snowflake in partnership with Fivetran & Bytecode IO

ZERO TO SNOWFLAKE IN 90 MINUTES

To participate in the hands-on portion sign up for a free account.
Select Enterprise Edition:
https://trial.snowflake.com/

Charlie Killian, Bytecode IO, Snowflake Certified Consultant
https://bytecode.io/

Hands on sign up

- The majority of this workshop is participating in guided exercises

- Please let me know if you don't have an account
- Sign up for a free account ($400 worth of free credits for 30 days)
- Select Enterprise Edition (needed for feature demonstration)
  - https://trial.snowflake.com/

- Fivetran
  - https://fivetran.com/signup

- This presentation
  - https://tinyurl.com/zero-to-snowflake-201901

Agenda

- Introductions
  - Partner
  - Participants
- Snowflake Introduction
- Hands on Snowflake
  - Data loading
  - Scaling
  - Semi-structured data query
  - Multi-clustering
  - Caching
  - Time travel
- Walk-through
  - Looker connectivity
  - Data Sharing
  - Security

Bytecode IO + Introductions

BYTECODE IO

Expert Team of Data Consultants

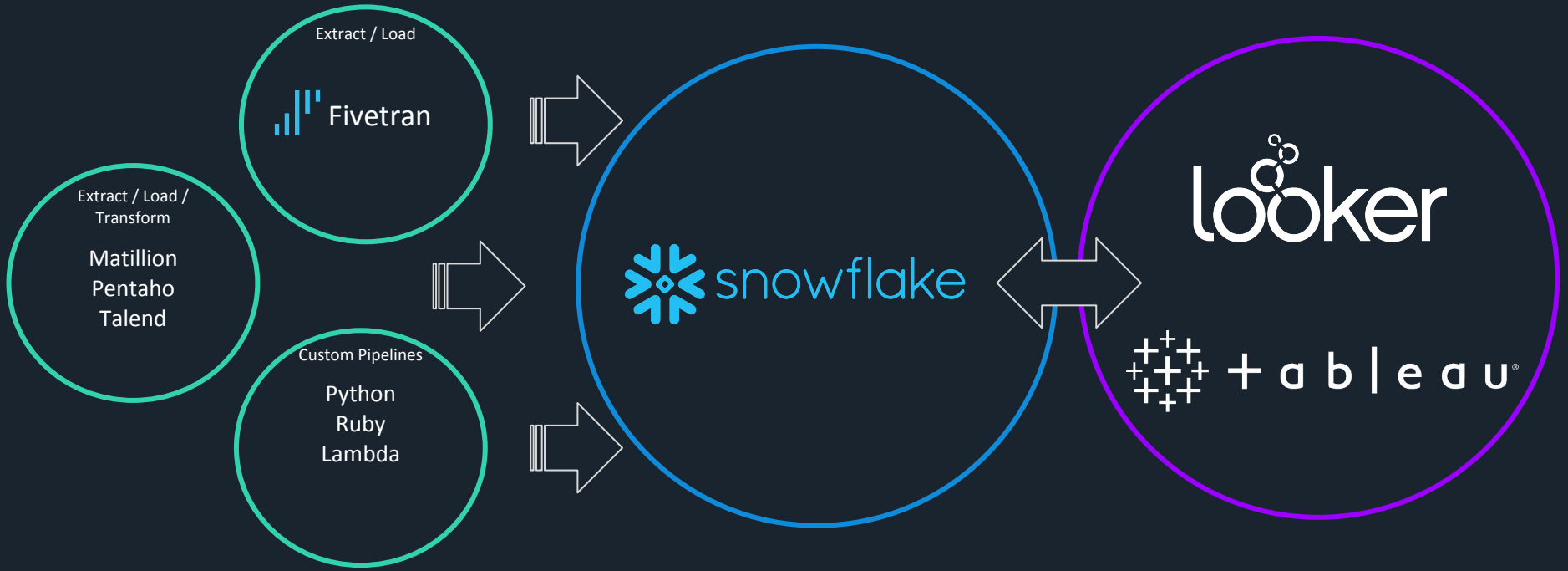
Solution Architecture
Implementing best in class technologies custom fit to you

Data Engineering
Real-time pipelines to data warehouse optimization

Snowflake Experts
Maximizing Snowflake value and performance

Business Intelligence
Design, develop and deploy breathtaking analytics products

Enabling the Technology Ecosystem



Experienced Snowflake Consultants

- Deployed and actively manage Snowflake for the last 3 years
- Mongo and 3rd party data sources
- Leveraging Snowflake's native JSON capabilities
- Running Looker for internal analytics on Snowflake

- Developed usage and cost attribution analytics to determine department utilization
- Terabytes of data stored and queried in Snowflake

Snowflake Collaborations

tapClicks

REGENT

KORTX

Malwarebytes

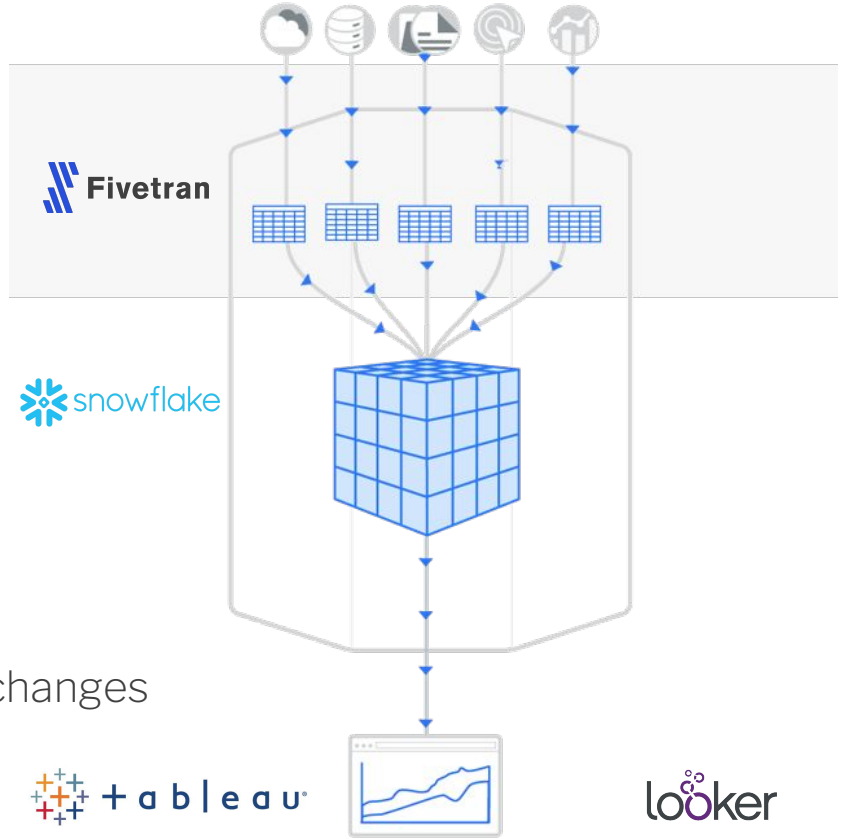
Participant introductions

- Name
- Company
- Do you have a data warehouse?
- Do you have a data lake?
- Cloud or on-premise? Which cloud?
- What type of use cases are you looking to solve?
- Which ETL tools are used?
- Which BI and analysis tools?


Fivetran

WHAT IS FIVETRAN?

- Fully managed data pipeline
- Fully automated
- Zero maintenance
- Zero configuration
- Hundreds of Sources
- Dynamic to API changes & schema changes
- 5 minute setup, 24/7 support

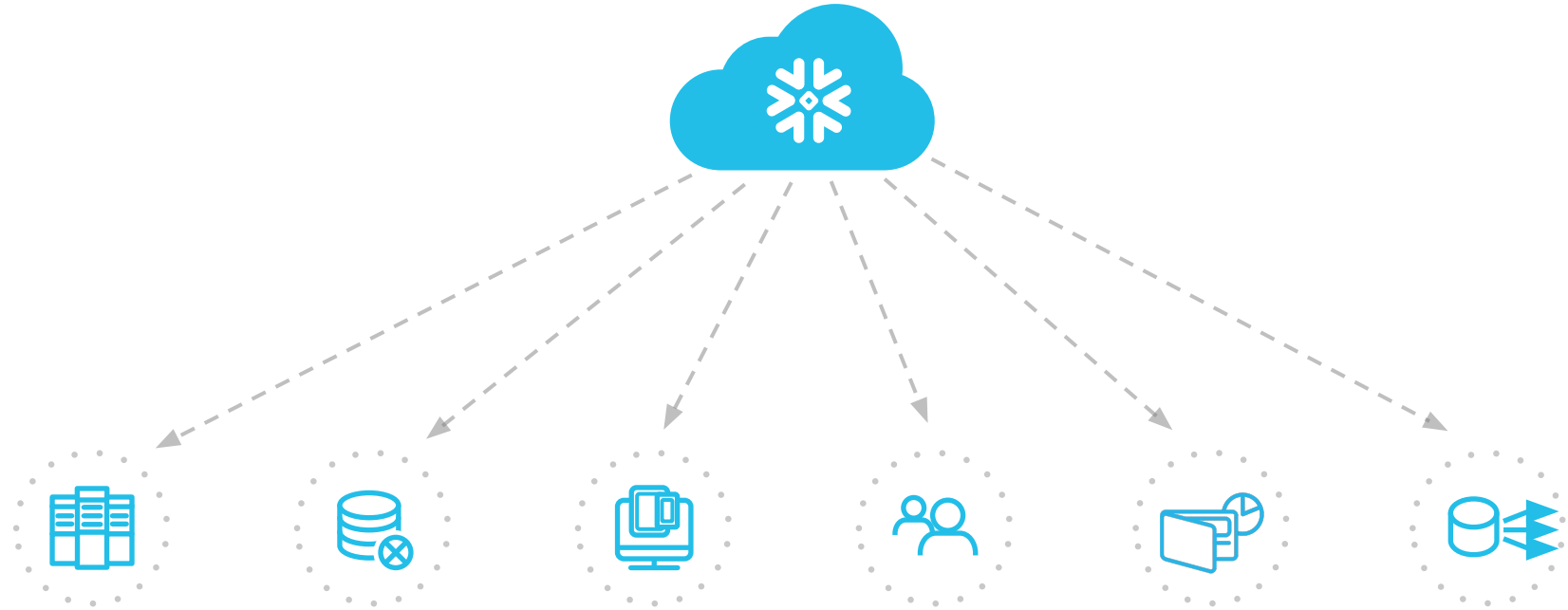


Snowflake introductions

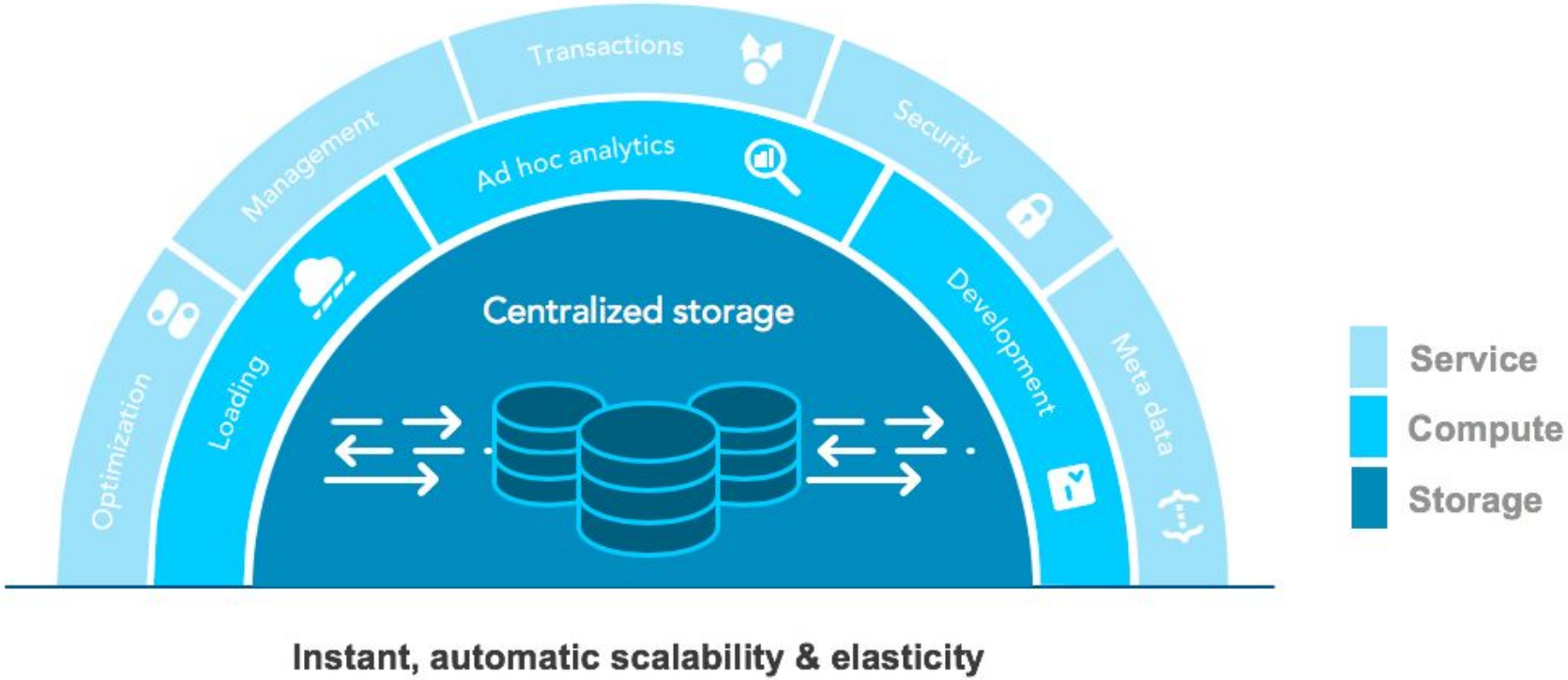
- Savannah English
-

Snowflake Introduction

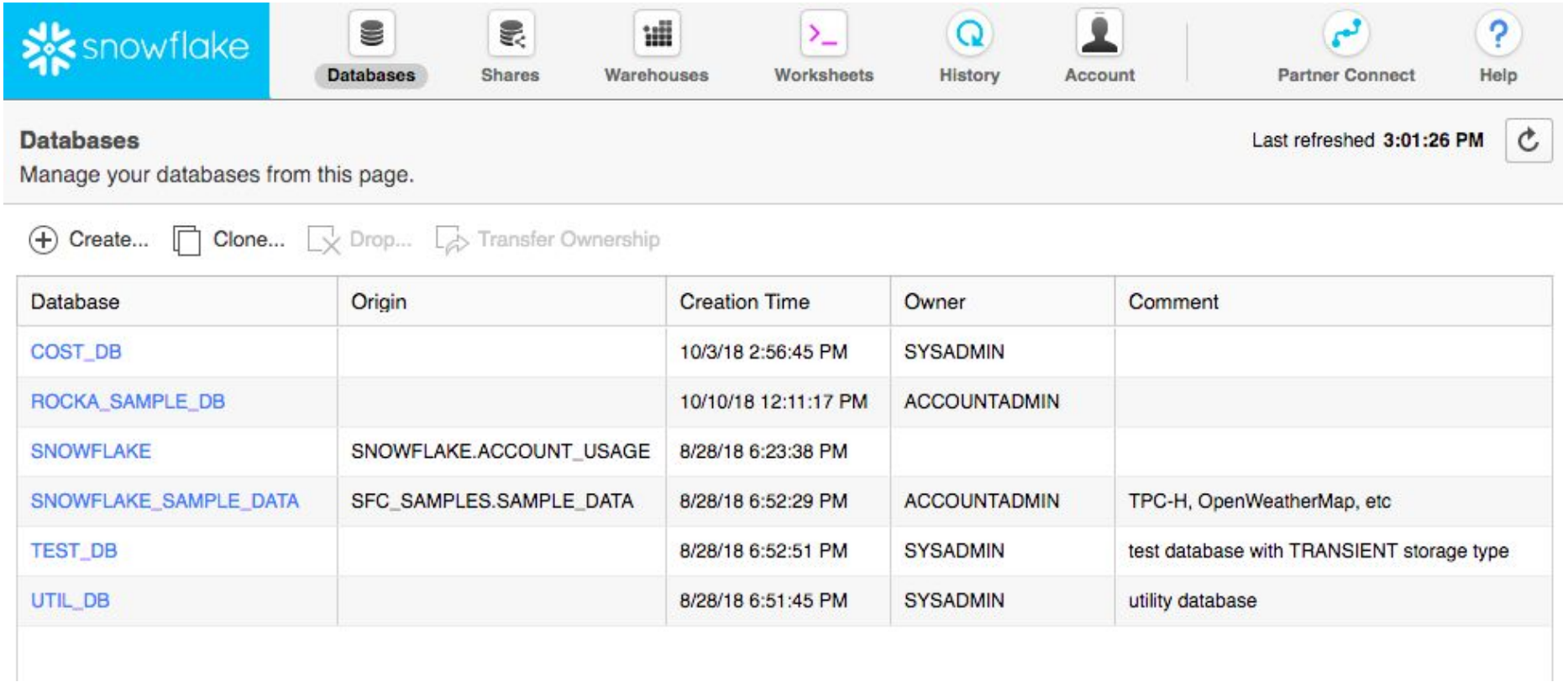
DATA WAREHOUSE BUILT FOR THE CLOUD



SNOWFLAKE'S MULTI-CLUSTER, SHARED DATA ARCHITECTURE

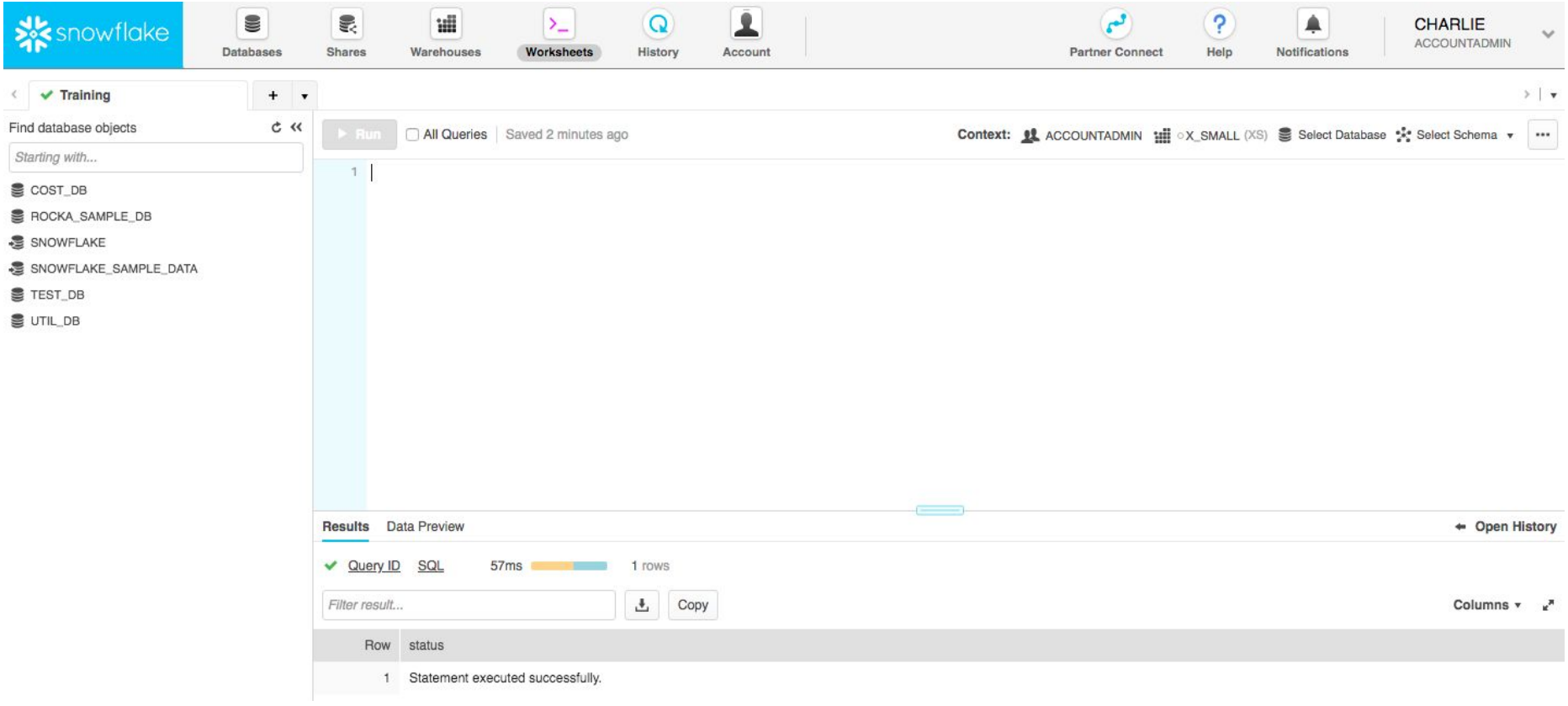


UI: ACCOUNTADMIN role in use



Database	Origin	Creation Time	Owner	Comment
COST_DB		10/3/18 2:56:45 PM	SYSADMIN	
ROCKA_SAMPLE_DB		10/10/18 12:11:17 PM	ACCOUNTADMIN	
SNOWFLAKE	SNOWFLAKE.ACCOUNT_USAGE	8/28/18 6:23:38 PM		
SNOWFLAKE_SAMPLE_DATA	SFC_SAMPLES.SAMPLE_DATA	8/28/18 6:52:29 PM	ACCOUNTADMIN	TPC-H, OpenWeatherMap, etc
TEST_DB		8/28/18 6:52:51 PM	SYSADMIN	test database with TRANSIENT storage type
UTIL_DB		8/28/18 6:51:45 PM	SYSADMIN	utility database

UI: ACCOUNTADMIN role in use



LET’S DIVE INTO SNOWFLAKE!

Hands on activity: what we are going to do today

- Guided exercises can be done via the UI or SQL
  - I will be using using the UI

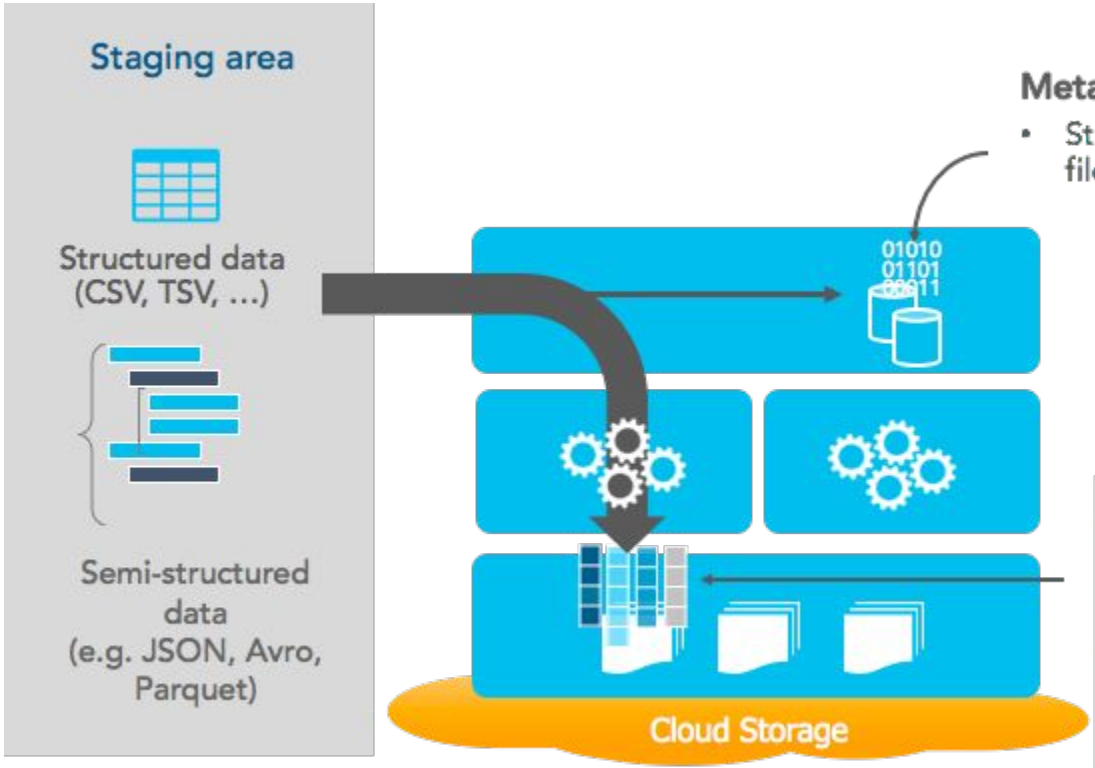
- Data loading
  - Creating all objects needed to load data
- Scaling
- Semi-structured data query
- Multi-clustering
- Time travel
- Cloning
- Caching

Links to open during workshop

- This presentation
  - https://tinyurl.com/zero-to-snowflake-201901
  - Follow along and copy SQL statements

Loading Data

Loading data into Snowflake



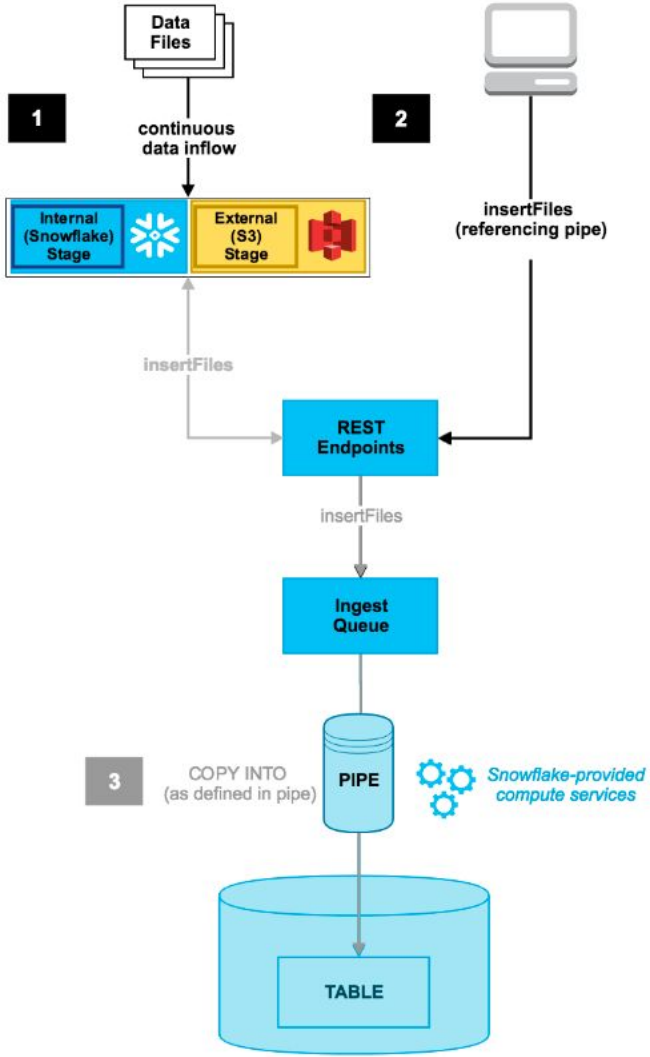
Metadata created

- Statistics for databases, tables, columns, & files calculated & stored in cloud services layer

Database data stored

- Columns in structured data separated
- Repeated elements extracted from semi-structured data into "columns"
- Columnar data chunked into optimized files, compressed, written to Amazon S3
- No change to logical view of data

Snowpipe



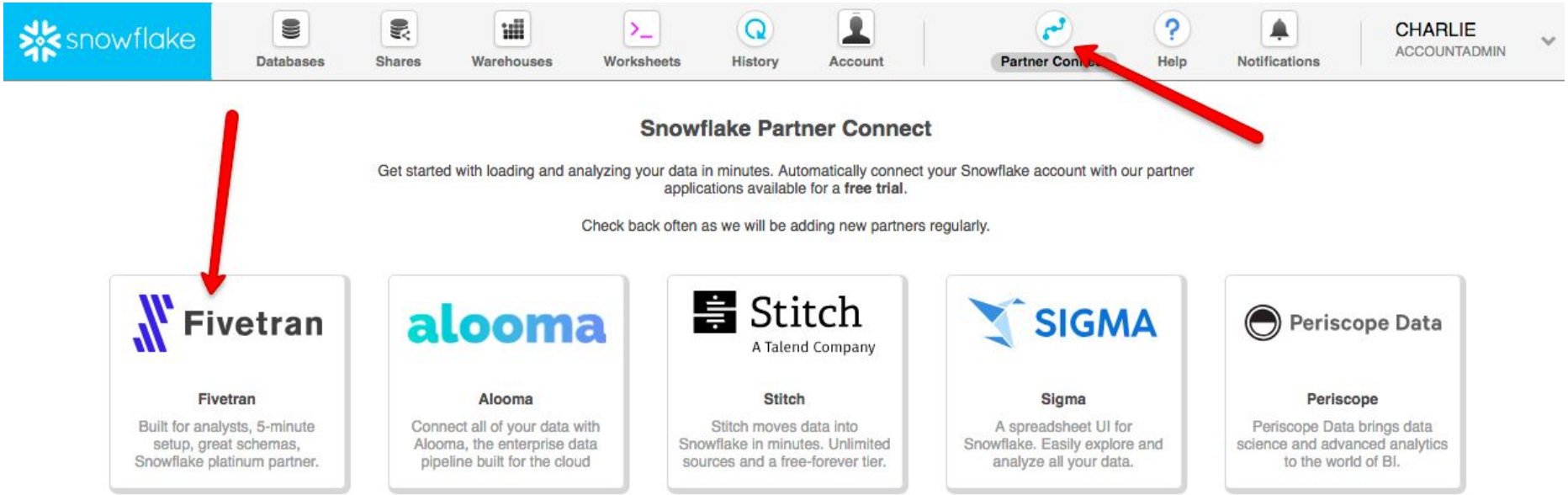
Manually loading data vs using existing cloud tools

- In this workshop, we will be manually loading data from a AWS S3 bucket

- Existing cloud based tools with connectivity to Snowflake
  - Extract and Load
    - Fivetran (https://fivetran.com/)
    - Stitch (https://www.stitchdata.com/)
  - Extract, Load, Transform
    - Matillion (https://www.matillion.com/)
    - Talend (https://www.talend.com/)
    - Alooma (https://www.alooma.com/)

- Snowflake's Partner Connect
  - Automatically connect your Snowflake account with our partner applications available for a free trial

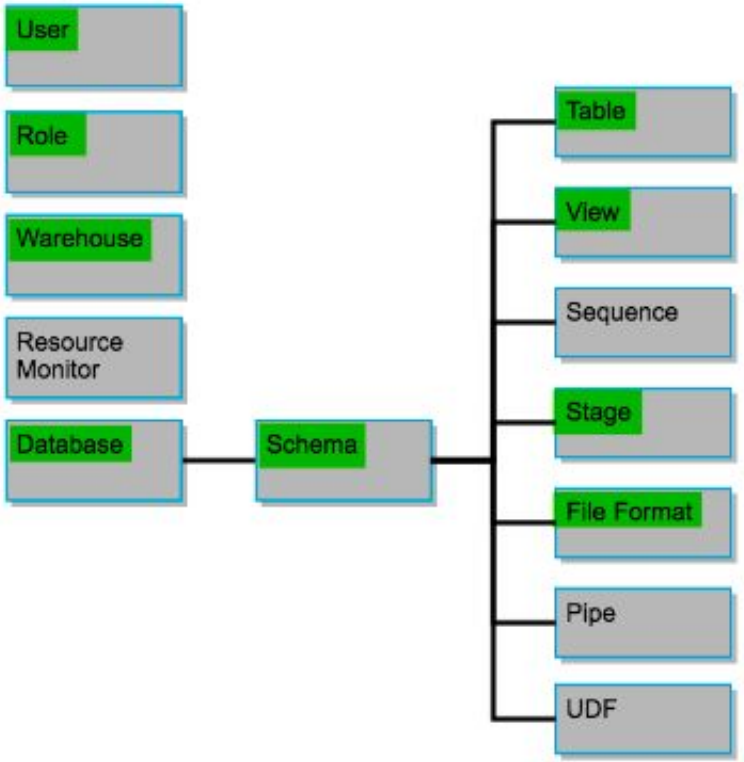
Fivetran on Snowflake Partner Connect



Objects needed to load data

1. USER
2. ROLE
3. VIRTUAL WAREHOUSE
4. DATABASE
5. FILE FORMAT
6. STAGE
7. TABLE

Securable objects



Access control

- Snowflake's approach to access control combines aspects from both of the following models:
  - Discretionary Access Control (DAC): Each object has an owner, who can in turn grant access to that object.
  - Role-based Access Control (RBAC): Access privileges are assigned to roles, which are in turn assigned to users.

https://docs.snowflake.net/manuals/user-guide/security-access-control.html

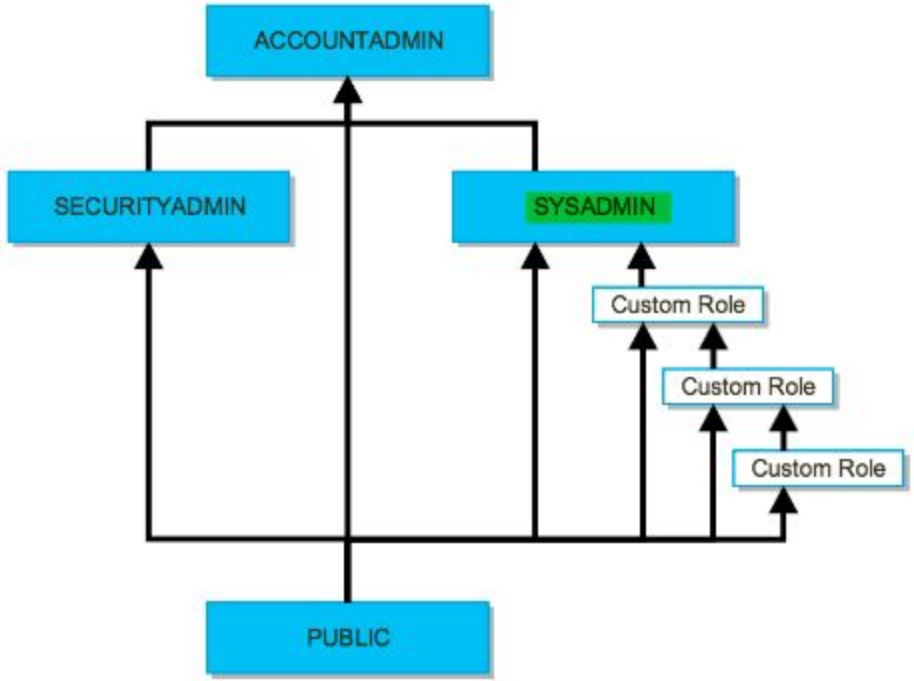
Access control best practices

- Avoid Using the ACCOUNTADMIN Role to Create Objects
- Avoid Using the ACCOUNTADMIN Role for Automated Scripts
- Create a role hierarchy that ultimately assigns all custom roles to the SYSADMIN role, this role also has the ability to grant privileges on warehouses, databases, and other objects to other roles

- Use the SYSADMIN role for this training

https://docs.snowflake.net/manuals/user-guide/security-access-control-considerations.html

Roles best practices diagram



User and role

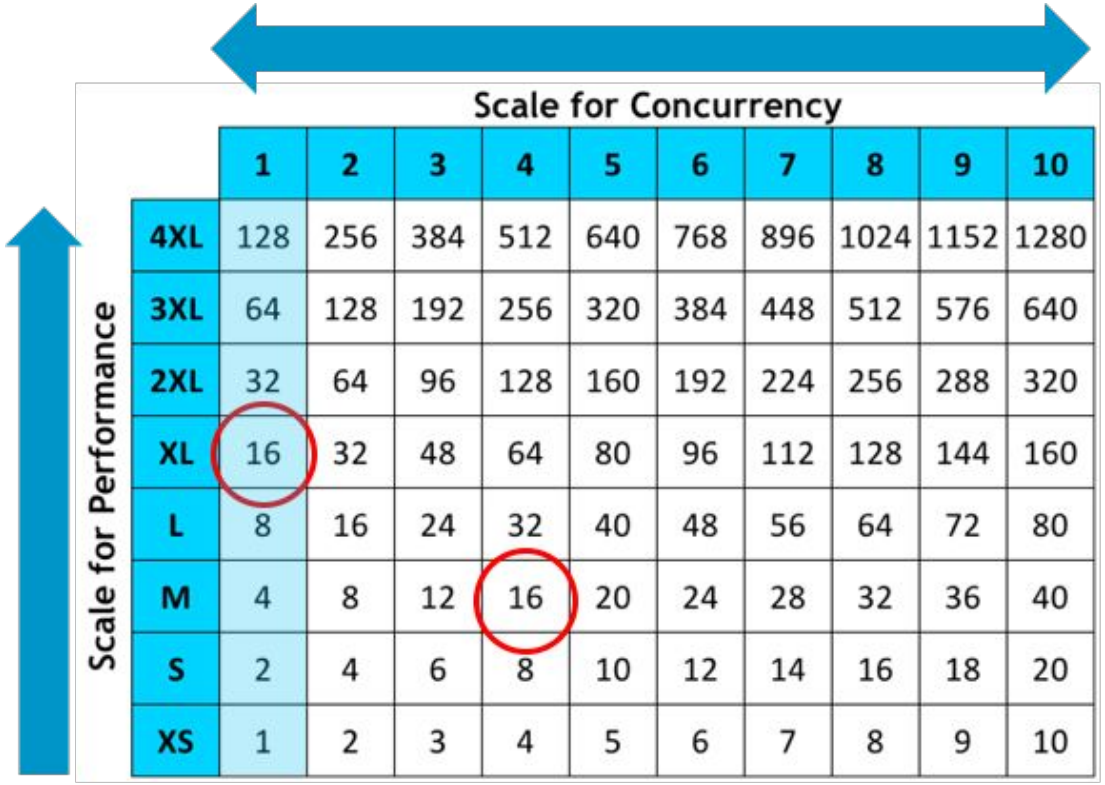
- Verify USER
  - In UI display in top right corner
  - SQL
    - SELECT CURRENT_USER();
- Verify Role
  - In UI display in top right corner
  - SQL
    - SELECT CURRENT_ROLE();
    - SHOW ROLES;
- Change Role to SYSADMIN to create objects
  - Using UI -> click on User -> Switch Role -> SYSADMIN
  - SQL
    - USE ROLE SYSADMIN;
    - SELECT CURRENT_ROLE();

Does anyone not have the SYSADMIN role granted?

- If you have ACCOUNTADMIN then grant SYSADMIN to your user
- Only execute if you don't have the SYSADMIN role
- SQL
  - USE ROLE ACCOUNTADMIN;
  - GRANT ROLE SYSADMIN TO USER CHARLIE; -- Change to your user

https://docs.snowflake.net/manuals/sql-reference/sql/grant-role.html

Warehouse sizing



	1	2	3	4	5	6	7	8	9	10
4XL	128	256	384	512	640	768	896	1024	1152	1280
3XL	64	128	192	256	320	384	448	512	576	640
2XL	32	64	96	128	160	192	224	256	288	320
XL	16	32	48	64	80	96	112	128	144	160
L	8	16	24	32	40	48	56	64	72	80
M	4	8	12	16	20	24	28	32	36	40
S	2	4	6	8	10	12	14	16	18	20
XS	1	2	3	4	5	6	7	8	9	10

These figures indicate the number of credits consumed for an hour's worth of compute

Scale each warehouse independently to provide additional compute power

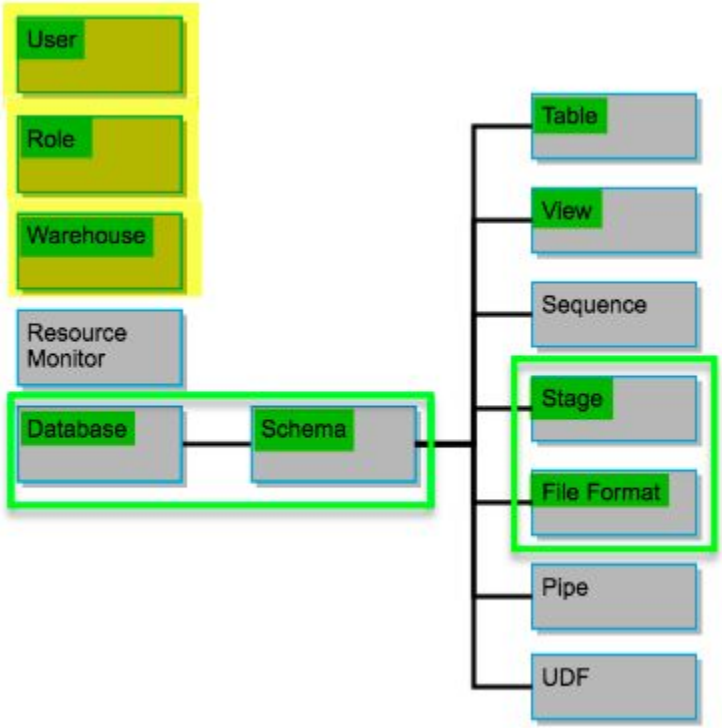
Automatically scale concurrent workloads with multi-cluster warehouses

Create the virtual warehouse

- Create a Virtual Warehouse
  - Using UI -> Warehouses -> Create
    - Name: TRAINING_WH
    - Size: X-Small
    - Auto Suspend: 300 seconds
    - Auto Resume: true
  - SQL
    - USE ROLE SYSADMIN;
    - CREATE OR REPLACE WAREHOUSE TRAINING_WH WITH
    - WAREHOUSE_SIZE='X-SMALL'
    - AUTO_SUSPEND = 300
    - AUTO_RESUME = TRUE
    - INITIALLY_SUSPENDED=FALSE;
    - SHOW WAREHOUSES;
    - USE WAREHOUSE TRAINING_WH;
    - SELECT CURRENT_WAREHOUSE();

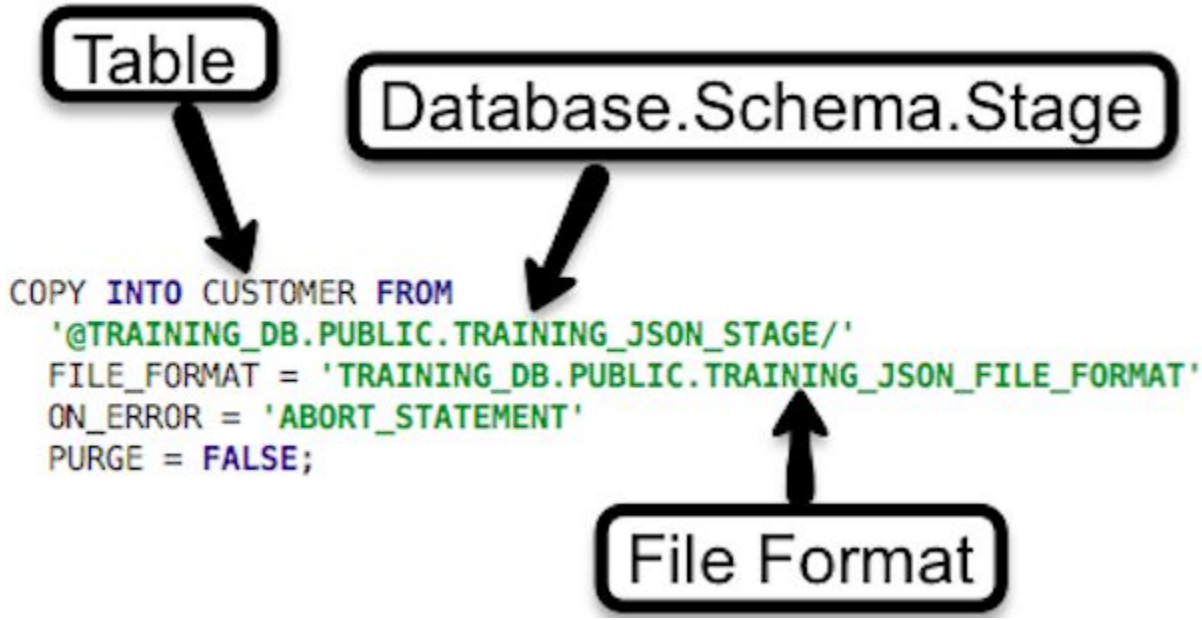
https://docs.snowflake.net/manuals/sql-reference/sql/create-warehouse.html

Objects created so far



COPY command

- We will be using the manual COPY command to manually load data
  - The COPY command needs the following objects



Create the database

- Using UI -> Databases -> Create
  - Name: TRAINING_DB
- SQL
  - USE ROLE SYSADMIN;
  - CREATE DATABASE TRAINING_DB;
  - SHOW DATABASES;
  - USE DATABASE TRAINING_DB;
  - SELECT CURRENT_DATABASE();

https://docs.snowflake.net/manuals/sql-reference/sql/create-database.html

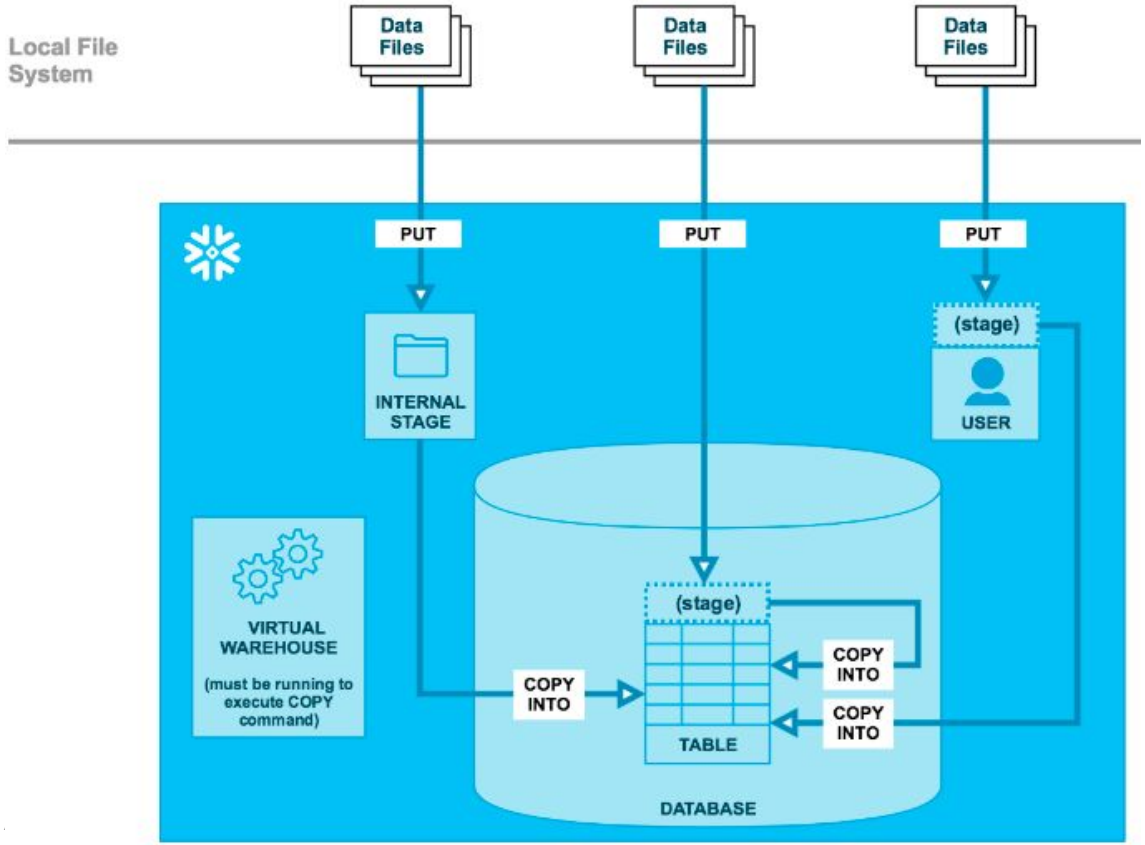
Schema

- The public schema is created by default
- We will use the public schema

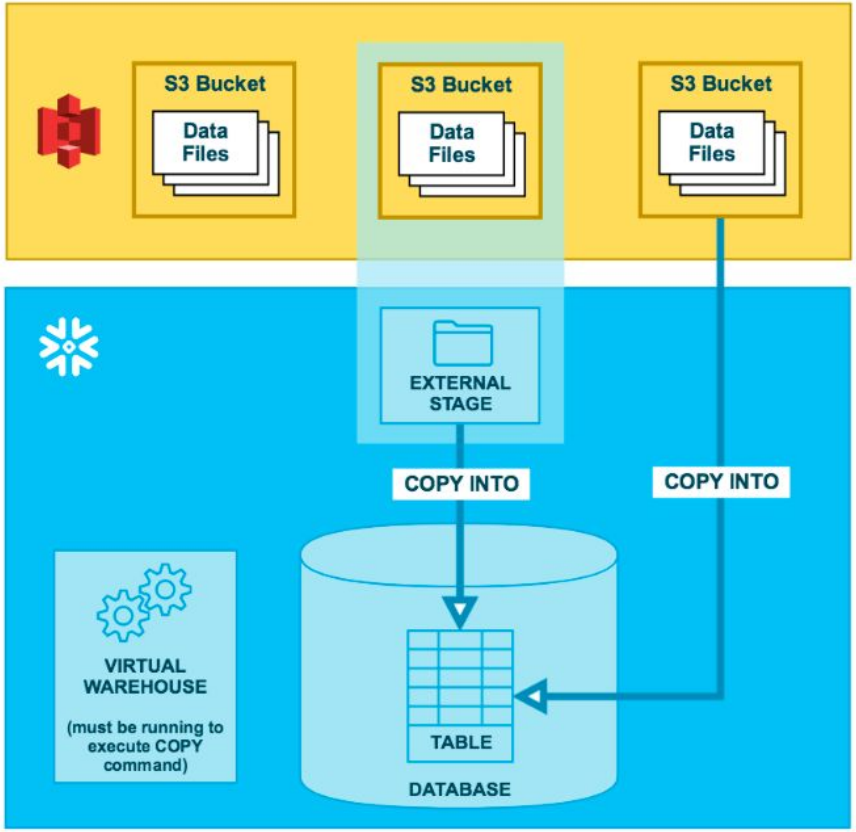
- Verify PUBLIC Schema exists
  - Using UI -> Databases -> click on TRAINING_DB -> click on Schemas tab
  - SQL
    - USE ROLE SYSADMIN;
    - USE DATABASE TRAINING_DB;
    - SHOW SCHEMAS;
    - USE SCHEMA PUBLIC;
    - SELECT CURRENT_SCHEMA();

https://docs.snowflake.net/manuals/sql-reference/sql/create-schema.html

Staging data: bulk loading from a local file system



Staging data: bulk loading from Amazon S3



Stages: user, table, named

- User stage
  - Internal
  - The @~ character combination identifies a user stage
    - put file:///data/data.csv @~/xyz_stage;
- Table stage
  - Internal
  - Same name as the table
  - The @% character combination identifies a table stage
    - put file:///data/data.csv @%xyz_table;
- Named Stage
  - Internal or External
  - Created before use
  - The @ character by itself identifies a named stage
    - put file:///data/data.csv @my_stage;

https://docs.snowflake.net/manuals/user-guide/data-load-local-file-system-create-stage.html

Stages: named internal or named external

- Named stages can be Internal or External
- An external stage references data files stored in a location outside of Snowflake. Currently, only AWS S3 buckets or Microsoft Azure containers are supported. The location can be either private/protected or public.
  - Named stages only have the @ in their names
- Named stages can specify the file_format
  - Then specifying the file_format in the COPY command isn't needed

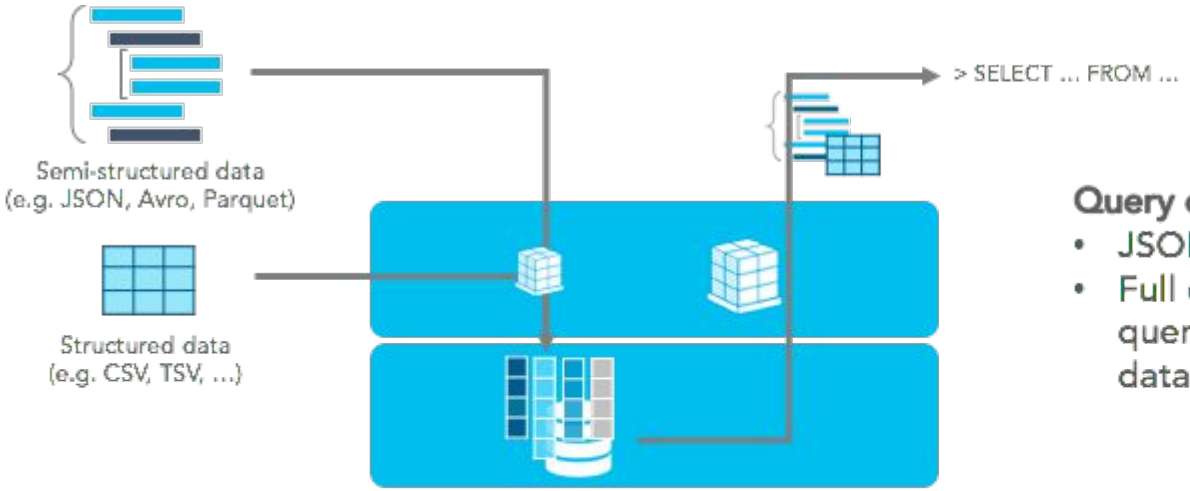
- An external S3 stage will be used for this exercise

https://docs.snowflake.net/manuals/user-guide/data-load-local-file-system-create-stage.html

Example data: JSON contents of data that will be loaded

```
{
  "customerprofile": {
    "customerinterests": [
      "sports",
      "movies"
    ],
    "gender": "female",
    "age": 19
  },
  "productline": "none",
  "uuid": "75e6e942-4f2d-482b-921d-0058856ddc88",
  "custkey": 5200001,
  "seqno": 0,
  "pagetype": "home",
  "useragent": {
    "platform": "windows",
    "devicetype": "pc",
    "experience": "browser"
  }
}
```

Native support for structured and semi-structured data



Query optimization

- JSON paths in SQL queries
- Full database optimization for queries on semi-structured data

Storage optimization

- Transparent discovery and storage optimization of repeated elements
- Data stored in same block-columnar format as structured data

Create file format

- Create a File Format
  - Using UI -> Databases -> click on TRAINING_DB -> click on File Formats tab -> Create
    - Name: TRAINING_JSON_FILE_FORMAT
    - Schema Name: PUBLIC
    - Format Type: JSON
    - Compression Method: Auto
  - SQL
    - USE ROLE SYSADMIN; USE DATABASE TRAINING_DB; USE SCHEMA PUBLIC;
    - CREATE FILE FORMAT TRAINING_JSON_FILE_FORMAT TYPE = 'JSON'
    - COMPRESSION = 'AUTO'
    - ENABLE_OCTAL = FALSE
    - ALLOW_DUPLICATE = FALSE
    - STRIP_OUTER_ARRAY = FALSE
    - STRIP_NULL_VALUES = FALSE
    - IGNORE_UTF8_ERRORS = FALSE;
    - SHOW FILE FORMATS;

https://docs.snowflake.net/manuals/sql-reference/sql/create-file-format.html

Create stage

- Create a Stage
  - Using UI -> Databases -> click on TRAINING_DB -> click on Stages tab -> Create
    - Existing Amazon S3 Location
    - Name: TRAINING_JSON_STAGE
    - Schema: PUBLIC
    - URL: s3://bytecodeio-zero-to-snowflake
  - SQL
    - USE ROLE SYSADMIN;
    - USE DATABASE TRAINING_DB;
    - USE SCHEMA PUBLIC;
    - CREATE OR REPLACE STAGE TRAINING_JSON_STAGE
    - FILE_FORMAT = TRAINING_JSON_FILE_FORMAT
    - URL = 'S3://bytecodeio-zero-to-snowflake';
    - SHOW STAGES;

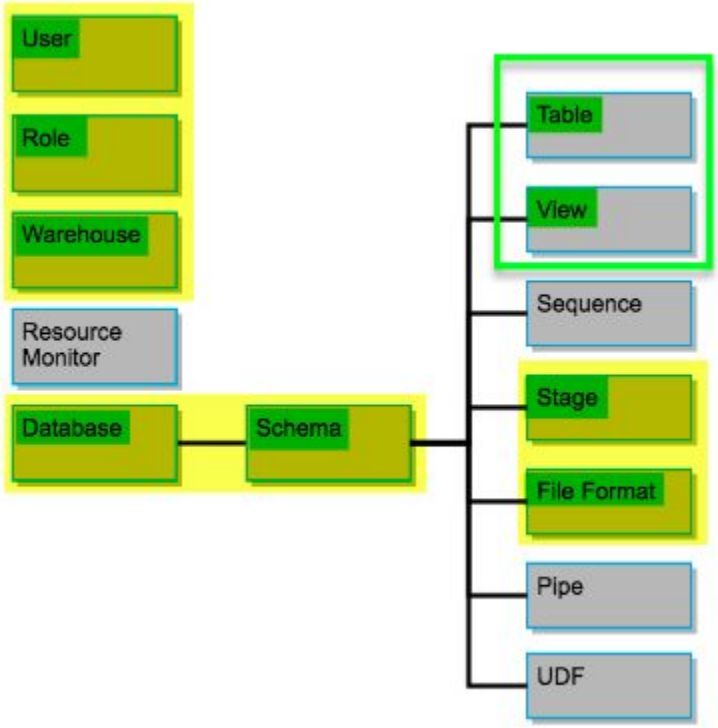
https://docs.snowflake.net/manuals/sql-reference/sql/create-stage.html

Confirm access to the staged files

- Create a Stage
  - SQL
    - USE ROLE SYSADMIN;
    - USE DATABASE TRAINING_DB;
    - USE SCHEMA PUBLIC;
    - SELECT METADATA$FILENAME, METADATA$FILE_ROW_NUMBER, PARSE_JSON($1)
    - FROM @TRAINING_JSON_STAGE (FILE_FORMAT => TRAINING_JSON_FILE_FORMAT)
    - LIMIT 10;

https://docs.snowflake.net/manuals/user-guide/querying-metadata.html

Objects created so far



Variant data type

- The variant data type is a tagged universal type, which can store values of any other type, including OBJECT and ARRAY, up to a maximum size of 16MB compressed
- We will use the variant type to store the JSON file contents in a single column of a table

https://docs.snowflake.net/manuals/sql-reference/data-types-semistructured.html

Create table

- Create a Table
  - Using UI -> Databases -> click on TRAINING_DB -> click on Tables tab -> Create
    - Table Name: CUSTOMER
    - Schema Name: PUBLIC
    - Columns:
      - Name: RAW_JSON
      - Type: Variant
  - SQL
    - USE ROLE SYSADMIN;
    - USE DATABASE TRAINING_DB;
    - USE SCHEMA PUBLIC;
    - CREATE TABLE CUSTOMER (RAW_JSON VARIANT);
    - SHOW TABLES;

https://docs.snowflake.net/manuals/sql-reference/sql/create-table.html

Loading the data

- Load the data
  - Using UI -> Databases -> click on TRAINING_DB -> highlight CUSTOMER -> click on Load data into table
    - Warehouse: TRAINING_WH
    - Load files from S3 bucket
      - Stage: TRAINING_JSON_STAGE
      - Path: blank
    - File Format: TRAINING_JSON_FILE_FORMAT
    -
  - SQL
    - USE ROLE SYSADMIN; USE DATABASE TRAINING_DB; USE SCHEMA PUBLIC;
    - COPY INTO CUSTOMER FROM
    - '@"TRAINING_DB"."PUBLIC"."TRAINING_JSON_STAGE"/'
    - FILE_FORMAT = '"TRAINING_DB"."PUBLIC"."TRAINING_JSON_FILE_FORMAT"'
    - ON_ERROR = 'ABORT_STATEMENT'
    - PURGE = FALSE;

https://docs.snowflake.net/manuals/sql-reference/sql/copy-into-table.html

Fixing load errors

- Locate the Query ID in History
  - Using UI -> History -> click on the Query ID
  - SQL

```
SELECT *
  FROM TABLE(INFORMATION_SCHEMA.QUERY_HISTORY())
  ORDER BY START_TIME
  LIMIT 10;
-- replace JOB_ID below with QUERY_ID from results
SELECT * FROM
  TABLE(VALIDATE(CUSTOMER, JOB_ID=>'63f08a0b-9d0c-48e8-ae43-cea4db3b434f'));
```

https://docs.snowflake.net/manuals/user-guide/data-load-considerations-ts.html

Verifying CUSTOMER table data

- Checking the count
  - Using UI -> Databases -> TRAINING_DB
    - The number of rows is displayed at 60M
  - SQL
    - SELECT COUNT(*) FROM CUSTOMER;
    - -- 60000000
- Analysing the data
  - SQL
    - SELECT * FROM CUSTOMER LIMIT 10;

Querying the JSON

- Insert a colon : between the VARIANT column name and any first-level element: <column>:<level1_element>
- Use dot notation to traverse a path in a JSON object: <column>:<level1_element>.<level2_element>.<level3_element>
- Alternatively, use bracket notation to traverse the path in an object: <column>['<level1_element>']['<level2_element>']
- Cast the values to the desired data type using :: notation)
- Example:
  - RAW_JSON:customerprofile.age::integer

https://docs.snowflake.net/manuals/user-guide/querying-semistructured.html

Example data: JSON contents

```
{
  "customerprofile": {
    "customerinterests": [
      "sports",
      "movies"
    ],
    "gender": "female",
    "age": 19
  },
  "productline": "none",
  "uuid": "75e6e942-4f2d-482b-921d-0058856ddc88",
  "custkey": 5200001,
  "seqno": 0,
  "pagetype": "home",
  "useragent": {
    "platform": "windows",
    "devicetype": "pc",
    "experience": "browser"
  }
}
```

Flatten the JSON

- Analysing the data
  - SELECT RAW_JSON:custkey::integer
  - FROM CUSTOMER LIMIT 10;
  - SELECT RAW_JSON:customerprofile.age::integer AS age,
  - RAW_JSON:customerprofile.gender::string AS gender
  - FROM CUSTOMER LIMIT 10;

Flatten the JSON array

- Analysing the data
  - SELECT
  - RAW_JSON:uuid::string AS uuid,
  - F.value
  - FROM CUSTOMER,
  - LATERAL FLATTEN(INPUT => RAW_JSON:customerprofile.customerinterests) F
  - WHERE RAW_JSON:uuid::string = 'ecbd6b7d-b2b7-4c30-8f39-ed18339ebb34';

https://docs.snowflake.net/manuals/sql-reference/functions/flatten.html

Create a CUSTOMER_VW

- Analysing the data
  - CREATE OR REPLACE VIEW CUSTOMER_VW AS
  - SELECT
  - RAW_JSON:uuid::string AS uuid,
  - RAW_JSON:custkey::integer AS custkey,
  - RAW_JSON:customerprofile.age::integer AS age,
  - RAW_JSON:customerprofile.gender::string AS gender,
  - RAW_JSON:productline::string AS productline,
  - RAW_JSON AS raw_json
  - FROM CUSTOMER ;

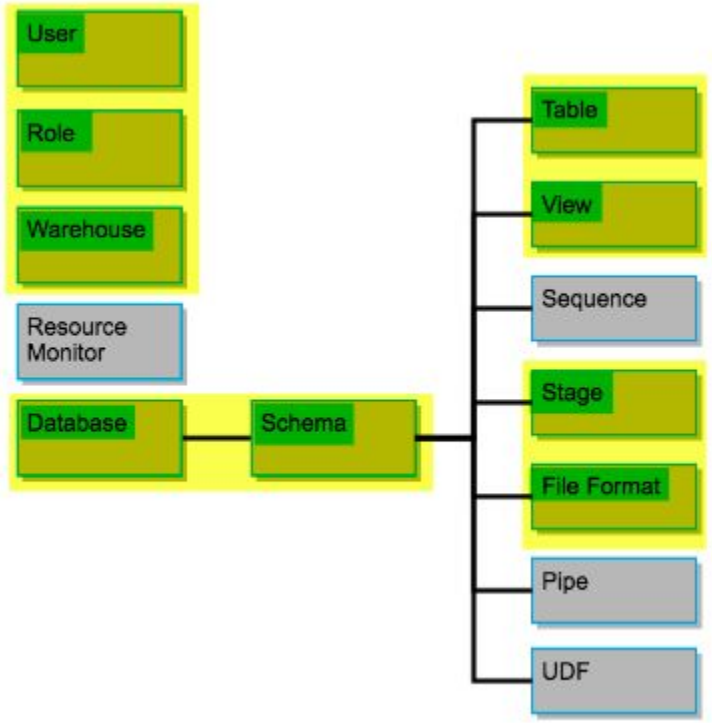
  - SELECT * FROM CUSTOMER_VW LIMIT 10;

Create a CUSTOMER_INTERESTS_VW

- Analysing the data
  - CREATE OR REPLACE VIEW CUSTOMER_INTERESTS_VW AS
  - SELECT
  - RAW_JSON:uuid::string AS uuid,
  - F.value::string AS interest
  - FROM CUSTOMER,
  - LATERAL FLATTEN(INPUT => RAW_JSON:customerprofile.customerinterests) F ;

  - SELECT * FROM CUSTOMER_INTERESTS_VW LIMIT 10;

All objects created!



Warehouse performance: scaling vertically

- Resize warehouse can be resized mid-query, without affecting existing workloads
- Resize warehouse to gain performance
  - Using UI -> Warehouses -> highlight Warehouse Name -> Configure ...
    - Change the Size
  - SQL
    - USE ROLE SYSADMIN;
    - ALTER WAREHOUSE TRAINING_WH SET WAREHOUSE_SIZE = 'LARGE';
    - SHOW WAREHOUSES like 'TRAI%';
    - ALTER WAREHOUSE TRAINING_WH SET WAREHOUSE_SIZE = 'X-SMALL';

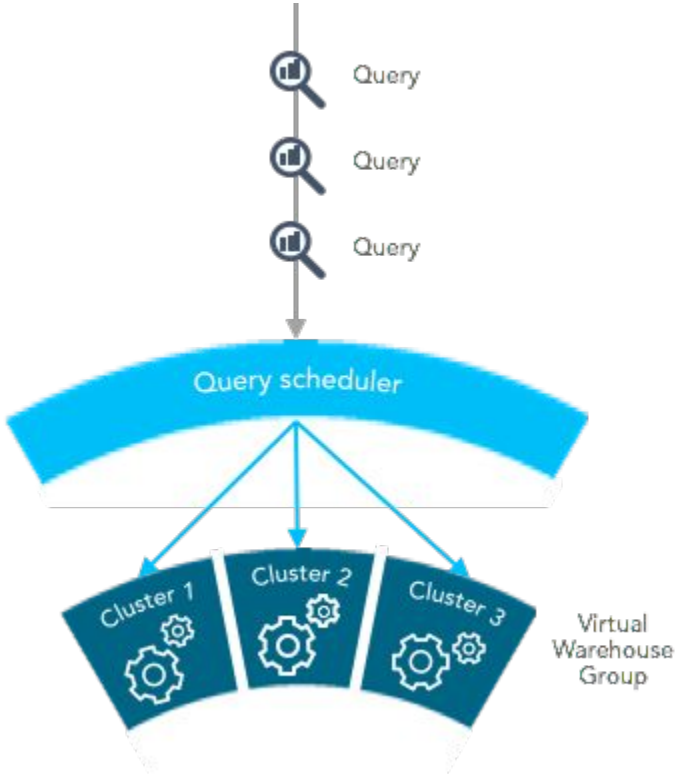
https://docs.snowflake.net/manuals/sql-reference/sql/alter-warehouse.html

JOIN views to get distinct count of customer interests

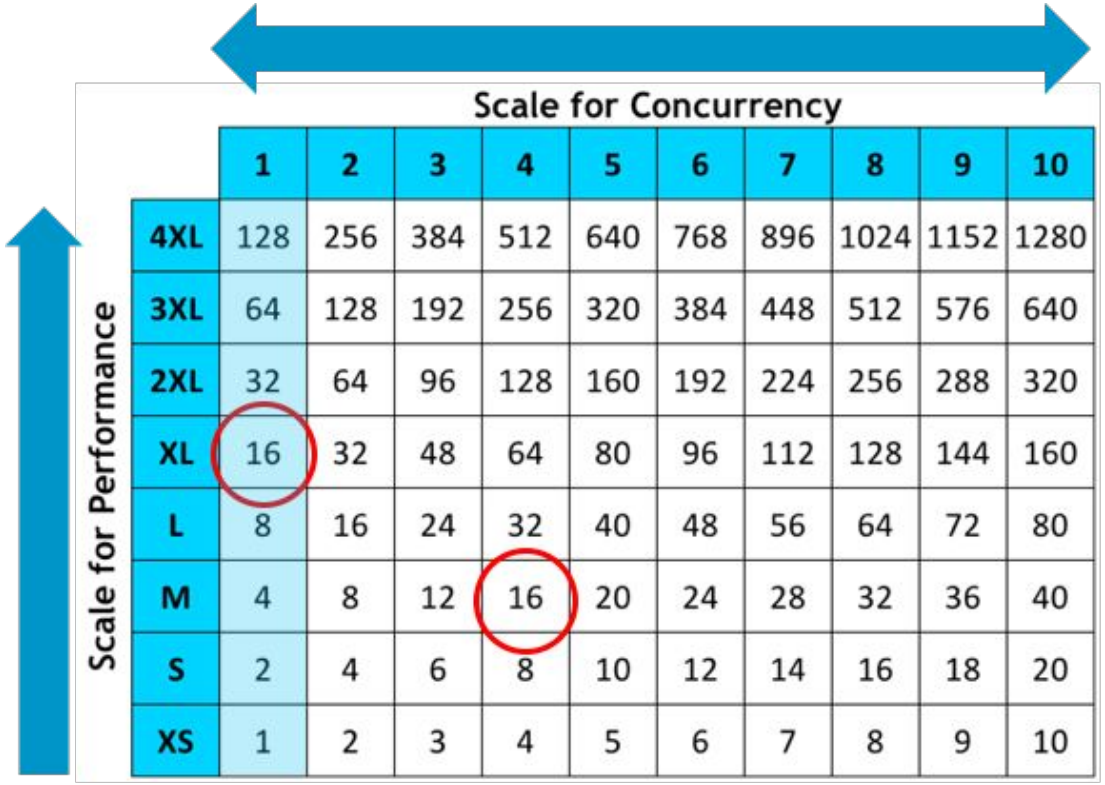
- Analysing the data
  - SELECT COUNT(DISTINCT C.custkey) AS distinct_customers, CI.interest AS interest
  - FROM CUSTOMER_VW AS C
  - JOIN CUSTOMER_INTERESTS_VW AS CI ON C.uuid = CI.uuid
  - GROUP BY CI.interest
  - ORDER BY COUNT(DISTINCT C.custkey) DESC;
- Run query and change warehouse size mid-query
  - Doesn't impact existing query
  - Run query again using new size
  - View queries in History
    - Execution time is reduced

Multi-cluster warehouse: scaling horizontally

- Automatically scales compute resources based on concurrent usage
- Single virtual warehouse of multiple compute clusters
- Queries are load balanced across the clusters in a virtual warehouse
- Split across availability zones for high availability



Warehouse sizing



	1	2	3	4	5	6	7	8	9	10
4XL	128	256	384	512	640	768	896	1024	1152	1280
3XL	64	128	192	256	320	384	448	512	576	640
2XL	32	64	96	128	160	192	224	256	288	320
XL	16	32	48	64	80	96	112	128	144	160
L	8	16	24	32	40	48	56	64	72	80
M	4	8	12	16	20	24	28	32	36	40
S	2	4	6	8	10	12	14	16	18	20
XS	1	2	3	4	5	6	7	8	9	10

These figures indicate the number of credits consumed for an hour's worth of compute

Scale each warehouse independently to provide additional compute power

Automatically scale concurrent workloads with multi-cluster warehouses

Query concurrency

- If queries are queuing then use a multi-cluster warehouse
  - Using UI -> Warehouses -> highlight Warehouse Name -> Configure ...
    - Change to multi-cluster
  - SQL
    - USE ROLE SYSADMIN;
    - ALTER WAREHOUSE TRAINING_WH
    - SET WAREHOUSE_SIZE='X-SMALL'
    - SCALING_TYPE = 'ECONOMY' -- Scales if system load will keep cluster busy for > 6 mins
    - MIN_CLUSTER_COUNT = 1
    - MAX_CLUSTER_COUNT = 10;
    - SHOW WAREHOUSES;

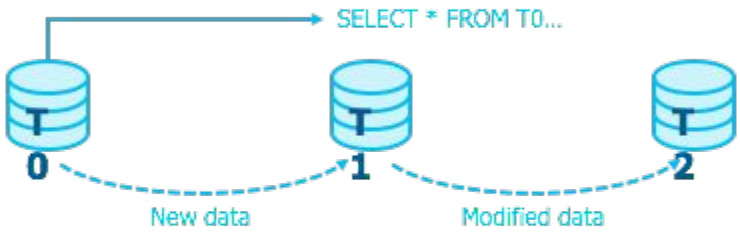
https://docs.snowflake.net/manuals/user-guide/warehouses-multicluster.html

Result cache

- All query results are stored for 24 hours unless underlying data changes
- Identical queries are returned instantly without requiring compute
- SQL
  - SELECT
  - RAW_JSON:uuid::string AS uuid
  - FROM CUSTOMER
  - WHERE RAW_JSON:uuid::string = 'ecbd6b7d-b2b7-4c30-8f39-ed18339ebb34';
- Reviewing history for second execution notice it doesn't consume compute

https://docs.snowflake.net/manuals/user-guide/querying-persisted-results.html

Time travel and cloning



Protection against infrastructure failures

All data transparently & synchronously replicated 3+ ways across multiple datacenters

Protection against corruption & user errors

"Time travel" feature enables instant roll-back to any point in time during chosen retention window

Long-term data protection

Zero-copy clones + optional export to S3 enable user-managed data copies

Time travel

See: https://docs.snowflake.net/manuals/user-guide/data-time-travel.html

Time travel

- The standard retention period is 1 day (24 hours) and is automatically enabled for all Snowflake accounts
- SQL
  - SHOW TABLES HISTORY;
  - DROP TABLE CUSTOMER;
  - SHOW TABLES HISTORY;
  - DESCRIBE TABLE CUSTOMER; -- This will error
  - UNDROP TABLE CUSTOMER;
  - SHOW TABLES HISTORY;
  - DESCRIBE TABLE CUSTOMER;

https://docs.snowflake.net/manuals/user-guide/data-time-travel.html

Cloning

- SQL
  - Clone a table before a specified query
  - CREATE TABLE CUSTOMER_CLONE_RESTORE
  - CLONE CUSTOMER
  - BEFORE (STATEMENT => '8e5d0ca9-005e-44e6-b858-a8f5b37c5726'); -- Query ID

  - Clone a table before a specified date
  - CREATE TABLE CUSTOMER_CLONE_RESTORE
  - CLONE CUSTOMER
  - AT (TIMESTAMP => TO_TIMESTAMP_TZ('04/05/2013 01:02:03', 'mm/dd/yyyy hh24:mi:ss'));

https://docs.snowflake.net/manuals/sql-reference/sql/create-clone.html

Data Sharing

Data sharing

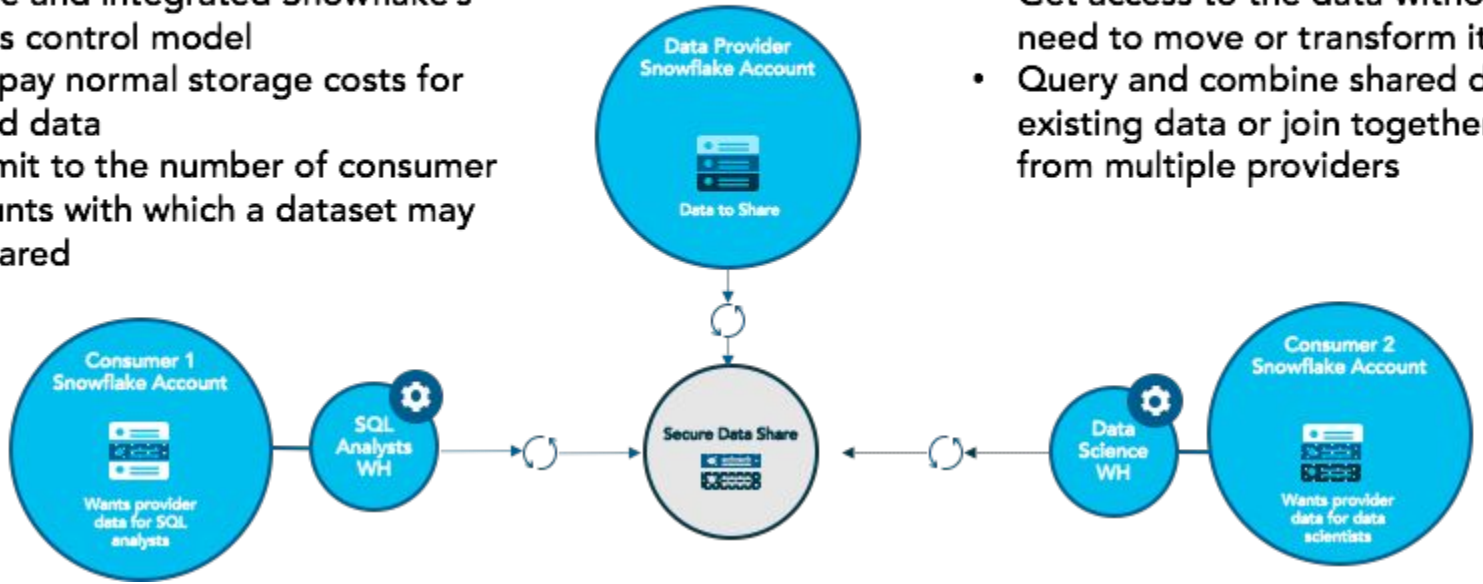
- Enabled by Snowflake's unique cloud architecture

Providers

- Secure and integrated Snowflake's access control model
- Only pay normal storage costs for shared data
- No limit to the number of consumer accounts with which a dataset may be shared

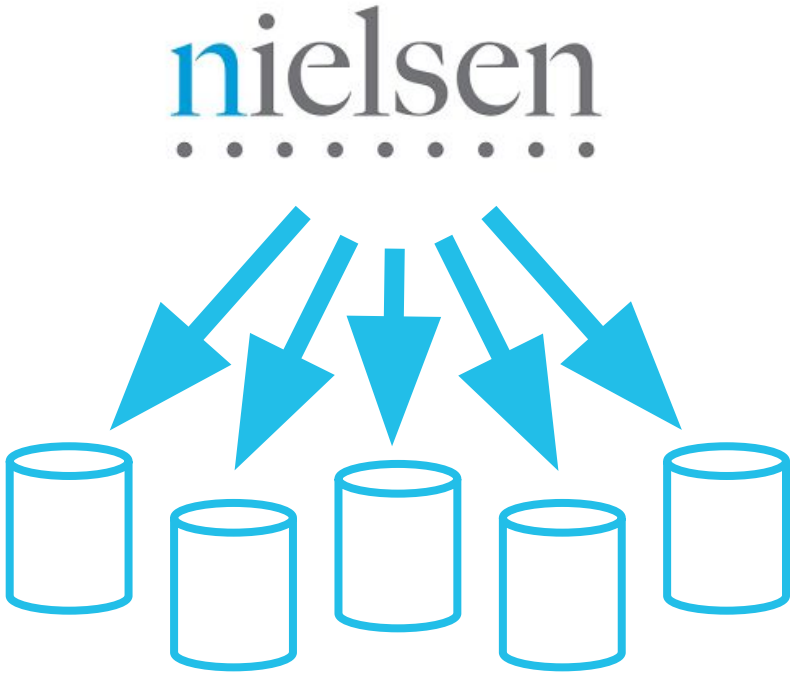
Consumers

- Get access to the data without any need to move or transform it.
- Query and combine shared data with existing data or join together data from multiple providers



USE CASES FOR SNOWFLAKE DATA SHARING

- Nielsen is a global information, data, and measurement company
  - Nielsen knows “What People Watch, Listen To, and Buy”
  - Nielsen Marketing Cloud includes eXelate DMP which provides unified consumer profiles
  - Nielsen sells selective slices of their DMP data available to advertisers for particular marketing campaigns
- Nielsen plans to use data sharing for making detailed datasets available to subscribers
  - Lower friction, lower cost solution
  - Scalable operations



Sharing: data provider

- The data provider creates the share and adds the data consumer account
- SQL
  - USE ROLE ACCOUNTADMIN;
  - CREATE SHARE CUSTOMER_S;
  - GRANT USAGE ON DATABASE TRAINING_DB TO SHARE CUSTOMER_S;
  - GRANT USAGE ON SCHEMA TRAINING_DB.PUBLIC TO SHARE CUSTOMER_S;
  - GRANT SELECT ON TABLE TRAINING_DB.PUBLIC.CUSTOMER TO SHARE CUSTOMER_S;
  - SHOW GRANTS TO SHARE CUSTOMER_S;
  - ALTER SHARE CUSTOMER_S ADD ACCOUNTS=NV67092; -- account to share
  - SHOW SHARES;

https://docs.snowflake.net/manuals/user-guide/data-share-providers.html

Sharing: data consumer

- The data consumer creates an object from the share
  - Shares are read-only
- SQL
  - SHOW SHARES;
  - DESCRIBE SHARE BYTECODE.CUSTOMER_S;
  - CREATE DATABASE BYTECODE_DB FROM SHARE BYTECODE.CUSTOMER_S;
- Query the shared data
  - USE DATABASE BYTECODE_DB;
  - USE SCHEMA PUBLIC;
  - SELECT
  - RAW_JSON:uuid::string AS uuid, *
  - FROM CUSTOMER
  - WHERE RAW_JSON:uuid::string = 'ecbd6b7d-b2b7-4c30-8f39-ed18339ebb34';
  - DROP DATABASE BYTECODE_DB

https://docs.snowflake.net/manuals/user-guide/data-share-consumers.html

Looker Connectivity

Different warehouses for different workloads

- BI Queries
  - Use a separate warehouse for BI workloads
  - Latest data is available as it is inserted
- ETL
  - Use a separate warehouse for ETL workloads
  - ETL doesn't impact BI performance
- Looker is connected to the LOOKER_WH

Leveraging LookML and Snowflake's JSON support

- Views can be maintained in LookML
- The views can contain JSON paths
  - dimension: platform {
  - type: string
  - sql: ${TABLE}.RAW_JSON:useragent.platform::string ;;
  - }
- Show Looker

Security

Enterprise grade security

Authentication

Multi-factor authentication

Federated authentication support

Identity propagation

Access control

Role-based access control model

Granular privileges on all objects & actions

Data encryption

All data encrypted, always, end-to-end

Encryption keys managed automatically

Tri-secret secure
Customer managed keys

Secure

AICPA SOC

SANS INSTITUTE

All communication via TLS

Private Link available in AWS

Certified against enterprise-class requirements

Numerous industry certifications

August Smart Home

Leveraging LookML and Snowflake's JSON support

- Start up moving fast
- Mongo as application db
  - Engineering had no time for change control
  - Leveraged Snowflake's JSON support
  - Able to update json paths in Looker
    - Technical business user could add new json fields
- Used different warehouses for different workloads
  - ETL and Looker
- Kustomer (customer service) data shared using Data Sharing
- Cloud source data acquisition using Fivetran
  - Shopify, Recurly, Delighted
- Secure

What was accomplished...

- Spun up multiple warehouses
- Instantly improved query performance
- Elastically scaled up, down and off for varying levels of activity
- Exhibited Multi-Clustering
- Discussed benefits and use cases for Time Travel
- Natively ingested JSON
- Cloned a database
- Connected to tools in a data ecosystem
- Learned how to easily share data, without the need for complex or tedious methods

Questions?

bytecode.io
charlie.killian@bytecode.io

FINANCE

"We are really excited about the opportunities technology gives us to deliver a personalized experience."

Linda Apsley

VP of Data Engineering

Capital One

THE IMPACT OF DATABASE PROBLEMS AT CAPITAL ONE

Painfully slow analytics cycles

Concurrency

With massive concurrency across dozens of business units, database performance was strained to the maximum, slowing analysis times and limiting data to a few.

Limited ability to answer complex questions

Scale

Data scientists were unable to scale the service to larger queries, significantly inhibiting their ability to find new ways to deliver personalized experiences.

Inability to provide business continuity

Resiliency

Without a way to failover between regions, the existing solution lacked a critical capability in delivering and protecting data.

Time consuming custom security

Security

Without the built-in security they needed, Capital One had to painstakingly secure their data warehouse from scratch.

THE IMPACT OF SNOWFLAKE AT CAPITAL ONE

Hundreds of newly empowered analysts

Skyrocketing analytics ROI

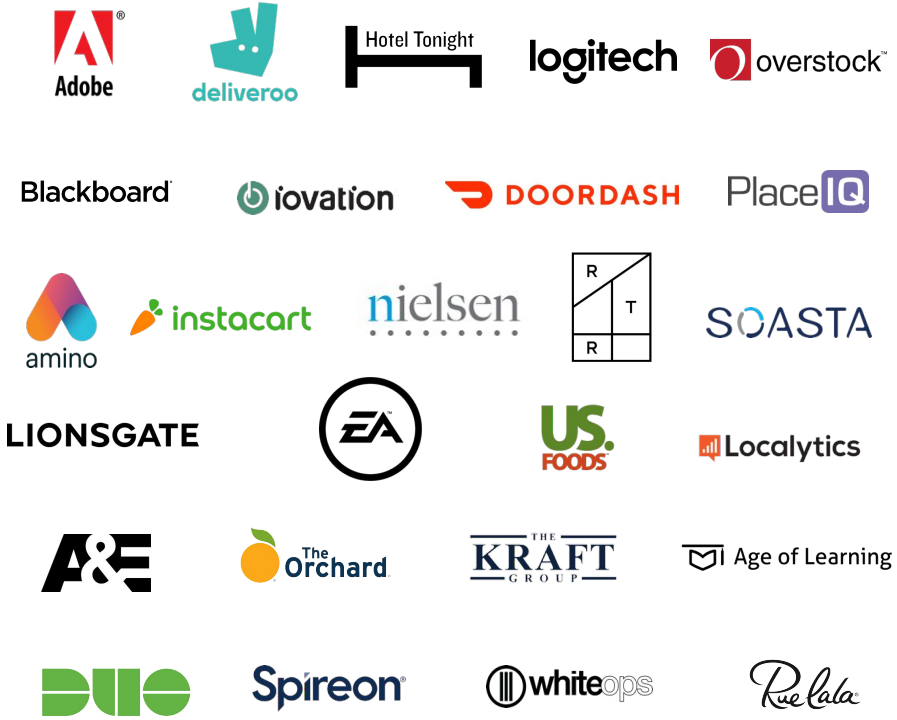
Guaranteed data availability, 24/7/365

Peace of mind for customers and employees

Concurrency	+	Scale	+	Resiliency	+	Security
Delivering customized experience requires a great deal of analytics. Now that the entire team has their own compute resources, anyone can finally ask any question at any time.		The data science team is now empowered with infinite scalability. Combined with pay as you go billing, they can answer every question quickly, at dramatically lower cost.		With built-in failover from US-West to US-East, Capital One has ensured their customer data will always be available, accurate, and ready for analysis.		Snowflake's built in security features have delivered the safety that Capital One demands and that their customers deserve.

CONSUMER APPS

1,500+ CUSTOMERS



Moving to the cloud

Capital One — Using Snowflake to move data analytics to the cloud

Accelerating enterprise BI and analytics

SONY — Moved from legacy data warehouse systems (appliance & cloud) to Snowflake

Modernizing data platforms

IAC PUBLISHING — Replaced data warehouse appliance + Hadoop with Snowflake

Building new analytic applications

PDX — Delivered new analytic application to pharmacies using Snowflake

MEDIA & ADVERTISING

STREAMLINING THE ANALYTICS PIPELINE

sharethrough

Scenario

Analytics to support monetization and optimization for native advertising

Pain Points

- Complicated stack with 14 technologies
- Slow report performance
- Single resource for deployment

Solution

Deployed Snowflake, FiveTran, Microstrategy to support analytics

Results

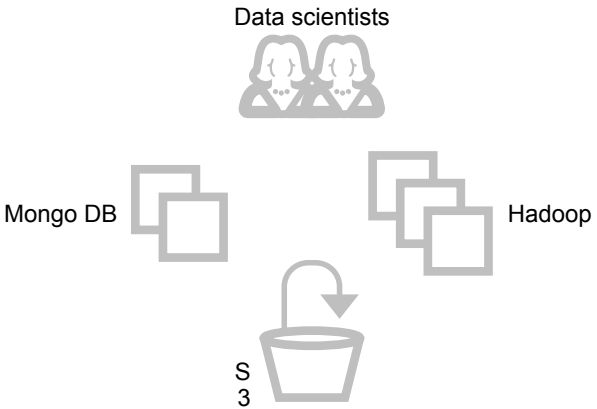
- Increased performance by **2,000X**
- Reduced analytics stack to four technologies
- Simplified management of data warehouse & analytics environment

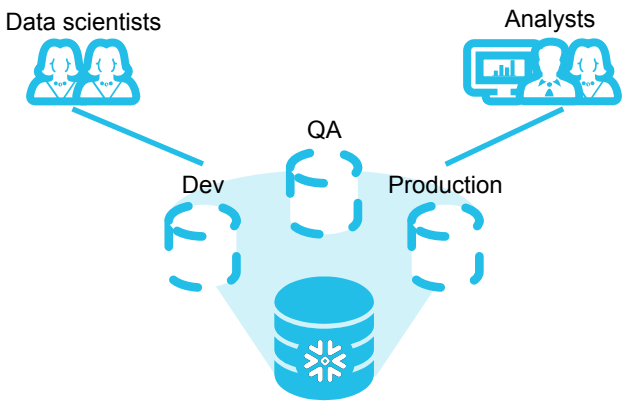
We went from using 14 technologies to a stack of four powerhouses—AWS, Snowflake, Fivetran and MicroStrategy—that can be managed by a single person.

Joseph Bates
Head of Analytics

SOFTWARE & INTERNET

DATA SCIENCE AND EXPLORATION





Scenario

- Security organization forced to use delayed reporting

Pain Points

- Many data requests unaddressed
- 24 hour turnaround time on requests
- 24 hours to push new models live

Solution

- Deploy Snowflake to accommodate analytics workloads

Snowflake Value

- 2 hours to push new models live
- Generated new research report
- Analysts can use data directly

EDUCATION

CUSTOMER EXAMPLE: BLACKBOARD

Blackboard

Scenario
Provide and perfect over 14 different data products that help universities facilitate learning online

Pain Points
Disparate data
Challenges integrating data
Semi-structured data

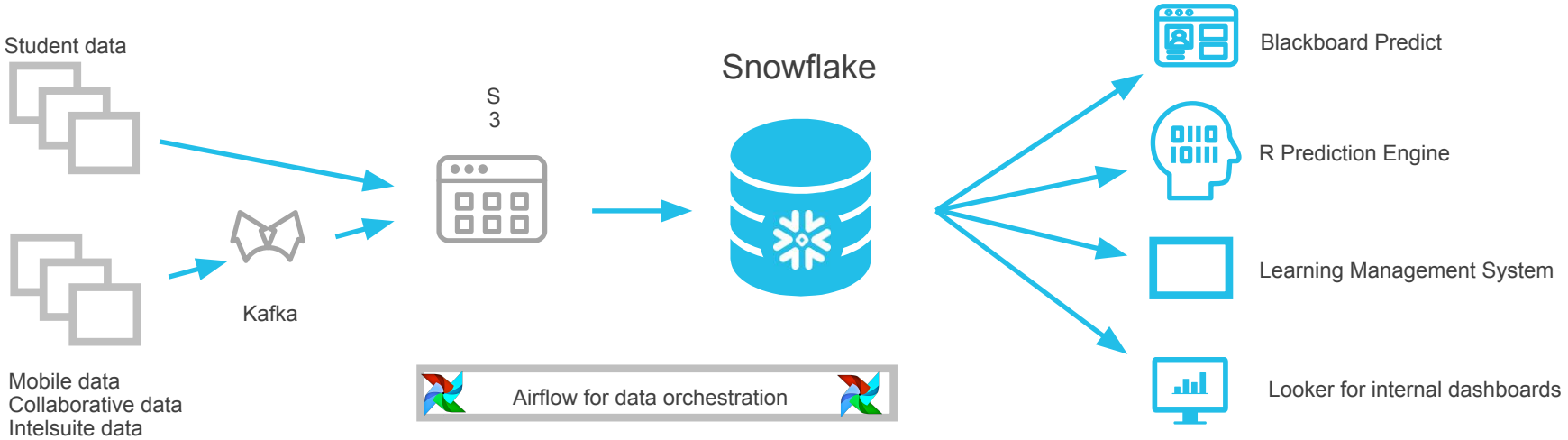
Solution
Replace existing Hadoop and RDBMS system with Snowflake

> *Everything that we did left our jaw on the table. 'Wait – we've never done anything like that.' Or, 'How did that just run so fast.' We are getting 16x performance from Snowflake.*

Jay White
Director, Software Engineering

A NEW DATA PIPELINE FOR BLACKBOARD

Blackboard



Unified data
Simplified data transformation
Existing tools integrate seamlessly

- 16x performance improvement over SQL
- 1 PB by the end of 2017