Banian

Datavault BUILDER

UNTERSCHIEDE & GEMEINSAMKEITEN IM KONTEXT VON DATA VAULT

snowflake

Exasol

DDVUG, 15.04.2021

ÜBER UNS

DOMINIK IMARK

dominik.imark@banian.ch
https://banian.ch

Banian

PETR BELES

petr.beles@2150.ch
https://datavault-builder.com

Datavault BUILDER

ÜBERSICHT

01 Unterschiede und Gemeinsamkeiten

02 Praxis Vergleich anhand des Data Vault Builder

03 Q & A

ANFORDERUNGEN DATA VAULT

ZUGANG ZU QUELLSYSTEMEN

- Möglichkeiten zum Import von Daten
- Möglichkeit zur virtuellen Anbindung von externen Quellen

ZUGANG ZU META DATEN

Zugriff auf Meta Informationen zu allen Objekten und Eigenschaften der Datenbank zur Unterstützung von Automatisierungs- und Deploymentlösungen

PERFORMANTE VERARBEITUNG

- Verwendung von Parallelität
- Join-Performance
- Streaming- / Triggerfunktionialitäten

KOMBINATION MIT "DATA SCIENCE" ARCHITEKTUREN

- Anbindung von Data Lake Umgebungen und Datenformaten
- Verwendung von diversen Skriptsprachen

TRENNUNG VON RESSOURCEN

- Sicherstellung von Ressourcen für die Verarbeitung
- Optimierte Ressourcenbereitstellung je nach Anwendungsfall (Scale-Up / Scale-Out)

VERWENDUNG VON HASH KEYS

- Optimierung von Hashing Verfahren und Speicherung

GRUNDSÄTZLICHES

Analytische Datenbank

STARK VEREINFACHT

Technologie um hohe Datenvolumen in einer kurzen Zeit zu filtern, zu aggregieren und zu selektieren

01 Pre-Calculation / OLAP

Vorberechnung von Aggregation über mehrdimensionalen "Würfel" oder auch „Materialized Views“.

02 In-Memory

Speicherung der Daten optimiert und komprimiert direkt im Memory.

03 Massive Parallel Processing

Verteilung der Daten über ein Cluster von Servern (Nodes) um die Arbeitslast bei der Abfrageverarbeitung zu teilen.

04 Spaltenorientierung

Speicherung der Daten nach Spalten anstatt nach Zeilen um dadurch die Anzahl der Datenelemente zu reduzieren, welche durch die Datenbank gelesen werden müssen.

GRUNDSÄTZLICHES

Deployment-Szenarien

01 On-Premise

Verwendung der Datenbank auf der eigenen Infrastruktur

- Vollständige Hoheit über die Umgebung
- Hoher Pre-Invest in Infrastruktur
- Hohe Laufzeiten für neue Infrastruktur

→ Kann heute durch Virtualisierung ebenfalls optimiert werden

02 Database in the Cloud

Verwendung der Datenbank auf der Infrastruktur bereitgestellt durch Cloud Provider

- Installation, Konfiguration und Verwaltung / Betrieb der Datenbank durch den Benutzer
- Einfache und schnelle Skalierbarkeit durch verfügbare Infrastruktur

03 Database as a Service (DBaaS)

Vollständig durch den Cloud Anbieter organisierte und verwaltete Datenbank

- Schnelle Verfügbarkeit
- Tiefe bis keine Betriebsaufwände
- Einfache und schnelle Skalierbarkeit durch verfügbare Infrastruktur
- Hohe Abhängigkeit zum Anbieter

GRUNDSÄTZLICHES

Massive Parallel Processing-Architektur (MPP)

01 Shared-Nothing

Jeder Node / Server besitzt seine eigene CPU, Memory und Speicher. Dadurch ist jede Node für sich selbstständig und das Cluster kann einfach erweitert werden.

→ Netzwerk ist immer noch shared

02 Shared-Disk

Jeder Node / Server besitzt seine eigene CPU und Memory (Compute), jedoch wird der Speicher von allen Nodes geteilt und gemeinsam verwendet.

STECKBRIEFE

Kurze Übersicht zu den Datenbanken

	Exasol	snowflake
Gründungsjahr	2000	2012
Land	Deutschland	USA
Markteintritt	2008	2014
Claim	Schnellste in-memory Datenbank gebaut für Analytics	Cloud Data Platform gebaut für die Cloud
Deployment-Szenarien	On-Prem, DB in the Cloud, DBaaS	DBaaS only
Architektur	MPP, in-memory, Shared Nothing	MPP, Shared Disk

STORAGE

	Exasol	snowflake
Grundlage	Daten verteilt über Nodes	Daten in einem zentralisiertem Speicher
Speicher	Memory und Disk	Disk, SSD (Local Cache), Memory
Orientierung	Spaltenbasierte Speicherung	Spaltenbasierte Speicherung
Clustering	Distributionsschlüssel und Partitionierung	Micropartitions (Clustering Keys)
Fail-Safety	Datenreplikation über Nodes, Reserve Node, Backups	Time Travel und SF Fail-Save

BACKUP / RESTORE / DATAOPS

01 Backup

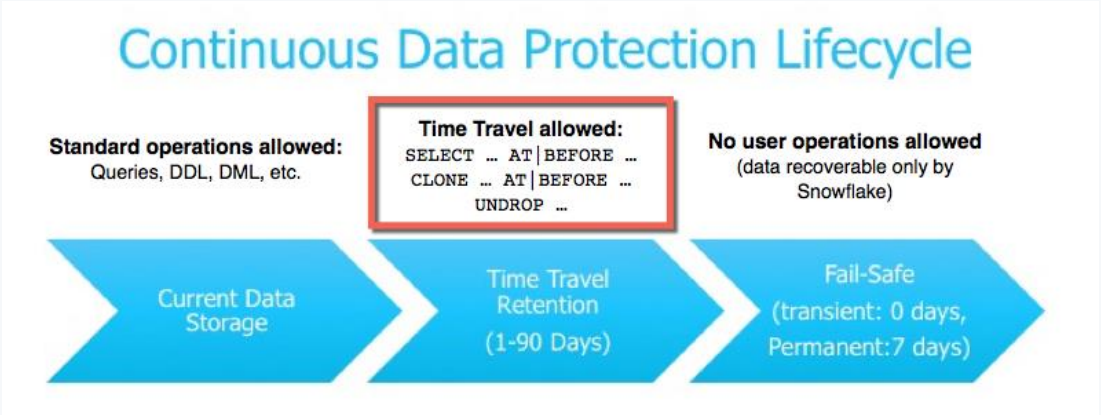
- Komplett automatisch
- Bis zu 90 Tage
- Fail Safe von 7 Tagen

02 Restore

- Point in Time Restore
- Cloning basierend auf Objekten und Zeit
- Abfragen mit Zeitbezug (PIT)

03 DataOps

- Database / Object Cloning
- Sandboxes (Cloning)

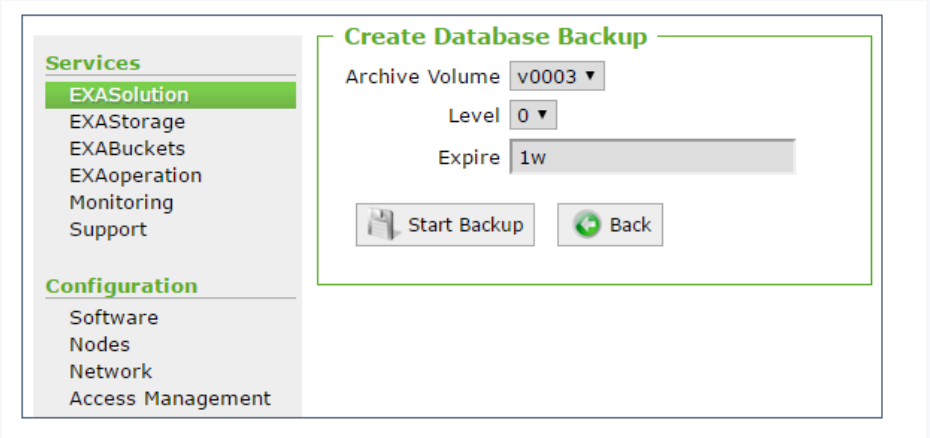


* Understanding & Using Time Travel — Snowflake Documentation

BACKUP / RESTORE / DATAOPS

01 Backup

- Manuell bzw. konfiguriert über Scheduler
- Local oder Remote (Cloud)
- Full oder Incremental Backup



02 Restore

- Immer basierend auf vorhandenen Backups (No PIT)
- Virtual Access Restore, Non-Blocking Restore und Blocking Restore

03 DataOps

- Docker Unterstützung für Sandboxes / Clon / Dev-Umgebung

* Backup and Restore - On Premise | Exasol Documentation

IMPORT DATA

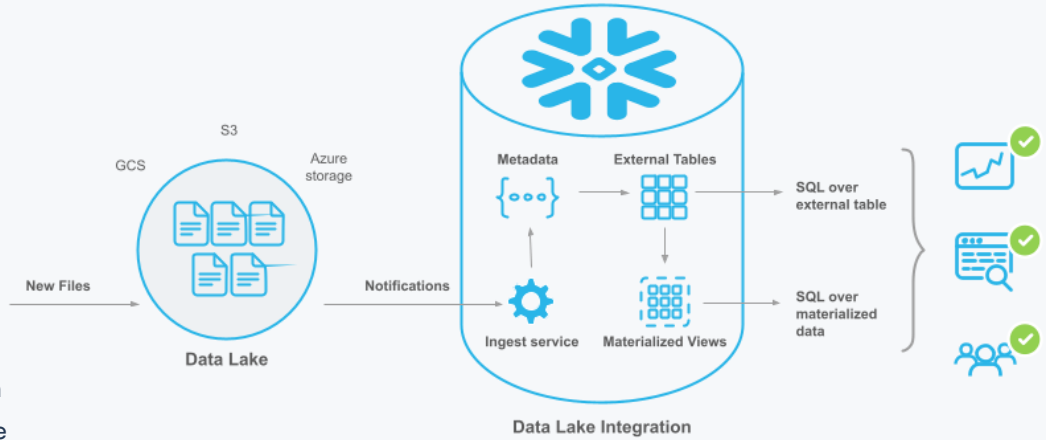
01 Source System Integration

- File Upload (SnowSQL CLI) – PUT
- ETL Tools

02 Data Lake Integration / External Tables

- Anbindung von diversen Cloud Storage Anbietern
- S3, GC Storage, Azure Blob Storage, Apache Hive
- Integration über Kopie oder Select (View)

Snowflake as Query Engine for Datalakes



* https://www.snowflake.com/blog/external-tables-are-now-generally-available-on-snowflake
* https://docs.snowflake.com/en/user-guide/tables-external-intro.html

IMPORT DATA

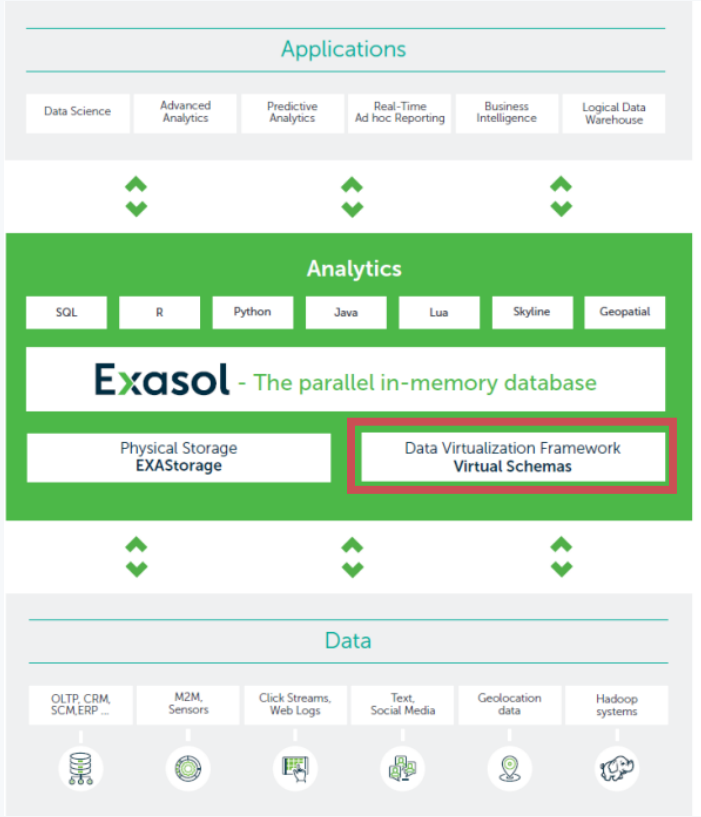
Exasol

01 Source System Integration

- Integration über Import SQL Befehl (File, JDBC DB, Skript)
- ETL Tools

02 Data Lake Integration / External Tables

- Flexible Open Source Connectoren zur direkten Anbindung von Quellen
- Bspw. DB2, BigQuery, S3, Hive, Impala, MySQL, PostgreSQL, Redshift, SAP HANA, SQL Server, ...
- Integration über Kopie oder Select (View)



* https://docs.exasol.com/6.1/get_started/exasol_overview.htm

STORED PROCEDURES / SKRIPT

01 Scripting

- Erstellung von Multi Statement Skripten inkl. Error Handling
- Erstellung von Kundenspezifischen Funktionen für SQL
- SQL (UDF) oder Javascript (UDF, Procedurs)

02 Data Science / in-memory Analytics

- Python Connector → Externe Bearbeitung über SQL

```
CREATE FUNCTION validate_ID(ID FLOAT)
RETURNS VARCHAR
LANGUAGE JAVASCRIPT
AS $$
    try {
        if (ID < 0) {
            throw "ID cannot be negative!";
        } else {
            return "ID validated.";
        }
    } catch (err) {
        return "Error: " + err;
    }
$$;
```

* UDFs (User-Defined Functions) — Snowflake Documentation

STORED PROCEDURES / SKRIPT

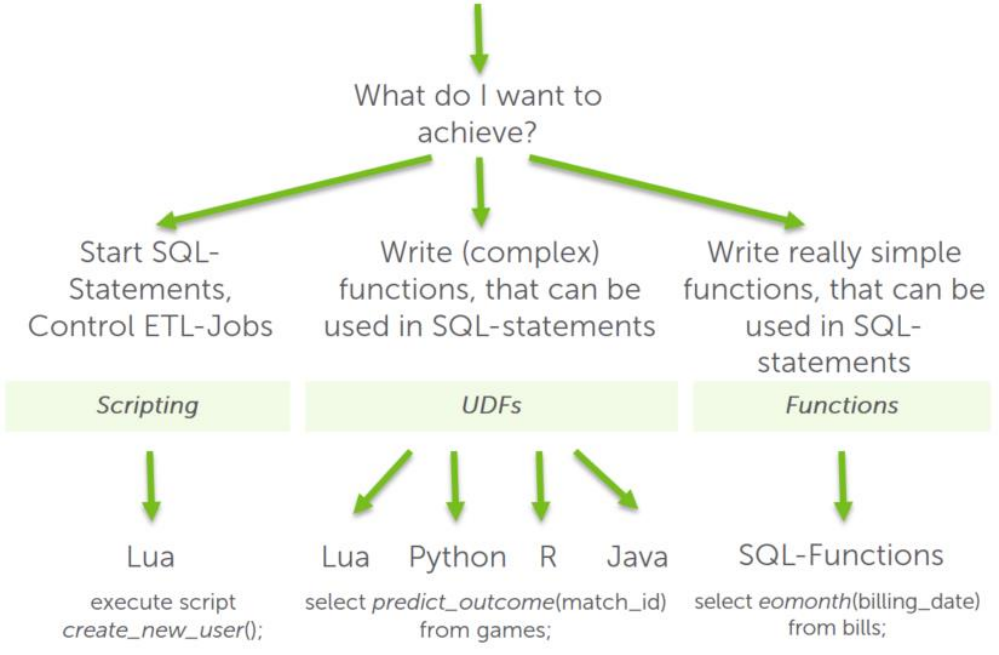
Exasol

01 Scripting

- Erstellung von Multi Statement Skripten inkl. Error Handling
- Erstellung von Kundenspezifischen Funktionen für SQL
- Lua
- Kombination von UDF und Script für ETL
- Verwendung von SQL Preprocessor

02 Data Science / in-memory Analytics

- Nutzung der MPP Funktionalität
- Verwendung gängiger Programmiersprachen (Python / R)



* https://community.exasol.com/t5/database-features/lua-python-and-all-the-others-when-to-use-which-programming/ta-p/593

DATA TYPES

Database Migration – Data Type Alias

For compatibility reasons, Exasol has implemented several aliases. The following table provides you with the list of aliases.

Alias	Exasol Type Equivalent	Note
BIGINT	DECIMAL(36,0)	
BOOL	BOOLEAN	
CHAR	CHAR(1)	
CHAR VARYING(n)	VARCHAR(n)	$1 \leq n \leq 2{,}000{,}000$
CHARACTER	CHAR(1)	

Exasol

Snowflake supports most SQL data types:

Category	Type	Notes
Numeric Data Types	NUMBER	Default precision and scale are (38,0).
	DECIMAL	Synonymous with NUMBER.
	NUMERIC	Synonymous with NUMBER.
	INT, INTEGER, BIGINT, SMALLINT	Synonymous with NUMBER except precision and scale cannot be specified.
	FLOAT, FLOAT4, FLOAT8 [1]	
	DOUBLE [1]	Synonymous with FLOAT.
	DOUBLE PRECISION [1]	Synonymous with FLOAT.
	REAL [1]	Synonymous with FLOAT.

snowflake®

DATA TYPES

Hash Data Type

Exasol

HASH Data Type

The hashtype can be used to store hash values, for example, MD5 hashes or Universally Unique Identifiers (UUID).

Exasol Type	Note
`HASHTYPE[(n BYTE | m BIT)]`	1<=n<=1024 BYTE 8<=m<=8192 BIT

Input

As input, hex strings (0-F) are accepted. Leading/trailing curly braces and dashes are ignored. The following input is valid for HASHTYPE (16 BYTE):

```
550e8400-e29b-11d4-a716-446655440000
{550e8400-e29b-11d4-a716-446655440000}
550E8400E29B11D4A716446655440000
550e-8400-e29b-11d4-a716-4466-5544-0000
```

DATA TYPES

Semi-Structured Data

```
SELECT id,
       JSON_VALUE(json, '$.name' NULL ON EMPTY DEFAULT 'invalid name' ON ERROR) as "JSON value"
FROM json_input;

| ID | JSON value
| -- | ------------
| 1  | Smith
| 2  | NULL
| 3  | invalid name
| 4  | Doe
```

Exasol

01 DV Satellites (Document, Multi-Active)

Speichere json strukturierte Daten / Dokumente direkt als Spalte in einem Satellit. Damit können Szenarien wie Multi-Active Satelliten ohne Änderung am Basis Pattern ermöglicht werden oder auch dynamische Dokumentstrukturen (Wechselnde Spalten) abgebildet werden.

```
select
  src:device_type::string as device_type
, src:version::string     as version
, value                   as src
from
  raw_source
, lateral flatten( input => src:events );
```

snowflake

```
select *
    from demonstration1;
+----+-------------+---------------------+--------------------------+
| ID | ARRAY1      | VARIANT1            | OBJECT1                  |
|----+-------------+---------------------+--------------------------|
|  1 | [           | {                   | {                        |
|    |   1,        |   "key1": "value1", |   "outer_key1": {        |
|    |   2,        |   "key2": "value2"  |     "inner_key1A": "1a", |
|    |   3         | }                   |     "inner_key1B": "1b"  |
|    | ]           |                     |   },                     |
|    |             |                     |   "outer_key2": {        |
|    |             |                     |     "inner_key2": 2      |
|    |             |                     |   }                      |
|    |             |                     | }                        |
```

02 References

https://datavault-builder.com/de/2018/11/07/on-multi-active-satellites/

https://danlinstedt.com/allposts/datavaultcat/datavault-2-0-supports-dynamic-data-warehousing/

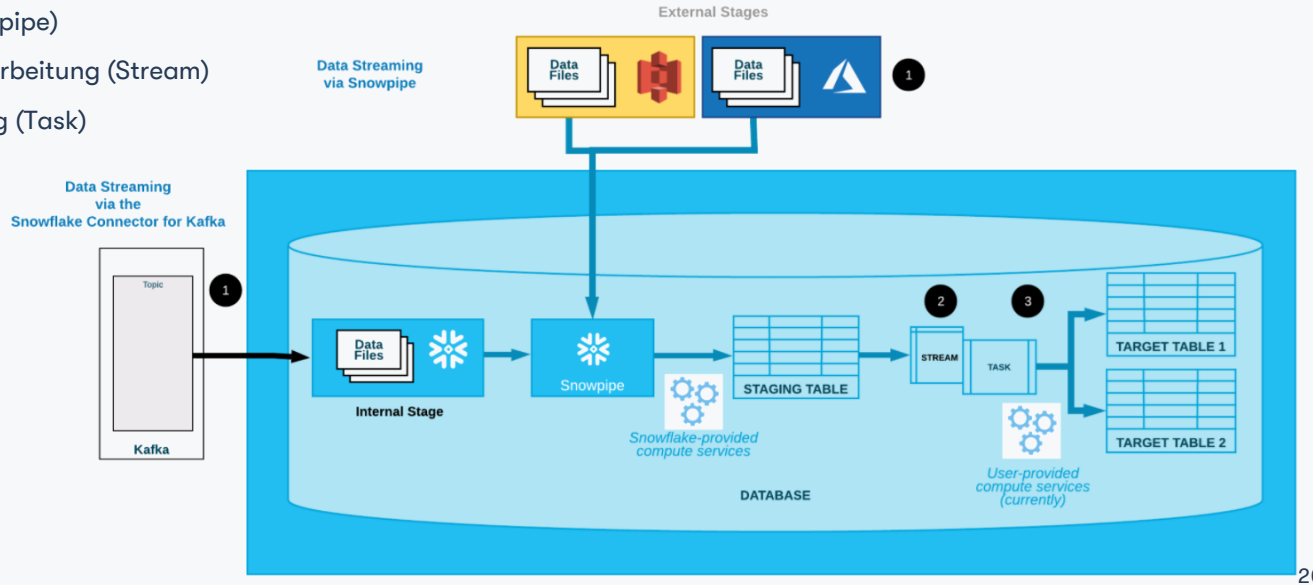
https://www.snowflake.com/blog/tips-for-optimizing-the-data-vault-architecture-on-snowflake-part-3

LOADING PATTERNS

01 Tasks / Streams

Integrierte Data Pipeline Fähigkeiten zur automatisierten Verarbeitung von Daten.

- Import von Quellen (Snowpipe)
- CDC Erkennung und Verarbeitung (Stream)
- Zeitgesteuerte Ausführung (Task)



* Einführung in Datenpipelines — Snowflake Documentation

LOADING PATTERNS

01 Multi-Table Insert

Schreibe mit einem SQL Statement in diverse Tabellen basierend auf einem Select.

ZB. Gleichzeitige Befüllung von Hub, Link und Satellites in einem Statement basierend auf einer Stage Tabelle.

```
insert all
  into t1
  into t1 (c1, c2, c3) values (n2, n1, default)
  into t2 (c1, c2, c3)
  into t2 values (n3, n2, n1)
select n1, n2, n3 from src;
```

```
insert all
  when n1 > 100 then
    into t1
  when n1 > 10 then
    into t1
    into t2
  else
    into t2
select n1 from src;
```

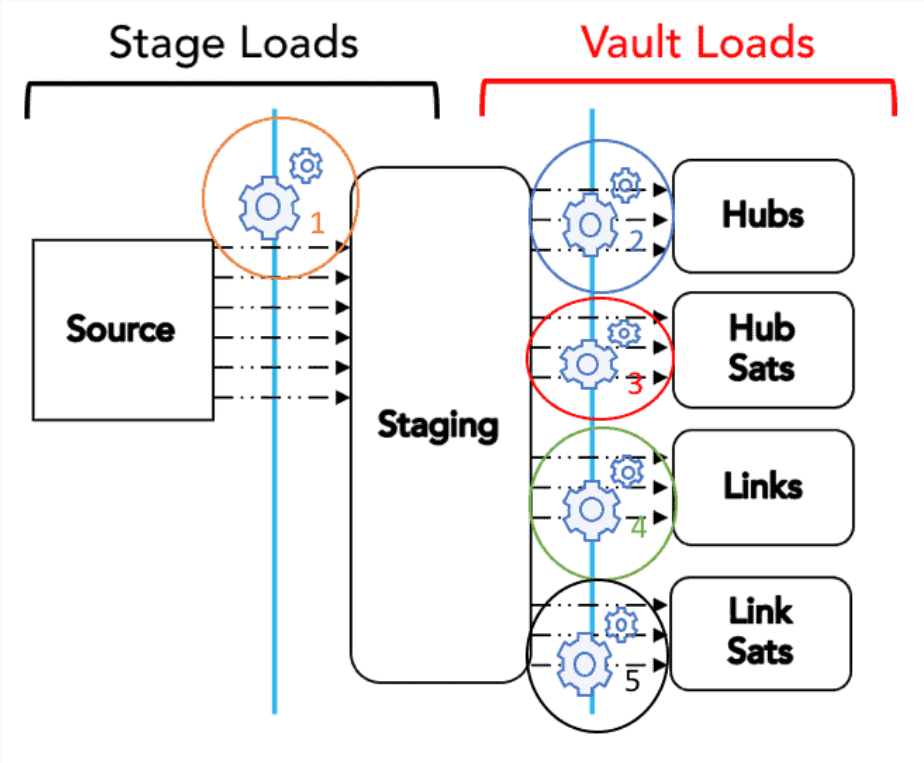
* INSERT (multi-table) — Snowflake Documentation

LOADING PATTERNS

01 Multiple Compute Cluster

Verwendung von getrennten Multi-Cluster Warehouses (Compute Cluster) für die verschiedenen Ebenen / Layer oder Bereiche im Vault.

- Keine Abhängigkeiten der Ressourcenverfügbarkeiten
- Automatisches Scaling über die Cluster



* Blog from Kent Graziano:
https://www.snowflake.com/blog/tips-for-optimizing-the-data-vault-architecture-on-snowflake-part-2

LICENSING

Exasol

01 On-Premise

Raw Data License
Database Ram License

02 Cloud

Pay as you go - (Cloud Infrastruktur + Software Kosten)
Bring your own license – (Cloud Infrastruktur + Lizenz)

→ Kalkulierbare Kosten (unabhängig von der Auslastung)

snowflake

01 On-Premise

-

02 Cloud

Pay as you go – On Demand (monatliche Abrechnung)
Pre-Purchased Capacity (Bessere Preise)

→ Basierend auf tatsächlicher Verwendung (Credits)

“IN DER THEORIE IST PRAKTISCH ALLES MÖGLICH.”

Q & A

Banian