DISCLAIMER

Other than statements of historical fact, all information contained in the presentations and accompanying oral commentary made available as part of this event (collectively, the "Materials"), including statements regarding (i) Snowflake's business strategy and plans, (ii) Snowflake's new or enhanced products, services, and technology offerings, including those that are under development, (iii) market size and growth, trends, and competitive considerations, and (iv) the integration, interoperability, and availability of our products with and on third-party platforms, are forward-looking statements. These forward-looking statements are subject to a number of risks, uncertainties and assumptions, including those described under the heading "Risk Factors" and elsewhere in the Quarterly Reports on Form 10-Q and Annual Reports on Form 10-K that Snowflake files with the Securities and Exchange Commission. In light of these risks, uncertainties, and assumptions, the future events and trends discussed in the Materials may not occur, and actual results could differ materially and adversely from those anticipated or implied in the forward-looking statements. As a result, you should not rely on any forwarding-looking statements as predictions of future events.

Any future product or roadmap information (collectively, the "Roadmap") is intended to outline general product direction; is not a commitment, promise, or legal obligation for Snowflake to deliver any future products, features, or functionality; and is not intended to be, and shall not be deemed to be, incorporated into any contract. The actual timing of any product, feature, or functionality that is ultimately made available may be different from what is presented in the Roadmap. The Roadmap information should not be used when making a purchasing decision. Further, note that Snowflake has made no determination as to whether separate fees will be charged for any future products, features, and/or functionality which may ultimately be made available.

The Materials may contain information provided by third-parties, including those participating in this event. Snowflake has not independently verified this information, and usage of this information does not mean or imply that Snowflake has adopted this information as its own or independently verified its accuracy.

snowflake®

Introduction to Data Clean Room

Joel Brunger
Senior Sales Engineer

DATA CLOUD WORLD TOUR

Speakers

Joel Brunger

Senior Sales Engineer

Joel is a Senior Sales Engineer focusing on Media, Entertain and Telecommunications industries with extensive deep technical knowledge and hands-on experience of managing many different data platforms from transactional to analytical technologies over the last 28 years with many different roles in service delivery, consultancy and technical pre-sales.

DATA CLOUD WORLD TOUR

AGENDA

- **Data Collaboration with Protections**
  - Benefits of Snowflake Collaboration
  - Basic Data Sharing
  - Data Protection
- **Secure Multi-party Computation**
  - Yao's Millionaire Practice
- **Global Data Clean Room**
  - Cross-Industry Use Cases
  - Example: Advertiser Direct Media Buy
  - Snowflake Enabling Features

DATA CLOUD WORLD TOUR

Data Collaboration with Protections

DATA CLOUD WORLD TOUR

ELEMENTS OF THE DATA CLOUD

PLATFORM

+

CONTENT

BENEFITS OF SNOWFLAKE COLLABORATION

Across Cloud & Region

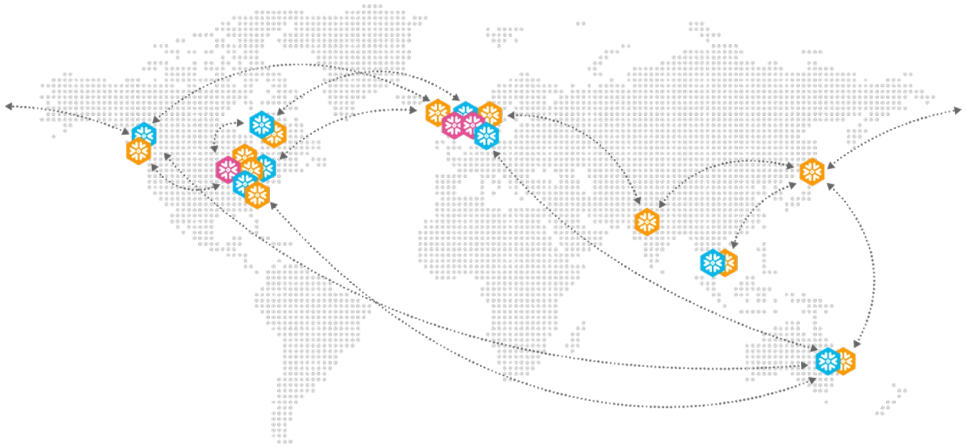
The only platform that delivers direct access to live, ready-to-query data across clouds and regions with on-demand fulfillment and no ETL

More than Data

Snowflake enables customers to collaborate with data, data services and applications including built-in usage based monetization.

Robust Data Governance

Achieve privacy-preserving collaboration with targeted discovery, revocable access and use case level auditing.



BENEFITS OF SNOWFLAKE COLLABORATION

Across Cloud & Region

The only platform that delivers direct access to live, ready-to-query data across clouds and regions with on-demand fulfillment and no ETL

More than Data

Snowflake enables customers to collaborate with data, data services and applications including built-in usage based monetization.

Robust Data Governance

Achieve privacy-preserving collaboration with targeted discovery, revocable access and use case level auditing.

DATA

Enable direct access to live, ready-to query data without ETL

DATA SERVICES

Deliver insights without exposing underlying data

APPLICATIONS

Discover, build and distribute apps that run natively within your Snowflake account

BENEFITS OF SNOWFLAKE COLLABORATION

Across Cloud & Region
The only platform that delivers direct access to live, ready-to-query data across clouds and regions with on-demand fulfillment and no ETL

More than Data
Snowflake enables customers to collaborate with data, data services and applications including built-in usage based monetization.

Robust Data Governance
Achieve privacy-preserving collaboration with targeted discovery, revocable access and use case level auditing.

BASIC DATA SHARING

```
use role accountadmin;
create share sales_share;
grant usage on database sales_db to share sales_share;
grant usage on schema sales_schema to share sales_share;
create secure view merchant_view as select * from sales;
grant select on secure view merchant_view to share sales_share;
alter share sales_share add accounts=xy12345, yz23456;
```

DATA PROTECTION: COLUMN-LEVEL

Dynamic Data Masking for column-level Data

Dynamically Mask Protected (PII, PHI)

Column Data at Query Time

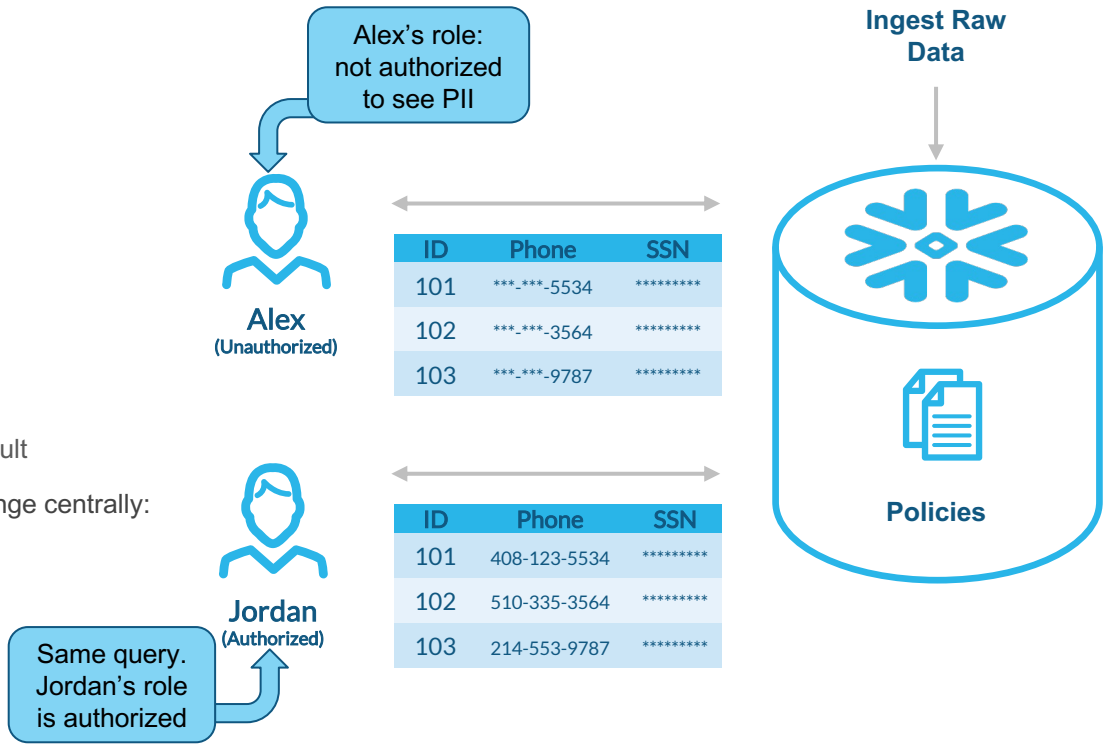
- No change to the stored data
- Mask or partial mask using constant value,
- hash, and custom functions
- Unmask for authorized users only

Policy Based Control

- Table/View owners and privileged users (such as account admin) unauthorized by default
- **Centralised** policy management, Make a change centrally: applied across any number of columns

Ease of Management

- Apply single policy to multiple columns
- Prevent secure view explosion



DATA PROTECTION: ROW ACCESS POLICIES

Dynamically Filter Rows

Centralise policies

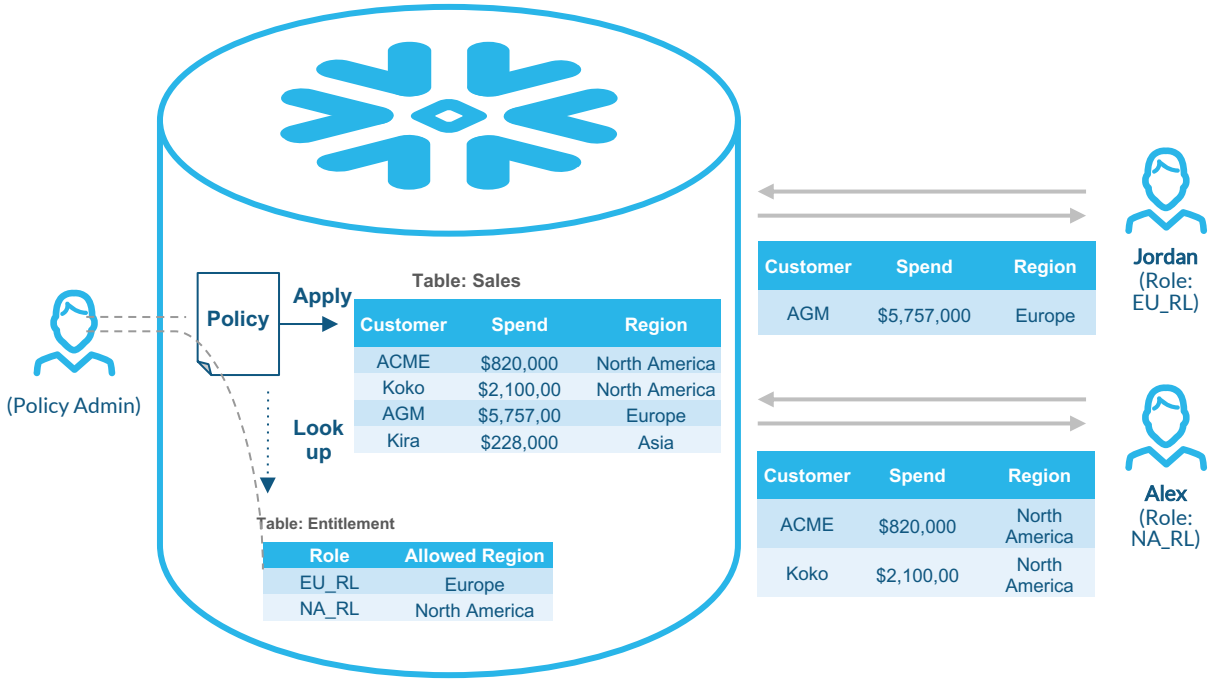
- Consistently enforced on tables, views, data shares, and external tables

Easily manage access controls

- Prevent secure view explosion
- Support diverse workloads such as External Tables and Data Sharing

Eliminate data silos

- Consolidate data and dynamically control access based on user authorization

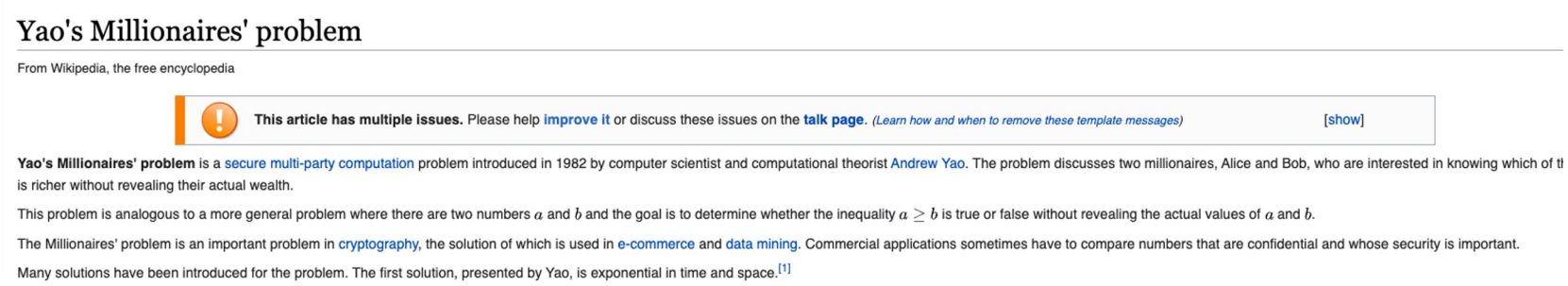


Secure Multi-party Computation

DATA CLOUD WORLD TOUR

YAO'S MILLIONAIRE PRACTICE

Yao's Millionaire:



Yao's Millionaires' problem

From Wikipedia, the free encyclopedia

This article has multiple issues. Please help **improve it** or discuss these issues on the **talk page**. *(Learn how and when to remove these template messages)* [show]

Yao's Millionaires' problem is a secure multi-party computation problem introduced in 1982 by computer scientist and computational theorist Andrew Yao. The problem discusses two millionaires, Alice and Bob, who are interested in knowing which of th is richer without revealing their actual wealth.

This problem is analogous to a more general problem where there are two numbers a and b and the goal is to determine whether the inequality $a \geq b$ is true or false without revealing the actual values of a and b.

The Millionaires' problem is an important problem in cryptography, the solution of which is used in e-commerce and data mining. Commercial applications sometimes have to compare numbers that are confidential and whose security is important.

Many solutions have been introduced for the problem. The first solution, presented by Yao, is exponential in time and space.[1]

Load the Yao's Millionaire Script from Github to your account (Enterprise or higher)

- This sample shows how to use row access policies from within one account using different roles to protect underlying data.
- Here we use Current_role() vs current_account() so the sample can be executed simply in one account (The full HOL is cross account)

Yao's Millionaires' Problem in SQL
Row Access Policy Solution

ALICE

1

```
use role alice1;
create or replace table alice1.data.my_wealth
as select 1110000 as wealth;
```

3

```
select case
when bob.wealth > alice.wealth then 'bob is
richer'
when bob.wealth = alice.wealth then 'neither
is richer'
else 'alice is richer' end
from bob1.data.my_wealth bob,
alice1.data.my_wealth alice
where exists (select table_name from
alice1.information_schema.tables where
table_schema = 'DATA' and table_name =
'MY_WEALTH' and table_type = 'BASE TABLE');
```

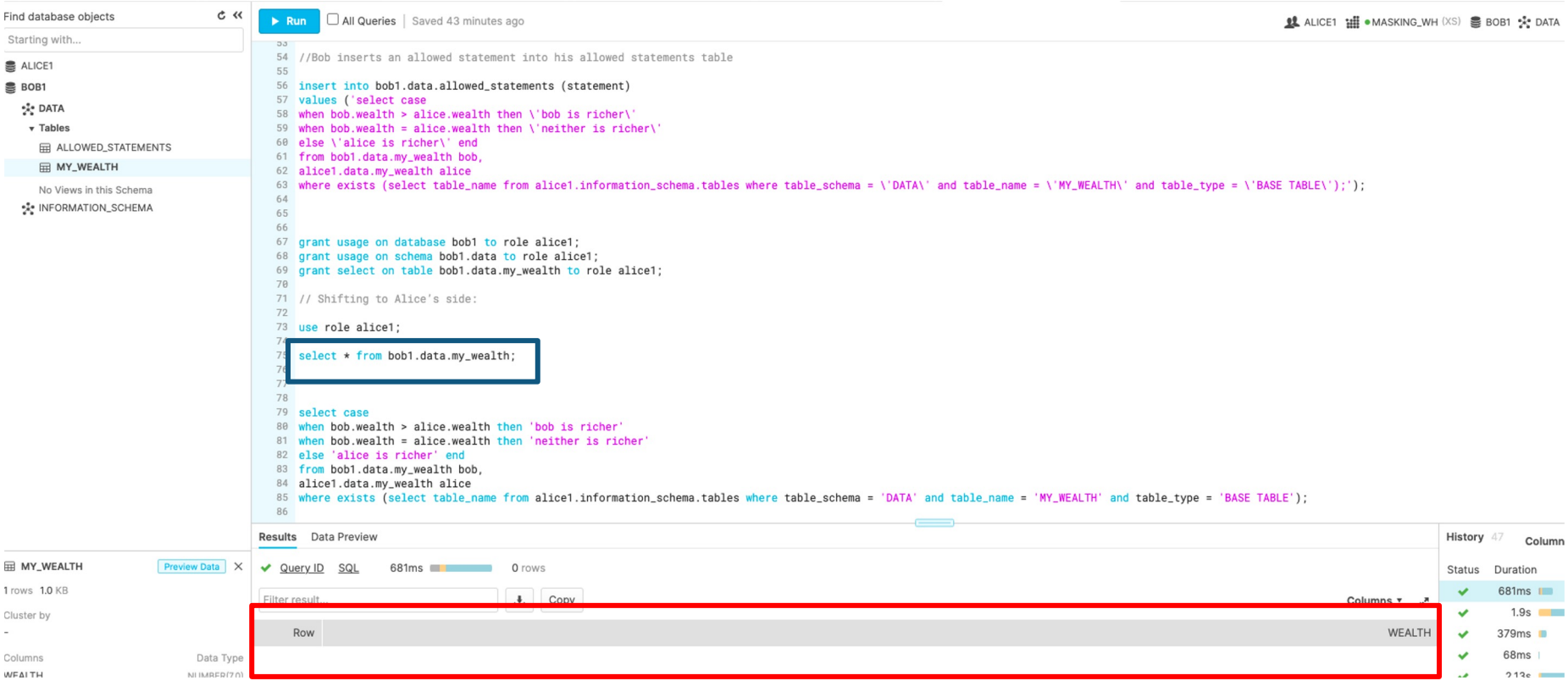
Row	CASE WHEN BOB.WEA
1	bob is richer

BOB

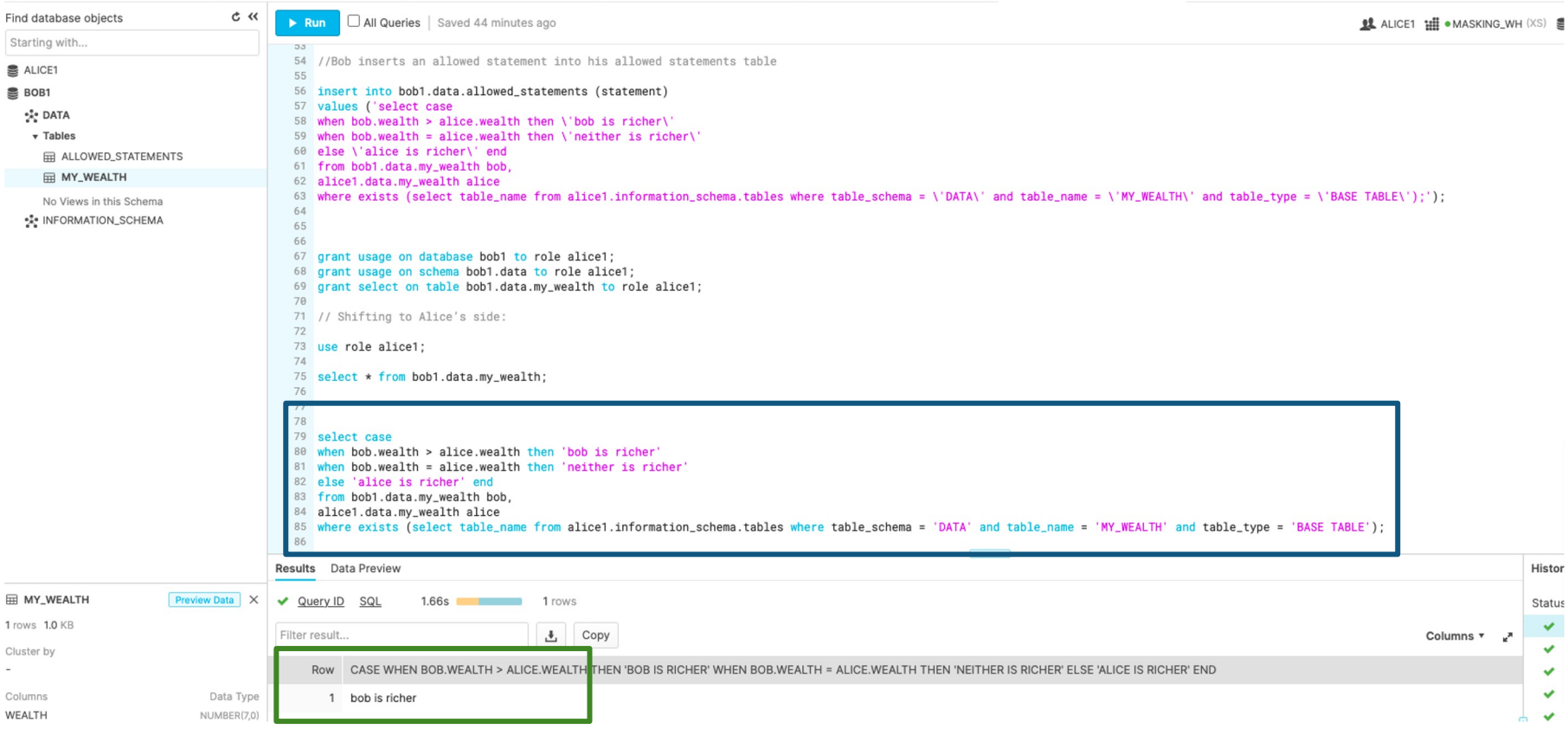
2

```
use role bob1;
create or replace table bob1.data.my_wealth as select 1250000 as
wealth;


create or replace table bob1.data.allowed_statements (
statement varchar(20000) );

CREATE OR REPLACE ROW ACCESS POLICY bob1.data.num_mask AS (wealth
integer) returns boolean ->
   current_role() IN ('ACCOUNTADMIN','BOB1')
     or exists  (select statement from
bob1.data.allowed_statements where statement=current_statement()
);


alter table bob1.data.my_wealth add row access policy
bob1.data.num_mask on (wealth);


insert into bob1.data.allowed_statements (statement)
values ('select case
when bob.wealth > alice.wealth then \'bob is richer\'
when bob.wealth = alice.wealth then \'neither is richer\'
else \'alice is richer\' end
from bob1.data.my_wealth bob,
alice1.data.my_wealth alice
where exists (select table_name from
alice1.information_schema.tables where table_schema = \'DATA\'
and table_name = \'MY_WEALTH\' and table_type = \'BASE
TABLE\');');
```

ALICE CAN'T SEE BOB'S PII



ALICE CAN RUN AN APPROVED QUERY



Yao's Millionaires' Problem in SQL Masking Policy Solution *with Defenses*

full example here

ALICE

1

```
use role alice1;
create or replace table alice1.data.my_wealth
as select 1110000 as wealth;
```

2

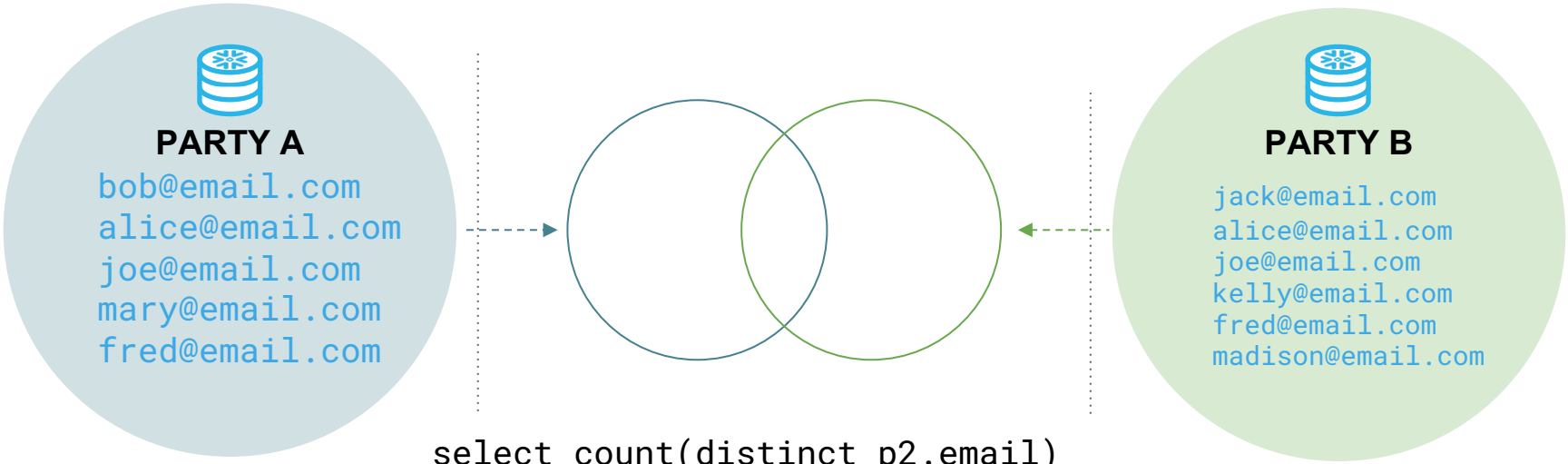
BOB

```
use role bob1;
create or replace table bob1.data.my_wealth as select 1250000 as
wealth;
create or replace secure view bob1.data.my_wealth_v as select *
from bob1.data.my_wealth;

create or replace masking policy bob1.data.millionarires_mask as
(val integer) returns integer ->
  case
    when current_statement() =
'select case
when bob.wealth > alice.wealth then \'bob is richer\'
when bob.wealth = alice.wealth then \'neither is richer\'
else \'alice is richer\' end
from bob1.data.my_wealth_v bob, alice1.data.my_wealth
at(timestamp => '2022-10-01 09:03:00 HST') alice
where exists (select table_name from
alice1.information_schema.tables where table_schema = \'DATA\' and
table_name = \'MY_WEALTH\' and table_type = \'BASE TABLE\');'
then val else 1/0 end;

alter view bob1.data.my_wealth_v modify column wealth set masking
policy bob1.data.millionarires_mask;

grant select on view bob1.data.my_wealth_v to role alice1;
```

3

```
use role alice1;

select case
when bob.wealth > alice.wealth then 'bob
is richer'
when bob.wealth = alice.wealth then
'neither is richer'
else 'alice is richer' end
from bob1.data.my_wealth_v bob,
alice1.data.my_wealth
at(timestamp => '2022-10-01 09:03:00 HST')
alice
where exists (select table_name from
alice1.information_schema.tables where
table_schema = 'DATA' and table_name =
'MY_WEALTH' and table_type = 'BASE TABLE')
// bob is richer

update my_wealth set wealth = 3000000;

// re-run above select case...
// bob is richer (still!)
```

WHAT IF I WANT TO KNOW HOW MANY CUSTOMERS WE HAVE IN COMMON?




```
select count(distinct p2.email)
   from party1.data.customers p1
   join party2.data.customers p2 on
    (p1.email = p2.email)
   having count(distinct p2.email) >= 2;
// result: 3
```

Global Data Clean Room

DATA CLOUD WORLD TOUR

SNOWFLAKE FOR DATA CLEAN ROOMS

SECURE

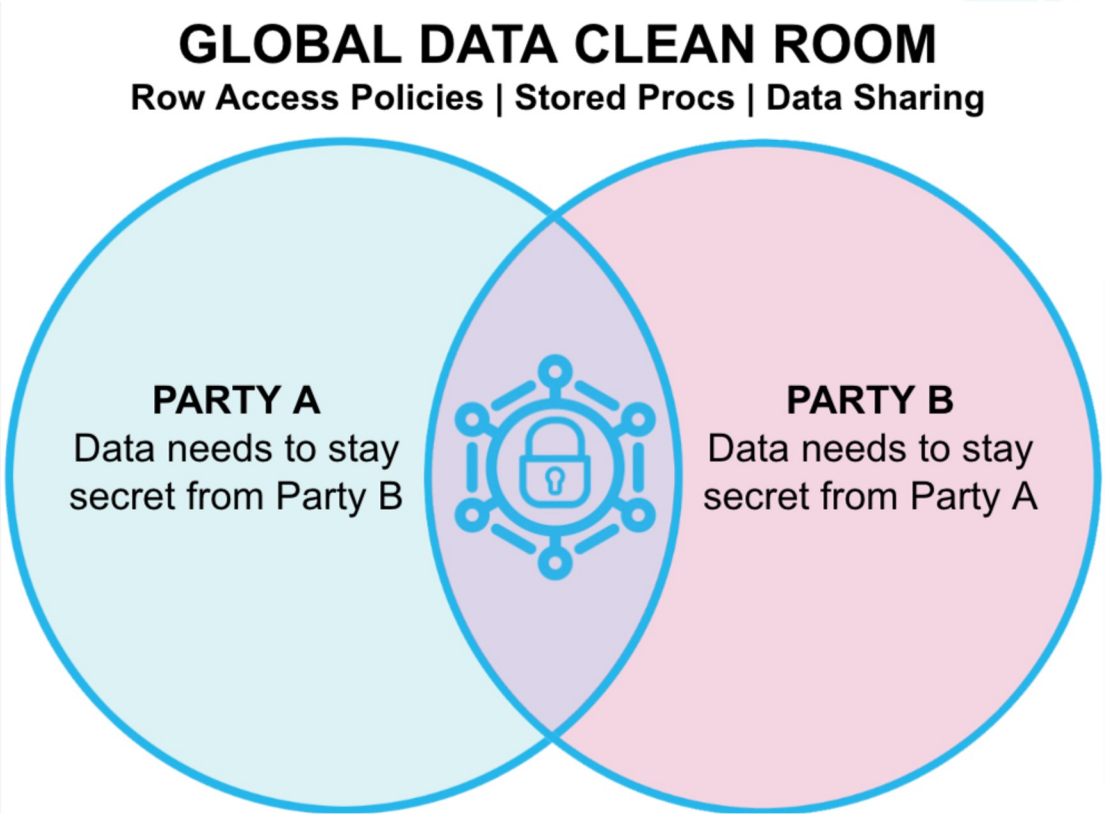
Answer questions of data while preventing drill down to PII

EASY

Overlap & join without copying, moving, or sharing the underlying data

POWERFUL

Enable large amounts of compute against other company's data



CROSS-INDUSTRY USE CASES

Advertising

- Audience insights
- Segmentation
- Measurement (reach & frequency analysis)
- Multi-touch attribution
- Activation

Retail & CPG

- Optimize their marketing and promotional activities
- Product decisions (new product categories to enter)
- Supply chain decisions (how much to distribute to which distribution center).

Life Sciences & Healthcare

- Clinical trials analysis
- Population selection
- Efficacy measurement
- Safety analysis
- Drug discovery

Financial Services

- Fraud detection modeling
- Anti-Money Laundering (AML) compliance.
- Securely share sensitive position and portfolio data for risk analysis, exposure analysis, regulatory, and settling activities

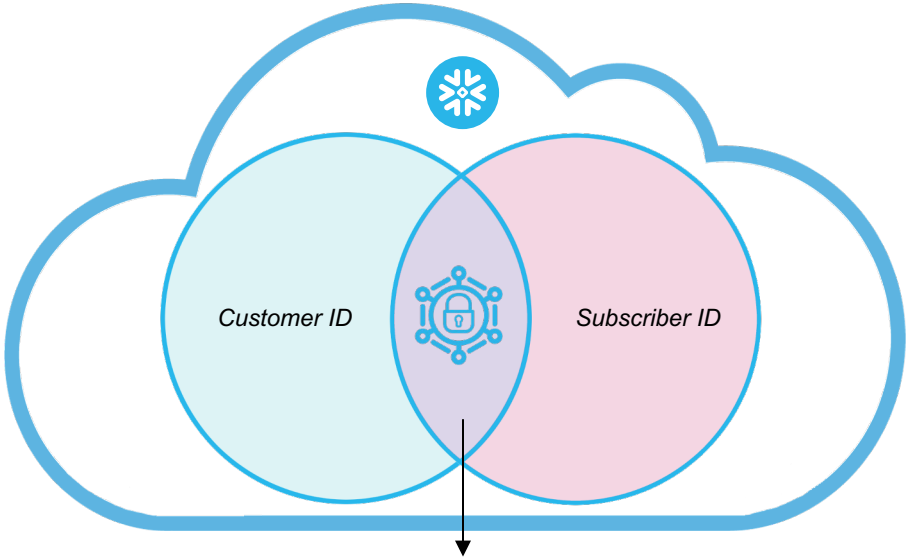
Government

Allow different agencies to share data with each other, e.g. surveillance, communications, watch-lists, monitoring, aerial images, video, audio, language translation.

EXAMPLE: ADVERTISER DIRECT MEDIA BUY

Brand / Advertiser

- **Objective**: Better predict return on advertising spend and attain timely, granular attribution analytics
- **Data**: Sales, web traffic, customer segments



Media Publisher

- **Objective**: Sell targeted inventory w/privacy compliant, real time, scalable solution
- **Data**: Subscribers and viewing patterns

Program	High Mileage Runner	College Basketball	U18 Soccer Players	65+ Tennis	High School Football
Whale Week	2500	2000	500	3000	5500
Friendliest Catch	5500	3000	404	220	2000
Cyber Rush	2200	500	100	8000	500
Rulebusters	3300	7000	850	7100	150
Cakemakers	~~25~~	~~60~~	~~8~~	~~14~~	~~32~~

Redacted to reduce risk of re-identification

DCR ENABLING FEATURES

Row Access Policies And SQL Query Templates

Match customer data without exposing PII

Est GA: Q3 2023

Native Applications

Simplified deployment of application capabilities and protected insights

Stored Procedures

Generate and validate query requests
Languages: SQL, Javascript, Python

Est GA: Q2 2023

Python UDF enables JINJA Templates

Substitutable Query Templates

Secure Data Sharing

Automatically and securely share tables between multiple snowflake accounts

Est PrPr: Q4 2023

Project & Aggregation Constraints

Join on fields without being able to view them.
Allow queries about groups, not individuals

QUERY TEMPLATE EXAMPLES

- **Audience Overlap and Segment Creation** for analyzing aggregate groupings of overlapping customers and building segments for audience planning
- **Customer Enrichment** to receive enriched data on overlapping customers
- **Measuring Campaign Success** and return on advertising spend
- 

Finding Lookalike Segments to grow your target audience

EXAMPLE – JINJA CUSTOMER OVERLAP TEMPLATE

```
select
    identifier({{ dimensions[0] }})
    {% for dim in dimensions[1:] %}
    , identifier({{ dim  }})
    {% endfor %}
    , count(distinct p.email) as overlap
from
    {{ app_data | sqlsafe }}.cleanroom_provider_data_p
    {{ consumer_db | sqlsafe }}.{{ consumer_schema | sqlsafe }}.{{ consumer_table | sqlsafe }} at(timestam
where
    c.{{ consumer_join_field | sqlsafe }} = p.email
    and exists (select table_name from {{ consumer_db | sqlsafe }}.information_schema.tables where table_s
table_name = upper('{{ consumer_table| sqlsafe }}') and table_type = 'BASE TABLE')
    {% if  where_clause  %}
    and ( {{ where_clause | sqlsafe }} )
    {% endif %}
group by
    identifier({{ dimensions[0]  }})
    {% for dim in dimensions[1:] %}
    , identifier({{ dim }})
    {% endfor %}
having count(distinct p.email)  > 25
order by count(distinct p.email) desc;
```

EXAMPLE – SQL CUSTOMER OVERLAP TEMPLATE

```
select
    @dimensions
      count(distinct p.email) as overlap
from
    @app_data.cleanroom.provider_data p,
    @consumer_db.@consumer_schema.@consumer_table at(timestamp => '@at_timestamp'::timestamp_tz)
where
    c.@consumer_join_field = p.email
    and exists (select table_name from @consumer_db.information_schema.tables where table_schema
    and table_name = upper('@consumer_table') and table_type = 'BASE TABLE') @where_clause
@group_by
having count(distinct p.email)  > 25
order by count(distinct p.email) desc;
```

EXAMPLE – EXECUTE REQUEST

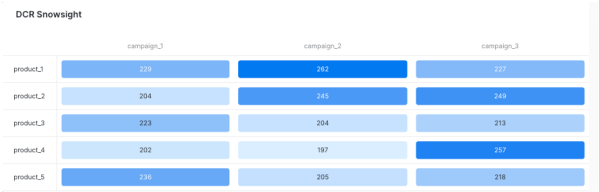
```
call dcr_JINJANDP_consumer.JNA54922_schema.request('customer_overlap',
        object_construct(
            'dimensions',array_construct('c.pets', 'p.status'),
            'where_clause',' c.PETS <> $$BIRD$$ '
            )::varchar, NULL, NULL);
```

EXAMPLE - CONFIGURED TEMPLATE

```
select
    identifier('c.pets')
    , identifier('p.status')
    , count(distinct p.email) as overlap
from
    dcr_JINJANDP_app.cleanroom.provider_data p,
    dcr_JINJANDP_consumer.mydata.customers at(timestamp => '2022-09-01 13:46:44.005 -070
where
    c.email = p.email
    and exists (select table_name from dcr_JINJANDP_consumer.information_schema.tables w
and table_type = 'BASE TABLE')
    and (  c.PETS <> $$BIRD$$  )
group by
    identifier('c.pets')
    , identifier('p.status')
having count(distinct p.email)  > 25
order by count(distinct p.email) desc;
```

GLOBAL DATA CLEAN ROOM DEMO

SNOWFLAKE SNOWSIGHT UI



STREAMLIT UI

snowflake
Data Clean Room

THANK YOU

DATA CLOUD WORLD TOUR

APPENDIX

DATA CLOUD WORLD TOUR