Grouper data field and subject source next generation

This is for a future v2.6 release

Terminology:

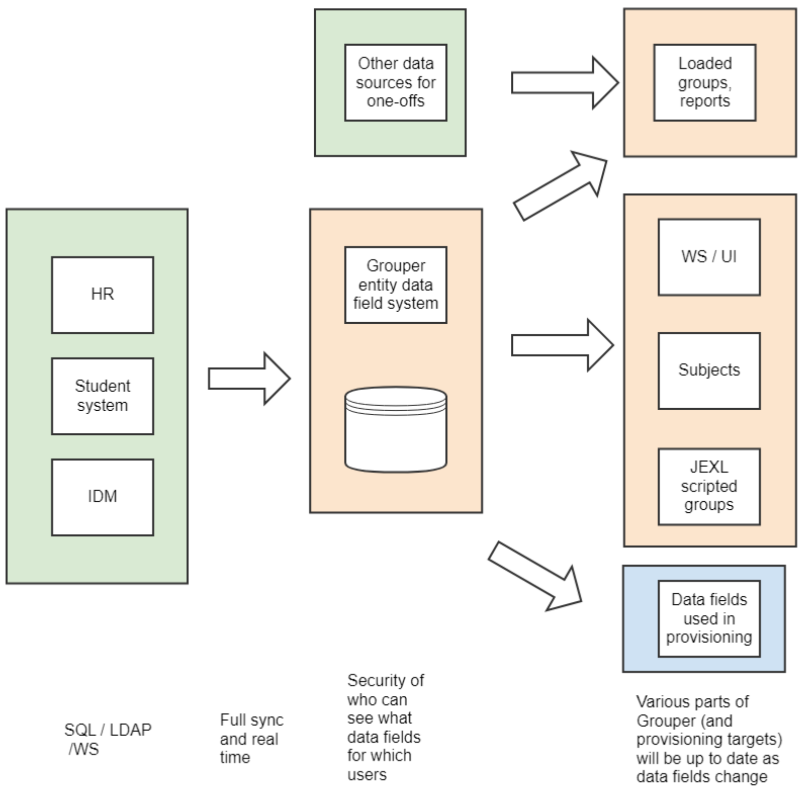
- "data field" is a user attribute, do not want use attribute since it overlaps with attribute framework

This is a suggestion for how user data could flow to Grouper in future state

The problem this is trying to solve

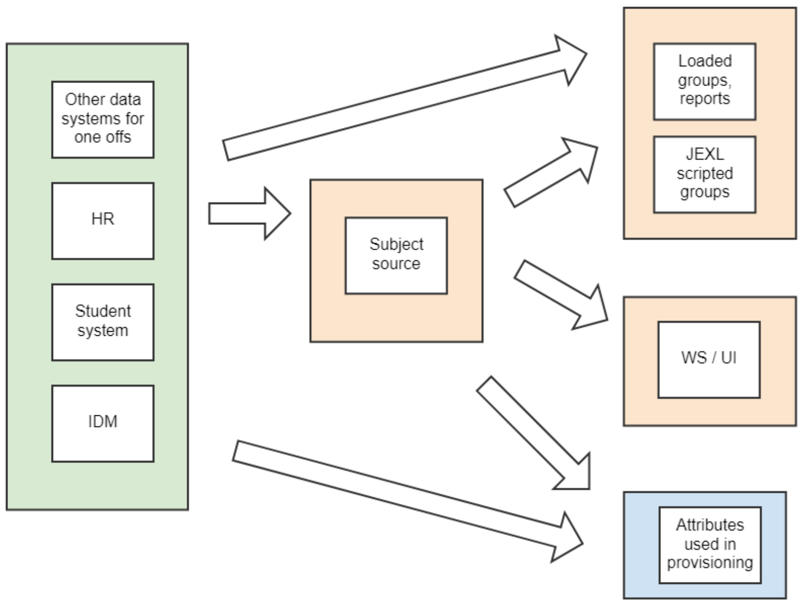
- User data in subjects, provisioners, and loaders solve similar problems
- Real time data solved in multiple ways
- Security of who is allowed to see what (by person aka row or data field aka value)
- Efficiency of being able to query data without reaching out to other resources
- Ability to use data from multiple sources at one time
- Reduce the number of network calls in various places
- Reduce the SQL and LDAP syncs required to make things work
- Troubleshooting access is difficult when the history of data field changes is not known
- Unresolvable subjects are a pain... history of data fields of users will help

Change JEXL to "expression"


Other data sources for one-offs
Loaded groups, reports
HR
Student system
IDM
Grouper entity data field system
WS / UI
Subjects
JEXL scripted groups
Data fields used in provisioning
SQL / LDAP /WS
Full sync and real time
Security of who can see what data fields for which users
Various parts of Grouper (and provisioning targets) will be up to date as data fields change

- In this diagram, the green data field resolvers are either cached (e.g. source systems), or not (one off which doesnt need the overhead of PIT etc)
    - A provisioner target could have entity data field values for users

Previous state

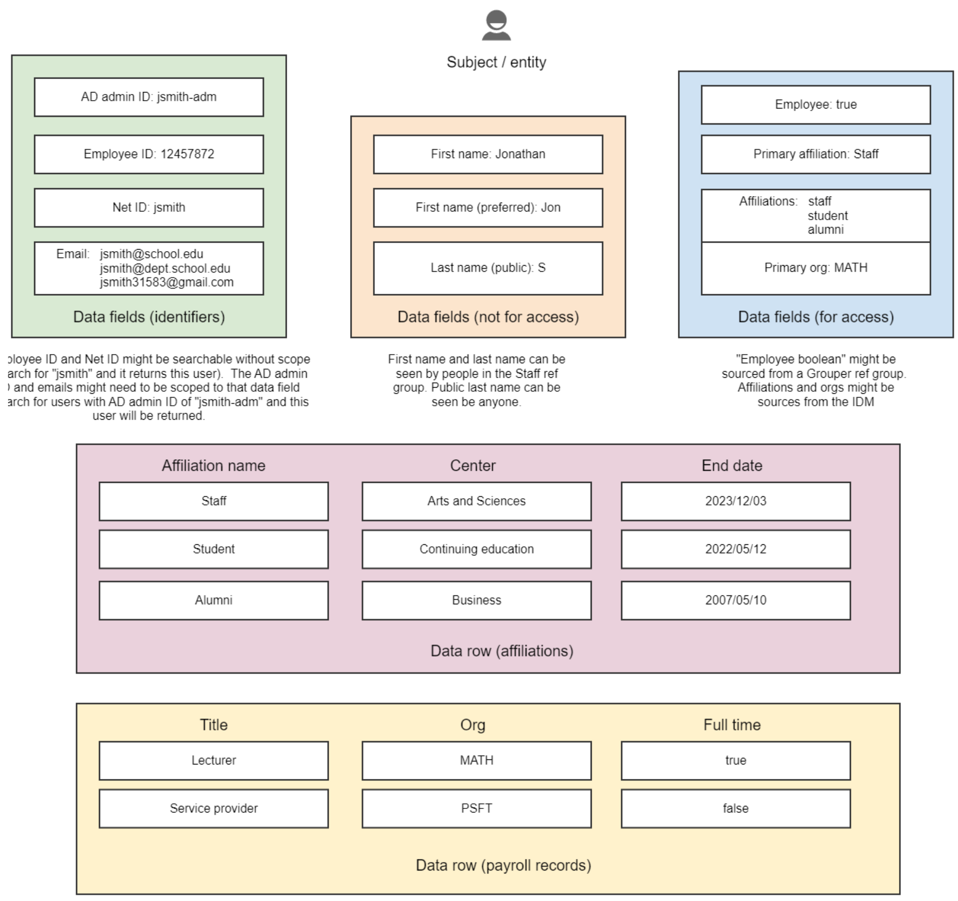


SQL / LDAP

Queries when needed, not cached, not secure. Subject sources difficult to work with

Various parts of grouper need to query out to external systems

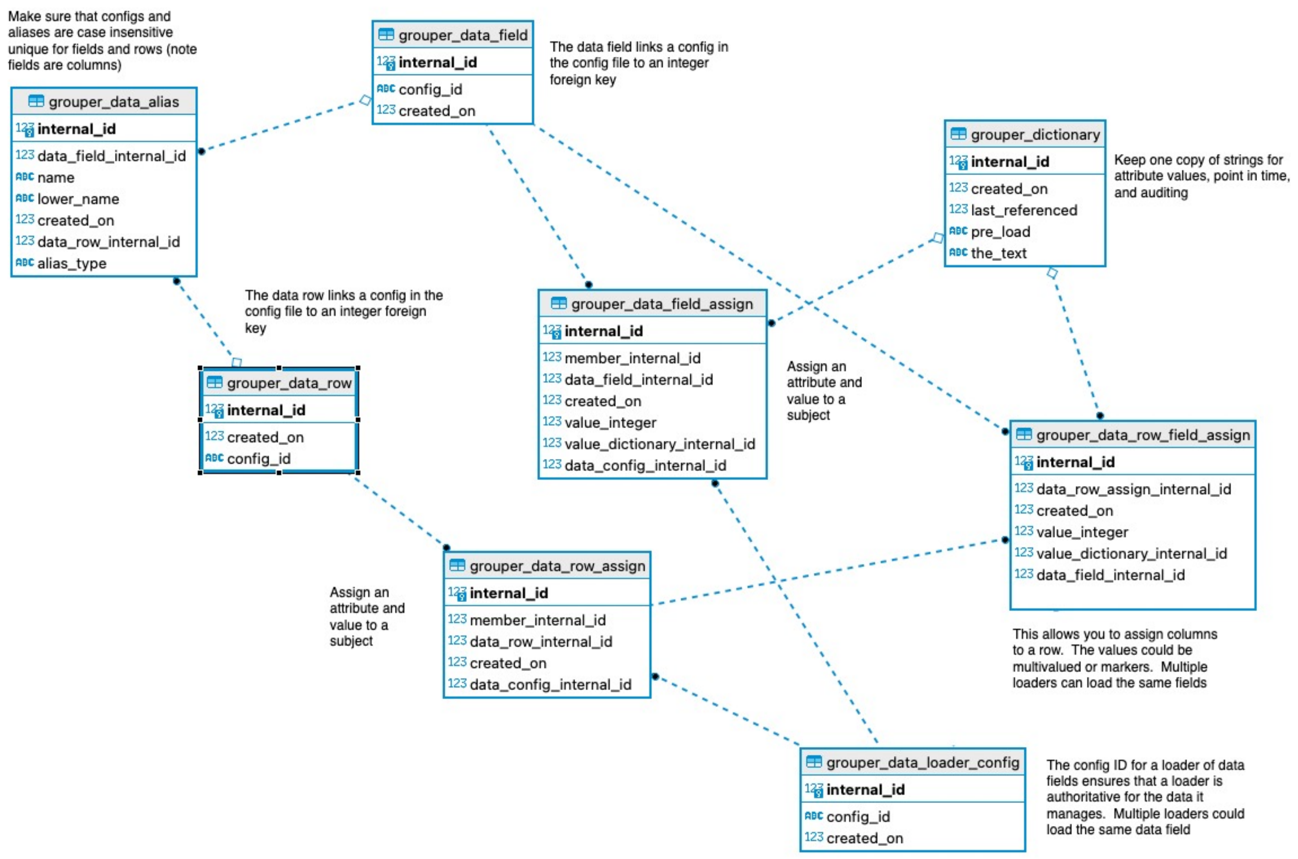
Data field and row diagram


Subject / entity
AD admin ID: jsmith-adm
Employee ID: 12457872
Net ID: jsmith
Email: jsmith@school.edu
jsmith@dept.school.edu
jsmith31583@gmail.com
Data fields (identifiers)
First name: Jonathan
First name (preferred): Jon
Last name (public): S
Data fields (not for access)
Employee: true
Primary affiliation: Staff
Affiliations: staff
student
alumni
Primary org: MATH
Data fields (for access)
ployee ID and Net ID might be searchable without scope
arch for "jsmith" and it returns this user). The AD admin
and emails might need to be scoped to that data field
arch for users with AD admin ID of "jsmith-adm" and this
user will be returned.
First name and last name can be
seen by people in the Staff ref
group. Public last name can be
seen be anyone.
"Employee boolean" might be
sourced from a Grouper ref group.
Affiliations and orgs might be
sources from the IDM
Affiliation name
Center
End date
Staff
Arts and Sciences
2023/12/03
Student
Continuing education
2022/05/12
Alumni
Business
2007/05/10
Data row (affiliations)
Title
Org
Full time
Lecturer
MATH
true
Service provider
PSFT
false
Data row (payroll records)

Example usages:

- Provision any identifier to a target without having to "resolve the subject"
- Make a JEXL scripted group: People who have a payroll data row where org is "MATH" and have an affiliation data row where affiliation name is Staff or Faculty with an End date in the next month. Put a rule on that group for the Business Analyst to review people who might need to be renewed.
- Load users from Zoom and match accounts to users in Grouper by email address
- A staff member creates a report where a column is if the user in the service is an Employee
- A help desk worker can the history of affiliations and troubleshoots access by seeing that the user's payroll org recently changed

Data model



Make sure that configs and aliases are case insensitive unique for fields and rows (note fields are columns)
grouper_data_alias
internal_id
data_field_internal_id
name
lower_name
created_on
data_row_internal_id
alias_type
grouper_data_field
internal_id
config_id
created_on
The data field links a config in the config file to an integer foreign key
grouper_dictionary
internal_id
created_on
last_referenced
pre_load
the_text
Keep one copy of strings for attribute values, point in time, and auditing
The data row links a config in the config file to an integer foreign key
grouper_data_row
internal_id
created_on
config_id
grouper_data_field_assign
internal_id
member_internal_id
data_field_internal_id
created_on
value_integer
value_dictionary_internal_id
data_config_internal_id
Assign an attribute and value to a subject
grouper_data_row_field_assign
internal_id
data_row_assign_internal_id
created_on
value_integer
value_dictionary_internal_id
data_field_internal_id
Assign an attribute and value to a subject
grouper_data_row_assign
internal_id
member_internal_id
data_row_internal_id
created_on
data_config_internal_id
This allows you to assign columns to a row. The values could be multivalued or markers. Multiple loaders can load the same fields
grouper_data_loader_config
internal_id
config_id
created_on
The config ID for a loader of data fields ensures that a loader is authoritative for the data it manages. Multiple loaders could load the same data field

Setup data fields

- Each has unique name (cant have same name as data field row)
- Is identifier?
- If is identifier, is it searchable without data field scope?
  - i.e. can you just search by netId and get a result (yes). or can you search by zoom user id and get a result without telling the search that the value is a zoom user id (no)
- Is access related?
  - Three types of data fields
    - Informational
      - e.g. name, description, email, etc
      - Needed for provisioning or UI or WS
    - Access related
      - e.g. dept, title, school, DN
      - Needed for loading groups, jexl scripted groups, provisioning events
    - Group - sourced
      - A group could be the basis of a data field
      - Imagine eduPersonAffiliation-like reference groups that drive flags on a user or a multi-valued set of affiliations: e.g. student, member, alumni
      - Makes exposing some access information over UI/WS/provisioning easier
      - PIT would still be stored in data field PIT since it is more efficient and consistent API
      - Should be a low number of groups
      - How to translate from group name to data field value
      - How to translate from data field value to group name
      - "like" SQL string to select all applicable groups
- Assignable to
  - Non-groups
  - Groups
    - Source query/filter would need a way to link to group e.g. by group-name or translation
- Stored locally?
- Stored in PIT?
  - If data is not huge volume and doesnt change frequently then yes
  - Policies could be based on history of attributes
  - NetId changes would be easier to deal with
- PIT retention in days
  - If stored in PIT, how many days should be stored. Default to 5 years. Blank means store forever

- Which privacy level
- Multi-valued?
- Calculated?
    - If the value needs to be translated from other data field in a script (e.g. description)
- Dynamic? (can be selected only if calculated)
    - If a script is needed to evaluate the field, and it depends on env or current user (GrouperSesssion), then it is dynamic. e.g. name is institutional_name if the user can see anything there, or public name if not
- From_sole_source: select the source if populated from one source only
- Data type:
    - String (in string lookup table)
    - Integer
    - Date (millis since 1970 integer)
    - Floating point (integer with precision of decimals)
- Could have LOV lookups. e.g. ID, name, display name, description

There are some built in data fields

- grouper_system_identifier (identifier for built in subjects) (single valued) (stored in PIT)
    - GrouperSystem
    - GrouperAll
- grouper_subject_source
    - Multivalued
    - Subject source identifier

Setup data field rows

- Each has unique name (cant have same name as data field)
- If data fields are grouped together and multivalued for a user
- One data field in the row can be an identifier or composite identifier (umich might not have identifiers)
- Could be from
    - SQL
    - Packed LDAP
    - WS
- E.g. affiliations, where each affiliation has a dept

Setup LOVs (list of values)

- There could static information about data field values
- Each column could be single or multi-valued
- Each column is a data field with a unique name across the namespace
- Could have an LOV about a datafield in an LOV (chaining)
- Linked to one or many data fields by one of the columns
- SQL only
- Not kept in point in time
- For instance
    - Affilition could have information
    - Information about an organization
    - Data about a campus

Setup data field sources (aka entity resolvers)

The first configuration step is to set up entity resolvers

For users

- SQL queries
- LDAP filters
- WS calls

Returns

- Single valued data fields for users
- Returns multi-valued data fields for users
- Multivalued rows of data fields for users (e.g. affiliation rows that have affiliation and dept)

Merge / match / split

- Similar to current subject sources, the upstream IDM needs to do the merging of entities. Identifier changes might create conflicts, but could also flow through without incident.
- We should have a "copy access from" function to copy from another user from a PIT, which could help with that though

A user search could trigger a query in a resolver for real time updates, though it would be better if users could be in Grouper when they exist so this is not necessary

Can use change log (LDIF, SQL, or messaging) to get data real time for all or certain users

Assumption

All institutions are either

- OK with full sync of user data fields on a schedule and thats how up to date they are (e.g. every 30 minutes, hourly, daily)
- or: Can get events of when data changes in source systems
- or: Queries to source systems have last updated dates or change logs for real time updates

Data field resolver loads the data

- Copies to Grouper database
- Could process the data a tad
- "Virtual data fields" can have logic and make a complicated description data field (across multiple resolver sources)
    - Its possible that this could help the problem of having too many subject sources though this isn't intended to be an identity system
- Can assign security so Grouper knows who is allowed to read which data
    - Each data field could have a group assigned who can see the data
- There are real time events or timestamps that ensures data is up to date
- Each attribute should only come from one resolver
- Active flag will signify if the user can be added to Grouper groups / privileges (i.e. "resolvable")

Subject source

- Points to Grouper's database
    - Instead being configured against sources would be configured against entity resolver data
    - Can use data from multiple sources
    - All identifiers must be unique for a data field
    - All identifiers should be unique across the identifier space, though they dont need to be.
- Note, if entity resolver data is secure and available over UI/WS then the subject doesnt need any fields... e.g. Penn would not need first name and last name etc in the subject configuration.
- Subject is really just a collection of prioritized identifiers (e.g. employeeId is highest priority) and attributes
- People who are allowed to see various entity resolver data fields would see description a certain way, name a certain way, and whatever data fields they can see when they need it
- Imagine a more detailed subject page for people who can see the data... easier to troubleshoot access
- If an employee ID does change (and no other conflicts), the user could be resolved by other identifiers and it might "just work"

When data fields are referenced, also a two part process. If a group (and user allowed to see), go to group table(s), if anything other than a group, then its the data field tables

If a subject is not found by identifier, search identifier history perhaps, then can be configured to reach out to certain sources

Will free-form searches go to grouper or also to source? Perhaps configurable

Loaders

- Loaders and jexl scripted groups can be written on top of entity data
- Non admins can securely use that data since Grouper knows who is allowed to see what
- When the entity resolver knows that data changed real-time, it knows which loader/jexl scripted group to update
- Not all data about users will be entity resolvers... more than what was in subject source, but not everything
- If there is peripheral data you can make SQL/LDAP loaders for that
- Privileges for loaded groups could be loaded with users who can see all the related data fields

UI/WS

- Imagine more data fields than subject data fields available over WS/UI securely in one query

Provisioning

- No more "subject link"
- You can provision any entity resolver data easily
    - Provisioners would not need their own data resolvers and can just use centrally cached or non-cached resolvers
- When data changes, Grouper can tell a provisioner to recalc a user

Summary

In summary here is a metaphor... we used to have SQL credentials in multiple places, then we made an external system layer to re-use that. This suggested is similar. Have a data layer that can we re-used across things. Includes real-time updates, security, and data manipulation configured centrally... why? if we want to be ABAC and data field-based, we need to organize our data fields

Miscellaneous

How to only load active people or people who change

Data field security (privacy level)

- Identify certain privacy levels
- Each privacy level has a group associated with it in the folder etc:dataField:privacyLevelGroups
- Table with list of privacy levels
- Includes public (no group used) or admin only (only grouper admins can see, no group used)

Subject security

Could have certain groups of subjects who are able to see certain other groups subjects. We will need to get requirements for this and see how it fits in with the data field and data row security

Data model

grouper_subject_source

- id_index
- name
- id
- description
- subject_id_data_field_id_index

grouper_members

Existing table can be stripped down since data is in the entity tables

- id (012)
    - idIndex
    - id_index_foreign_key (id_index of group, local entity, or misc (user)
    - subject_id (uuid for groups or entities, or some value of data field)
    - type (G = group, E = local entity, U = user (person or non person)
    - view_group_id_index (group which can view this subject, comes from privilege or data field view group)
    - search strings
    - sort strings
    - resolvable

grouper_members_identifiers

Make sure identifiers. (hopefully unique)

When subjects are looked up, it can be a two part process (instead of N-part for N subject sources).

1. Look at groups in group table,
2. Look at grouper local entities in group table (maybe move to entities table)
3. Look at subjects (including GrouperSystem, users, apps, things) in the data_field tables based on data fields that are marked as identifiers
4. Perhaps make external calls if configured

- id (737)
    - member_id (012)
    - subject_identifier_value (12345678) (indexed non unique)
    - subject_identifier_value_lower (if case insensitive?)
    - data_field_id

Unique index on data_field_id / subject_identifier_value tuple

grouper_data_field_privacy_level

List of privacy levels

- id (058)
    - system_name (e.g. institutional), this is also the groupExtension if applicable
    - display_name
    - description

grouper_data_field_namespace

- id
    - type (dataField, row, lov, etc)
    - name display name to use

grouper_data_field

Types of data fields for user or rows. Note: a lot of this is configuration and not in this table, maybe we just need a table with id and system name to link to config and be used for foreign keys

- id (234)
    - system_name (emailAddress)
    - display_name (Email)
    - data_type (boolean, string, integer)
    - type (user)
    - multi_valued? false
    - description
    - privacy_level_id 058
    - is_identifier? true
    - access_related? false
    - stored_locally? true
    - stored_in_pit? true
    - pit_retention: 5 * 365
    - group_can_see: ref:staff
    - from_sole_source: my_people
    - calculated: false
    - dynamic: false
    - case sensitive?
- id (567)
    - system_name (org)
    - display_name (Org)
    - data_type (boolean, string, integer)
    - type (row)
    - multi_valued: true
    - description
    - privacy_level_id 058
    - is_identifier? false
    - access_related? true
    - stored_locally? true
    - pit_retention: 5*365
    - group_can_see: ref:powerUsers
    - from_sole_source: my_payroll
    - calculated: false
    - dynamic: false

grouper_data_row

Type of data field rows available for users. Note: a lot of this is configuration and not in this table, maybe we just need a table with id and system name to link to config and be used for foreign key

- id (123)
    - system_name (affiliation)
    - display_name (Affiliation)
    - description
    - privacy_level_id 058

grouper_data_row_field

Which fields are in which rows

- id (538)
    - grouper_data_row_id (012)
    - grouper_data_field_id (567)

grouper_data_lov

Type of data field rows available for metadata about data field. Note: a lot of this is configuration and not in this table, maybe we just need a table with id and system name to link to config and be used for foreign key

- id (123)
    - system_name (affiliation_lov)
    - display_name (AffiliationLov)
    - description
    - privacy_level_id 058

grouper_data_lov_field

Which fields are in which rows

- id (538)
    - grouper_data_lov_id (012)
    - grouper_data_field_id (567)

TODO we need values of LOVs

grouper_data_member_field

Assignment of a data field to an entity.   When data is synced to the data field tables it will need to do some matching and assign a new grouper_members row if existing not found

- id (480)
    - member_id (012)
    - grouper_data_field_id (234)
    - value_id (789)

grouper_data_member_changelog

Events that happen to data fields to be processed by loaders/provisioners/etc. Keep data for a week then delete

- id (480)
    - member_id (012)
    - grouper_data_field_id (234)
    - old_value_id
    - new_value_id
    - date
    - action
    - grouper_data_row_id (123)

grouper_data_member_field_pit

History of data field to entity

- id (480)
    - member_id (012)
    - grouper_data_field_id (234)
    - value_id (789)
    - started_on 1/2/3
    - ended_on

grouper_data_member_row

Assignment of a row of data to an entity

- id (321)
    - member_id (012)
    - grouper_data_row_id (123)

grouper_data_member_row_pit

HIstory of assignment of a row of data to an entity

- id (321)
    - member_id (012)
    - grouper_data_row_id (123)
    - started_on 1/2/3
    - ended_on

grouper_data_member_row_field

Assignment of a field to a row assignment

- id (637)
    - grouper_data_row_field_id (538)
    - value_id (654)

grouper_data_member_row_field_pit

History of assignment of a field to a row assignment

- id (637)
    - grouper_data_row_field_id (538)
    - value_id (654)
    - started_on 4/5/2021
    - ended_on

grouper_string_table

Keep data field values here to reduce data redundancy

- id (789)
    - value (a@b.c)
    - delete_on (3 months in future if detected as not used, throw error if mistake)
- id (654)
    - value (math)
    - delete_on (3 months in future if detected as not used, throw error if mistake)

grouper_data_field_sec_group_mem_cache

Cache these memberships so lookups are fast. Cache this in memory too for long running processes. The groups that are cached are... any groups that secure fields, any groups that secure rows, etc

- id
    - sec_group_id
    - mem_id_index

grouper_data_field_row_sec

Row level security for data

- id (941)
    - grouper_data_field_id (234)
    - group_id_of_result_member
    - privacy_level_id 058

grouper_data_field_row_pop_group

- id
    - group_id_of_result_member
    - privacy_level_id 058