Flytt til SAS Viya og Snowflake på skyen med CICD

Bruk av CICD og SAS sin Snowflake aksessmodul for å få SAS DI jobber, EG prosjekter og SAS Datasett raskest mulig over til SAS Viya og Snowflake

Og se raskt på hva man kan gjøre med kombinasjonen av SAS, CAS og Snowflake

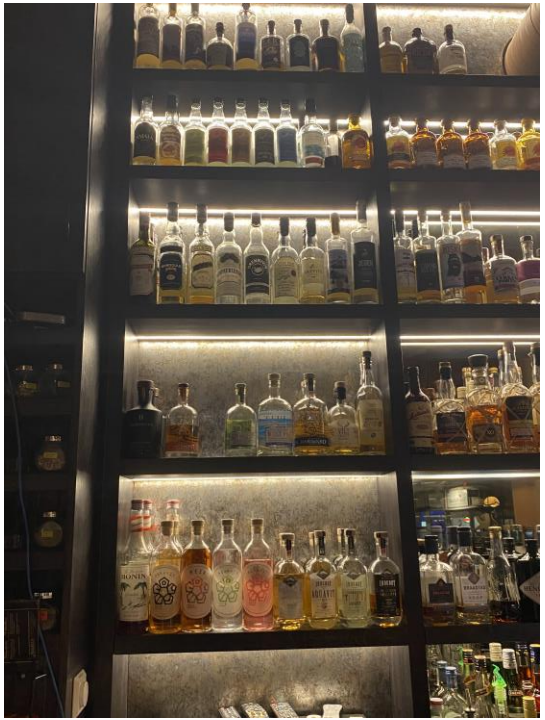
Demo – still gjerne spørsmål, del egne behov og ambisjoner

Lars Skaar – bryr seg om data og devops med CICD

Hvordan få med seg viktige trender og hva som skjer innenfor dataverden?

Gode samtaler hjelper:

- Etter FANS i fjor på en bar med imponerende utvalg av aquavit og gin – diskuterte vi bekymring over at SAS skulle erstattes med Snowflake
- På julesamling i fjor – Snowflake er standard datavarehus globalt – hva skjer med SAS?



FANS

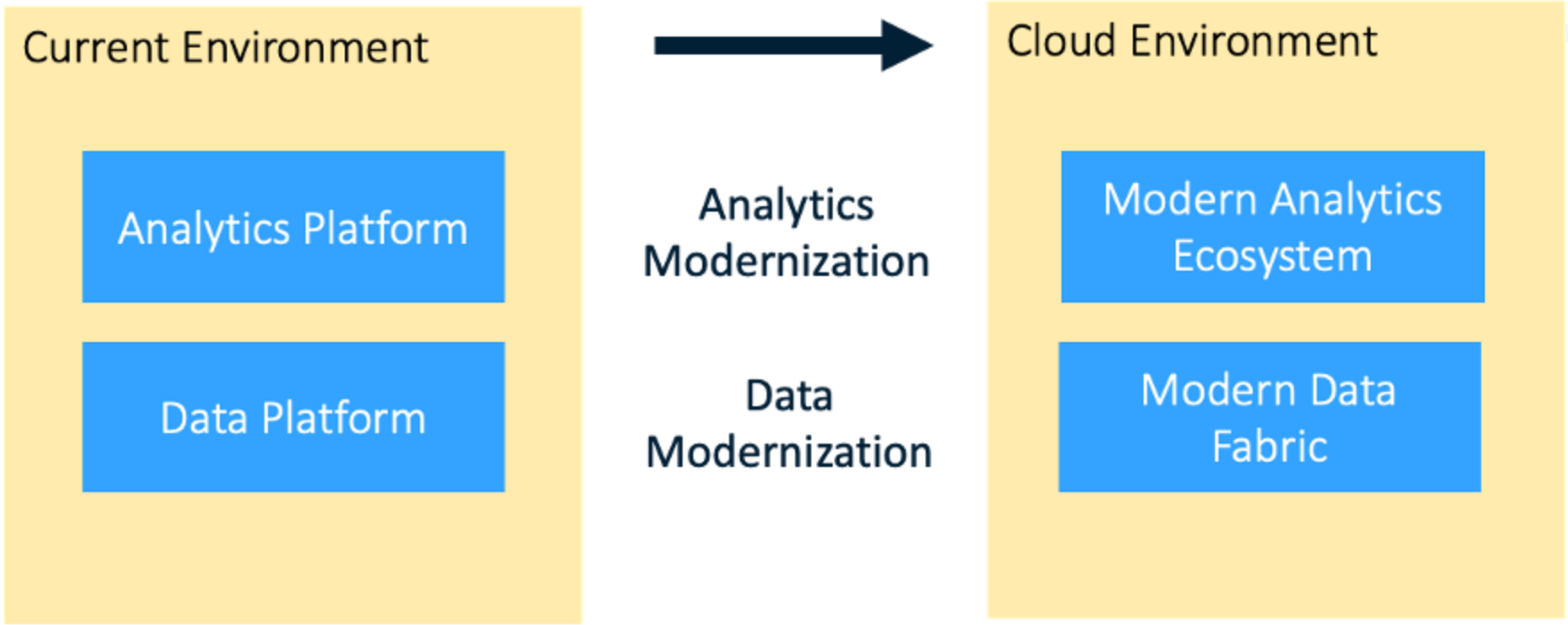
Viktig å spre:

- SAS og Snowflake har mer enn 400 felles kunder og samarbeider aktivt
- Det er både enkelt og veldig effektivt å bruke SAS med Snowflake
- Du kan gjenbruke det du har og oppnå effektiv bruk av både SAS og Snowflake
- Flytting til Snowflake med SAS til skyen er sannsynligvis billigste og raskeste vei

snowflake

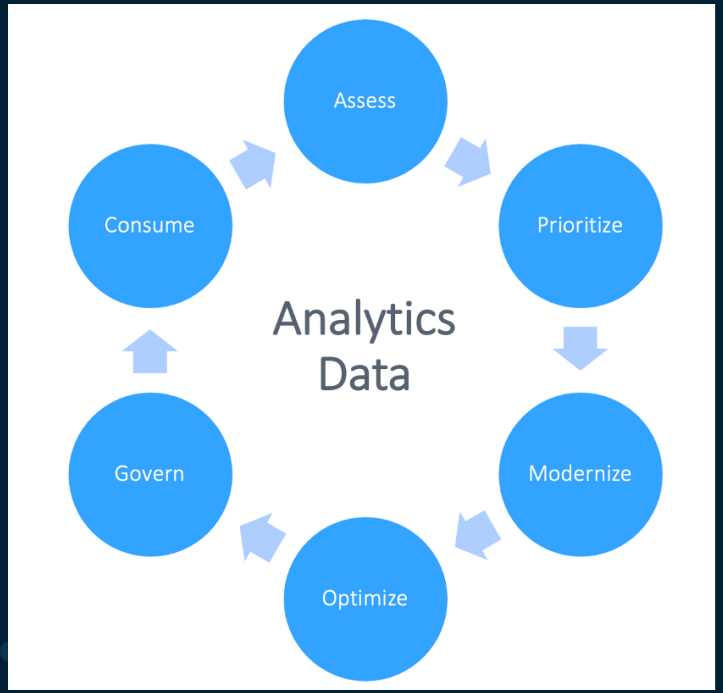
sas

Moving to Cloud



Jeff Stander's data modernization methodology

High level visual – of our internal methodology of approaching data modernization



Data trends in fashion

Data mesh, lakehouse, data fabric,

what is a «modern data platform»

What is Data Mesh really?

Wikipedia:

"Data mesh is a sociotechnical approach to building a decentralized data architecture by leveraging a domain-oriented, self-serve design (in a software development perspective), and borrows Eric Evans' theory of domain-driven design and Manuel Pais' and Matthew Skelton's theory of team topologies."

A way to organize and work with data to increase productivity

Inspired by domain driven design and product / domain based organization of work (agile transformation towards product/domain ownership)

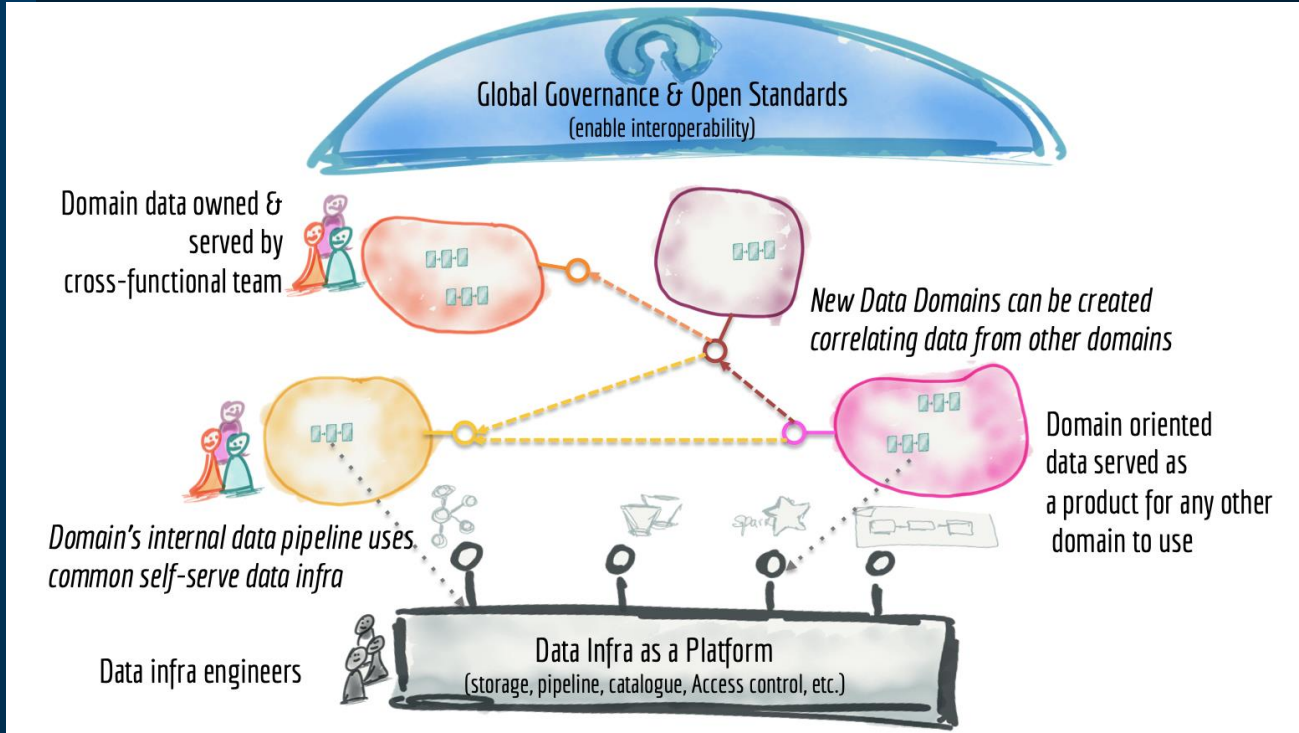
Great optimism in advantages to be gained by Data Mesh – similar to agile transformation with SW

Works well with a common data and analytics platform / infrastructure in the cloud – like Snowflake and SAS Viya

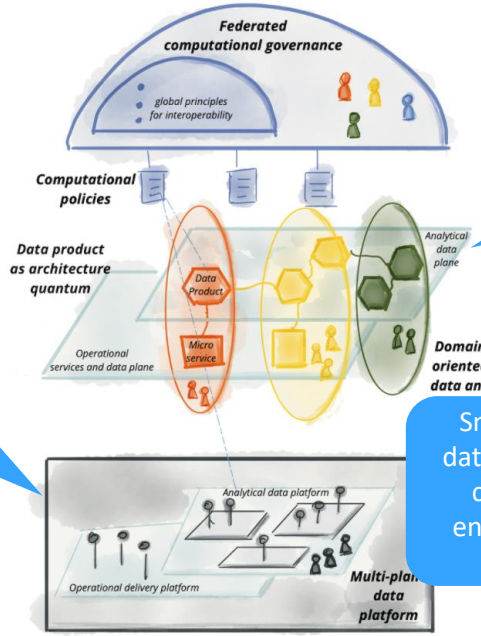
Data Mesh – silver bullet for data productivity?

Source - https://martinfowler.com/articles/data-monolith-to-mesh.html

Author: Zahmak Dehghani



So what do data mesh, data lakehouse and data fabric mean to SAS and Snowflake?



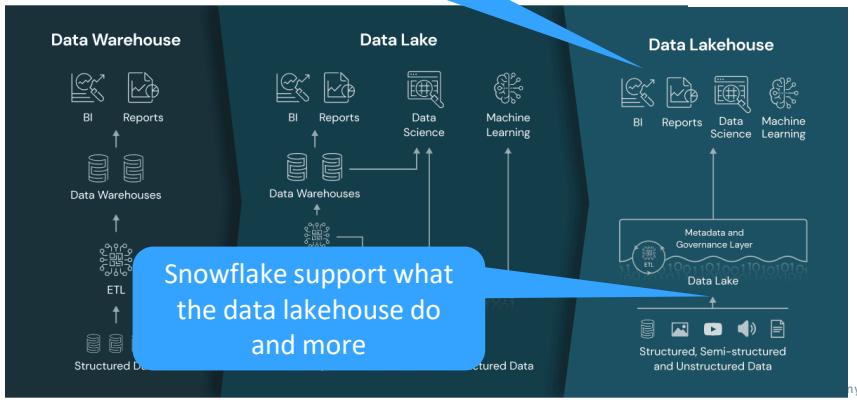
Data Mesh in practice needs to deal with culture and organization – AND relies on common data services to function – ideally on the cloud

SAS and Snowflake can enable a data mesh «culture»

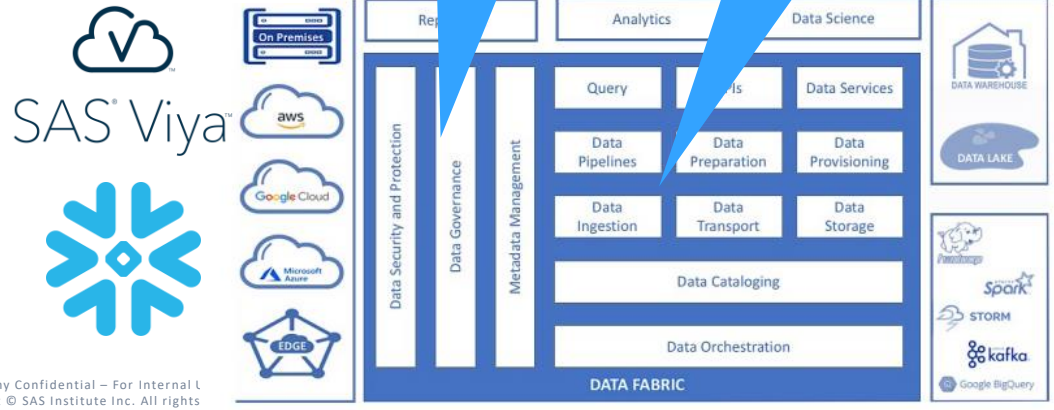
SAS Viya can work with most data sources – so we effectively work well with a data lakehouse

Snowflake supports data fabric capabilities close to storage – enabling governance and security

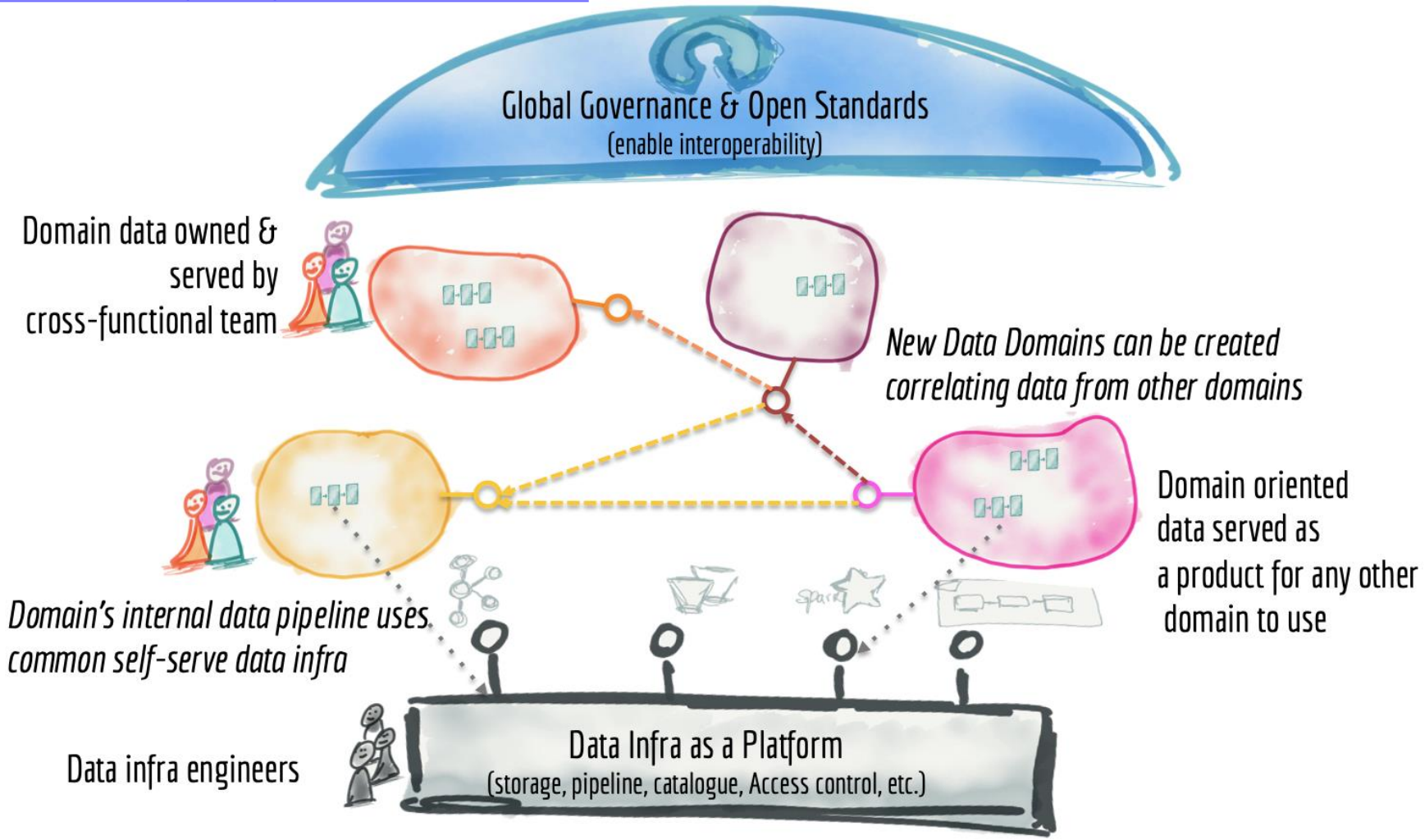
SAS Viya already supports most data fabric capabilities and have the ambition to support all



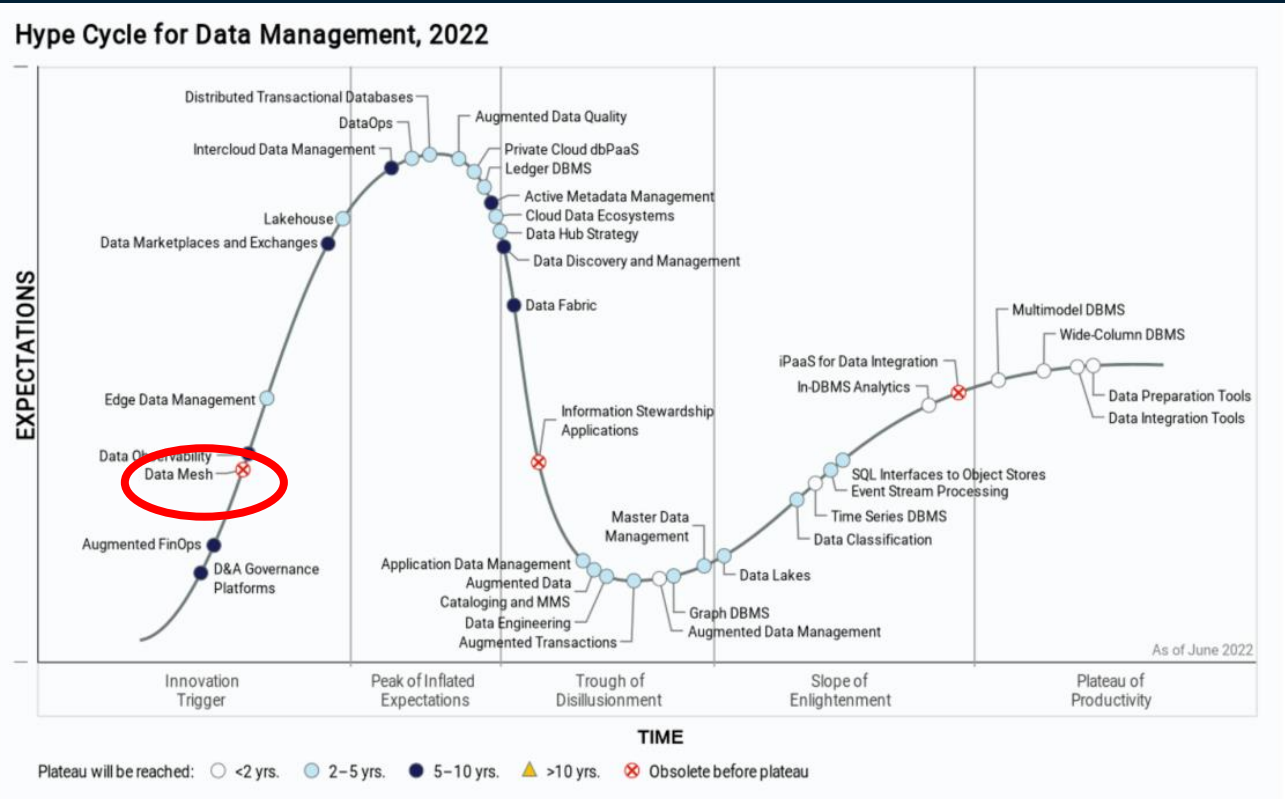
Snowflake support what the data lakehouse do and more



https://martinfowler.com/articles/data-monolith-to-mesh.html



Data mesh is dead – just doesnt know it yet…



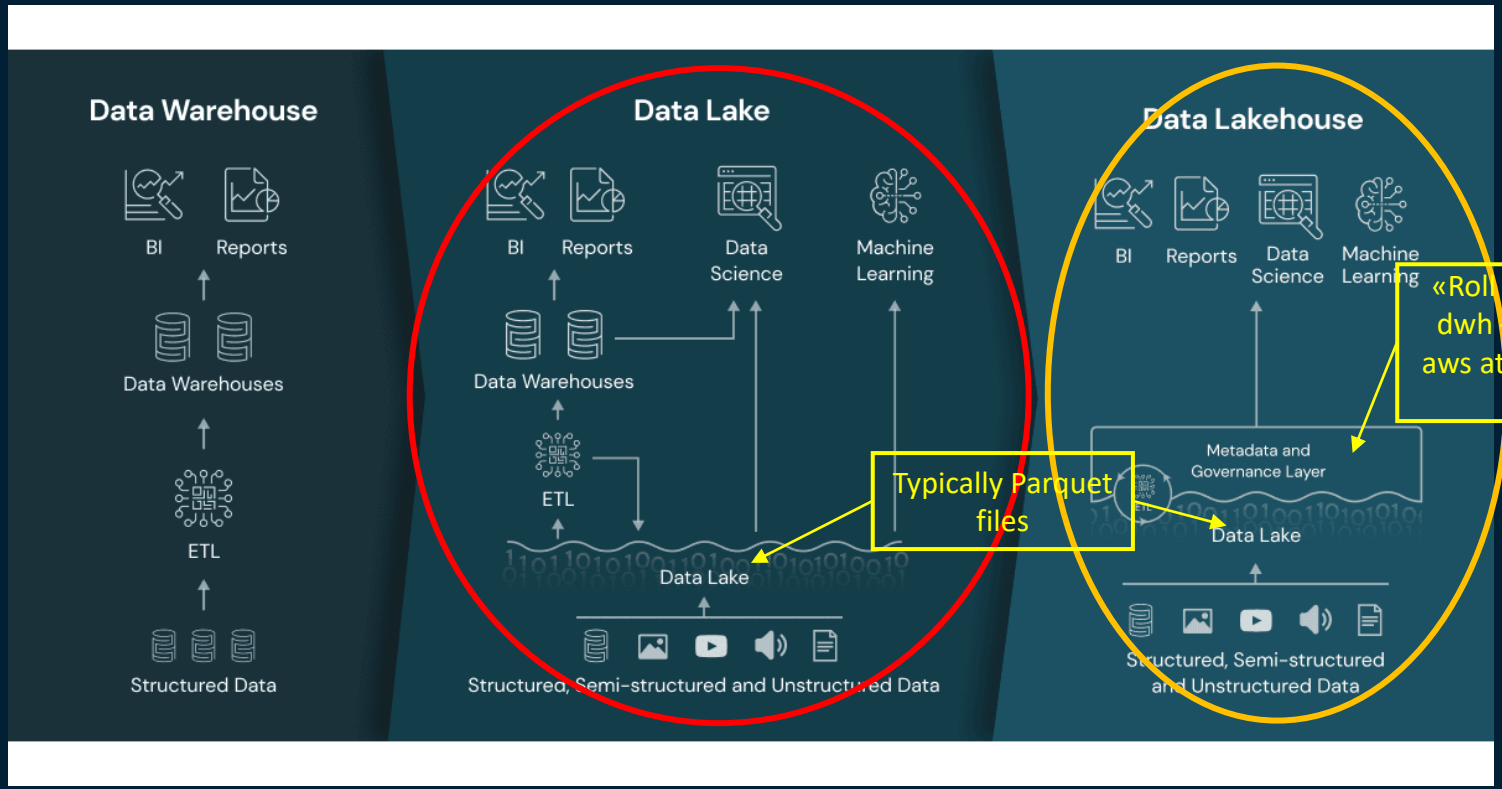
Who uses Data Mesh?

Quite a few actually ☺

- JP Morgan Chase (Wholesale Credit Risk) A1 V1 V2
- HSBC A1
- ABN Amro A1 A2
- Fifth Third Bank
- Cox Automotive
- Intuit V1 A1 V2
- Disney V1
- Michelin A1
- AutoZone V1 A1 S1
- Adidas A1 A2 A3 A4
- Department for Work and Pensions V1 A1
- Fannie Mae A1
- PayPal A1
- Roche V1 V2
- Post Italiane A1
- RTL Deutschland A1
- Saxo Bank V1 V2 A1 A2 A3
- United States Army A1
- US Dept of Veterans Affairs A1
- Source (even more found here): (https://datameshlearning.com/user-stories/)

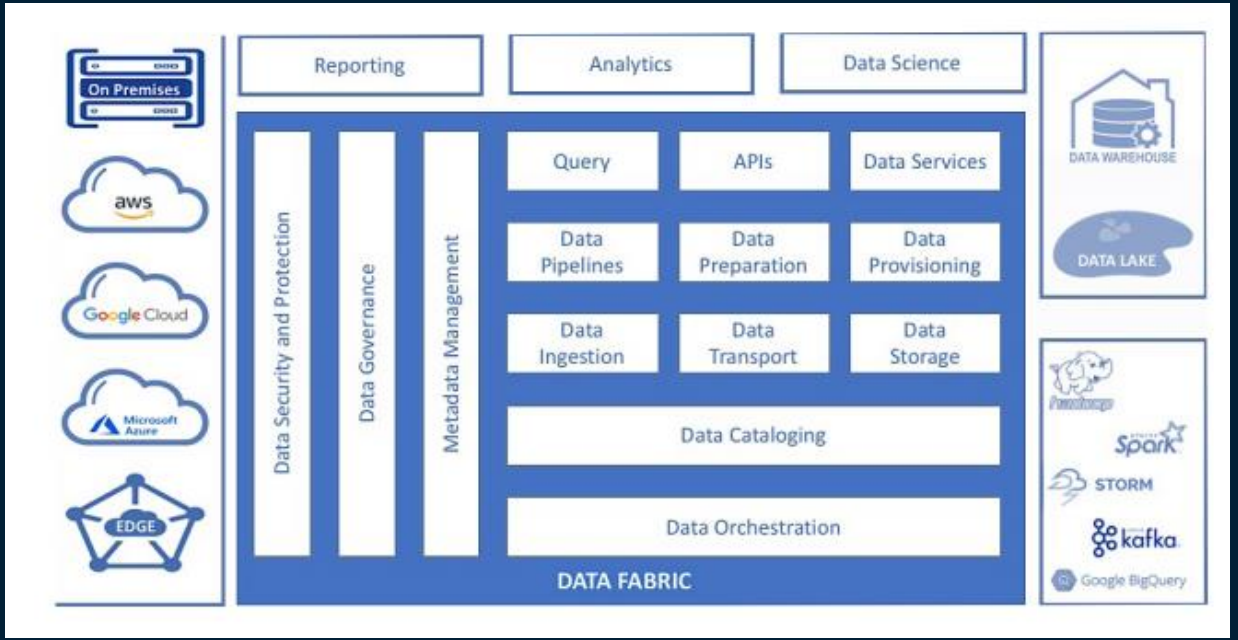
What is a data lakehouse?

https://www.databricks.com/blog/2020/01/30/what-is-a-data-lakehouse.html

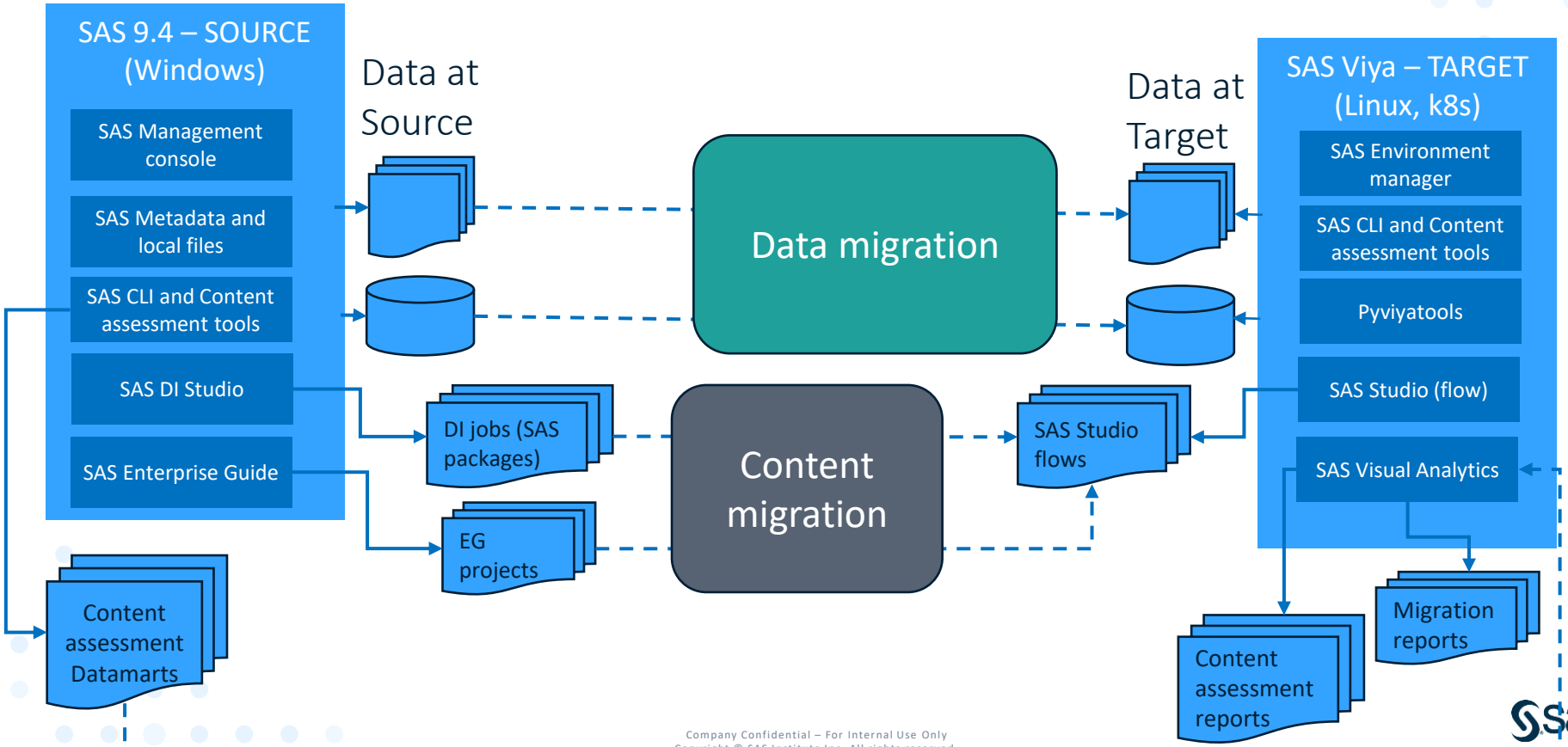


Data fabric

SAS Viya have many of these capabilities already and they are continuously evolving



Migration from SAS 9.4 to SAS Viya – what and how, what do we need?



Why automate and what to automate?

Good reasons at your place?

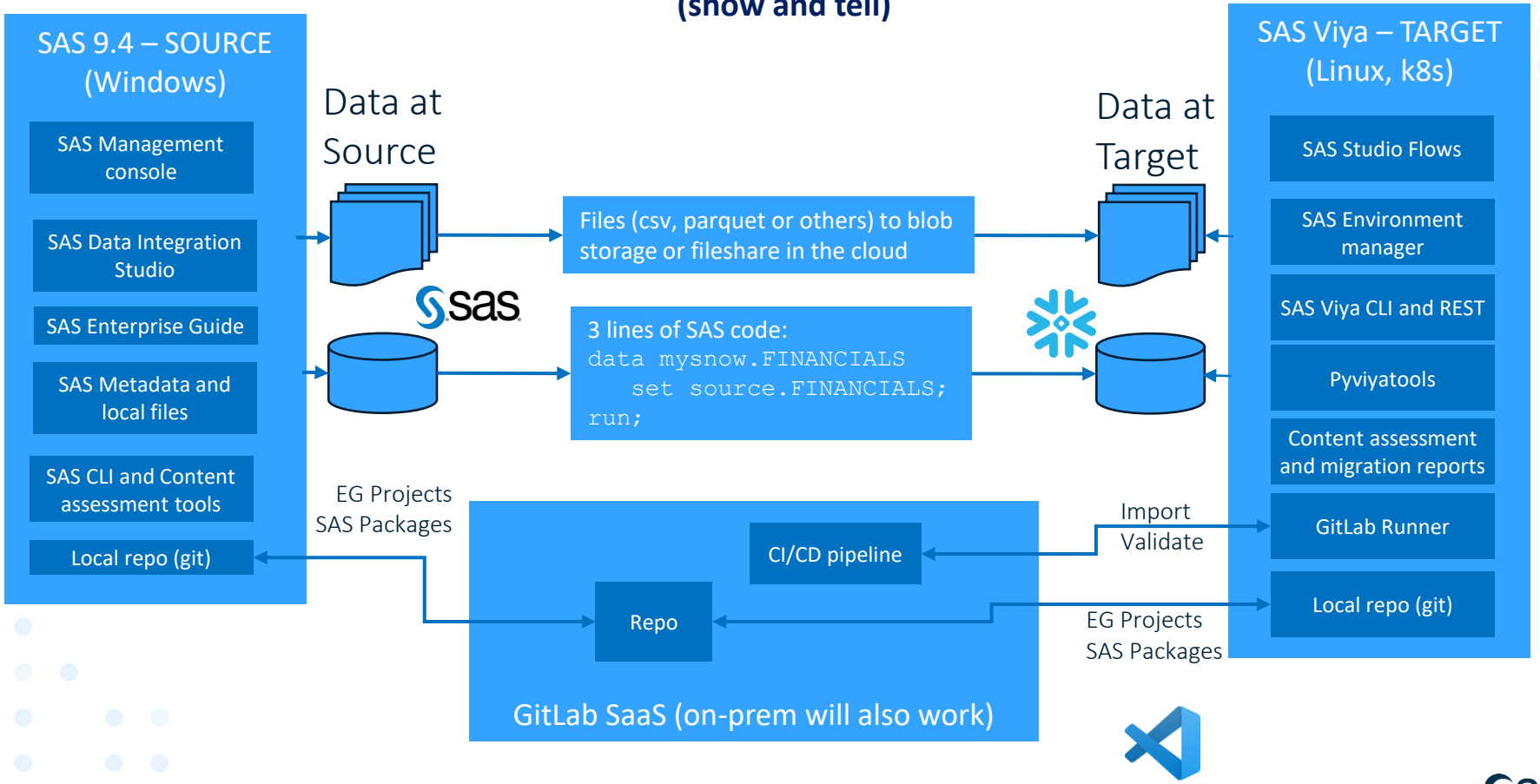
Why?

- Save time and effort
- Accuracy
- Validation
- It is possible

What to automate?

- Stuff we are likely to do often
- Stuff we want to be done exactly the same way each time
- Report / visualize outcome
- Assessing outcome of validation (automate decision making)
- Stuff that is easy / possible to automate

Migration of data and SAS Content using CICD to automate (show and tell)



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

Connecting to Snowflake

Libname and CASLib options

Connecting to Snowflake

Compute – SAS Studio

Libname statement :

```
libname mylib snow
server="<account>.snowflakecomputing.com"
user=myusr1 password=mypwd1 warehouse=compute_wh
database= robin schema=robin role=user  ;
```

Note: The USER= and PASSWORD= options remain valid. Better security practice, it is recommended that you use the AUTHDOMAIN= option instead

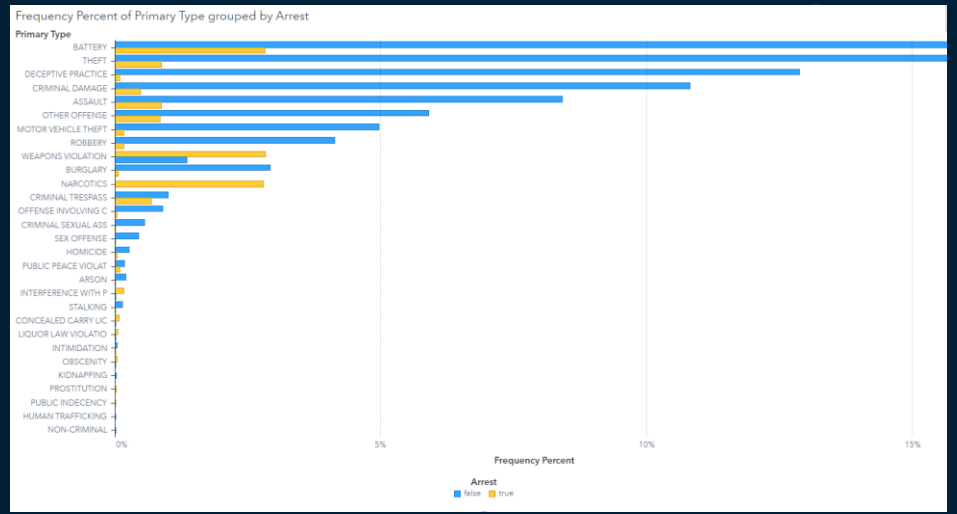
Include the READBUFF=32000, INSERTBUFF=3200 and DBCOMMIT = 0 options on every Snowflake Libname you create. Feel free to experiment with the first two, but always set the DBCOMMIT=0.

DBCOMMIT=0 - new rows are committed at the end of the insert operation

Connecting to Snowflake

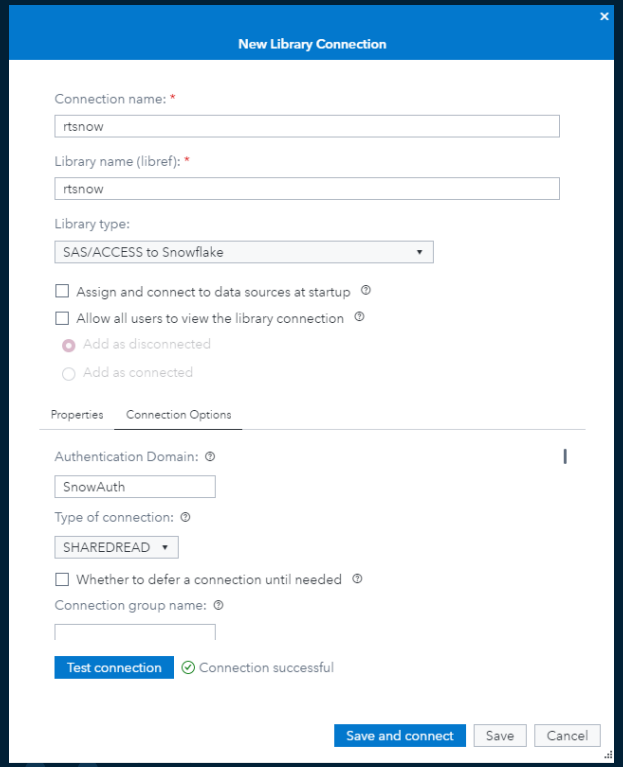
Loading data to CAS

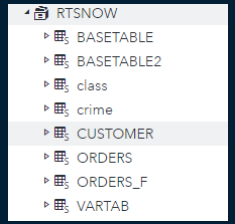
```
cas mysession ;

caslib snow drop ;
caslib snow dataSource=(srctype="snowflake",
                        server="[redacted].snowflakecomputing.com",
                        authdomain="SnowAuth",
                        database="ROBIN",
                        schema="ROBIN",
                        readbuff=2048,
                        insertbuff=32767,
                        scanStringColumns=true) global libref=cassnow ;

proc casutil incaslib="snow" ;
    list files ;
quit ;

proc casutil incaslib="snow" outcaslib="snow" ;
    load casdata="crime" casout="crime" promote;
quit ;
```



SAS & Snowflake

Use UI to create connections





	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5
2	Alice	F	13	56.5	84
3	Barbara	F	13	65.3	98
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84
10	John	M	12	59	99.5
11	Joyce	F	11	51.3	50.5
12	Judy	F	14	64.3	90
13	Louise	F	12	56.3	77
14	Mary	F	15	66.5	112
15	Philip	M	16	72	150
16	Robert	M	12	64.8	128
17	Ronald	M	15	67	133
18	Thomas	M	11	57.5	85
19	William	M	15	66.5	112

Connecting to Snowflake

Authentication options

- Username & Password
- SAS Viya AuthDomains
- Single Sign On via Azure Active Directory
- Use of conopts option in libname to support other methods (not documented)
  - Key Pairs

In Database Processing

Pushing processing to the data

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

Implicit SQL Passthrough

SAS Studio Code

```
proc sql;
   select avg(age)
   from csnow.class
   where upcase(sex) = 'F';
quit;
```

AVG(TXT_1."AGE")
13.22222

Use Options statements to check what is happening….

```
Options sastrace=',,,d' sastraceloc=saslog nostsuffix sql_ip_trace=note;
```

```
80   proc sql;
81   select avg(age)
82   from csnow.class
83   where upcase(sex)='F';
SQL_IP_TRACE: The SELECT statement was passed to the DBMS.
84    quit;
```

```
80   proc sql;
81   select upcase(sex),avg(age)
82   from csnow.class
83   group by upcase(sex)
84   ;
SQL_IP_TRACE: None of the SQL was directly passed to the DBMS.
NOTE: The query requires remerging summary statistics back with the original data.
```

Snowflake Query History

```
1  select AVG(TXT_1."Age") from "ROBIN"."class" TXT_1 where UPPER(TXT_1."Sex")
= 'F'
```

Passing joins to DB

Tip! Design your joins to avoid data movement

- Typically inner, left, right, full and cross joins are passed to the DB.

- If unsure, do a test with small data and validate using options sastrace=…!

In-Database Processing

SQL Passthrough – Supported Functions

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

Bulk Data Movement

Performance

Bulk Loading & Unloading

Bulk loading is the fastest way to insert large numbers of rows into a Snowflake table, or bulk unloading to retrieve from Snowflake into a SAS table.

The SAS/ACCESS engine handles bulk loading and bulk unloading through one of these methods.

- Snowflake **internal stage**: The engine uses PUT and GET commands to upload files to the internal stage. Bulk loading and unloading use the Snowflake COPY command to load staged files into or from a table.
- Amazon S3 bucket (**external stage**): The bulk-load facility uses the Amazon Simple Storage Service (S3) tool to move data to and from the client to the Snowflake database. Files are copied using S3 utilities.

Snowflake supports different types of stages.

- user stages: `@~` is the name of the stage of the current user.
- table stages: `@[<namespace>.]%<tablename>` is the name of the stage for a table.
- internally named stages: `@[<namespace>.]<internal_stage_names>` is the name of the internal stage.
- external stages: These are supported only through the Amazon S3 bucket.

Performance

Bulk Loading

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

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;
```

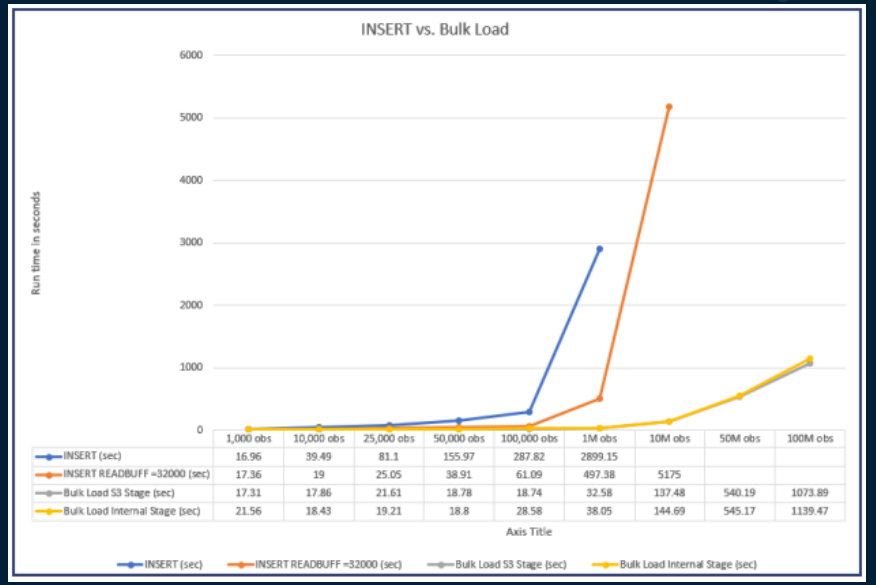
Snowflake Query History

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 Viya and Snowflake

Features & Value

Embedded connectivity

Bulk-loading

Integrated Security

In-Database SAS functions

Key findings so far

Moving to Snowflake and blob storage with SAS

- Snowflake access is solid and mature – works well from SAS 9.4 as well as SAS Viya
- Moving data from sas datasets to Snowflake takes 3 lines of code and with compression it will be fairly effective – moving directly from SAS 9.4 may be a practical approach
- When understanding data formats across – data movement is also stable
- Snowflake – as all other RDMBS behaves differently from SAS datasets
  - Dont assume data is sorted for instance – proc compare needs to get data sorted
- Migrating from EG and DI works well towards Snowflake – just change the lib to point to Snowflake and it works the same
- EG still has better migration support than DI – but seems to incrementally improve
- May want to consider mounting blob storage to cas and compute to simplify the coding – have the same mount to where it is being produced (a popular scenario is that CSV files is written from Excel and then processed in EG – do this with a mount and you can do the same in the cloud)

Snowflake WITH SAS

Key capabilities

- It just works – moving data between SAS and snowflake is easy and effective – one of the most productive tools to support this?
- EG and DI can already effectively be migrated to SAS Viya towards snowflake – certain steps in DI still gets migrated to SAS code
- SAS supports SQL push–down to Snowflake, so SQL will effectively take advantage of Snowflake
- We already have several integration steps that support Snowflake in SAS Studio Engineer
- Migrating DI and EG to SAS Viya with Snowflake is by far the most cost effective way of moving data and analytics from on-prem to the cloud