Real ID in Snowflake Application

January 2024

Table of Contents

Copyright Information

Real ID - PREREQUISITES

The following information is needed to set up Acxiom's Real ID:

- A Snowflake account in one of the supported regions
- Only U.S.-based names and addresses are supported
- **Role**: The instructions in this guide must be performed using the **accountadmin** role, or a role with the following privileges:
    - Create application, databases, shares, schema, table, and role
    - Grant privileges
    - Import shares
    - Execute tasks
    - Install and manage applications
- **Cloud and Region**: Your data needs to be in **AWS US East 1** to use Real ID. If you're on a different cloud or region, let your Acxiom representative know as we are rolling out support for other regions.
- **Account Locator**: Have the Account Locator number ready for your Snowflake account (referred to as **{consumer_account_locator}**).
- **Input View:** Must be able to conform to Acxiom's input structure. A view on top of your data that conforms to our structure is the recommended approach.

Field	Type	Required	Description
recordId	String	True	This is the primary key back to this table / view
first	String	False	This is the first name field name in the input table
middle	String	False	This is the middle name field name in the input table
last	String	False	This is the last name field name in the input table
suffix	String	False	This is the name suffix field name in the input table
line1	String	False	This is the first street address line field name in the input table
line2	String	False	This is the second street address line field name in the input table
city	String	False	This is the city field name in the input table
state	String	False	This is the state field name in the input table
zip	String	False	This is the zip field name in the input table
email	String	False	This is the email field name in the input table
phone	String	False	This is the phone field name in the input table

- **Event Table**: Event table should be configured for the account where the application is installed. This allows you to share event logging to the provider. See additional details: https://other-docs.snowflake.com/en/native-apps/consumer-enable-logging
- **Run Database Creation Script:** Run Acxiom's usage tracking script that creates a database for usage tracking and passing of user selected IDs back to Acxiom for various functions.
- **Share Usage Database:** Share the usage database back to Acxiom for various downstream functions.

Client Reference Chart

Below you will see latency testing at different levels of input data and warehouse configurations.

This data is provided to help you decide what type of warehouse you should utilize.
Your times will vary.

Input Records	1,000	10,000	100,000	1,000,000	10,061,200	100,637,906	201,278,247	1,006,391,235
Warehouse	Overall	Overall	Overall	Overall	Overall	Overall	Overall	Overall
Small	0:40:43		0:44:13	0:46:01	1:17:11	6:01:51		
Medium	0:20:49	0:21:51	0:21:35	0:23:48	0:38:35	3:07:22	5:44:21	
Large	0:10:45	0:11:15	0:11:05	0:12:27	0:20:56	1:39:02	3:09:43	14:38:51
X-Large	0:04:53	0:05:04	0:05:13	0:06:26	0:11:11	0:56:42	1:48:10	8:40:21
2X-Large	0:01:16	0:02:57	0:03:26	0:04:36	0:06:41	0:37:43	1:12:03	5:40:57
3X-Large					0:04:42	0:27:53	0:53:23	4:16:11
4X-Large								3:28:22
Medium - Snowpark			0:23:58	0:27:22	1:14:40	9:02:20	17:53:33	
Large - Snowpark			0:11:31	0:13:50	0:38:11	4:41:31	9:16:56	23:06:31
X-Large - Snowpark			0:05:01	0:06:45	0:19:57	2:28:13	4:51:43	18:00:36
2X-Large - Snowpark			0:03:00	0:04:18	0:11:11	1:22:06	2:42:00	13:05:47

3X-Large - Snowpart					0:08:28	0:53:15	1:41:08	7:48:16
4X-Large - Snowpark								5:08:24

Findings

- SQL Acceleration did not appear to help in our testing with this work load
- Snowpark Optimized warehouses did not appear to help in our testing with this work load.
- You will note that we canceled two of the jobs after they ran longer than the normal warehouse. Items in Red

Real ID - SET UP

Snowflake documentation on installing an application from a listing can be found here, for general knowledge: https://other-docs.snowflake.com/en/native-apps/consumer-installing#installing-an-application-from-a-privately-shared-listing

The following set-up process only needs to be completed for initial set-up.

REQUEST AND INSTALL REAL ID FROM MARKETPLACE

1. Navigate to Snowflake Marketplace and search for "Acxiom Real ID."
2. Select the Acxiom Real ID tile listing and select "Request App."
3. Acxiom will receive and process the request.
4. Upon approval of the request, Acxiom will proceed with required setup steps to enable use of the Real ID application.
    1. This process may take up to 5 business days to complete.
5. Once setup is complete, visit 'Recently Shared With You' and navigate to 'Acxiom Real ID.'
6. Select **Install App** and follow the click through instructions to complete the installation https://other-docs.snowflake.com/en/native-apps/consumer-installing#installing-an-application-from-a-privately-shared-listing

    Run the following commands:

    1. Install the application
    2. Grant application 'create database on account' privilege

    Execute the following queries.

```
--/ Set the variable to what you call the application when you installed it
SET APPLICATION_NAME = NAME_OF_APP;


GRANT CREATE DATABASE ON ACCOUNT TO APPLICATION
IDENTIFIER($APPLICATION_NAME);
```

    3. Run initialization stored procedure to create a usage database for sharing back to Acxiom

    Execute the following queries.

```
--/ Set the variable to what you call the application when you installed it
SET APPLICATION_NAME = NAME_OF_APP;
USE IDENTIFIER($APPLICATION_NAME);

CALL REALID.INITIALIZE_ACXIOM_SHARE_DB();
```

4. Update permissions on the usage database and create share ***This step requires accountadmin role or permissions as outlined in the prerequisites section of this document.*
    i. Execute the following queries.

```
--/ Set the variable to what you call the application when you installed it

SET APPLICATION_NAME = NAME_OF_APP;

SET REALID_SHARE_DB_NAME = CONCAT($APPLICATION_NAME, '_SHARE')

GRANT OWNERSHIP ON DATABASE IDENTIFIER($REALID_SHARE_DB_NAME) TO ROLE
NAME_OF_ROLE REVOKE CURRENT GRANTS;

GRANT OWNERSHIP ON ALL SCHEMAS IN DATABASE IDENTIFIER($REALID_SHARE_DB_NAME)
TO ROLE NAME_OF_ROLE REVOKE CURRENT GRANTS;

GRANT OWNERSHIP ON ALL TABLES IN DATABASE IDENTIFIER($REALID_SHARE_DB_NAME)
TO ROLE NAME_OF_ROLE REVOKE CURRENT GRANTS;


EXECUTE IMMEDIATE $$

    DECLARE

        my_realid_app_name_v VARCHAR;

    BEGIN

        my_realid_app_name_v := $APPLICATION_NAME;


        EXECUTE IMMEDIATE 'CREATE OR REPLACE DATABASE ROLE ' ||
:my_realid_app_name_v || '_SHARE.REALID_SHARE_ROLE';


        EXECUTE IMMEDIATE 'GRANT USAGE ON ALL SCHEMAS IN DATABASE ' ||
:my_realid_app_name_v || '_SHARE TO DATABASE ROLE ' || :my_realid_app_name_v
|| '_SHARE.REALID_SHARE_ROLE';

        EXECUTE IMMEDIATE 'GRANT SELECT ON ALL TABLES IN DATABASE ' ||
:my_realid_app_name_v || '_SHARE TO DATABASE ROLE ' || :my_realid_app_name_v
|| '_SHARE.REALID_SHARE_ROLE';


        EXECUTE IMMEDIATE 'GRANT USAGE ON DATABASE ' || :my_realid_app_name_v
|| '_SHARE TO APPLICATION ' || :my_realid_app_name_v;

        EXECUTE IMMEDIATE 'GRANT USAGE ON ALL SCHEMAS IN DATABASE ' ||
:my_realid_app_name_v || '_SHARE TO APPLICATION ' || :my_realid_app_name_v;

        EXECUTE IMMEDIATE 'GRANT SELECT, INSERT, UPDATE, DELETE, TRUNCATE ON
ALL TABLES IN DATABASE ' || :my_realid_app_name_v || '_SHARE TO APPLICATION '
|| :my_realid_app_name_v;

    CREATE SHARE ACXIOM_REALID_USAGE;

    GRANT USAGE ON DATABASE IDENTIFIER($REALID_SHARE_DB_NAME) TO SHARE
ACXIOM_REALID_USAGE;

    EXECUTE IMMEDIATE 'GRANT DATABASE ROLE ' || :my_realid_app_name_v ||
'_SHARE.REALID_SHARE_ROLE TO SHARE ACXIOM_REALID_USAGE';

    END;

$$ ;


--/ The share has been created and executing this still will create share to
Acxiom's snowflake account for usage information

ALTER SHARE ACXIOM_REALID_USAGE ADD ACCOUNTS = todo
```

5. Grant application read access to input tables.

Ex

ecute the following queries.

```
--/ Set the variable to what you call the application when you installed it
SET APPLICATION_NAME = NAME_OF_APP;


--/ Grant Access to the database that contains the input data
GRANT USAGE ON DATABASE MY_DATABASE TO APPLICATION
IDENTIFIER($APPLICATION_NAME);


--/ Grant Access to the schema in the above database that contains the input
data
GRANT USAGE ON SCHEMA MY_DATABASE.SCHEMA_NAME TO APPLICATION
IDENTIFIER($APPLICATION_NAME);


--/ Grant Access to the actual table that contains the input
--/ You will change "TABLE" to "VIEW" if you are passing in a view
GRANT SELECT ON TABLE MY_DATABASE.SCHEMA_NAME.TABLE_WITH_DATA TO APPLICATION
IDENTIFIER($APPLICATION_NAME);
```

6. Call the application stored procedure.

Execute the following queries.

```
USE IDENTIFIER($APPLICATION_NAME);


CALL REALID.IDGENERATION('MY_DATABASE.SCHEMA_NAME.TABLE_WITH_DATA');


--/ The output table uses the input table name and adds _realids to the end
SELECT * FROM REALID_RESULTS.TABLE_WITH_DATA_realids;
```

Real ID - Use ID Generation

The following steps walk you through how to use ID Generation.

Generate Real IDs for Your PII Data

1. Use this query to create a view of a dataset within your Snowflake account to pass through to the generate ID function. All fields listed below must be provided exactly as outlined, regardless of data availability. If you do not have data to populate a given field, it must still be defined in your view or table in order to pass through empty or null fields.

```
CREATE VIEW <YourViewName> AS
SELECT
    <recordId> STRING AS recordId,
    <first> STRING AS first,
    <middle> STRING as middle,
     <last> STRING as last,
     <suffix> STRING as suffix,
     <line1> STRING as line1,
     <line2> STRING as line2,
     <city> STRING as city,
     <state> STRING as state,
     <zip> STRING as zip,
     <email> STRING as email,
     <phone> STRING AS phone
FROM <YourTableName>;
```

This query creates the view in the required format to pass through the idGeneration function to get your Real IDs.

NOTE
Your recordId must be unique per record, this will act as your primary key in specific scenarios.

Below are explanations of each of the values used in the query above.

- **recordID:** This will be your primary key for your PII records. For enterprise use cases, you can utilize this field to join the RealIDs provided by the idGeneration function back to your PII records.

- Note: The output of the idGeneration function will exclude this field in free trial use cases.

- **first**: first name field, ex: Jane
- **middle:** middle name or initial, ex: R or Renee
- **last**: last name field, ex: Smith
- **suffix:** name suffix such as Jr. Sr, III.
- **line1**: This is the first street address line field, ex: 123 E. Main St.
- **line2**: This is the second street address line field, ex: Apt. 101
- **city:** city field name, ex: Atlanta
- **state**: State field, accepts abbreviation or state name, ex: CO or Colorado
- **zip**: 5-digit postal zip code, ex: 80922
- **email**: email field, ex: example@email.com
- **phone**: phone number field, remove parentheses, ex: 123-456-7812

1. Pass your view as a parameter to the idGeneration() stored procedure:

```
idGeneration(<YourViewName>);
```

The output of the idGeneration function will be a table that carries your view name, appended with _realids. Example: idGeneration(SampleView), your table of generated Real IDs will be called SampleView_realids.

The resulting table will contain the following information:

Field Name	Type	Description	Conditions
recordid *not available to all accounts	Varchar2(2000)	The record id of the input table record (primary key)	This field will only be returned for 'Enterprise' subscriptions. Trial subscriptions will not receive this field back.
personId		Acxiom's person id assigned to this record	Minimum required fields must be provided to generate a personId. Name and at least one of the following must be provided to generate a personId: Address, email or phone.
addressId		Acxiom's address id assigned to this record	Minimum required fields must be provided to generate an addressId. If address fields are not populated, an addressId will not be provided.
householdId		Acxiom's household id assigned to the record	Minimum required fields must be provided to generate householdId. Name and one of the following fields: Address, email or phone.

This output table will be available in your account, and the IDs provided will be needed to utilize the additional feature functions available in the Real ID application.

NOTE
For trial users, records will not be returned in the same order they were provided in, nor will the recordID be returned.

Real ID – Data Exploration

The following steps explain how to use the Data Exploration feature functionality.

Prerequisites

- idGeneration() function must already be complete, and output table from that function available
- personID field from idGeneration() function must be provided in the input table/view passed into the function
- viewname must be alphanumeric or contain only the following special characters:
    - a dash (-)
    - an underscore (_)
    - a period (.)
- a minimum of 10 records are required

Run the Data Exploration Stored Procedure

1. Pass your table name as a parameter to the EXPLORATION() stored procedure:

```
EXPLORATION(<SampleView_realids>);
```

NOTE
The REAL IDs output EXPLORATION() stored procedure will be unique to this use case and will not match back to the REAL IDs that were input into the function. The IDs will also be scrambled and provided back in a random order in output to intentionally prevent re-identification of the data attributes provided within this application.

Explore the Data Attributes

The output of the EXPLORATION() stored procedure will be a table that maintains the input dataset name, followed by _exploration. Example: EXPLORATION(data); would output a table called data_exploration.

The table will include:

Field Name	*Type*	*Description*
personID	*string*	*Acxiom's anonymous person ID assigned to this record. This ID will be uniquely encoded and not match the personIDs provided in the idGeneration() function, nor will the records return in the same order.*
Data attributes …		*Each attribute from Acxiom authorized for use for anonymous data analysis*

Real ID – Activation

The following steps walk through how to use the Activation feature functionality.

Prerequisites

- idGeneration() function must already be complete, and output table from that function available
- personID field from idGeneration() function must be provided in the input table/view passed into the function.
- viewname must be alphanumeric or contain only dashes, underscores or periods.
- Destination setup steps have been completed per the vendors' requirements.
- A minimum of 10 records is required.

xRun the Activation Stored Procedure

1. Below are the parameters for activation stored procedure:

Parameter	Data Type	Required	Description
realIDView	Varchar	True	The name of a view or table containing the personIDs generated by the idGeneration() function
Name	Varchar	True	Distribution name
Description	String	True	Description of Distribution
Destination	Varchar	True	Destination
Lookalike	Boolean	True	Defaults to False
Lookalike_Min	Integer	False	
Lookalike_Max	Integer	False	
Lookalike_rank_threshold	Integer	False	1-100. 1= most likely, 100= least likely

2. Pass your table name and distribution options as parameters to the activation(inputTable,distributionOptions) stored procedure:

<value> = a placeholder for value.
<app dataset> = a placeholder for dataset name.
<integer value> = a placeholder for an integer value
<> are part of the placeholder and should not be included

Facebook
Parameter da5c425d2e1947419eaea0b1a1fd20c2 = Destination Account ID

Yahoo
Parameter da5c425d2e1947419eaea0b1a1fd20c2 = Destination Account ID
Parameter 59dfea60fd9e4e539b5de41ff7cca1da = Campaign end date in UNIX epoch
(e.g. 1704085200 for Jan 1, 2024 - do not exceed 90 days)

Amazon
Parameter 33429c3a494940c785fde53f3fda63ce = Price
Parameter c111a2df8ef34b56ac02e76f414e9f40 = Publisher Account ID

Facebook

```
call REALID_MATCH.REALID.ACTIVATION(
-- realIDView
'REALID_RESULTS.<app dataset>',
-- destination
'{ \"destinations \": [
    {\"id \":  \"92f11b6061114609a9eee5732bfec3f6",
      \"parameters\": [
          {\"id \":  \"da5c425d2e1947419eaea0b1a1fd20c2",
           \"values \": [
               {\"value \":  \"<value>\"}]
          }]
       }]
}',
-- name
'name',
-- description
'description');
```

Yahoo

```
call REALID_MATCH.REALID.ACTIVATION(
-- realIDView
'REALID_RESULTS.<app dataset>',
-- destination
'{ \"destinations \": [
 {\"id \":  \"9a74fe9a2f8343e989b5553ae8ec5f97",
    \"parameters\": [
     {\"id \":  \"59dfea60fd9e4e539b5de41ff7cca1da",
        \"values \": [
            {\"value \":  \"<value>\"}]
     },
     {\"id \":  \"da5c425d2e1947419eaea0b1a1fd20c2",
```

```
          \"values\": [
              {\"value\":   \"<value>\"}]
       }]
    }]
}',
-- name
'name',
-- description
'description');


-------------------------------------------------------------------------------
Amazon
-------------------------------------------------------------------------------
call REALID_MATCH.REALID.ACTIVATION(
-- realIDView
'REALID_RESULTS.<apptaset>',
-- destination
'{ \"destinations   \": [
        {\"id \":   \"ed63b58c462c4146bbc672d5530c0f94",
         \"parameters\": [
          {\"id \":   \"33429c3a494940c785fde53f3fda63de",
             \"values \": [
                  {\"value \":   \"<value>\"}]
               },
             {\"id \":   \"c111a2df8ef34b56ac02e76f414e9f40",
                \"values \": [
                  {\"value \":   \"<value>\"}]
             }]
     }]
}',
-- name
'name',
-- description
'description');


-------------------------------------------------------------------------------
```

The following procedure calls utilize our look - alike modeling and accept additional input parameters.

Minimum Records - The result of the model/score needs to have at least these many records, otherwise the job fails

Maximum Records The result of the model/score needs to not contain more than these many records

RankThreshold - The include flow will model, then score. Based on that score, records will be pulled from the first several tile buckets until all records have been claimed from all ventiles up to and including the worst case OR maximum records value has been reached

```
-------------------------------------------      ----------------------------------------
Facebook look- alike call
-------------------------------------------------------------------------------
call REALID_MATCH.REALID.ACTIVATION(
-- realIDView
'REALID_RESULTS.<app dataset>',
```

```
-- destination
'{ \"destinations   \": [
      {\"id \":   \"235ece9e0cec423b985d448a623a1e77
         \"parameters\": [
            {\"id \":   \"da5c425d2e1947419eaea0b1a1fd20c2
               \"values \": [
                  {  \"value \":   \"<value>\"}]
            } ]
      }
   ]
}',
-- name
'name',
-- description
'description',
-- lookALike
true,
-- minRecords
<integer value>,
-- maxRecords
<integer value>,
-- rankThreshold
<integer value>
);


--------------------------------------------------------------------------------
Yahoo look-alike call
--------------------------------------------------------------------------------
call REALID_MATCH.REALID.ACTIVATION(
-- realIDView
'REALID_RESULTS.<app dataset>',
-- destination
'{ \"destinations   \": [
  {\"id \":   \"9a74fe9a2f8343e989b5553ae8ec5f97,
     \"parameters\": [
      {\"id \":   \"59dfea60fd9e4e539b5de41ff7cca1da",
         \"values \": [
            {\"value \":   \"<value>\"}]
      },
      {\"id \":   \"da5c425d2e1947419eaea0b1a1fd20c2
         \"values \": [
            {\"value \":   \"<value>\"}]
      }]
   }]
}',
-- name
'name',
-- description
'description',
-- lookALike
true,
-- minRecords
<integer value>,
-- maxRecords
<integer value>,
```

```
-- rankThreshold
<integer value>
);


--------------------------------------------------------------------------------
Amazon look-alike call
--------------------------------------------------------------------------------
call REALID_MATCH.REALID.ACTIVATION(
-- realIDView
'REALID_RESULTS.<app dataset>',
-- destination
'{  \"destinations   \": [
     {\"id \":   \"35c213771814481492f88593a9c0a0a1\",
       \"parameters\": [
              {\"id \":   \"33429c3a494940c785fde53f3fda63de\",
                \"values \": [
                    {\"value \":   \"<value>\"}
                ]},
           {\"id \":   \"c111a2df8ef34b56ac02e76f414e9f40\",
            \"values \": [
                {\"value \":   \"<value>\"}
            ]}
     ]}
]}',
-- name
'name',
-- description
'description',
-- lookALike
true,
-- minRecords
<integer value>,
-- maxRecords
<integer value>,
-- rankThreshold
<integer value>
);
```

NOTE
Activation is available to the following distribution platforms through Acxiom's Real ID application:
Facebook
Yahoo
Amazon

Real ID – Data Portrait Analysis (DPA)

The following steps explain how to use the DPA feature functionality.

Prerequisites

- idGeneration() function must already be complete and output table from that function available.
- personID field from idGeneration() function must be provided in the input table/view passed into the function.
- viewname must be alphanumeric or contain only dashes, underscores or periods.
- A minimum of 15,000 records is required.
- If the file size exceeds 100,000 records, it will be sampled down to 100,000 records for processing into the DPA

Run the DPA Stored Procedure

1. Pass your table name as a parameter to the DPA() stored procedure:

```
DPA(<inputTable>);
```

NOTE
You will need to utilize the login details for your Looker account provided in your welcome email to access the DPA details. The DPA portal offers a robust, intuitive UI to navigate through the provided DPA.