SAS & Snowflake

A story on how to go from SAS9 to SAS Viya & Snowflake

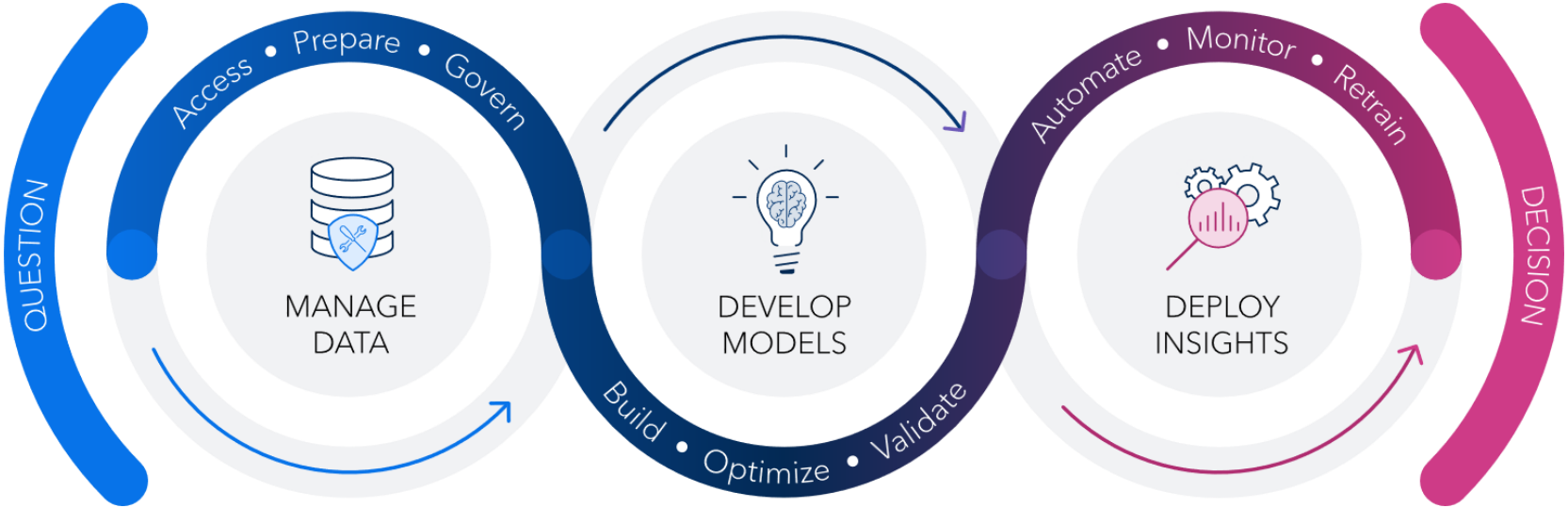
+ short SAS Viya with Singlestore update

Jonas Lie-Nielsen

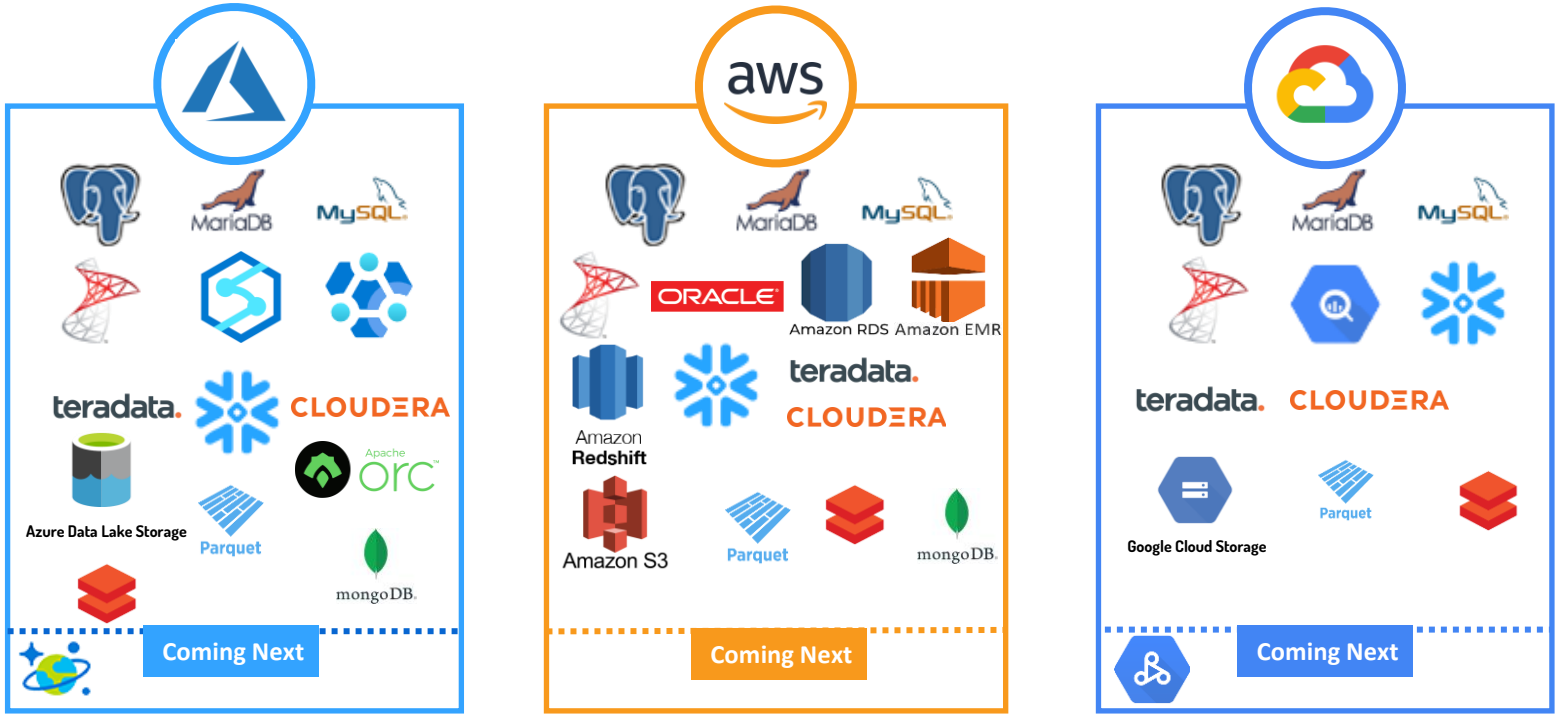
Advisory Solution Architect

Tromsø, May 11th, 2023

AI Lifecycle

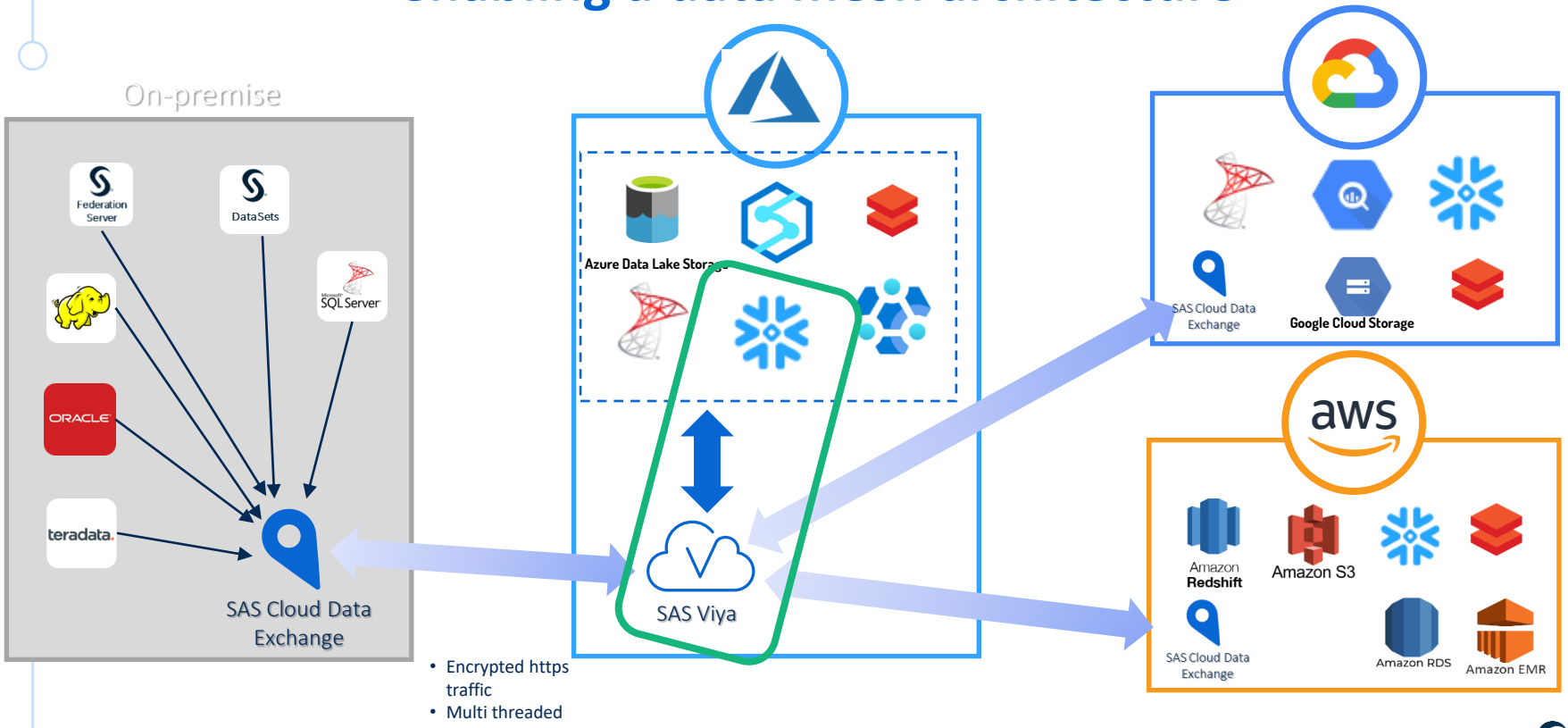


Cloud Platforms Connectivity Support



Using SAS Viya as a engine for cloud migration and enabling a data mesh architecture

Focus for today



- Encrypted https traffic
- Multi threaded

2029 target - $10B

AS OF JANUARY 31, 2023 - Q4 AND FULL YEAR FY23

PRODUCT REVENUE 1

Q4 FY23: **$555.3M**
+ 54% YoY Growth

FY23: **$1,938.8M**
+ 70% YoY Growth

NET REVENUE RETENTION RATE 2

158%

TOTAL CUSTOMERS 2

7,828
+ 31% YoY Growth

$1M CUSTOMERS 2

330
+ 79% YoY Growth
Customers with Trailing 12-Month Product Revenue Greater than $1M

FORBES GLOBAL 2000 CUSTOMERS 3

573
+ 16% YoY Growth

SNOWFLAKE MARKETPLACE LISTINGS 4

1,838
Total Listings
+ 8% QoQ Growth

CUSTOMER SATISFACTION

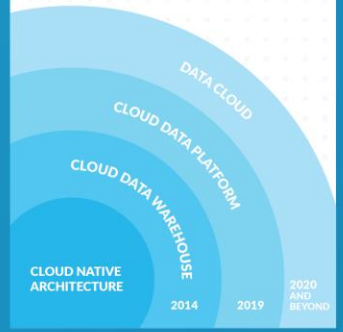
DRESNER CUSTOMER SATISFACTION SCORE 5

100%
Of Customers Recommend Snowflake for Sixth Consecutive Year

NET PROMOTER SCORE (NPS) 6

72
Most Customers Would Recommend Snowflake to a Friend or Colleague

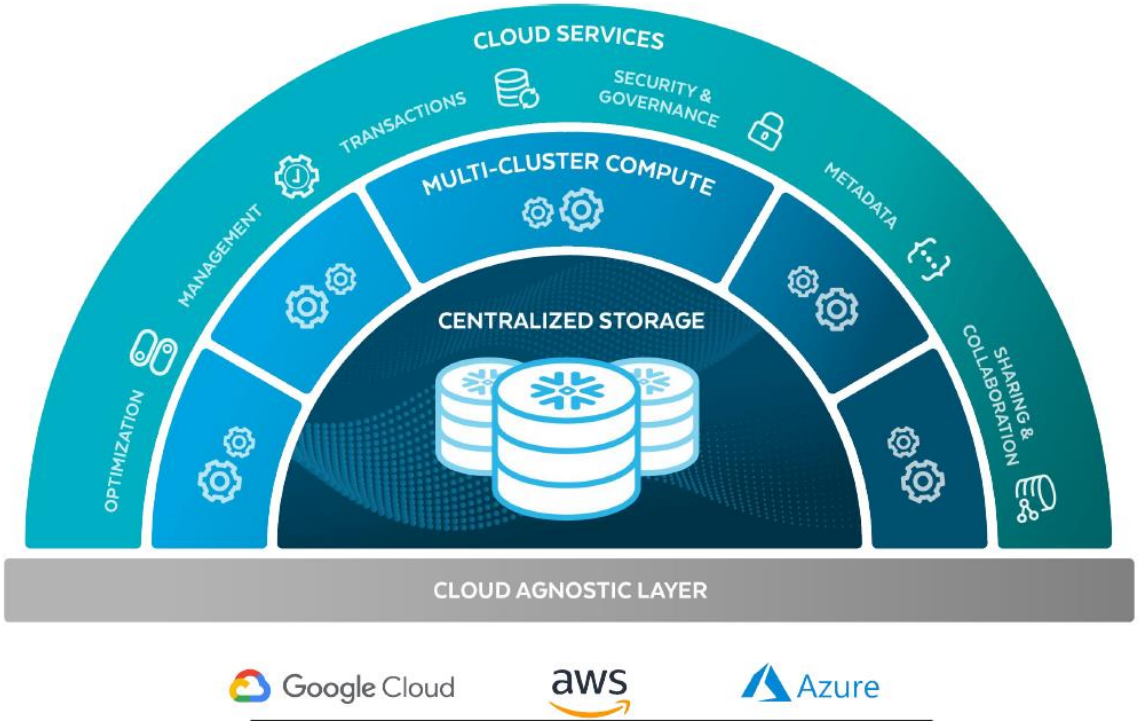
SNOWFLAKE DATA CLOUD



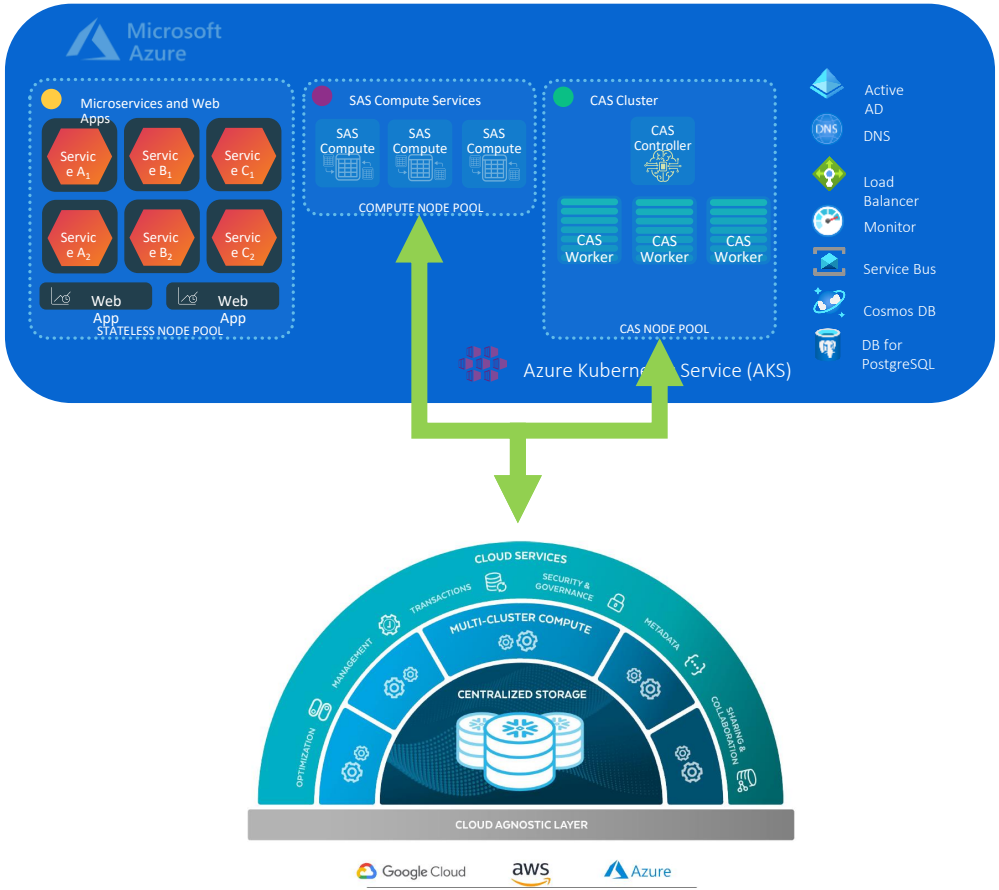
1. For the three and twelve months ended January 31, 2023, respectively. 2. As of January 31, 2023. Please see our Q4 FY23 earnings press release for definitions of net revenue retention rate, total customers, and customers with trailing 12-month product revenue greater than $1 million. 3. As of January 31, 2023. Based on the 2022 Forbes Global 2000 list. Our Forbes Global 2000 customer count is subject to adjustments for annual updates to the Global 2000 list by Forbes, as well as acquisitions, consolidations, spin-offs, and other market activity with respect to such customers, and we present our Forbes Global 2000 customer count for historical periods reflecting these adjustments. 4. As of January 31, 2023. Each live dataset, package of datasets, or data service published by a data provider as a single product offering on Snowflake Marketplace is counted as a unique listing. A listing may be available in one or more regions where Snowflake Marketplace is available. 5. Dresner Advisory Services: 2023 Wisdom of Crowds® Analytical Data Infrastructure (ADI) Market Study, January 2023. 6. As of June 2022. If a customer fails to (i) respond to each required question in the survey or (ii) submit a complete set of responses by the end of the survey period, we consider that customer's survey incomplete. Starting with our NPS as of June 2022, we exclude incomplete survey responses from the calculation.

SAS Viya and Snowflake

Snowflake Overview



Architecture Overview



SAS/ACCESS Interface to Snowflake

Supported Features

Category	Feature	Supported
In-Database Processing	SQL Passthrough (PROC SQL)	Yes
	Procedures	Yes
	Processing with PROC FEDSQL and PROC DS2	Yes
	SQL Functions	Yes
Performance	Bulk load data to Snowflake (from SAS Compute Server)	Yes
	Bulk unload data from Snowflake (to SAS Compute Server)	Yes
CAS Related	Serial Data Transfer (including multi-node support)	Yes
	Parallel Data Transfer	No

In-Database Processing

SQL Passthrough

- SAS/ACCESS translates SAS statements into the SQL of the DB and passes down for processing (**implicit** passthru).

```
proc sql;
    drop table snowlib.europe_cars;
    create table snowlib.europe_cars as
    select * from snowlib.cars
    where origin='Europe';
quit;
```

```
proc delete data=snowlib.europe_cars;
run;
data snowlib.europe_cars;
   set snowlib.cars;
   where origin='Europe';
run;
```

- You submit SQL that is native to the DB and it is passed directly (**explicit** passthru).

```
proc sql noerrorstop;
    connect to snow as x1(server=&server
    user=&user password=&pw database=&db schema=&schema warehouse='&warehouse' );
    execute ( CREATE TABLE mycars ( no int primary key, make varchar(20) ) ) by x1;
    execute ( INSERT INTO mycars values (1, 'Audi') ) by x1;
    execute ( INSERT INTO mycars values (2, 'Tiguan') ) by x1;
    select * from connection to x1 (SELECT * FROM mycars ORDER BY no);
    disconnect from x1;
```

In-Database Processing

SQL Passthrough – Supported Functions

ABS
ARCOS (ACOS)
ARSIN (ASIN)
ATAN
ATAN2
CAT (CONCAT)
CEIL
COALESCE
COS
COSH
COT
DAY (DAYOFMONTH)
DTEXTDAY (DAYOFMONTH)
DTEXTMONTH (MONTH)
DTEXTYEAR (YEAR)
DTEXTWEEKDAY (DAYOFWEEK)*
EXP
FLOOR
HOUR
INDEX (CHARINDEX)
LEFT (LTRIM)
LENGTH (OCTET_LENGTH(RTRIM()))** ***
LENGTHC (LENGTH)
LOG (LN)
LOG10 (LOG(10,*n*))
LOG2 (LOG(2,*n*))
LOWCASE (LOWER)
MINUTE
MOD
MONTH
QTR (QUARTER)
REPEAT
SECOND
SIGN
SIN
SINH
SQRT
STD (STDDEV)
STRIP (TRIM)
SUBSTR
TAN
TANH
TRANWRD (REGEXP_REPLACE)
TRIMN (RTRIM)
UPCASE (UPPER)
VAR (VARIANCE)
WEEKDAY (DAYOFWEEK)*
YEAR

Use libname option to check list of supported functions:
sql_functions_copy=saslog
Add sql_functions=all for extended view of mapped functions

CAT	CONCAT
DATE	CURRENT_DATE
TODAY	CURRENT_DATE
TIME	CURRENT_TIME
DATETIME	CURRENT_TIMESTAMP
DATEPART	TO_DATE
TIMEPART	TO_TIME
COMPRESS	TRANSLATE

In-Database Processing

PROC Pushdown

In-Database SAS procedures:

- FREQ
- MEANS
- RANK
- REPORT
- SORT
- SUMMARY
- TABULATE

SAS Studio Code

```
proc means data=csnow.ORDERS;
class O_ORDERSTATUS;
var O_TOTALPRICE;
run;
```

The MEANS Procedure

Analysis Variable : O_TOTALPRICE O_TOTALPRICE						
O_ORDERSTATUS	N Obs	N	Mean	Std Dev	Minimum	Maximum
F	73072502	73072502	150229.98	88595.33	811.7300000	586945.44
O	73086053	73086053	150243.63	88608.91	821.1800000	591036.15
P	3841445	3841445	184767.36	79509.06	1914.25	550128.18

Snowflake Query History

SQL Text

```
1  select COUNT(*) as "ZSQL1", MIN(TXT_1."O_ORDERSTATUS") as "ZSQL2", COUNT(*)
as "ZSQL3", COUNT(TXT_1."O_TOTALPRICE") as "ZSQL4", MIN(TXT_1."O_TOTALPRICE")
as "ZSQL5", MAX(TXT_1."O_TOTALPRICE") as "ZSQL6", SUM(TXT_1."O_TOTALPRICE")
as "ZSQL7", COALESCE(VARIANCE(TXT_1."O_TOTALPRICE")*
(COUNT(TXT_1."O_TOTALPRICE")-1),0) as "ZSQL8" from "ROBIN"."ORDERS" TXT_1
group by TXT_1."O_ORDERSTATUS"
```

Performance

Bulk Loading

SAS Studio Code

```
/*bulk load - move data from SAS Viya to snowflake */
data csnow.crime (bulkload=yes bl_internal_stage='@~');
set rtdata.crime;
run;
```

Snowflake Query History

01a637a3-...	SELECT "ID", "Case Number", "Date", "Block", "IUCR", "Primary Type", "Description", "Location Des...	SUKROT	COMPUTE_...	1	X-Small	1373304094...	2:55:32 PM	2:55:33 PM	495ms	2.3MB
01a637a3-...	commit	SUKROT	COMPUTE_...			1373304217...	2:55:24 PM	2:55:24 PM	31ms	
01a637a3-...	REMOVE '@~/SASSNBL_9F106A93-3E23-C943-94A5-272FE38159E7-01.dat'	SUKROT	COMPUTE_...			1373304217...	2:55:24 PM	2:55:24 PM	162ms	
01a637a3-...	REMOVE '@~/SASSNBL_9F106A93-3E23-C943-94A5-272FE38159E7-00.dat'	SUKROT	COMPUTE_...			1373304217...	2:55:23 PM	2:55:24 PM	196ms	
01a637a3-...	commit	SUKROT	COMPUTE_...			1373304217...	2:55:23 PM	2:55:23 PM	162ms	
01a637a3-...	COPY INTO "ROBIN"."crime" ("ID","Case Number","Date","Block","IUCR","Primary Type","Description"...	SUKROT	COMPUTE_...	1	X-Small	1373304217...	2:55:22 PM	2:55:23 PM	908ms	7.4MB
01a637a3-...	PUT 'file:///opt/sas/viya/config/var/tmp/compsrv/default/dc3e7421-31d6-4810-8b9c-03b2d6ff54...	SUKROT	COMPUTE_...			1373304217...	2:55:20 PM	2:55:21 PM	80ms	
01a637a3-...	PUT 'file:///opt/sas/viya/config/var/tmp/compsrv/default/dc3e7421-31d6-4810-8b9c-03b2d6ff54...	SUKROT	COMPUTE_...			1373304217...	2:55:19 PM	2:55:19 PM	73ms	
01a637a3-...	alter session set autocommit=false	SUKROT	COMPUTE_...			1373304217...	2:55:19 PM	2:55:19 PM	45ms	
01a637a3-...	CREATE TABLE "ROBIN"."crime" ("ID" double,"Case Number" VARCHAR(8),"Date" TIMESTAMP,"Blo...	SUKROT	COMPUTE_...			1373304217...	2:55:19 PM	2:55:19 PM	171ms	

Performance

Bulk Unloading

SAS Studio Code

```
/*bulk unload - move data from snowflake to viya */
data work.basetable2;
set
csnow.BASETABLE2(bulkunload=yes bl_internal_stage='@~');
run;
```

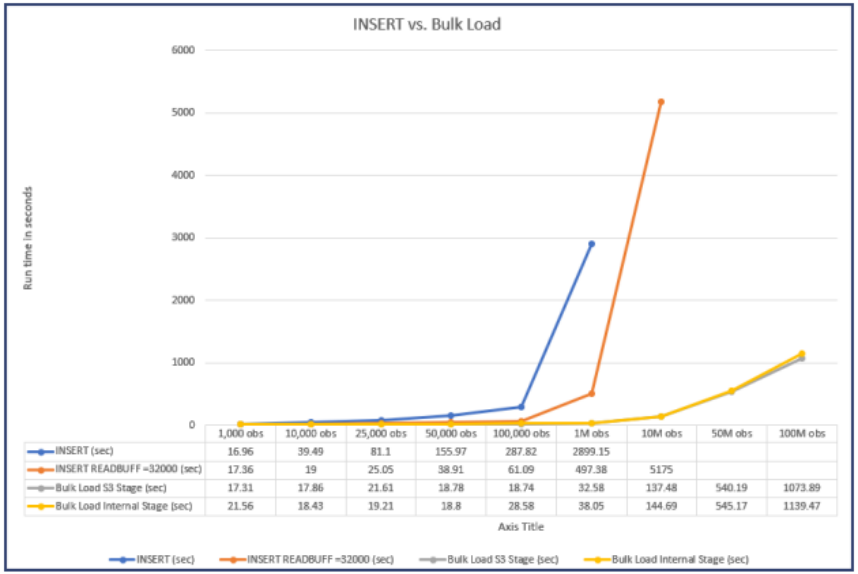
Snowflake Query History

Status	Query ID	SQL Text	User	Warehouse	Clust...	Size	Session ID
✔	01a63c69-...	REMOVE '@~/SASSNBL_41C0415A-7944-7245-A3B1-AF5D40...	SUKROT	COMPUTE_...			1373304095...
✔	01a63c69-...	GET '@~/SASSNBL_41C0415A-7944-7245-A3B1-AF5D409F3...	SUKROT	COMPUTE_...			1373304095...
✔	01a63c69-...	COPY INTO '@~/SASSNBL_41C0415A-7944-7245-A3B1-AF5D...	SUKROT	COMPUTE_...	1	X-Small	1373304095...

Performance

Bulk Loading

- Conclusion:
  - Bulk loading much more performant with >100k records
- See Global Forum paper for more details
  - [Paper SAS4103-2020 - An Insider's Guide to SAS/ACCESS® Interface to Snowflake by Jeff Bailey]



	1,000 obs	10,000 obs	25,000 obs	50,000 obs	100,000 obs	1M obs	10M obs	50M obs	100M obs
INSERT (sec)	16.96	39.49	81.1	155.97	287.82	2899.15			
INSERT READBUFF =32000 (sec)	17.36	19	25.05	38.91	61.09	497.38	5175		
Bulk Load S3 Stage (sec)	17.31	17.86	21.61	18.78	18.74	32.58	137.48	540.19	1073.89
Bulk Load Internal Stage (sec)	21.56	18.43	19.21	18.8	28.58	38.05	144.69	545.17	1139.47

SAS Snowflake Partnership Update

300+ SAS customers *already* integrate SAS with Snowflake data (mostly through SAS Access to Snowflake). This happened organically – no campaign or active GTM.

SIs (and even AWS & **Azure)** see the demand for Snowflake and have expressed interest in how SAS works with Snowflake.

Going from Good to Great

- **Snow Services** workstream (Bringing SAS AI/ML to Snowflake data) SAS is one of a small handful chosen for this
- **CI 360 workstream** Increasing SAS integration with Snowflake based on customer demand
- **SAS Access/EP workstream** Exploring how to best enable EP capability

What does the customer get from this

- Better TCO for customers
- More SAS Viya awareness among Millennials and cloud community
- 'Better together'- Snowflake and SAS collaborating solving customer challenges
- No need to reimplement all SAS code for using Snowflake

A migration example

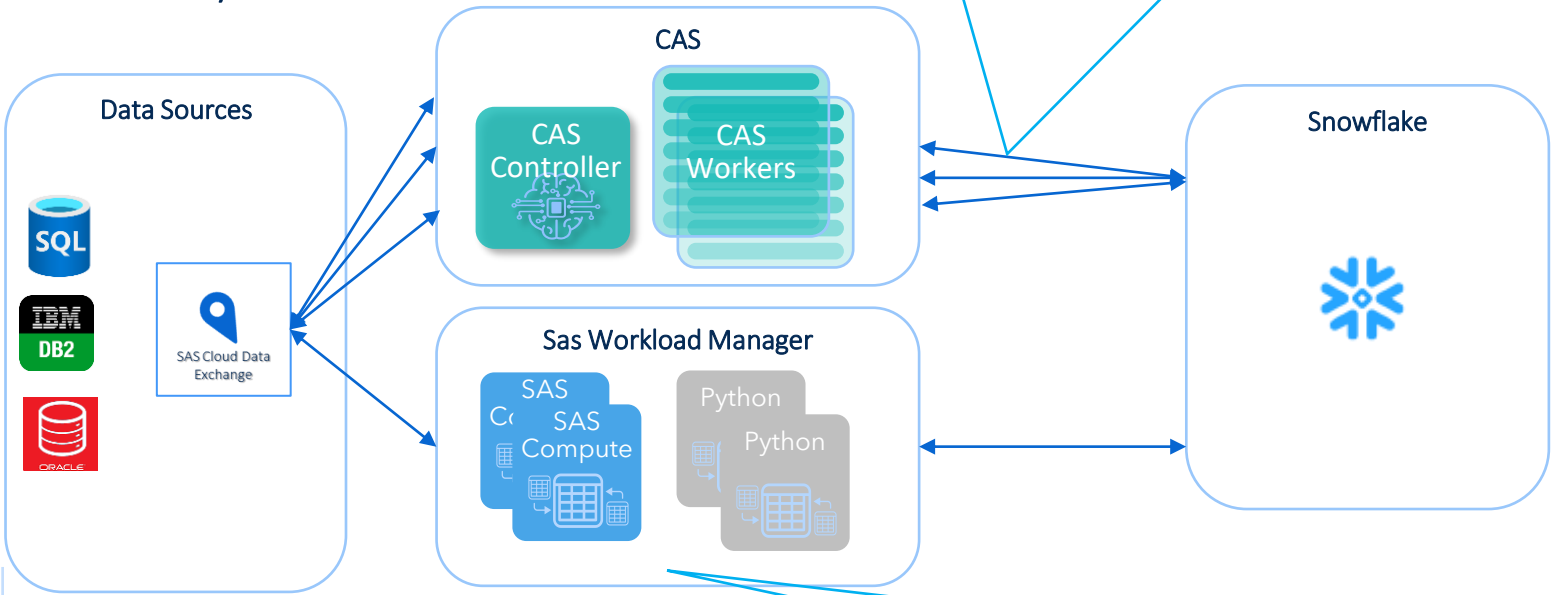
How to migrate to Snowflake and reuse your SAS assets?

- Take an example customer having their DWH in SAS datasets and:
  - 3000 SAS DI ETL jobs
  - 2000 Datastep code jobs
  - 1000 EG projects
- What is the typical approach of migrating this to SAS Viya / Snowflake and what is the implementation cost of migrating SAS versus rebuild with an alternative tool like dbt?
- The example showcase typical target architecture, migration method and cost

Example architecture

SAS Viya & Snowflake

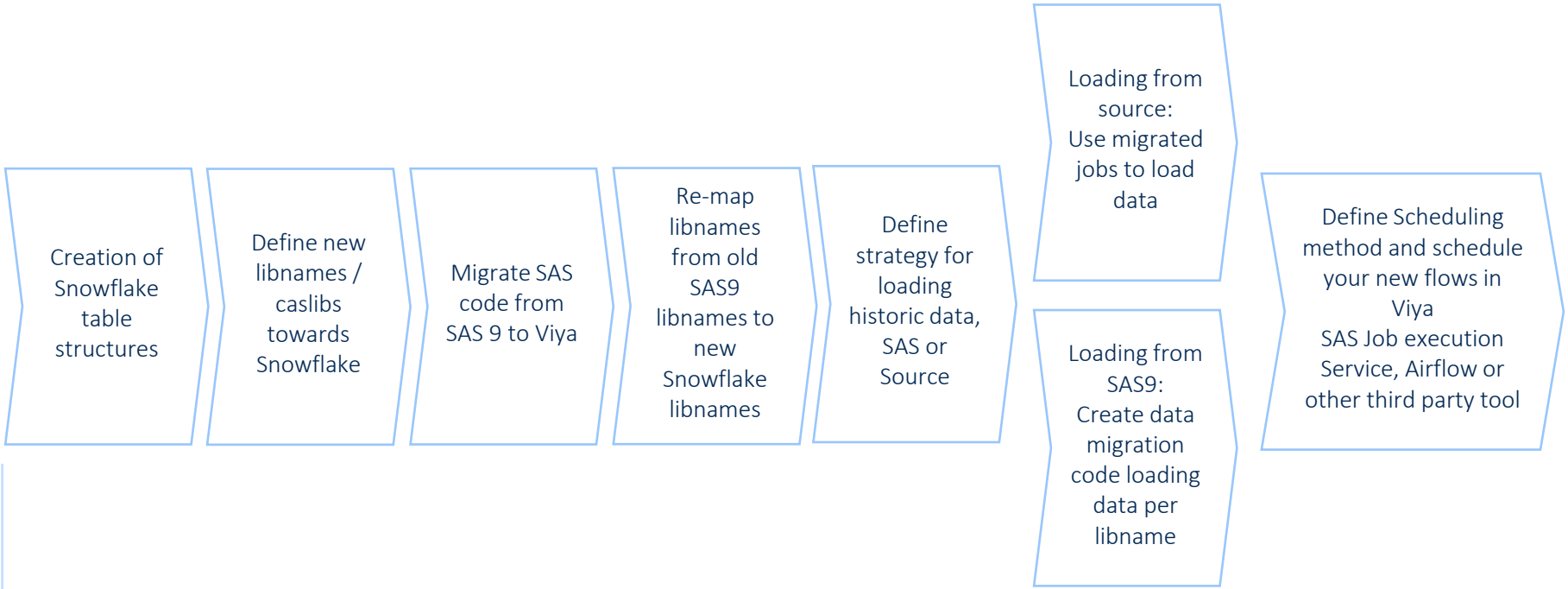
- CAS uses Snowflake as data storage
- Tables lifted into CAS in parallell using multi-node loading with bulkloading
- Tables written back using bulk unloading to Snowflake



- All SAS DI or EG jobs are migrated to SAS Studio jobs
- Execution is done on SAS compute
- Data reading /writing to Snowflake is done using SAS Access to Snowflake with bulk loading/unloading

Migration steps

Key elements to consider for a sucessfull migration



Project cost example calculation

Comparing SAS Migration vs rebuild using another tool like dbt

The calculations use the following assumptions:

1. Not all existing code is used, the migration scope is set to 2000 (of 3000) DI jobs, 1000 (of 2000) datastep jobs and 500 (of 1000) EG jobs
2. The total number of tables in use is 5000
3. Uses same method of automated DDL generation
4. Cost per hour 1500 NOK

A) SAS Migration

Asumes a refactor percentage of 10% of DI jobs, 10% needs to reimplement one step that is not supported yet.-> 5 hours per job

Calculation:

1. DDL creation (automated) –	2500 hours
2. Automated migration 1 hour pr job: 3300 * 1:	3300 hours
3. Refactoring: 200 * 5:	1000 hours
4. Testing	3500 hours
5. Project Management	1000 hours
Total	**10 300 hours**

Estimated cost: 10 300 * 1500 = 15 450 000 NOK

B) Rebuild

Implement one job – 25 hours – this is probably to small
Testing and PM gets bigger due to the size and length of the project

Calculation:

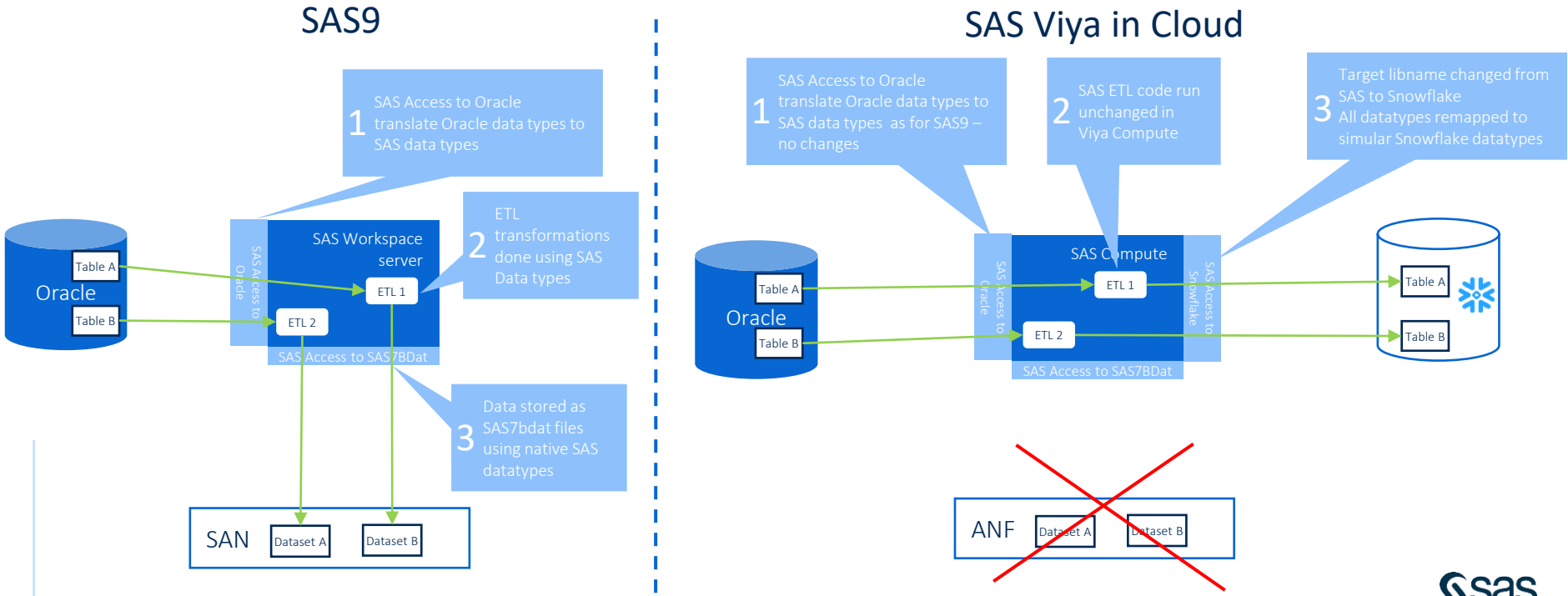
1) DDL creation (automated) –	2500 hours
2) Job implementation 3500 jobs * 25:	87 500 hours
3) Testing	10 000 hours
4) Project Management	5 000 hours
Total	**105 000 hours**

Estimated cost: 105 000 * 1500 = 157 500 000 NOK

Rebuilding from scratch becomes more than 10 times as expensive

SAS connectivity -> fueling cloud transisiton

SAS libnames makes your SAS code database agnostic



Handling Data Type transition

Let SAS Decide or control it yourself

A) Using the default data type conversion *1

Snowflake Data Type	SAS Data Type	Default SAS Format
ARRAY	character	$w., where w is the minimum of 32767 and the value of the DBMAX_TEXT= opt
BINARY(n)	character	$HEXw.
CHAR, CHARACTER	character	synonymous with VARCHAR except that the default length is VARCHAR(1)
OBJECT	character	$w., where w is the minimum of 32767 and the value of the DBMAX_TEXT= opt
STRING	character	synonymous with VARCHAR
TEXT	character	synonymous with VARCHAR
VARCHAR(n)	character	$w.
VARIANT	character	$w., where w is the minimum of 32767 and the value of the DBMAX_TEXT= opt
BIGINT	numeric	synonymous with NUMBER
BOOLEAN	numeric	1.
DECIMAL	numeric	synonymous with NUMBER
DOUBLE	numeric	synonymous with FLOAT
DOUBLE PRECISION	numeric	synonymous with FLOAT
FLOAT	numeric	none Snowflake uses double-precision (64-bit) IEEE 754 floating point numbers.
FLOAT4	numeric	none Snowflake uses double-precision (64-bit) IEEE 754 floating point numbers.
FLOAT8	numeric	none Snowflake uses double-precision (64-bit) IEEE 754 floating point numbers.
INT	numeric	synonymous with NUMBER
INTEGER	numeric	synonymous with NUMBER
NUMBER(p,s)	numeric	w.d If the precision or scale does not fit into SAS w.d format, use the default format.
NUMERIC	numeric	synonymous with NUMBER
REAL	numeric	synonymous with FLOAT
SMALLINT	numeric	synonymous with NUMBER
TINYINT	numeric	synonymous with NUMBER
DATE	numeric	DATE9.
DATETIME	numeric	DATETIMEw.d, where w and d depend on precision Alias for TIMESTAMP_NTZ.
TIME	numeric	TIME8.
TIMESTAMP	numeric	DATETIMEw.d, where w and d depend on precision*

B) Control it using DDL statements

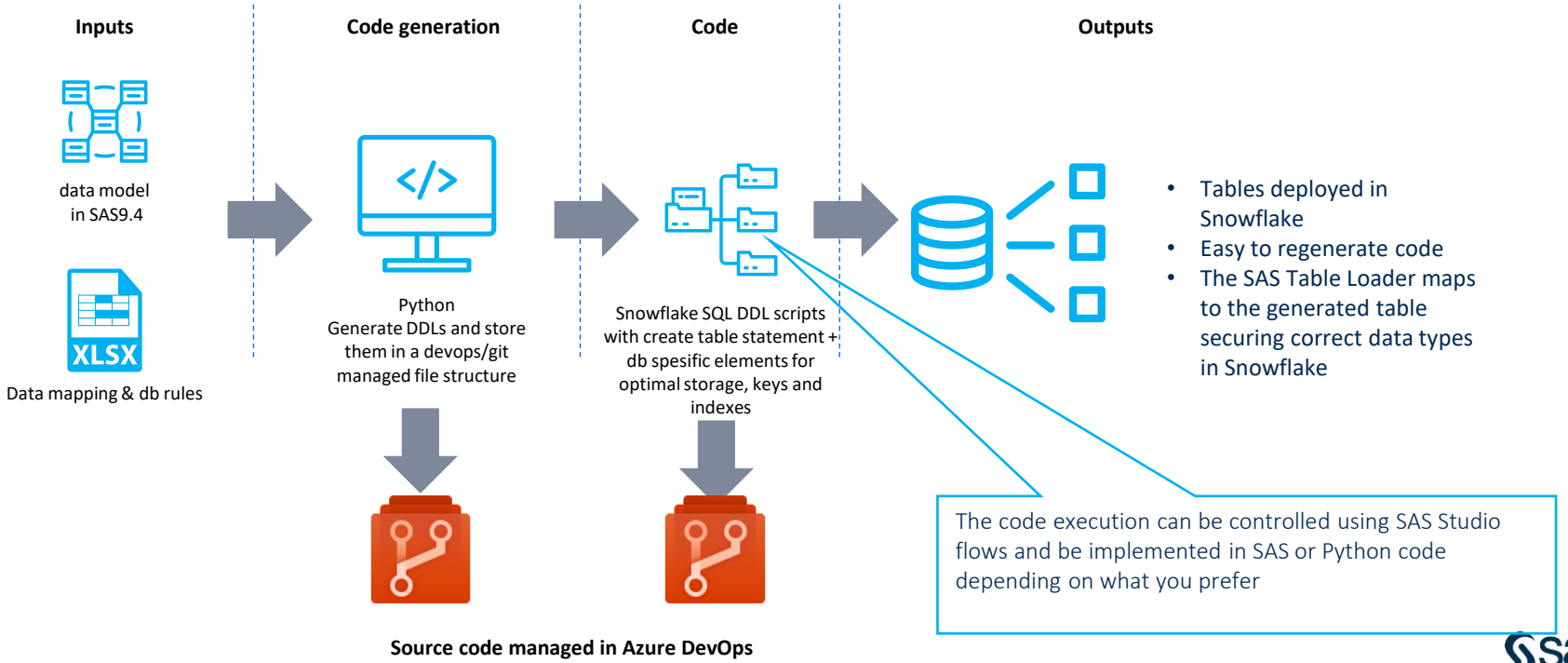
This can be done im multiple different ways:

1) Pre-generate tables using create table statements
2) Create tables using the create table object (May)
3) Create table using the DDL option in Table Loader

*1: https://go.documentation.sas.com/doc/en/pgmsascdc/v_038/acreldb/n0qnx5bs1vcs3in0zazami3levju.htm

Autogeneration of DDLs through Snowflake Rest API

Based on SAS9 metadata and Snowflake spesific rules



Roadmap

Connectivity Domain

Released

2022.11

SAS/ACCESS-Data Connectors

- Hadoop - Support for Azure HDInsight 5
- Hadoop - Support for Google DataProc
- Spark - Support for HDInsight 5 Spark

2022.12

SAS/ACCESS-Data Connectors

- Compute - Parquet - Support for Temporal Logical Types

2023.01

SAS/ACCESS-Data Connectors

- Compute - Parquet - Support for String Logical Types

2023.02

SAS/ACCESS-Data Connectors

- Compute - Parquet - Support for Numeric Logical Types

2023.03

SAS/ACCESS-Data Connectors

- SAP IQ - SAS/ACCESS & Data Connector
- Informix - SAS/ACCESS & Data Connector
- SingleStore - SAS/ACCESS & Data Connector
- Add DBTYPE to selected Data Connectors
- Google Big Query - Performance Improvements
- Compute - Parquet files - Embedded Logical Type Support

Hybrid Cloud Management

- SAS Cloud Data Exchange on Viya 4

2023.04

In-Database technologies

- In-DB Cloud Hadoop Services - HDInsight

Connectivity Domain

Now

Data Connectors & Open File Formats

SAS 9 M8

- Reintroduce Support for Access/Teradata on ZOS

2023.06

- SAP IQ - Bulk Load support
- CAS - Update Apache Arrow Library

2023.07

- S/A&DC Teradata - Implement SSO
- CosmosDB - SQL API

2023.08

- Parquet files support for partitioned data (Compute)

Hybrid Cloud Management

2023.05

- Research Publish CDE API
- Common Connectivity Dialog (C3) Release 1 (Internal)

2023.06

- ODBC driver for CDE - Enable 3rd party access

Next

Data Connectors & Open File Formats

23Q2

- Access - Performance reviews
- Spark - Databricks AAD Support
- Spark - Dataproc support
- SingleStore - Performance Enhancements

23Q3

- Google BigQuery - BIGNUMERIC type Support
- Vertica - Include ODBC Client
- SingleStore - Performance Enhancements

23Q4

- Compute - Parquet Files - Support for ADLS2
- CAS - Parquet – Add. Supported Features
- Oracle Wallet Support
- Oracle - Access - Bulk Unload

Hybrid Cloud Management

- CDE - Additional Access Engines
- CDE - Support for multiple Collocated Data Agents
- CDE Publish REST API
- Common Connectivity Dialog (C3) Release 2

Future

Data Connectors & Open File Formats

- General Performance Improvements based on Reviews
- Oracle - DC - Bulk Unload
- Teradata - Ship with Client
- MongoDB - Ship with Client
- Oracle - Ship with Instant Client
- Snowflake - Additional Features
- SalesForce - Support for Update/Delete
- CosmosDB - MongoDB API
- Performance Benchmarking

Hybrid Cloud Management

- CDE - UI
- Common Connectivity Dialog (C3) Release 3

Connectivity Domain - In-Database Technologies

Now

In-Database technologies

2023.05

- In-DB Teradata - Move EP to Viya 4 pipeline

Next

In-Database technologies

23Q2

- Parallel Load Azure Synapse Analytics
- Parallel Load Databricks
- Investigate potential options for In-DB functionality in Snowflake
- In-DB Synapse - Synapse AAD Support
- Review Moving EP to new Universal EP Design

23Q3

- Testing Tasks - Included CData Drivers - Hive/Spark/Databricks

23Q4

- Spark - Move to New Universal EP Design
- Cloud Hadoop Services - Support AWS EMR 5/6, Google DataProc

Future

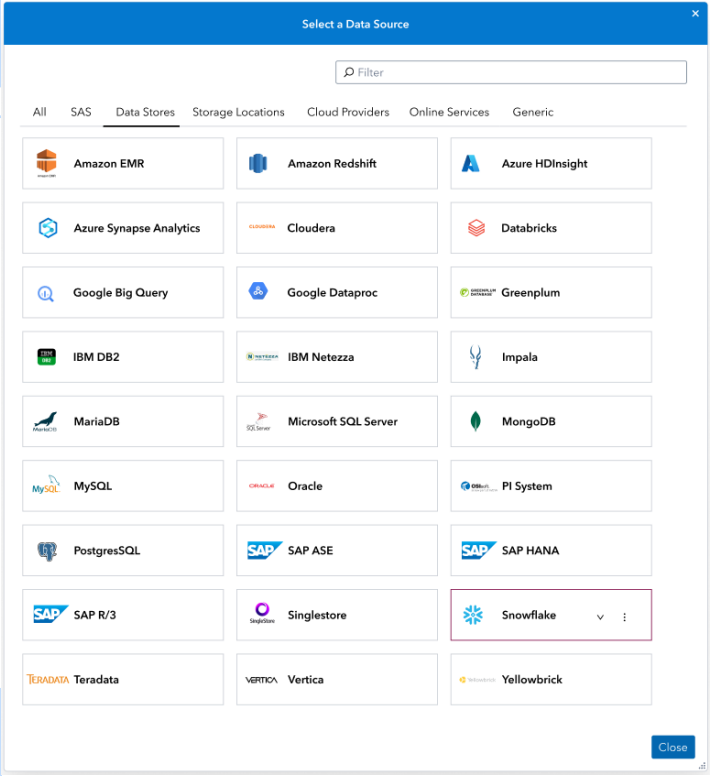
In-Database technologies

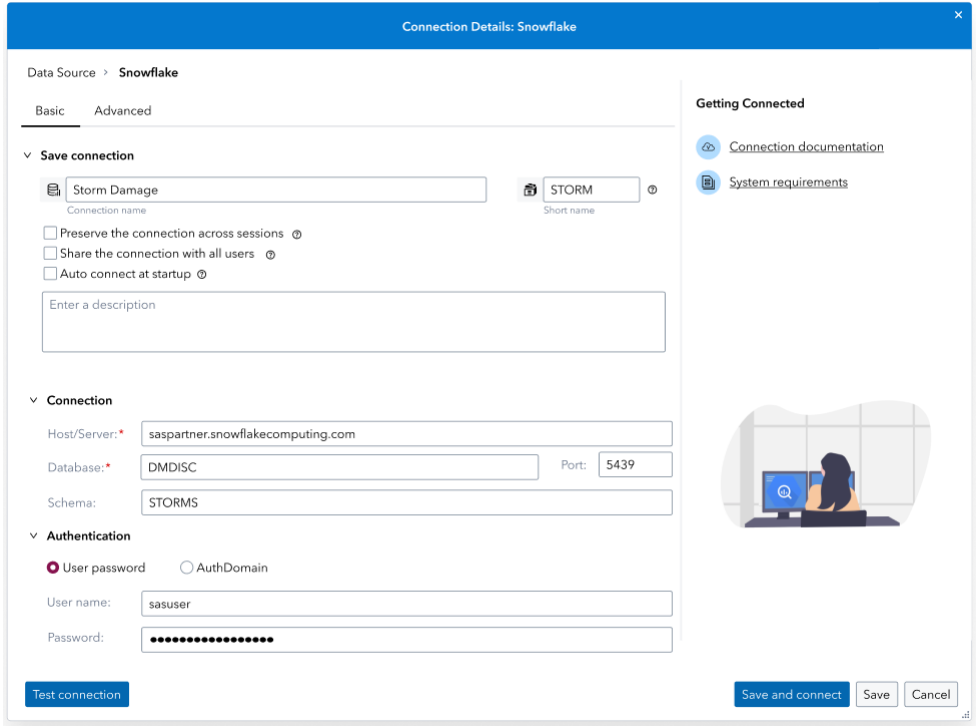
23Q4

- In-DB Snowflake - Scoring Accelerator/Parallel Load

Future

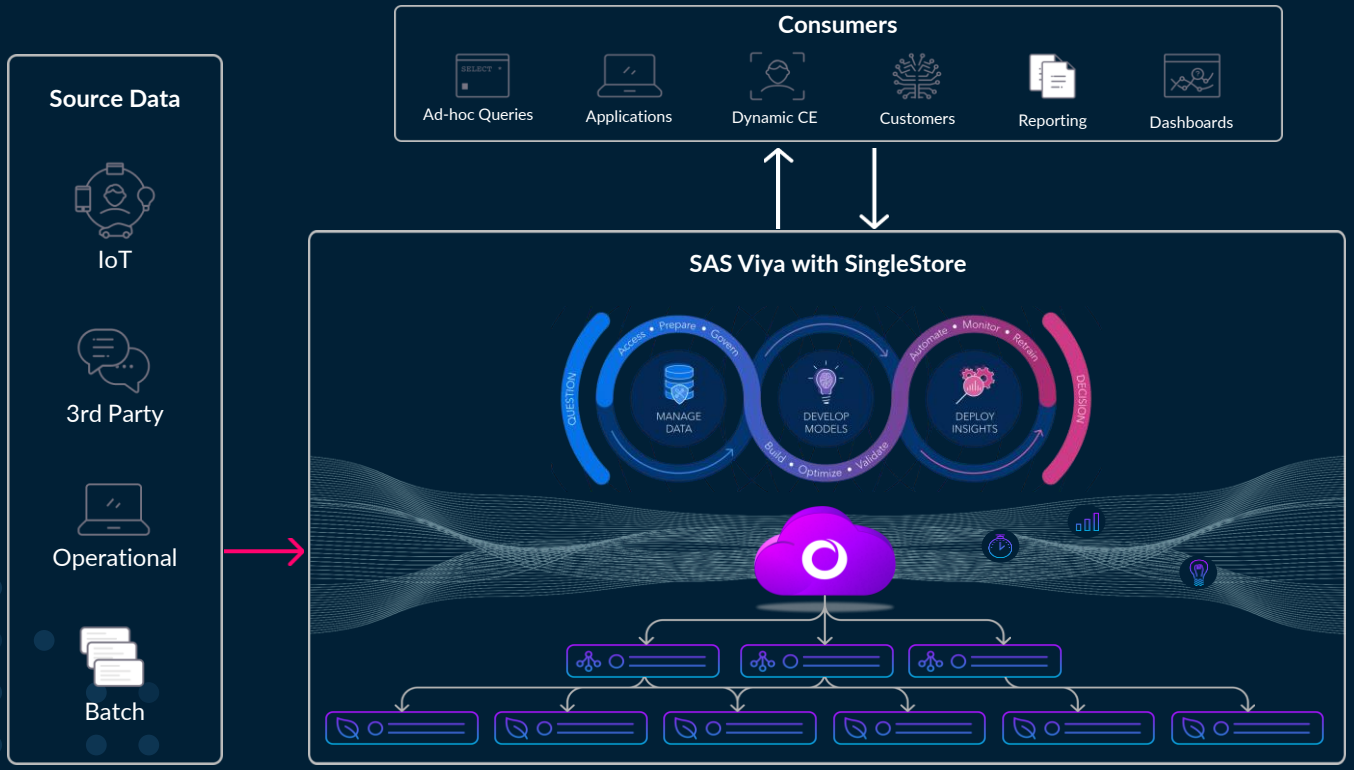
- Kerberos Support CDP7.2 Public Cloud & CDP7.1 Private
- Code & DQ Accelerators - All -In-DB Products
- Support for DataStep in Embedded Process





SAS Viya with Singlestore update

SAS Viya with SingleStore Approach

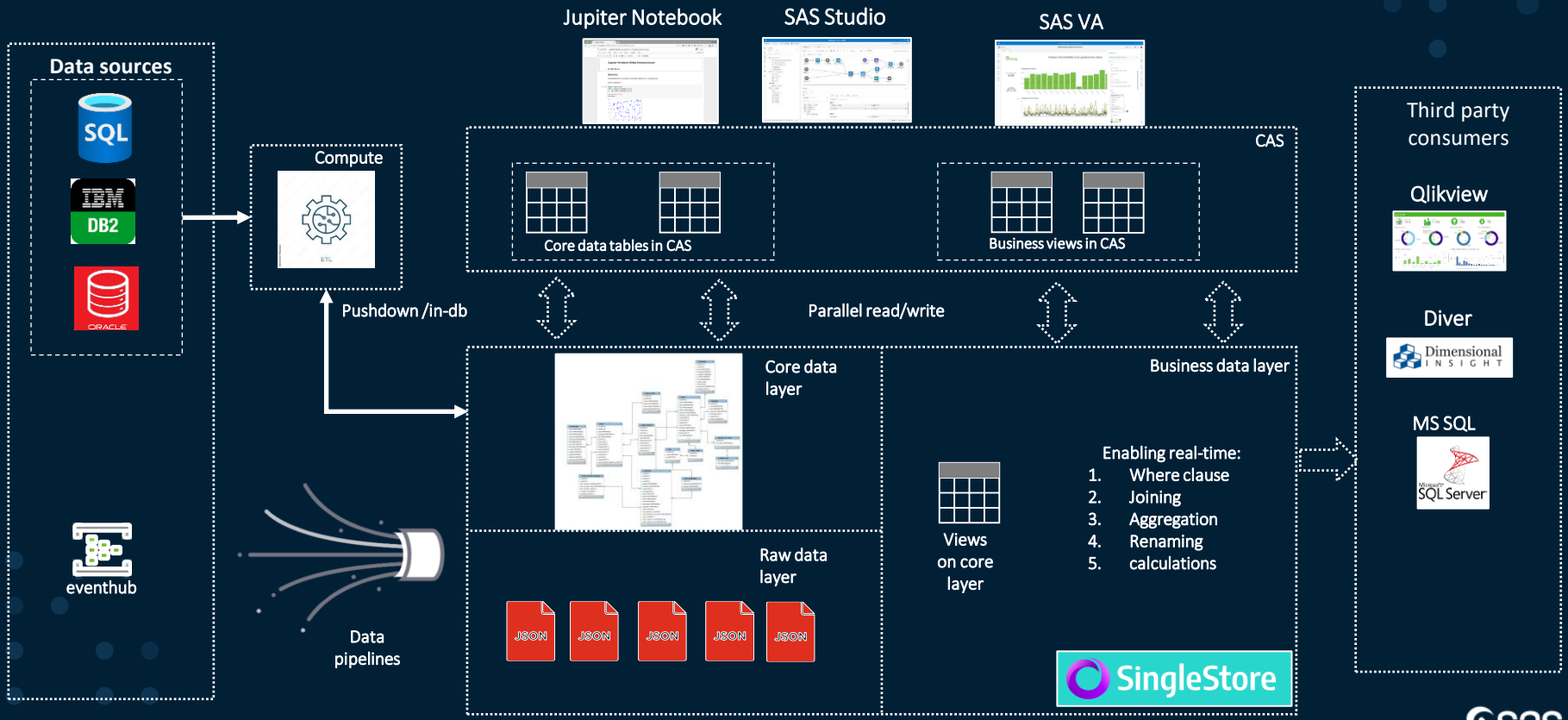


Consumers now have a single source of truth and easy to access curated enriched analytically driven data

Parallel read and writes horizontally scaled across an industry leading MPP data engine provides world class performance and compression

Engineered integrated solution to remove barriers to data access and minimizing data movement and cost across the analytic data landscape

Customer example



SAS Viya with SingleStore

Multiple Options for the Customer