Onboarding Applications, Correlation, and Data Transformation

Fundamentals of IdentityIQ Implementation

IdentityIQ

Overview

Onboarding Applications, Correlation, and Data Transformation

- Process of Onboarding Additional Applications
- Account Correlation
- Application Configuration
  - Connector Types
  - Connector Configuration
- Logical Applications
- Multiplex Applications

Next Steps

Data Collection

Identity Cubes have been created, next load additional non-authoritative applications

- Collect information for additional applications
  - Implementer and application owners will require several iterations of meetings on data
- Define and agree upon data format and connectivity mechanism
  - Direct Connect
  - Flat File
  - JDBC
- Ensure understanding and definition of entitlement data and hierarchy
  - What attributes should be Certified/used for Role Mining?
  - What attributes should get loaded into Entitlement Catalog?
- Capture and analyze data aggregation schedules and dependencies
- Leverage *Onboarding Applications Checklist* (available on Compass)

SailPoint

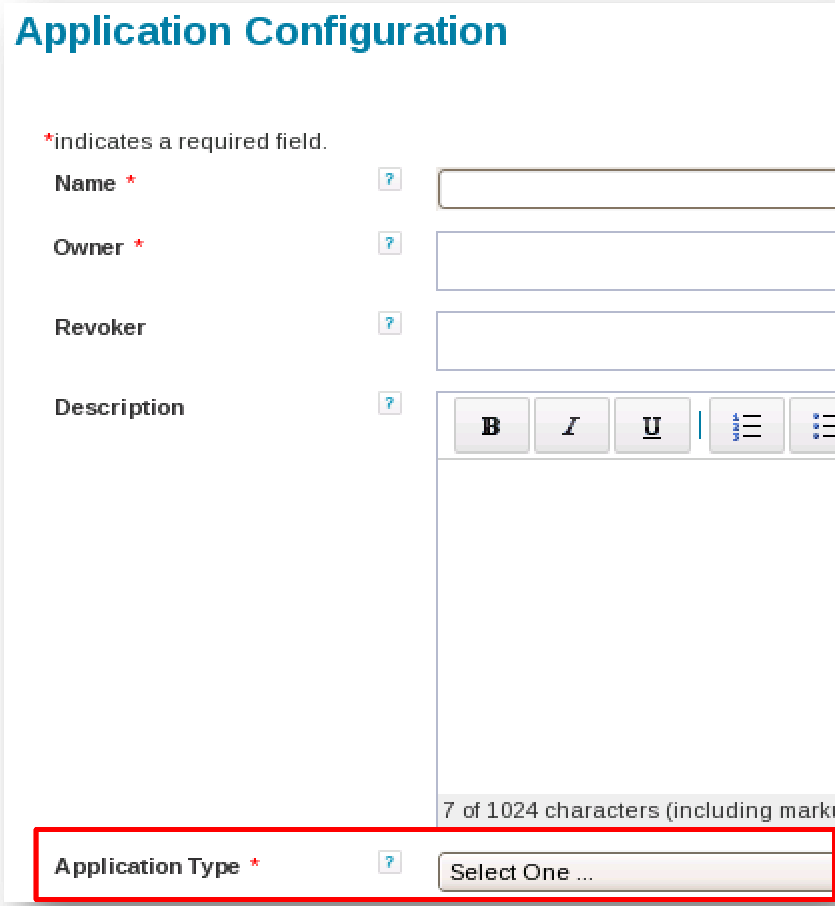
Next Steps

Define and Onboard Additional Applications

- Account Schema
- Account Group Schemas (if needed)
- Connector Type
- Rules
  - Connector Rules
    - Support Data Transformation operations
    - Connector Rules vary based on the connector type
  - Application Rules
    - Support treatment of Accounts/Account Groups (Resource Objects)
    - Consistent across all connector types

Applications – Specifying Connector

- Connectors
  - Provide for reading data from applications or ill-formed text feeds
  - May provide for writing data to applications
  - Vary significantly
- *Application Type* defines the connector used
- Connectors:
  - Read Only
    - Delimited File
  - Read/Write
    - AD, LDAP, JDBC
  - Rule Based
    - Multiplexed, Logical, Rule Based File Parser



Application Configuration

*indicates a required field.

Name *	
Owner *	
Revoker	
Description	7 of 1024 characters (including mark
Application Type *	Select One ...

For full connector listing, refer to *Compass*

Applications – Specifying Schema

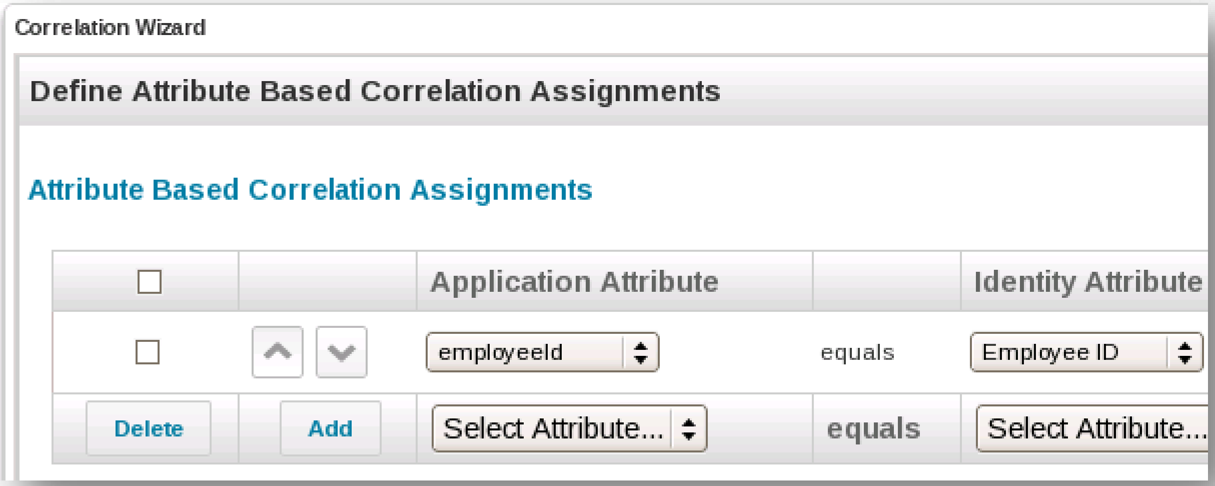
- Specifying Schemas
  - Account – Used to represent individual accounts
  - Group – Used to represent individual group account
- Specifying Connector (Application Type)
  - Dependent upon connector, schema attributes may be pre-defined
- Specifying Activity Data Sources
  - Drives activity tracking and monitoring (logins/logouts, etc.)

Applications – Group Schema

- Group Schema is an extensible feature of Connectors which makes group objects on a given system first-class objects
  - Allows IdentityIQ to support native account group object models
  - Provides framework for defining what group membership really means
    - I am a member of Group 920-100, I can access the financial planning file share
    - I am a member of AD group VPN, I can login to corporate VPN
- Similar to Permissions in that they are implementation dependent
  - Permissions are **direct**
  - Group-based permissions are **indirect**

Account Correlation

- Account correlation specifies how to match an account to an authoritative identity cube
- Determines whether native account results in creation of new identity cube or linkage to an existing one
- 3 correlation methods
  - Correlation Wizard
    - Set of ordered correlations
    - Reusable correlation configuration
    - Attribute based
    - Condition based
  - Correlation Rule
  - Manually


Correlation Wizard

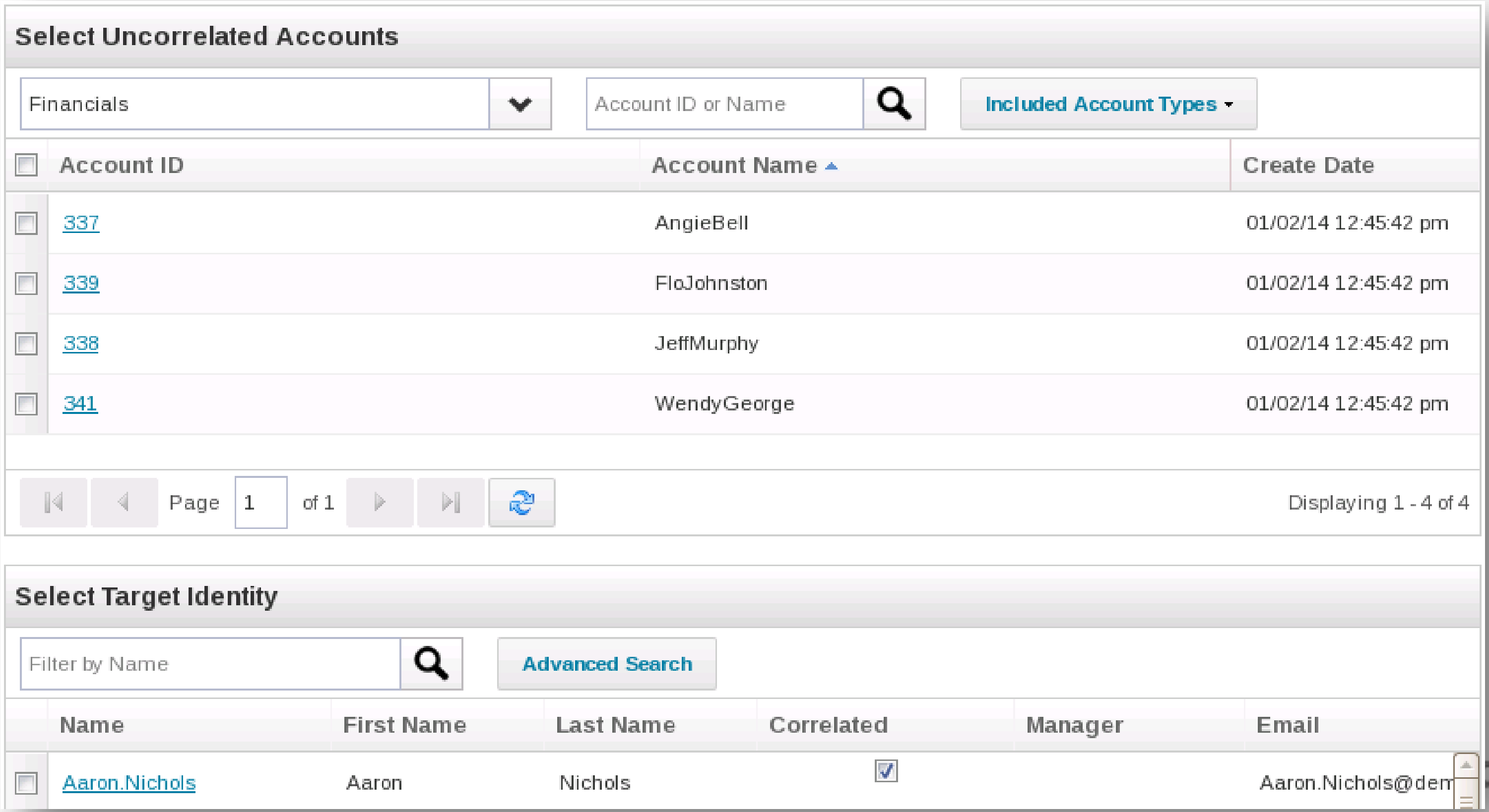
Define Attribute Based Correlation Assignments

Attribute Based Correlation Assignments

☐		Application Attribute		Identity Attribute
☐		employeeId	equals	Employee ID
Delete	Add	Select Attribute...	equals	Select Attribute...

Manual Correlation

- When Automatic Correlation falls short...
  - Manual Correlation provides correlation clean-up
    - Manage → Identity Correlation
  - Accounts that are uncorrelated can be assigned to identities
  - Correlation permanently retained



Select Uncorrelated Accounts

Financials | Account ID or Name | Included Account Types

☐	Account ID	Account Name ▲	Create Date
☐	337	AngieBell	01/02/14 12:45:42 pm
☐	339	FloJohnston	01/02/14 12:45:42 pm
☐	338	JeffMurphy	01/02/14 12:45:42 pm
☐	341	WendyGeorge	01/02/14 12:45:42 pm

Page 1 of 1 — Displaying 1 - 4 of 4

Select Target Identity

Filter by Name | Advanced Search

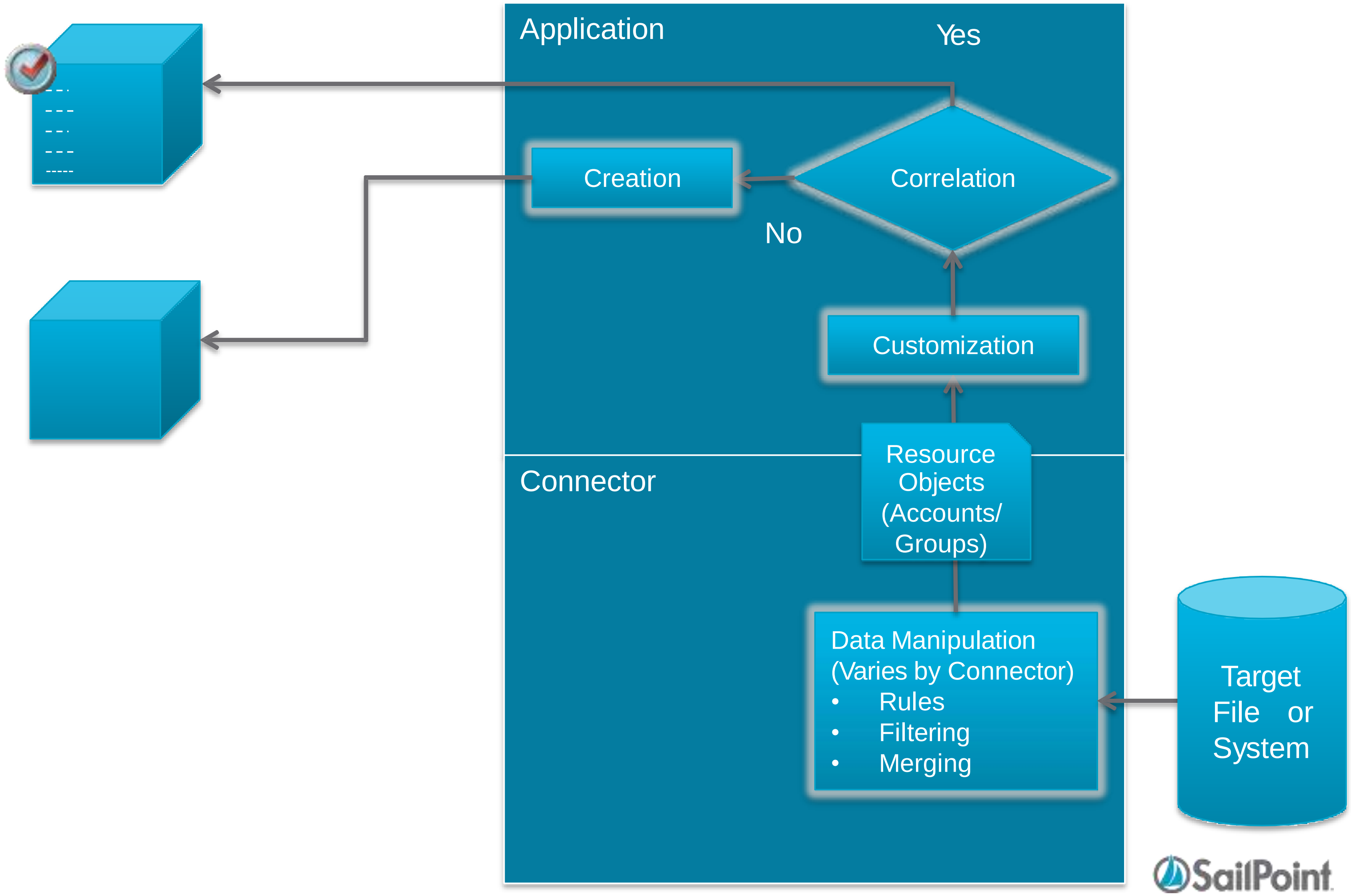
	Name	First Name	Last Name	Correlated	Manager	Email
☐	Aaron.Nichols	Aaron	Nichols	✓		Aaron.Nichols@dem

Applications – Rules

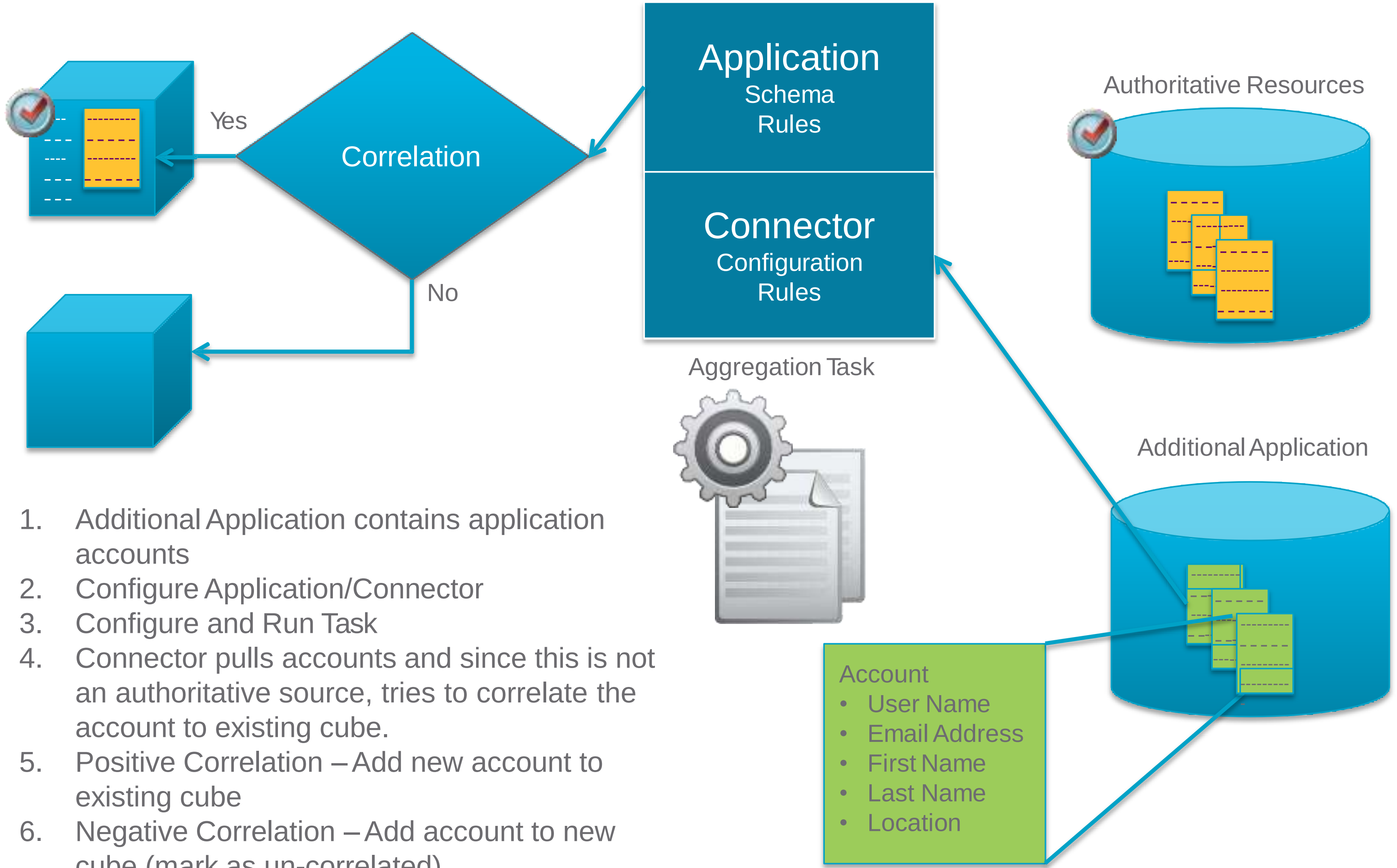
- Creation Rule
  - Hook for performing customizations at cube creation time
    - E.g. assigning IdentityIQ capabilities or setting default passwords
- Correlation Rule
  - Used to build and maintain account correlations
  - Alternatively configured through GUI
- Manager Correlation Rule
  - Used by IdentityIQ to build and maintain manager relationship
  - Alternatively configured through GUI
- Customization Rule
  - Used to modify incoming Accounts/Groups prior to saving to an Identity

Application/Connector Processing

Aggregation

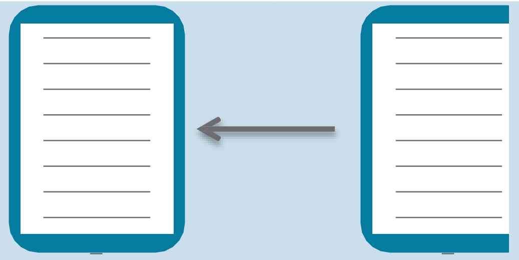
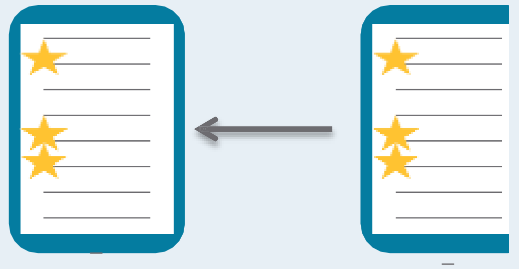
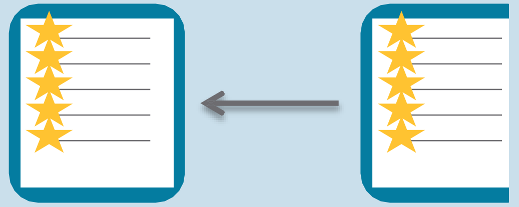
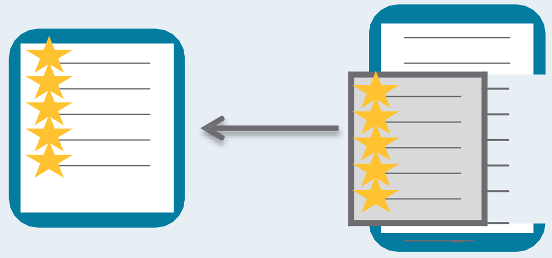


Overview – Account Correlation



1. Additional Application contains application accounts
2. Configure Application/Connector
3. Configure and Run Task
4. Connector pulls accounts and since this is not an authoritative source, tries to correlate the account to existing cube.
5. Positive Correlation – Add new account to existing cube
6. Negative Correlation – Add account to new cube (mark as un-correlated)

Aggregation Strategies

	IdentityIQ	Application
Process All • Every account read and processed • Task option ***Disable optimization of unchanged accounts*** = ***true***		
IdentityIQ-based Optimization (default) • Every account read • Only those with changes are processed • Task option ***Disable optimization of unchanged accounts*** = ***false***		
Custom Delta Processing • Manage own change (i.e. write changed accounts to a flat file and process flat file) • Task option ***Detect deleted accounts*** = ***false***		
Connector-based Delta Aggregation* • Read and process only accounts with changes that have taken place after benchmark • lastModData, usnChanged, etc. • Task option ***Enable Delta Aggregation*** = ***true***		

*Not supported by all connectors

Connectors

Overview

Connectors

- Planning
- Merging
- Highlighted Connectors
  - Delimited File
  - JDBC
  - LDAP
  - AD
  - Logical
  - Multiplex

Connector Planning

Parameters

- Define parameters for each connector/application instance.
    - Connection parameters
        - Login
        - Password
        - Etcetera
    - Schema
    - Groups
    - activity sources
    - Formatting
    - IdentityIQ rules
    - Application owners
- Leverage the SailPoint Functional Requirements Template (available on Compass)

Connector Planning

Merging

- Merging – Supported by Delimited File and JDBC Connectors
  - **Data needs to be merged** – indicates if connector needs to be aware of multiple rows.
  - **Index Column** – the column name which indicates how similar rows are correlated.
  - **Which columns should be merged?** – the columns that are used in the default merge.

Merging Example

- Delimited File or JDBC with the following result:

  username, firstname, lastname, scope

  bsmith, Bob, Smith, US

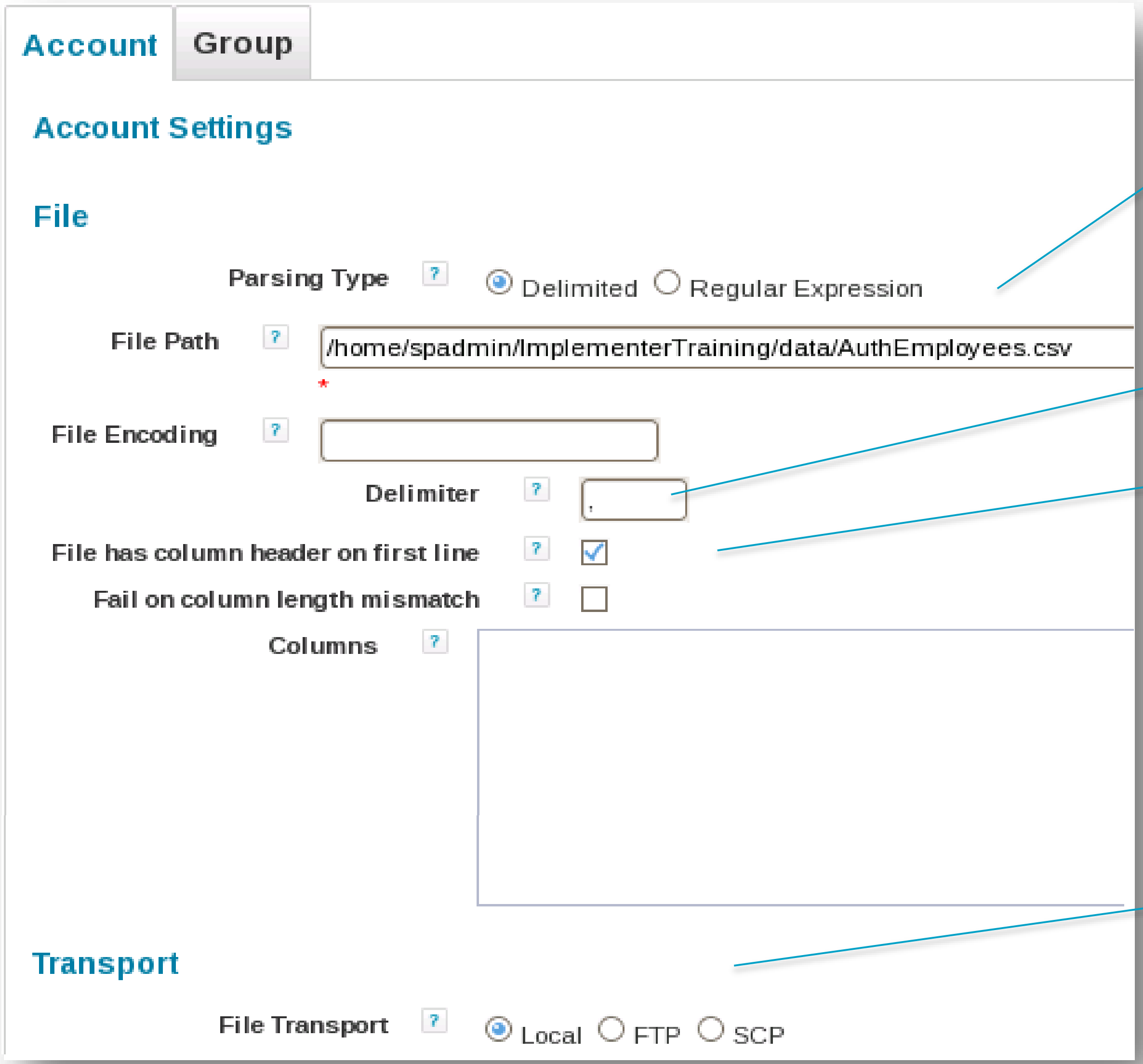
  bsmith, Bob, Smith, EMEA

- Set the merging to the following:
  - **Data needs to be merged** : true
  - **Index Column** : username
  - **Which columns should be merged?** : scope

- Results
  - One Account for Bob Smith (username=bsmith)
  - Scope attribute set to “US,EMEA”

Delimited File

Delimited File – File and Transport



Account | Group

Account Settings

File

Parsing Type: [✓ Delimited] Regular Expression

File Path *: /home/spadmin/ImplementerTraining/data/AuthEmployees.csv

File Encoding:

Delimiter: ,

File has column header on first line: [✓]

Fail on column length mismatch: [ ]

Columns:

Transport

File Transport: [✓ Local] FTP SCP

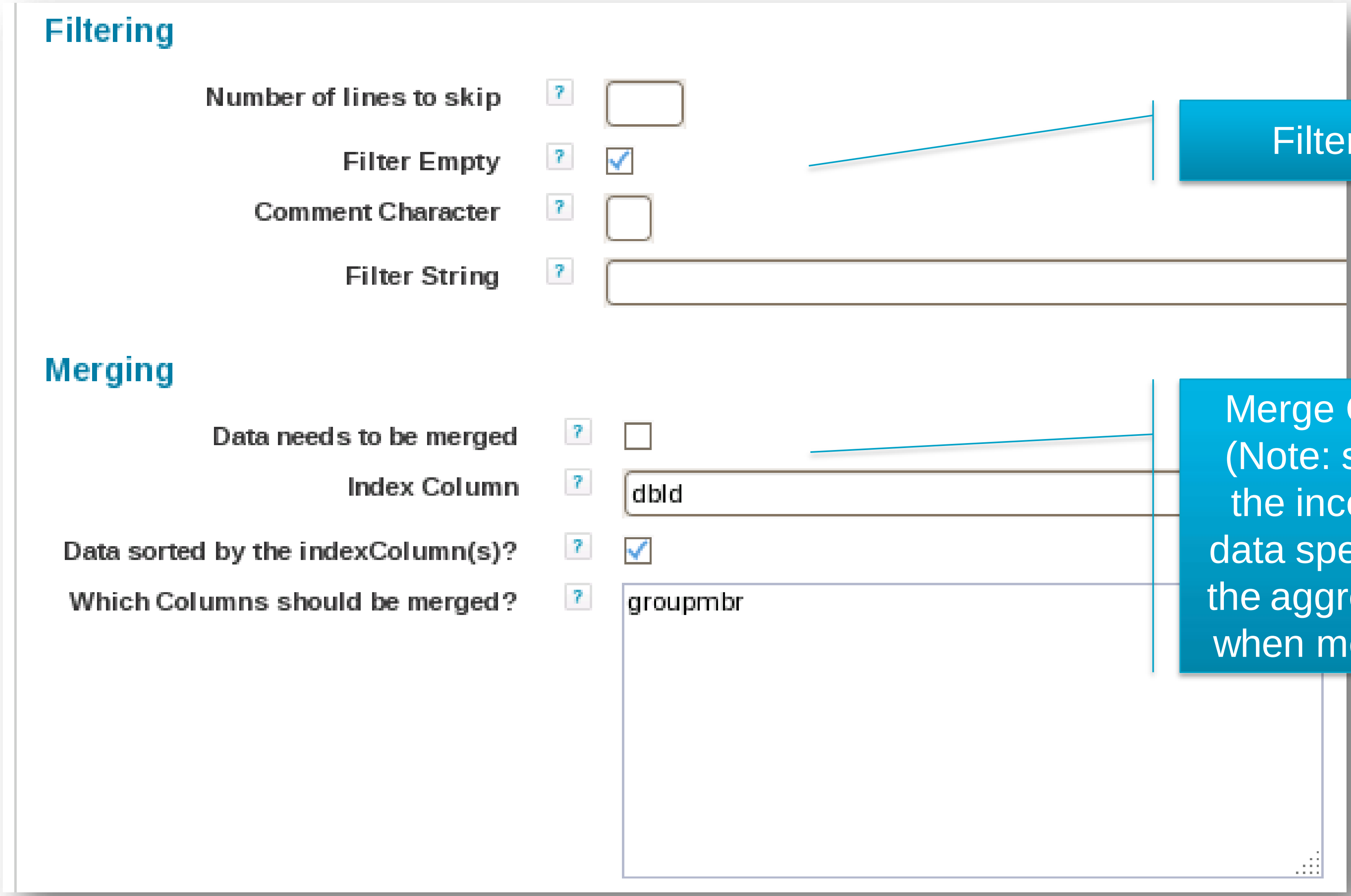
File Path

Delimiter

Column Header Present?

File Transport

Delimited File – Filtering and Merging



Filtering

Number of lines to skip	?	
Filter Empty	?	✓
Comment Character	?	
Filter String	?	

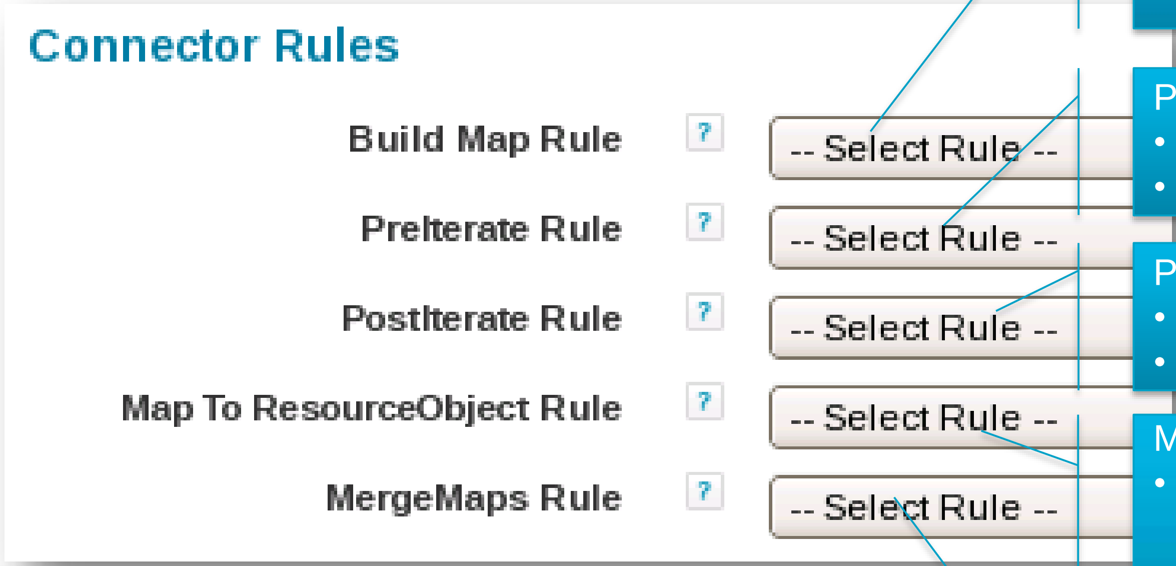
Merging

Data needs to be merged	?	☐
Index Column	?	dbId
Data sorted by the indexColumn(s)?	?	✓
Which Columns should be merged?	?	groupmbr

Filtering

Merge Config (Note: sorting the incoming data speeds up the aggregation when merging)

Delimited File – Connector Rules



Connector Rules

Build Map Rule	-- Select Rule --
PreIterate Rule	-- Select Rule --
PostIterate Rule	-- Select Rule --
Map To ResourceObject Rule	-- Select Rule --
MergeMaps Rule	-- Select Rule --

BuildMap Rule
- Runs for every line in the file
- Converts incoming data into map

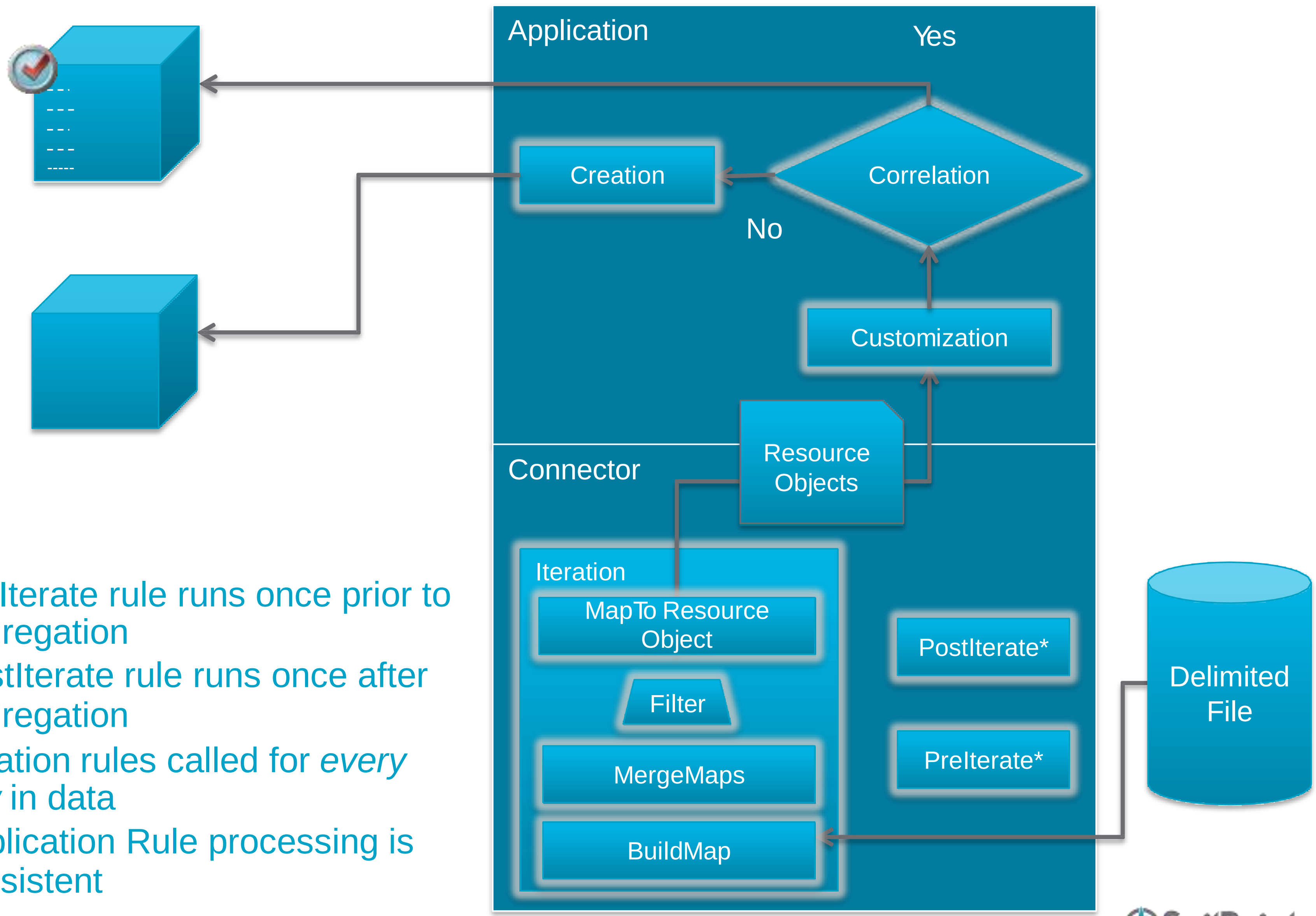
PreIterate Rule
- Runs once for each aggregation
- Can do any pre-processing

PostIterate Rule
- Runs once for each aggregation
- Can do any post-processing

Map To ResourceObject Rule
- Performs final conversion to Resource Object
- Runs once for each account or group
- Runs after merging

MergeMaps Rule
- Performs merging processing
- If default merge capabilities aren't enough, a rule here can control merging

Delimited File Processing



Notes:

- PreIterate rule runs once prior to aggregation
- PostIterate rule runs once after aggregation
- Iteration rules called for *every row* in data
- Application Rule processing is consistent

Writing to CSV Files

SQL Loader Connector

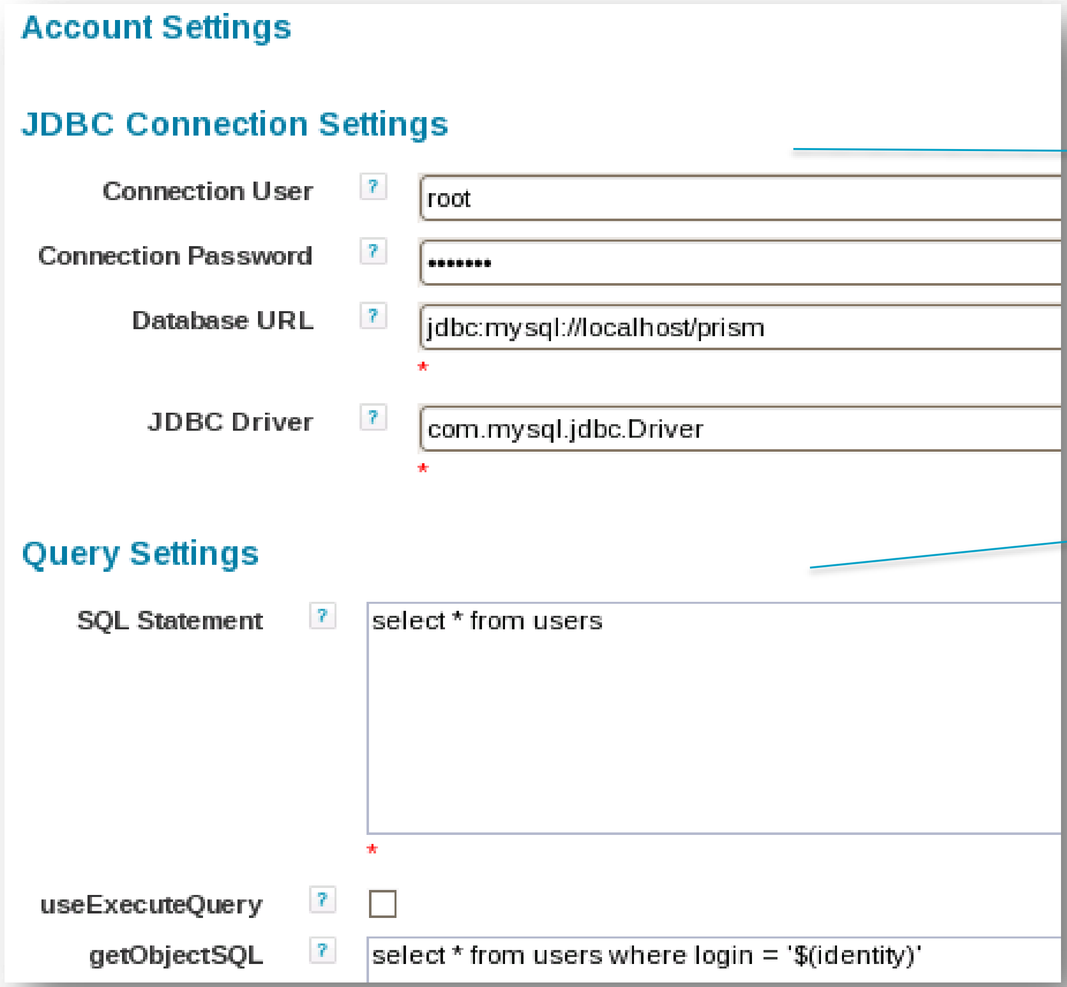
Overview

- Provides SQL query option to read/write data from CSV/Text files
- Based on JDBC Connector architecture
- Data can be pulled from multiple files
- Support direct Permission functionality

JDBC

JDBC Applications – Connection/Query

- Similar parsing to the Delimited File Connector
- Processing database rows (instead of lines in a file)



Account Settings

JDBC Connection Settings

Connection User	root
Connection Password	•••••••
Database URL	jdbc:mysql://localhost/prism *
JDBC Driver	com.mysql.jdbc.Driver *

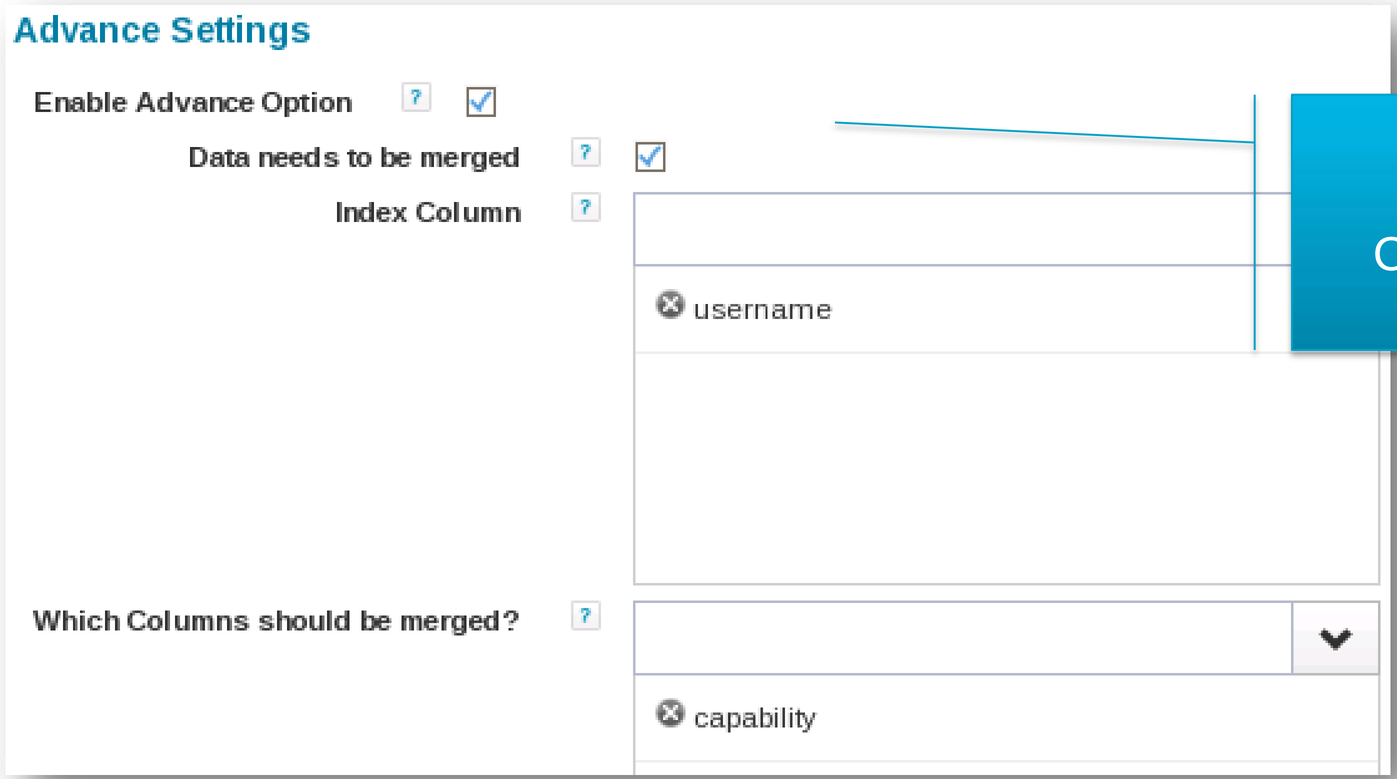
Query Settings

SQL Statement	select * from users *
useExecuteQuery	☐
getObjectSQL	select * from users where login = '$(identity)'

JDBC Connection Settings
- user/password
- DB URL
- JDBC Driver

Query Settings
- SQL Statement for pulling all accounts
- SQL Statement for pulling single account

JDBC Applications – Merging



Advance Settings

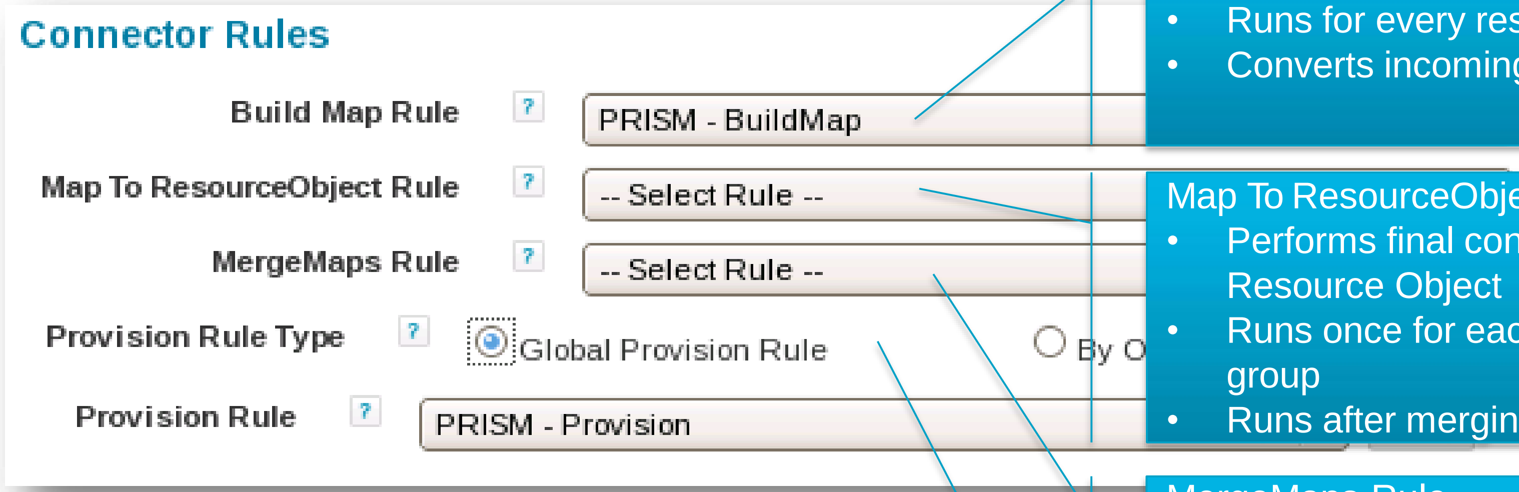
Enable Advance Option	✓
Data needs to be merged	✓
Index Column	username
Which Columns should be merged?	capability

Merge Configuration

Notes:

- No filtering support (filtering supported by query)
- Sorting incoming data speeds up aggregation when merging

JDBC Applications – Rules



Connector Rules

Build Map Rule	PRISM - BuildMap
Map To ResourceObject Rule	-- Select Rule --
MergeMaps Rule	-- Select Rule --
Provision Rule Type	[✓ Global Provision Rule] By O
Provision Rule	PRISM - Provision

BuildMap Rule
- Runs for every result row
- Converts incoming data into map

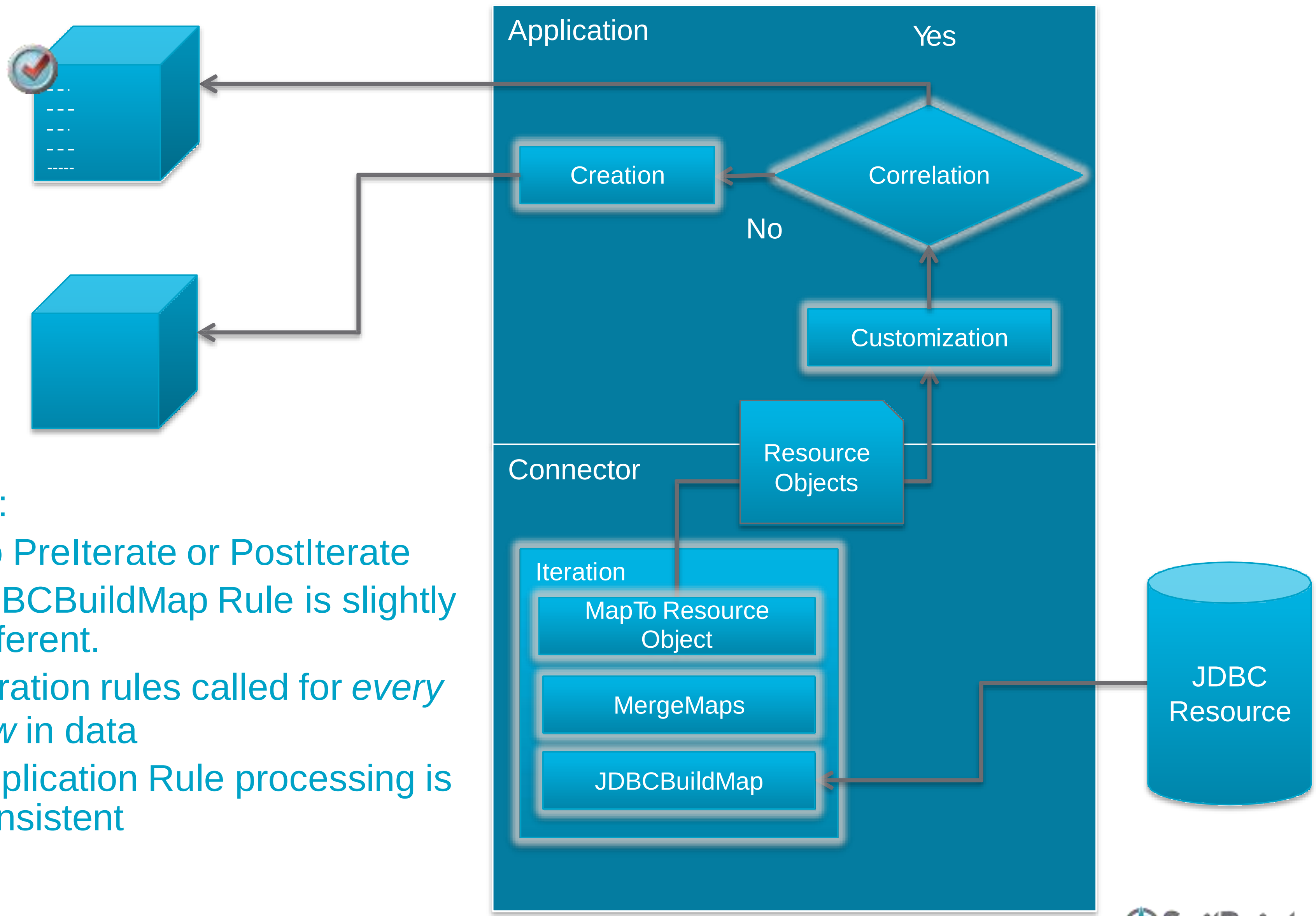
Map To ResourceObject Rule
- Performs final conversion to Resource Object
- Runs once for each account or group
- Runs after merging

MergeMaps Rule
- Performs merging processing
- If default merge capabilities aren't enough, a rule here can control merging

JDBC Provision Rule
- Handles Provisioning Operations
- All in single rule or per operation
- More on this later

Note: No PreIterate or PostIterate

JDBC Processing

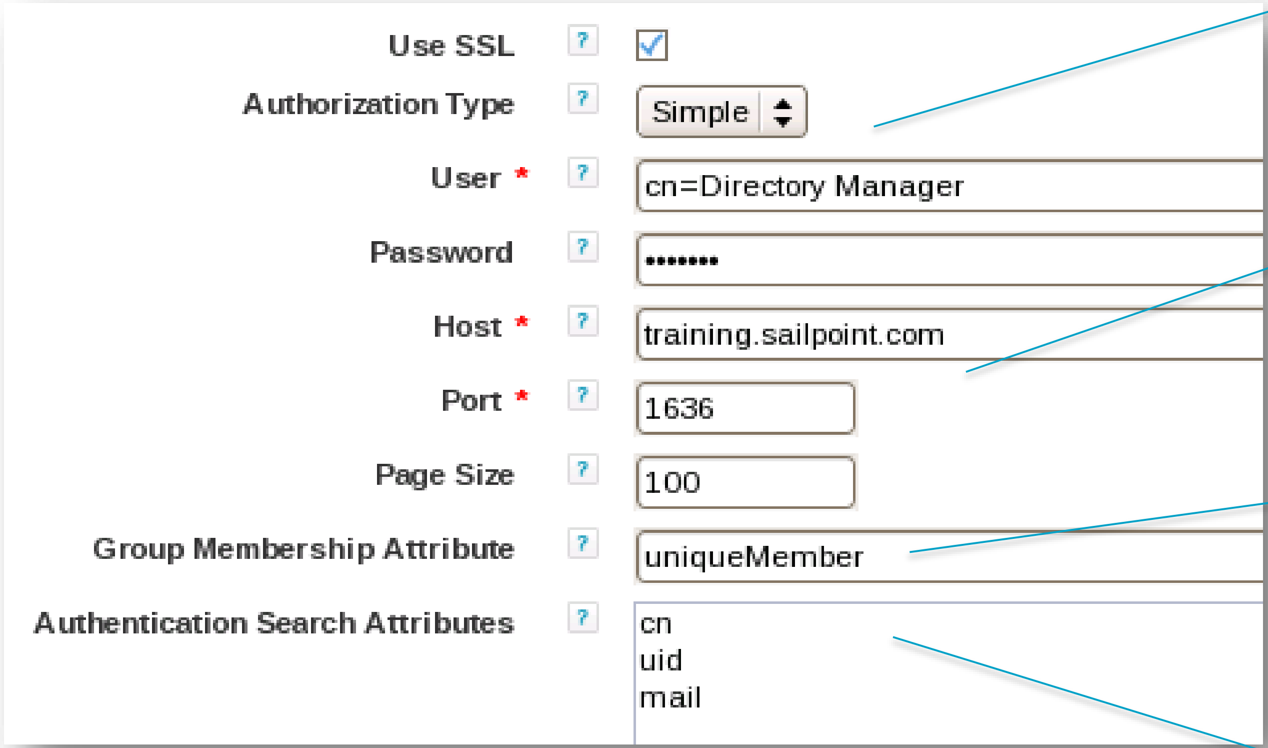


Notes:

- No PreIterate or PostIterate
- JDBCBuildMap Rule is slightly different.
- Iteration rules called for *every row* in data
- Application Rule processing is consistent

LDAP

LDAP Connector



Use SSL	✓
Authorization Type	Simple
User *	cn=Directory Manager
Password	•••••••
Host *	training.sailpoint.com
Port *	1636
Page Size	100
Group Membership Attribute	uniqueMember
Authentication Search Attributes	cn uid mail

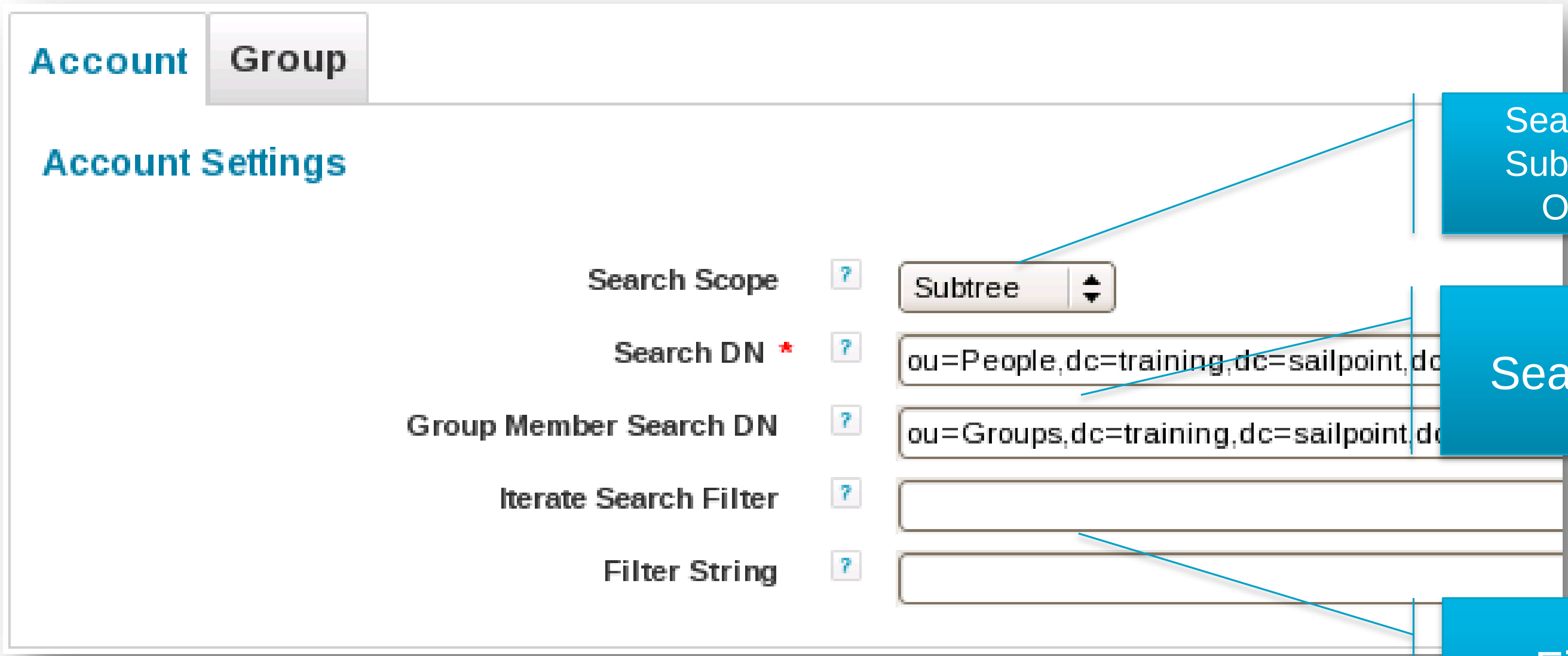
SSL/Auth Type Credentials

Host/Port

Which LDAP Attribute holds group

For Pass Through Authentication

LDAP Connector – DN and Filtering



Account | Group

Account Settings

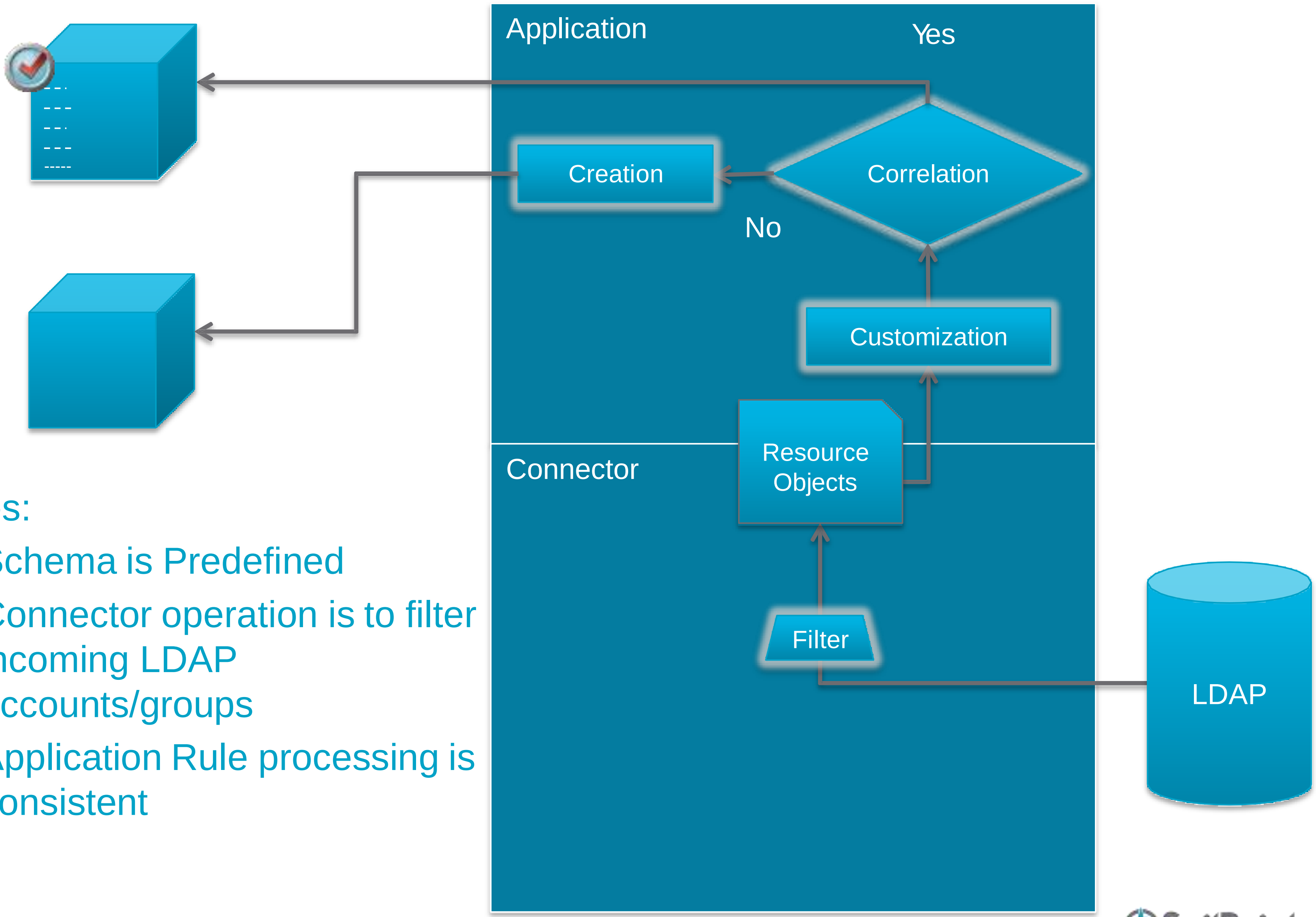
Search Scope	Subtree
Search DN *	ou=People,dc=training,dc=sailpoint,dc
Group Member Search DN	ou=Groups,dc=training,dc=sailpoint,dc
Iterate Search Filter	
Filter String	

Search Scope Subtree,Base, OneLevel

Search DNs

Filtering Information

LDAP Processing

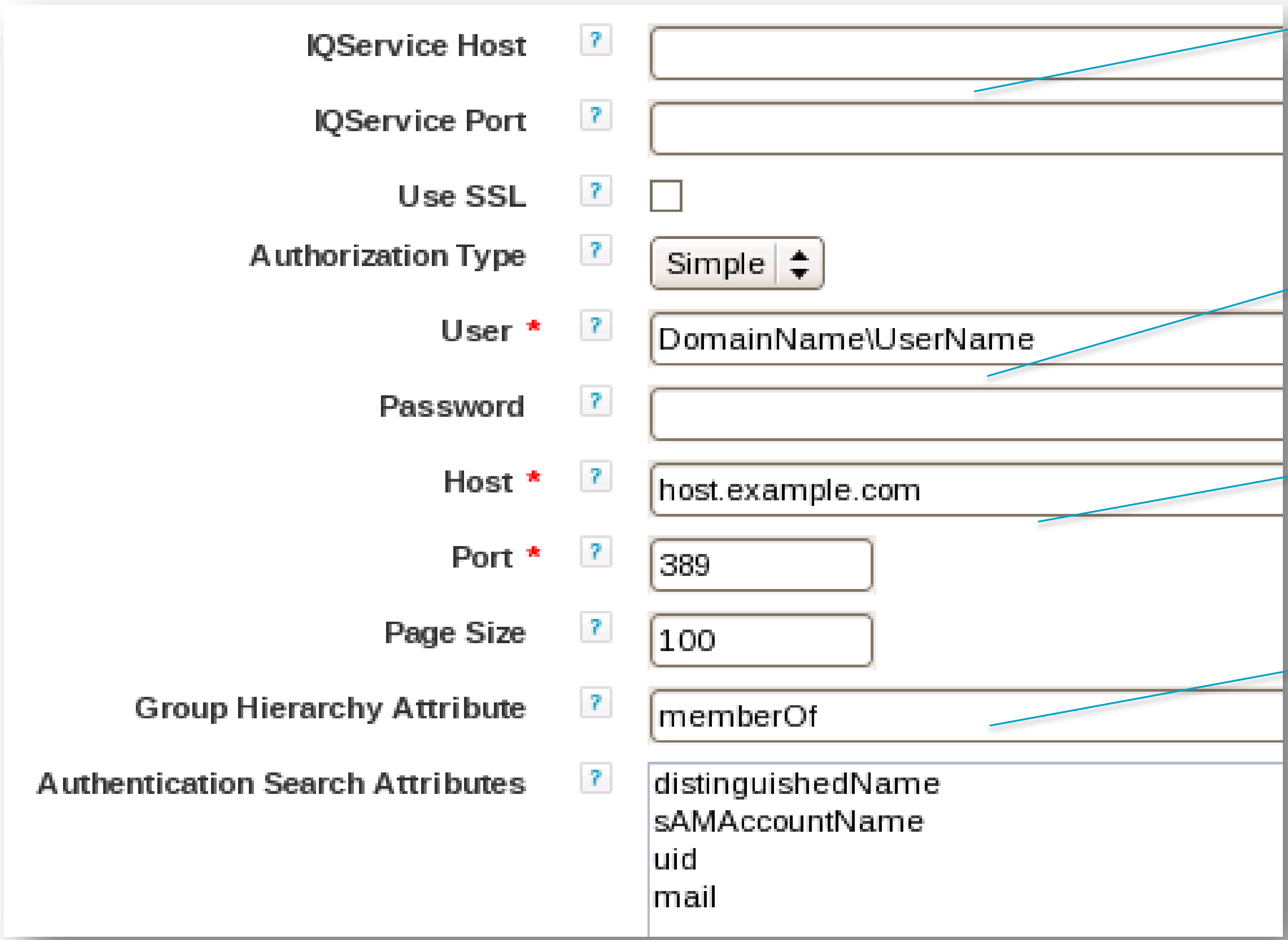


Notes:

- Schema is Predefined
- Connector operation is to filter incoming LDAP accounts/groups
- Application Rule processing is consistent

AD

AD Connector – Connection Information



IQService Host	
IQService Port	
Use SSL	☐
Authorization Type	Simple
User *	DomainName\UserName
Password	
Host *	host.example.com
Port *	389
Page Size	100
Group Hierarchy Attribute	memberOf
Authentication Search Attributes	distinguishedName sAMAccountName uid mail

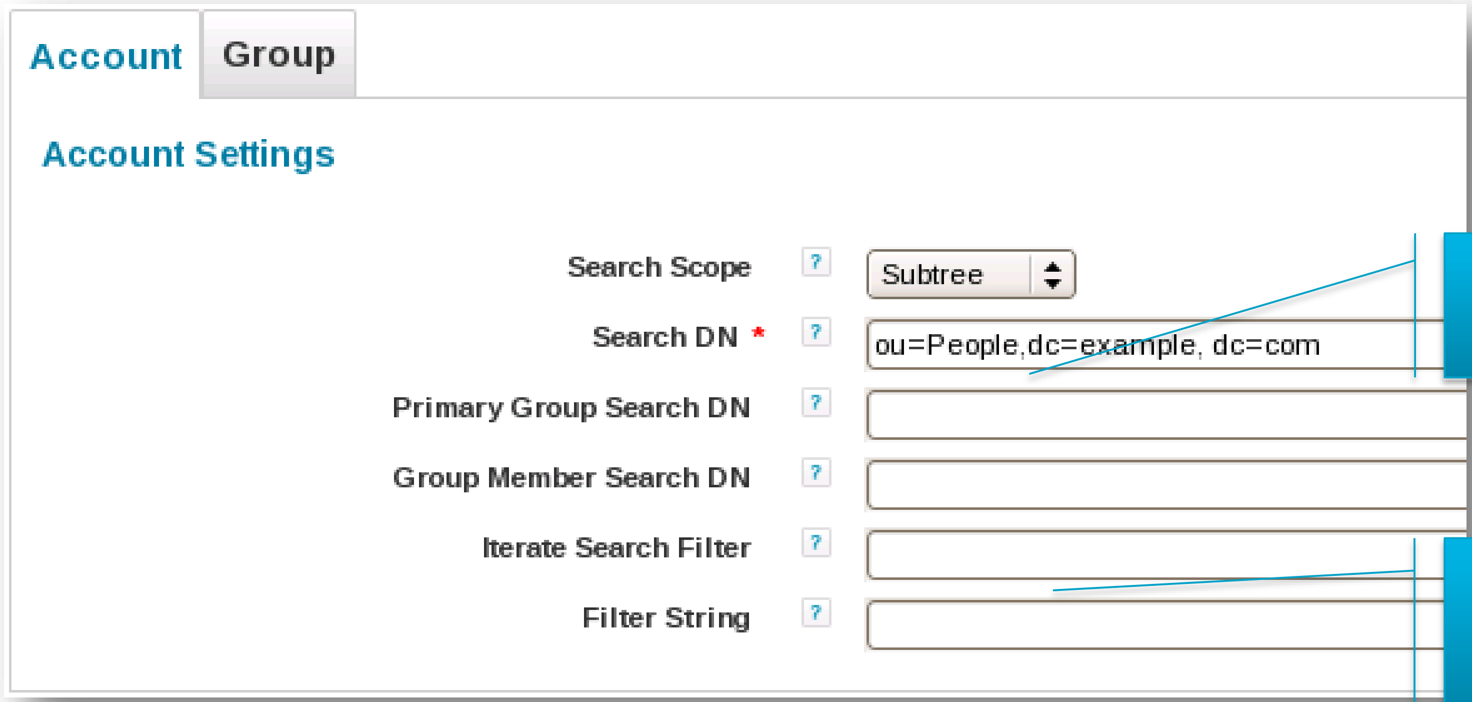
Connection Info for IQService for Provisioning

Auth Information

Host/Port

Group Attribute

AD Connector – DN and Filtering


Account | Group

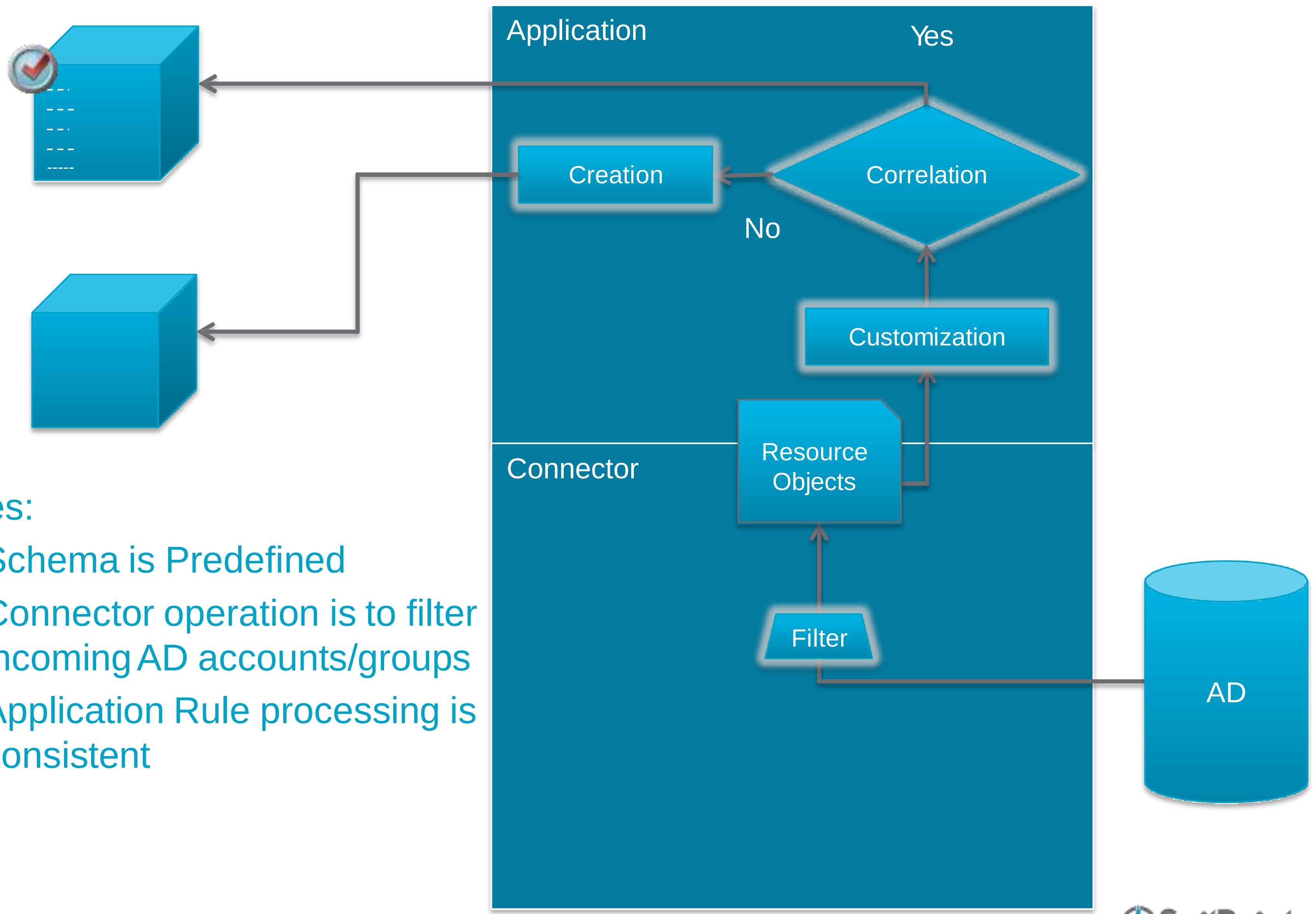
Account Settings

Search Scope	Subtree
Search DN *	ou=People,dc=example, dc=com
Primary Group Search DN	
Group Member Search DN	
Iterate Search Filter	
Filter String	

DN Information for Searching

Filtering Information

AD Processing



Notes:

- Schema is Predefined
- Connector operation is to filter incoming AD accounts/groups
- Application Rule processing is consistent

Other Connectors

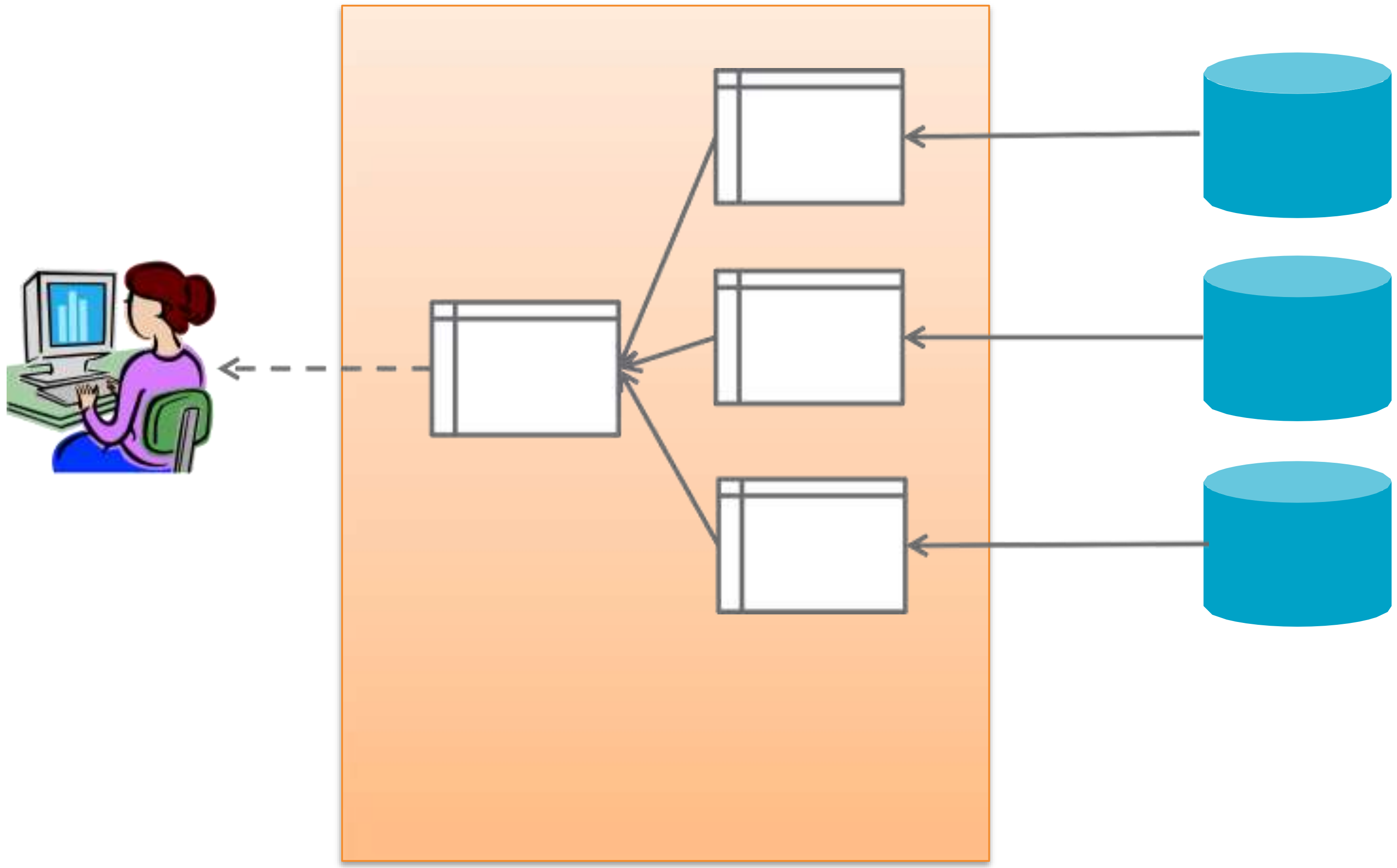
- Each connector will vary on:
  - Connector Settings
  - Connector Rules
- Each Connector is consistent with regard to:
  - Application Rules
    - How correlation, creation, customization is handled?
  - Schema (Account and Group)

Logical

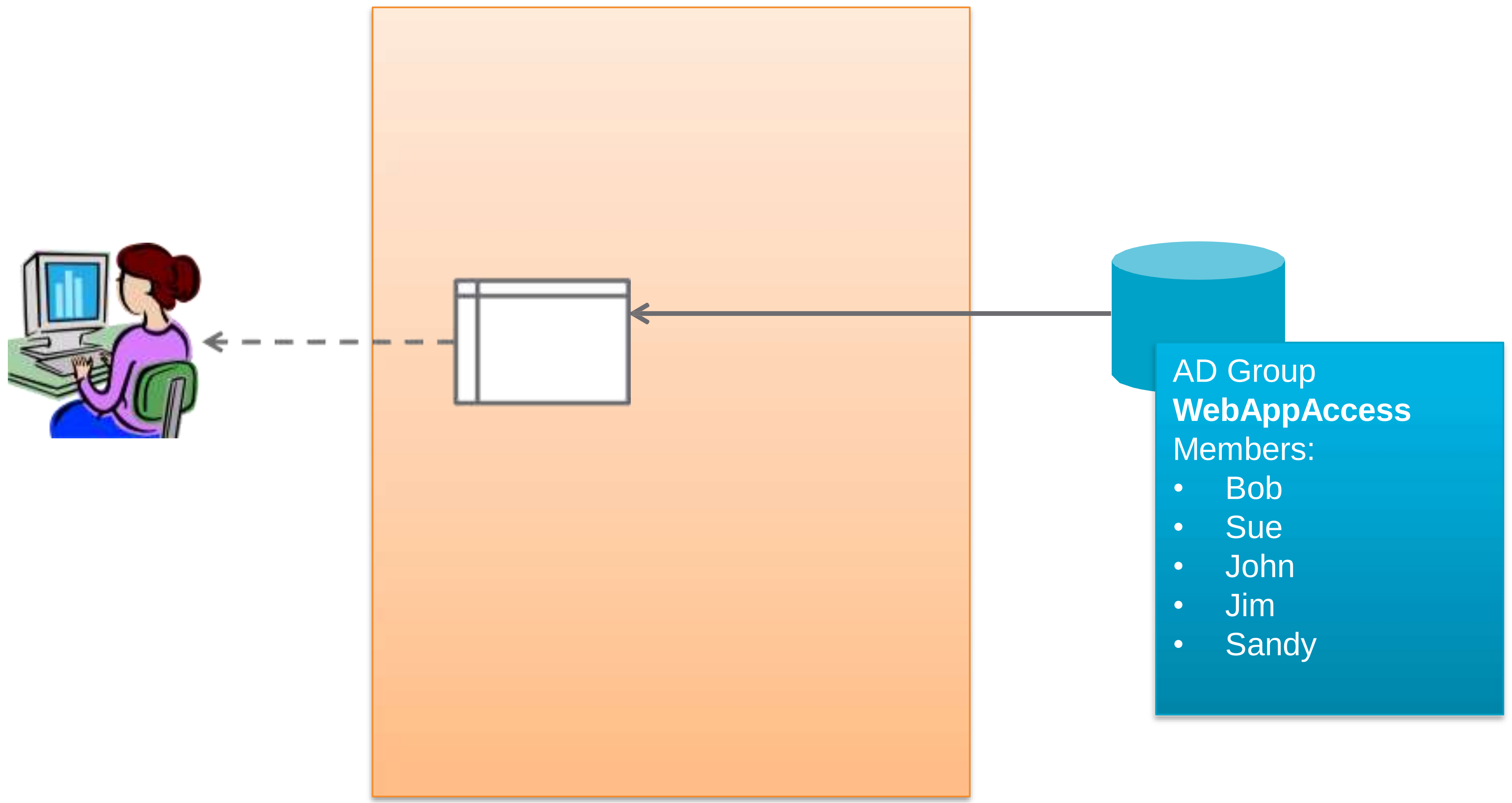
What is a Logical Application?

- A way to define an application "logically"
  - Logical apps allow for two things:
  - Combining (previous composite behavior)
    - Treating Multiple Applications as a single Logical Application
  - Subdividing
    - Treating users with a certain LDAP or AD group as account holders on an application.
  - Examples:
    - Application is defined by Web Application access, Mainframe application access and SQL database access (Composite)
    - AD group controls access to a web application at company XYZ. They want to treat this web application as a logical standalone application for requesting access and certifications (Subdividing)
- Simplifies searches, certifications, etc. by treating these special types of applications as a logical entity.

Logical Application – Combining



Logical Application – Subdividing



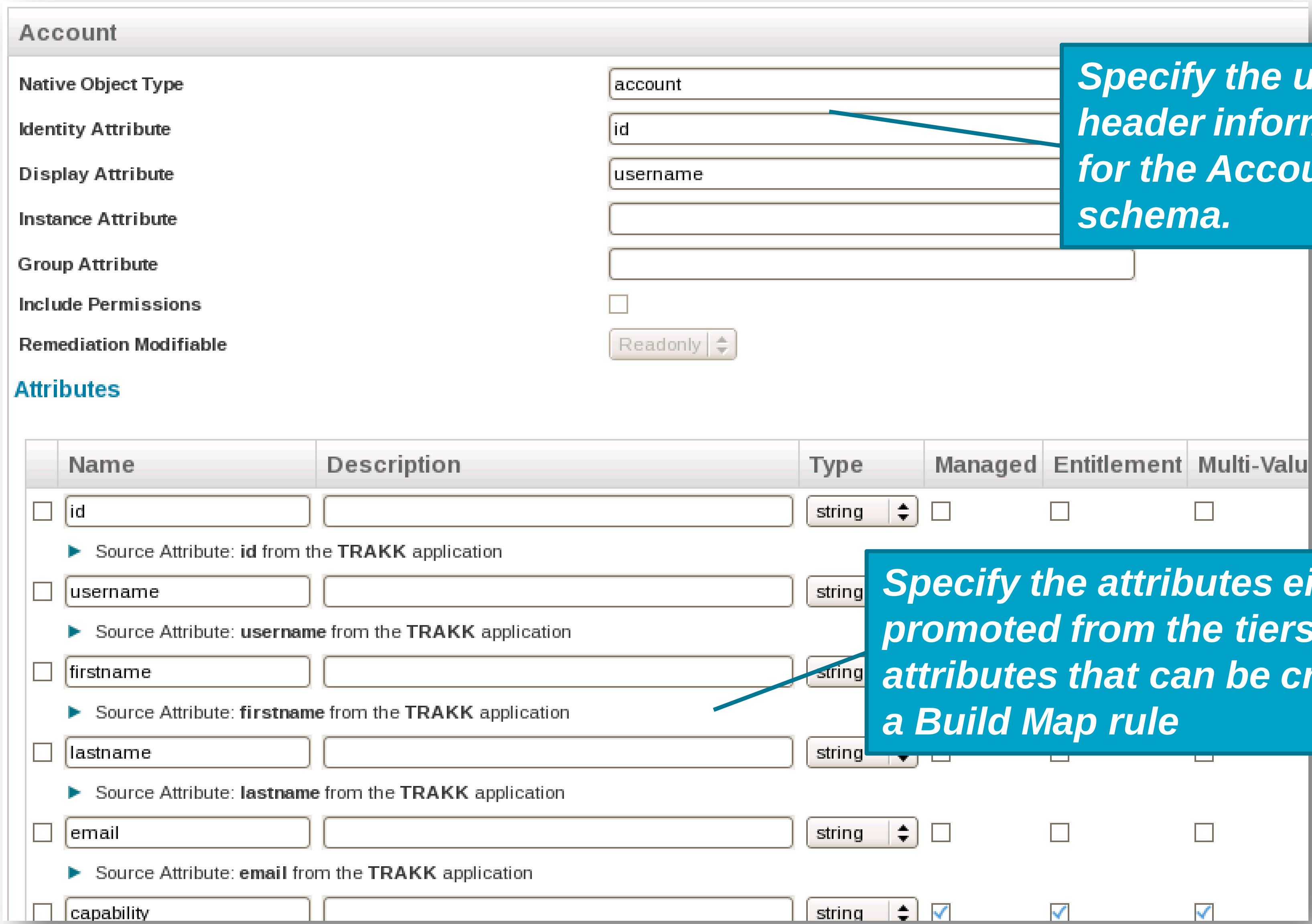
Required Pieces of a Logical Application

- The individual application(s) which need to be aggregated
- The logical application:
  - Tiers – the list of applications that make up this composite application with information on:
    - Tier attributes that define application membership
    - If you have more than one application
      - Which application is “primary” (e.g., defines the identities to be loaded)
      - Which attributes to use to correlate the non-primary applications to the primary application
  - Schema
    - Tier Attributes – From Primary tier or promoted from other tiers
    - New Attributes – Defined and then created using Build Map rule

Optional Pieces of a Logical Application

- Account Rule
  - Determines whether to create an account for an Identity
- Provisioning Rule
  - Called when provisioning needs to be performed for a Logical application. Can properly handle provisioning across the one or more tier applications
- Remove Tier entitlements on Account Removal (checkbox)
  - Remove those entitlements in the tier applications when an account on the logical application is removed.

Specifying the Schema



Account

Native Object Type	account
Identity Attribute	id
Display Attribute	username
Instance Attribute	
Group Attribute	
Include Permissions	☐
Remediation Modifiable	Readonly

Attributes

	Name	Description	Type	Managed	Entitlement	Multi-Valu
☐	id		string	☐	☐	☐
	▶ Source Attribute: **id** from the **TRAKK** application					
☐	username		string			
	▶ Source Attribute: **username** from the **TRAKK** application					
☐	firstname		string			
	▶ Source Attribute: **firstname** from the **TRAKK** application					
☐	lastname		string	☐	☐	☐
	▶ Source Attribute: **lastname** from the **TRAKK** application					
☐	email		string	☐	☐	☐
	▶ Source Attribute: **email** from the **TRAKK** application					
☐	capability		string	✓	✓	✓

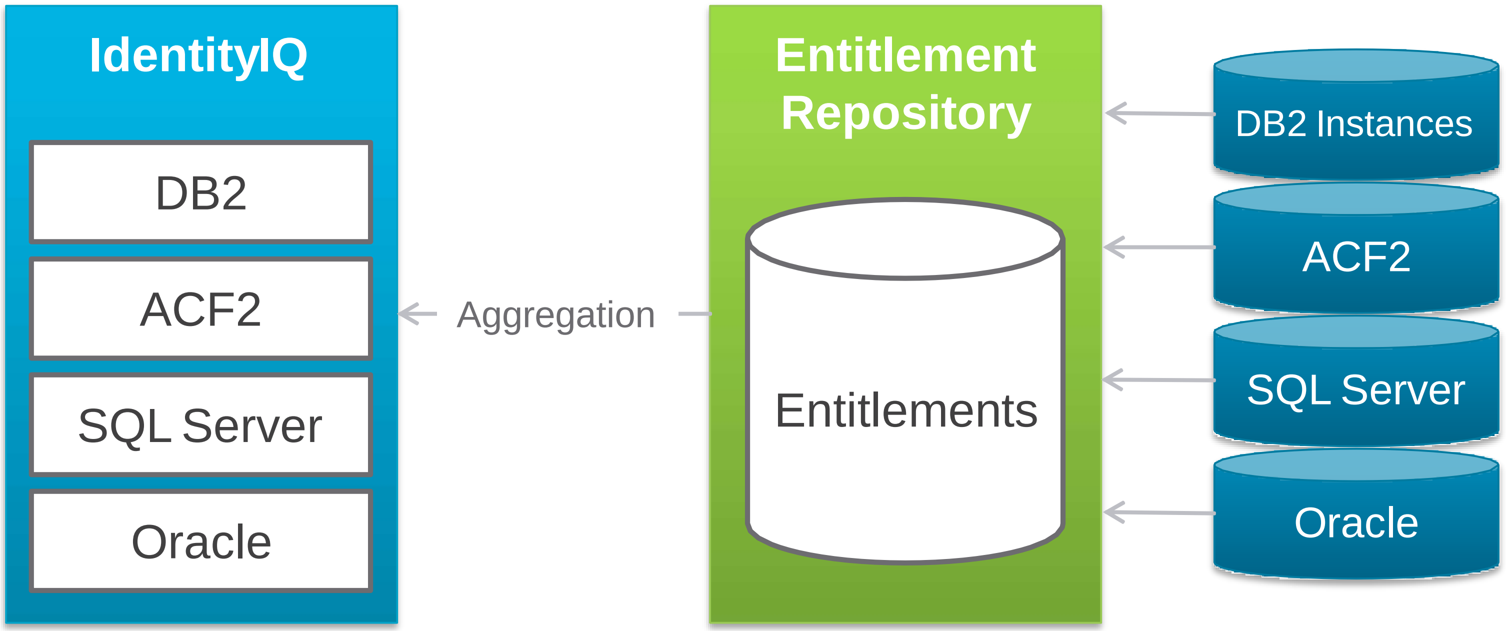
Specify the usual header information for the Account schema.

Specify the attributes either as promoted from the tiers, or new attributes that can be created in a Build Map rule

Multiplex

What is a Multiplex Application?

- Automatically create multiple applications based on a single data feed
- Primarily used with pre-existing entitlement repositories which contain multiple applications

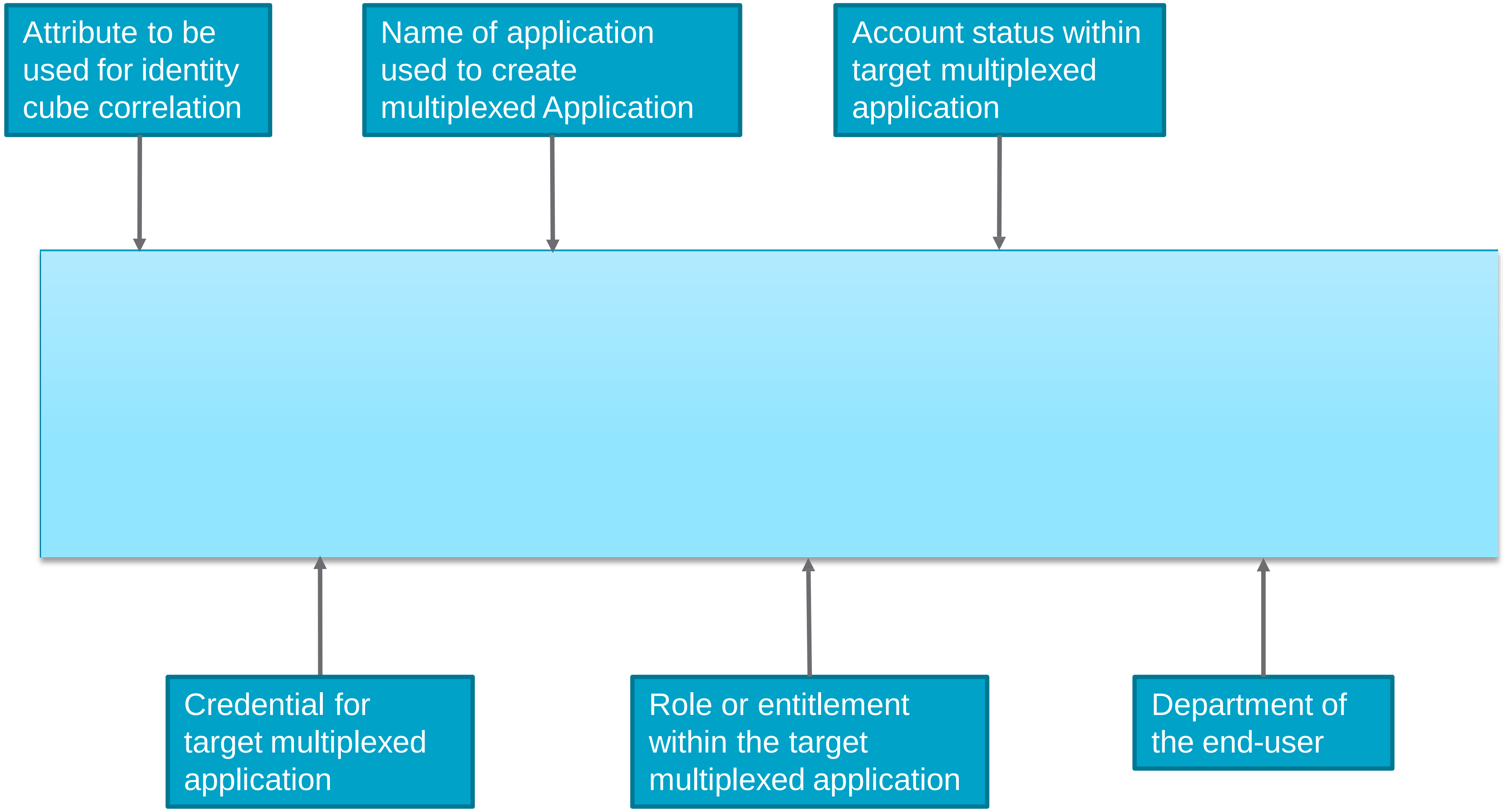


MultiPlex Requirements

- A single data feed containing user entitlements for multiple applications
- An IdentityIQ application to define the repository (the base multiplex*ing* application)
- Connector with a buildMap or customization rule
  - Rule adds two reserved attributes
    - IIQSourceApplication (required) – specifies an application
      - Application specified is created (if not already existing)
      - Account is created for application specified rather than base multiplexing application
      - If not set, Aggregator creates accounts for the base application
    - IIQMultiplexIdentity (optional) – specifies identity
      - Used when repository is sorted
      - Allows aggregator to skip correlation on all subsequent matching records
- All “IIQSourceApplication” applications conform to the same schema (default)

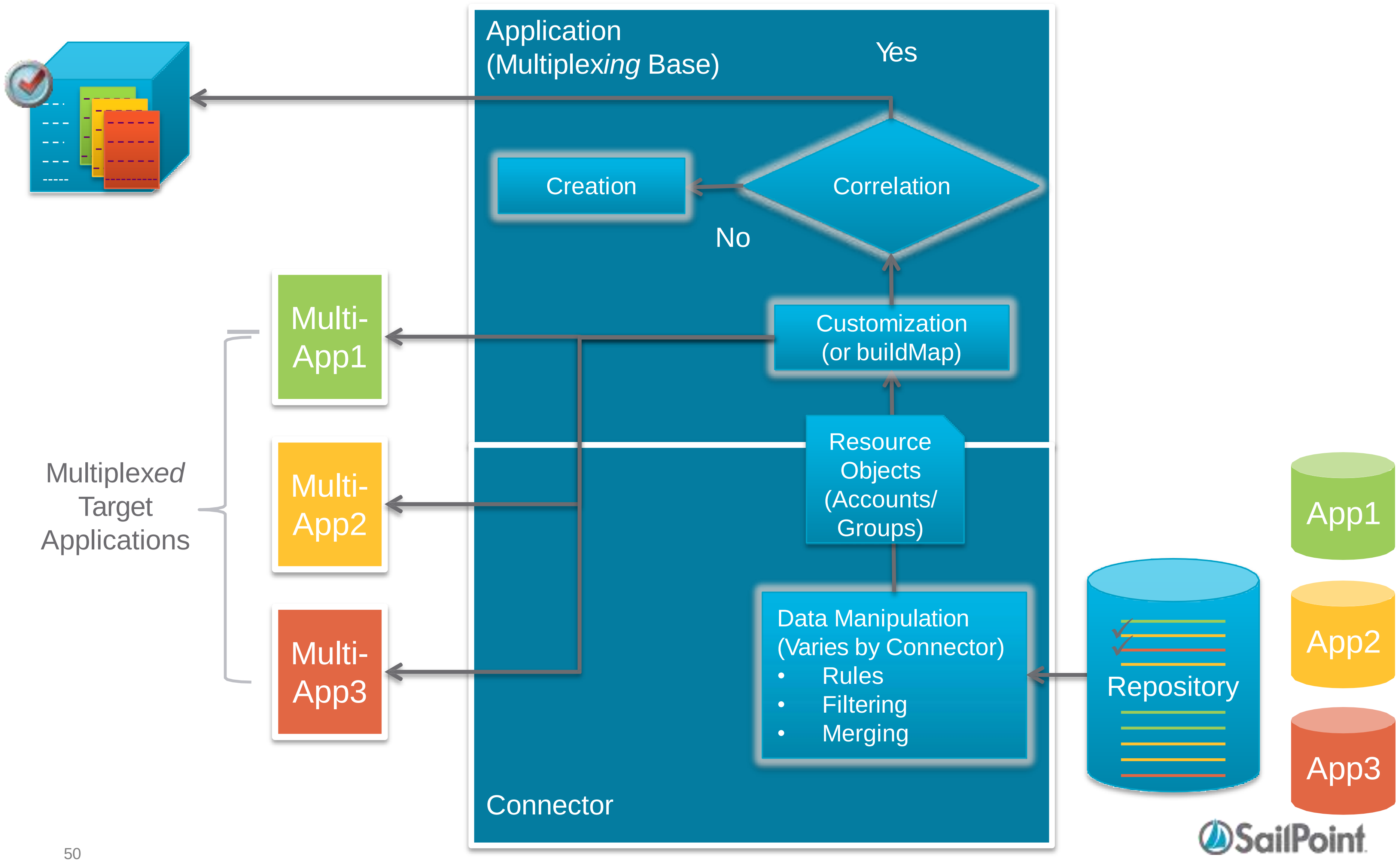
MultiPlex Data Structure

Data Source


Attribute to be used for identity cube correlation
Name of application used to create multiplexed Application
Account status within target multiplexed application
Credential for target multiplexed application
Role or entitlement within the target multiplexed application
Department of the end-user

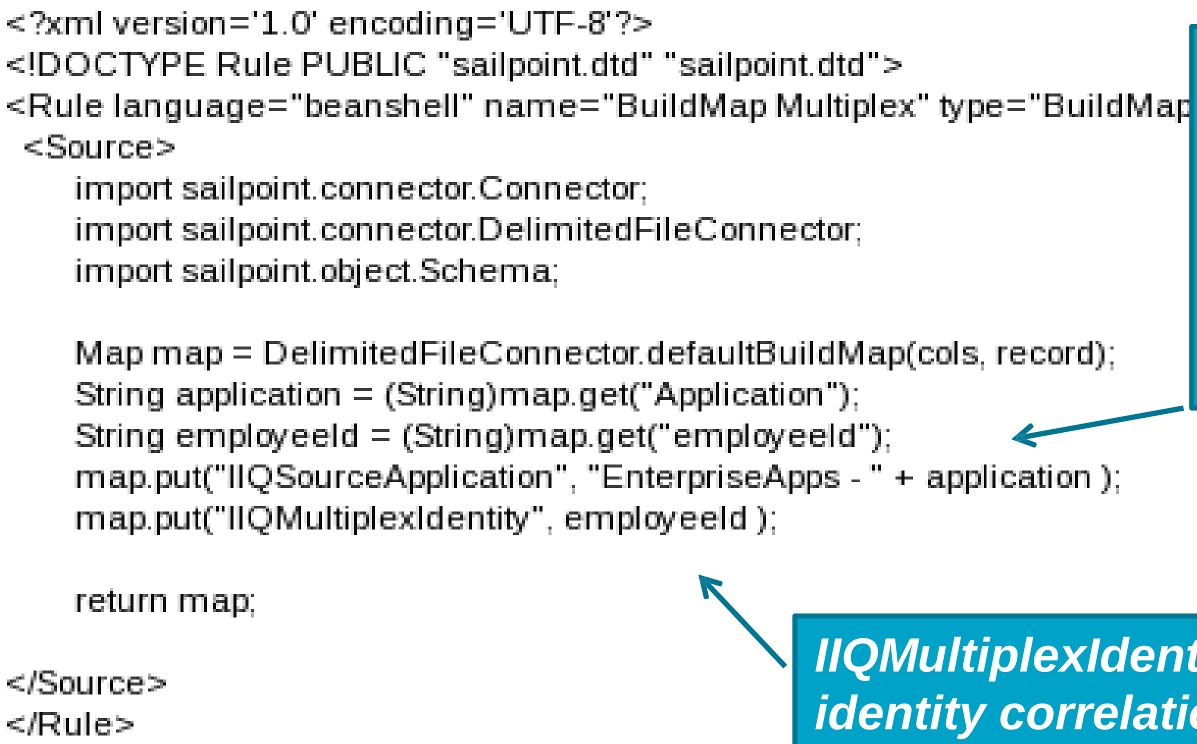
Multiplex Processing

Aggregation



Rule Example

Build Map Rule



```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE Rule PUBLIC "sailpoint.dtd" "sailpoint.dtd">
<Rule language="beanshell" name="BuildMap Multiplex" type="BuildMap
  <Source>
      import sailpoint.connector.Connector;
      import sailpoint.connector.DelimitedFileConnector;
      import sailpoint.object.Schema;

      Map map = DelimitedFileConnector.defaultBuildMap(cols, record);
      String application = (String)map.get("Application");
      String employeeId = (String)map.get("employeeId");
      map.put("IIQSourceApplication", "EnterpriseApps - " + application );
      map.put("IIQMultiplexIdentity", employeeId );

      return map;

</Source>
</Rule>
```

IIQSourceApplication: The name of the Multiplexed Application. We are using the "Application" attribute from the CSV file which contains the data.

IIQMultiplexIdentity: The identity correlation attribute. This is only useful if the repository is sorted by this value.

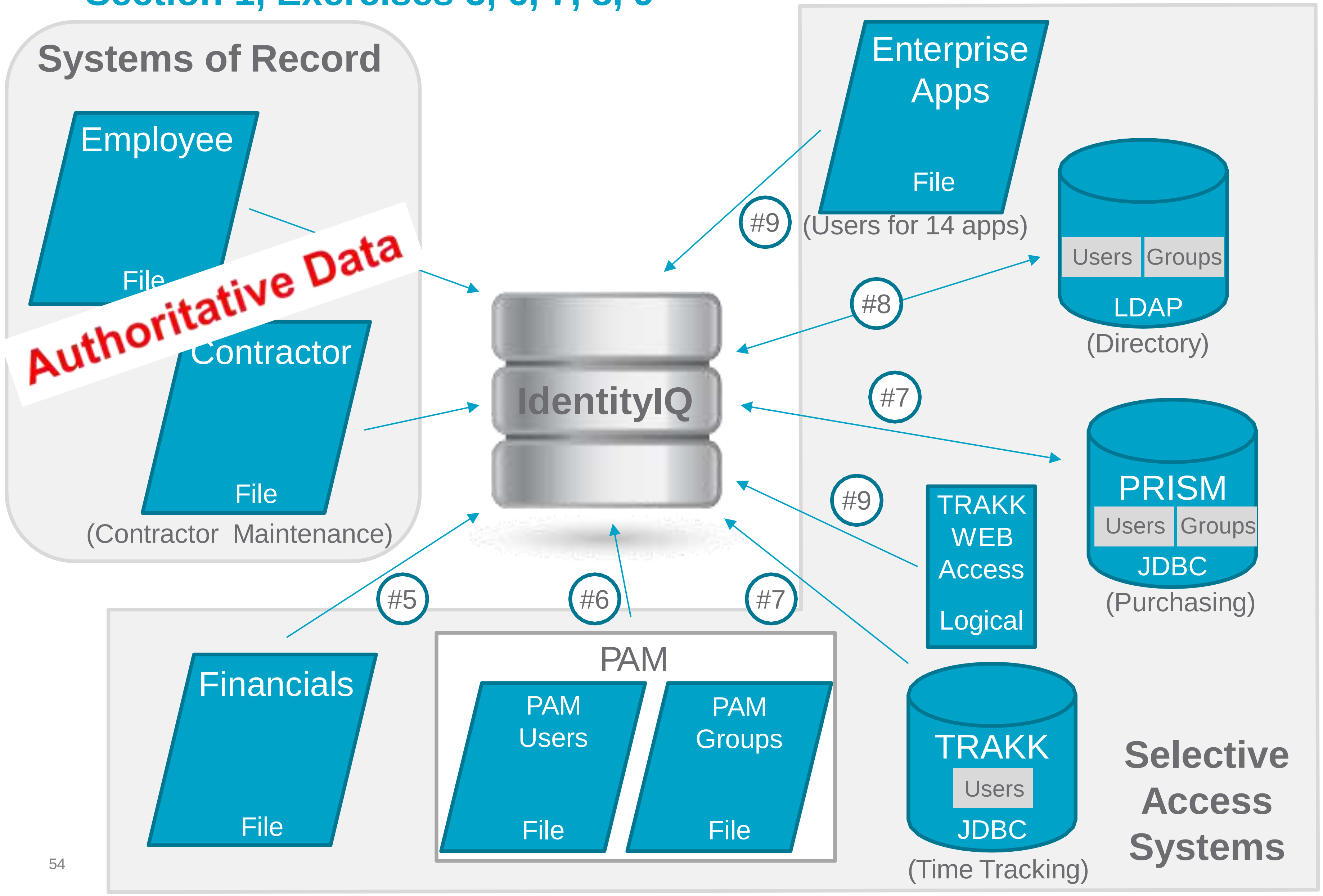
MultiPlex Details

- Schema and correlation rule
  - Copied from base multiplex*ing* application to each target multiplex*ed* application during creation of the target application
- Adding attributes – 2 methods
  - Manually add them to multiplexed schemas
  - Automatically read from repository
    - Create multiplexed apps with common attributes (as above)
    - Add additional attributes to base multiplexing application
      - Manually or with *Discover Account Schema*
    - In aggregation task XML, set *updateMultiplexedSchemas* = *true*
    - Aggregate
    - Attributes are only added to multiplexed applications where data exits for the new attribute for that application
    - When schemas are stable, set *updateMultiplexedSchemas* = false

Questions?

Exercise Preview

Section 1, Exercises 5, 6, 7, 8, 9



Exercise Preview

Section 1, Exercises 9 and 10

- Exercise #9: Onboarding Logical and Multiplexed Applications
  - Multiplexed application is optional
  - Strongly encouraged if you have a multiplex use case
- Exercise #10: Exploring the Identity Refresh Task