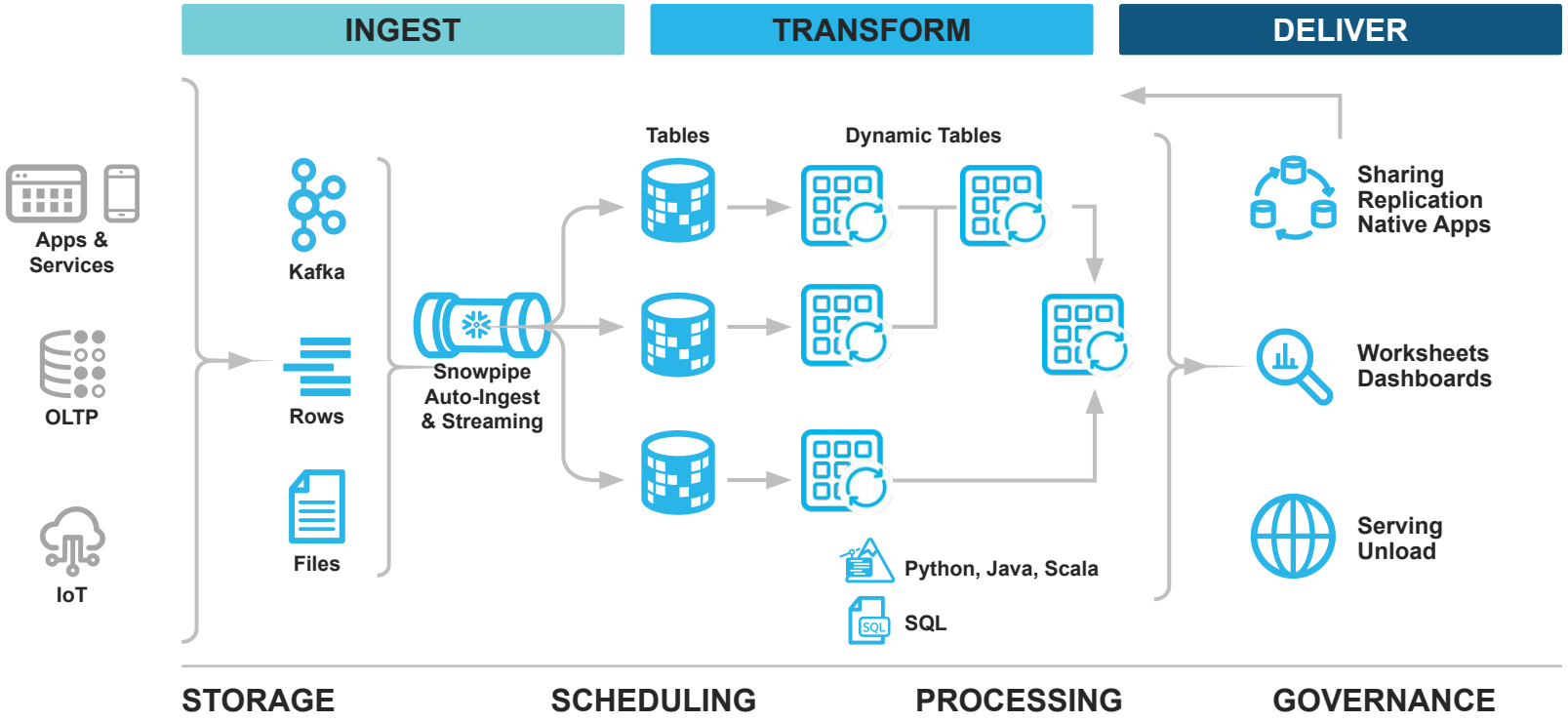
COST EFFECTIVE

Streaming ingest as much as 50% cheaper than files. Continuous incremental processing reduces wasted compute

NATIVE TO DATA CLOUD

Expanding ecosystem in the Data Cloud with consistent and strong security, governance, and scalability with Snowflake Horizon

Streaming Pipelines at a Glance



STREAMING USE CASES

Connectors

- Ingest CDC streams with lower latency
- Ensure exactly once semantics
- Sourced from OLTP DBs, SaaS apps
- Serverless so no clusters / stages to manage

Kafka / Kinesis sources

- Use existing event hubs to source data
- Flexible latency-cost profiles
- Run transformations with all reference data instead of just single row transforms (ELT & ETL)

Security & Log Analytics

- Powered-by-Snowflake apps
- Add full power of Snowflake analytics from day 1
- One place to query latest window of data & full history + reference data
- Proprietary (ISV-built) pipelines for continuous analysis

POWERED BY SNOWFLAKE

IoT / Device logs

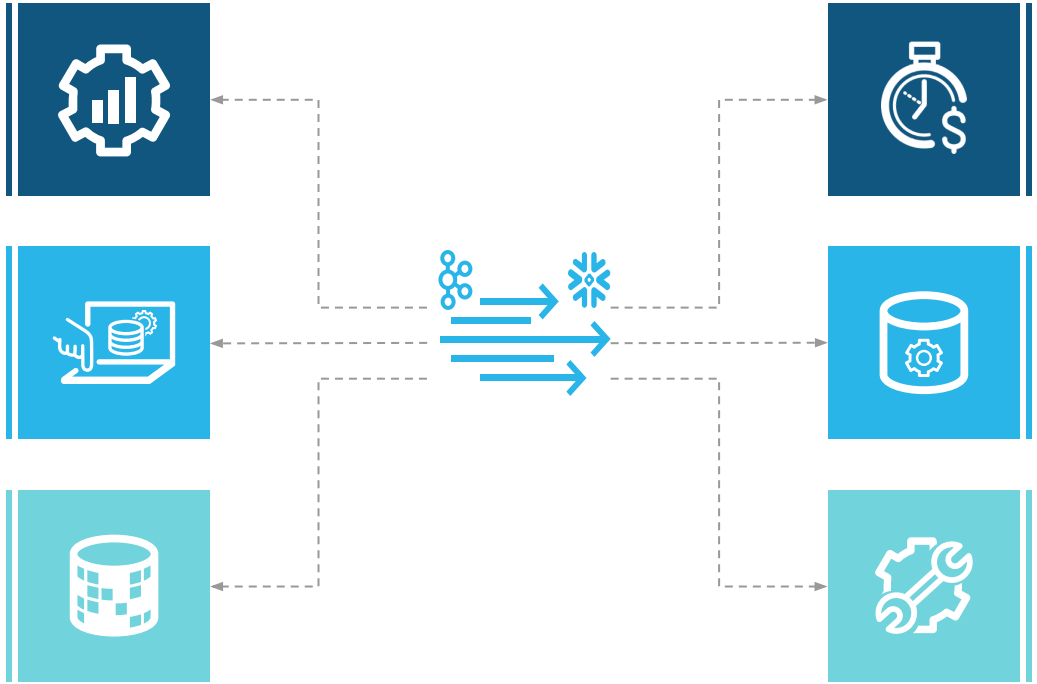
- Aggregated logs from devices
- No need to add event hubs if not needed
- Simple post-ingestion cleanup

BENEFITS OF SNOWPIPE STREAMING

Data queryable in < 5 seconds after Snowflake ack.

Direct Data Streaming: Ingestion to tables over HTTPs, No files or pipe to manage.

High Throughput: GB/sec ingestion rates supported.



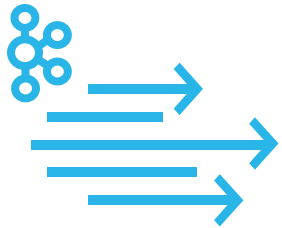
Low cost for both flood and trickle cases.

Exactly-Once and Per-Channel Ordering: Get data in once and process in order.

Low overhead: Minimal configuration and setup needed.

Kafka Connector with Snowpipe Streaming

- **Open Source:** Using new Snowpipe Streaming APIs starting from version 1.9.0
- **Semantics:** Exactly Once
- **Error Handling:** Sent to DLQ (Kafka Topic)
- **Low Cost + Low Latency:** Benefits of Snowpipe Streaming
- **Schematization:** Support for schema detection and evolution



Snowpipe Streaming Best Practices

Do

Consider your business requirements to determine the best ingestion cost/latency

Leverage Snowpipe Streaming for Snowflake's Kafka Connector with a simple upgrade

Provide usage feedback and feature requests

Don't

Build a streaming application just to read a large file or batch load existing data

Replace all batch ingestion; augment it instead with streaming

Connectors

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

Description

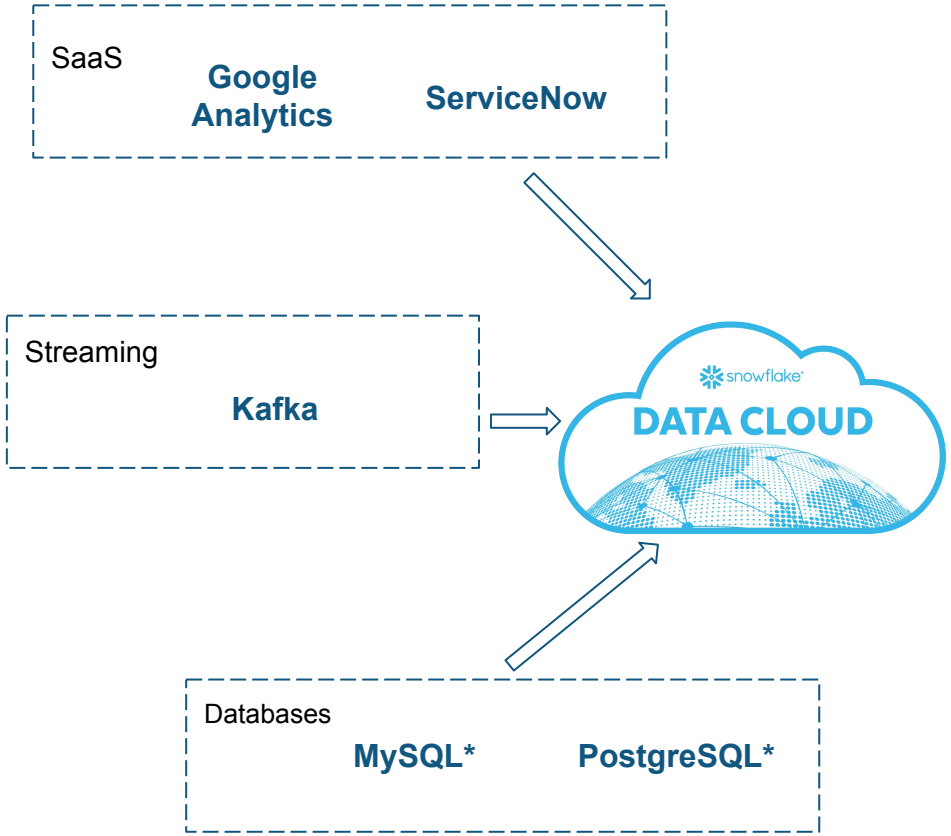
Native integration with databases, applications, and messaging.

Value

Live data with no delays through a fully managed service with governed, revocable access. Reliable and scalable.

Functionality

Natively built in the Data Cloud leveraging built-in security and reliability capabilities.



RESOURCES

https://www.snowflake.com/blog/best-practices-for-data-ingestion/

https://www.snowflake.com/blog/best-practices-for-data-ingestion-part-2/

https://www.snowflake.com/blog/data-ingestion-best-practices-part-three/

https://docs.snowflake.com/en/user-guide/data-load-snowpipe-streaming-overview