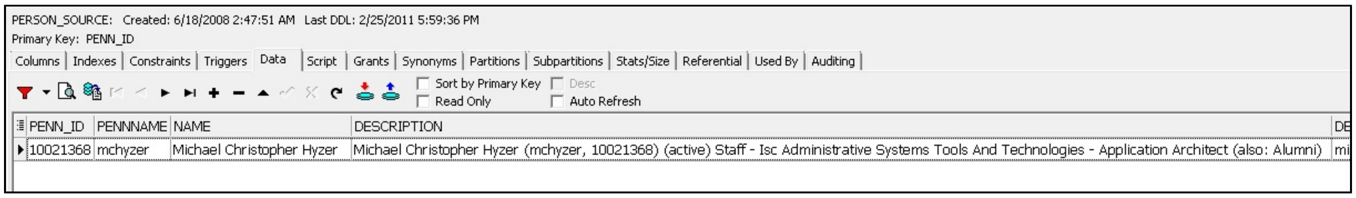
Penn subject source JDBC2 example

Penn is not a big LDAP shop, we have a person database, so we decided to have a feed from there to the Grouper DB into a subject table. The JDBC2 subject source provides more powerful querying (and we contributed it!) so we are using that. We have about 600k subjects. Generally we try not to take subjects out of resolvability once they are resolvable subjects.

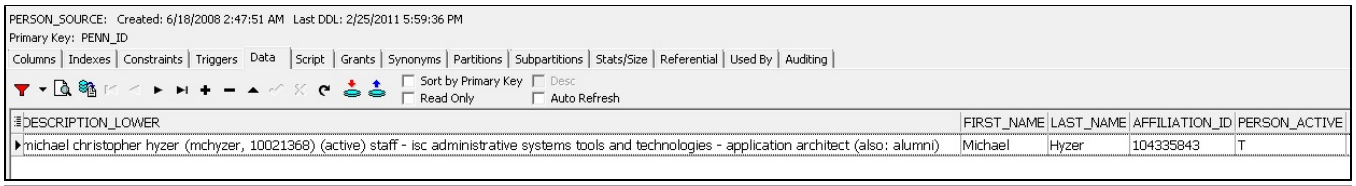
Here is a row from our table



PERSON_SOURCE: Created: 6/18/2008 2:47:51 AM Last DDL: 2/25/2011 5:59:36 PM
Primary Key: PENN_ID

Columns | Indexes | Constraints | Triggers | Data | Script | Grants | Synonyms | Partitions | Subpartitions | Stats/Size | Referential | Used By | Auditing
Sort by Primary Key
Desc
Read Only
Auto Refresh

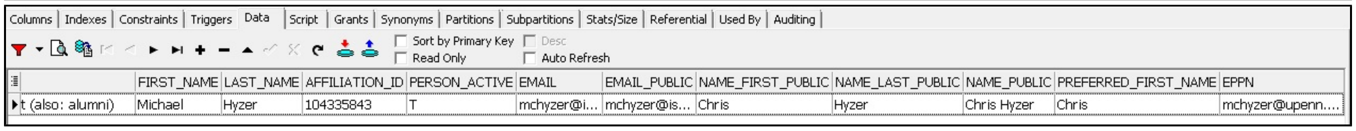
PENN_ID	PENNNAME	NAME	DESCRIPTION	DE
10021368	mchyzer	Michael Christopher Hyzer	Michael Christopher Hyzer (mchyzer, 10021368) (active) Staff - Isc Administrative Systems Tools And Technologies - Application Architect (also: Alumni)	mi



PERSON_SOURCE: Created: 6/18/2008 2:47:51 AM Last DDL: 2/25/2011 5:59:36 PM
Primary Key: PENN_ID

Columns | Indexes | Constraints | Triggers | Data | Script | Grants | Synonyms | Partitions | Subpartitions | Stats/Size | Referential | Used By | Auditing
Sort by Primary Key
Desc
Read Only
Auto Refresh

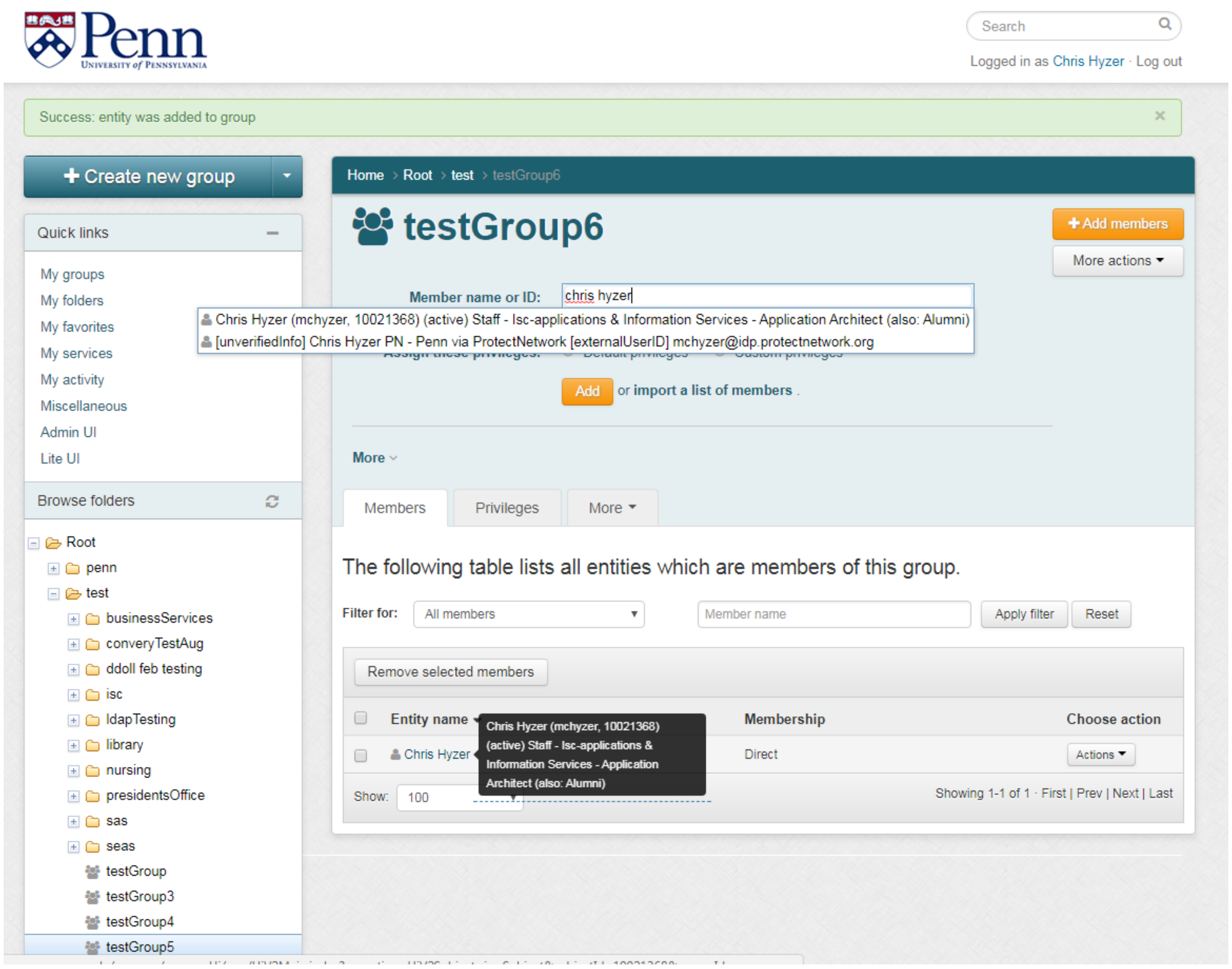
DESCRIPTION_LOWER	FIRST_NAME	LAST_NAME	AFFILIATION_ID	PERSON_ACTIVE
michael christopher hyzer (mchyzer, 10021368) (active) staff - isc administrative systems tools and technologies - application architect (also: alumni)	Michael	Hyzer	104335843	T



Columns | Indexes | Constraints | Triggers | Data | Script | Grants | Synonyms | Partitions | Subpartitions | Stats/Size | Referential | Used By | Auditing
Sort by Primary Key
Desc
Read Only
Auto Refresh

	FIRST_NAME	LAST_NAME	AFFILIATION_ID	PERSON_ACTIVE	EMAIL	EMAIL_PUBLIC	NAME_FIRST_PUBLIC	NAME_LAST_PUBLIC	NAME_PUBLIC	PREFERRED_FIRST_NAME	EPPN
t (also: alumni)	Michael	Hyzer	104335843	T	mchyzer@i...	mchyzer@is...	Chris	Hyzer	Chris Hyzer	Chris	mchyzer@upenn....

Screen showing search result, membership listing, and tooltip


Search

Logged in as Chris Hyzer · Log out

Success: entity was added to group

testGroup6

Member name or ID: chris hyzer

Chris Hyzer (mchyzer, 10021368) (active) Staff - Isc-applications & Information Services - Application Architect (also: Alumni)

[unverifiedInfo] Chris Hyzer PN - Penn via ProtectNetwork [externalUserID] mchyzer@idp.protectnetwork.org

Add or import a list of members .

The following table lists all entities which are members of this group.

Filter for: All members · Member name

Entity name	Membership	Choose action
Chris Hyzer	Direct	Actions

Chris Hyzer (mchyzer, 10021368) (active) Staff - Isc-applications & Information Services - Application Architect (also: Alumni)

Show: 100

Showing 1-1 of 1 · First | Prev | Next | Last

Attributes

- SubjectID is PennID which is an 8 digit opaque number which does not change or get reassigned
- Our subject identifiers are pennkey (netId), and EPPN
- The name is the name which puts all the name pieces together
- The description is a long form of the name which we show in search results. This takes a lot of logic to produce, it is the name, pennkey, pennid, if active, description of primary affiliation, and listing of short forms of other affiliations. This is so descriptive to assure that someone using Grouper does not pick the wrong person. There are privacy issues which we address by using information from the user which they display in the authenticated directory, and we clamp down on which users can access the UI
- We have some subject attributes to help applications which are Grouper enabled get access to user attributes. Things like name, email, eppn, etc.
- The search field for the source is the description in lower-case.

Sync process

- We wrote a Java daemon which runs on real time and nightly to compute the subject source table
- There is a change log on the source table which we use to know which rows to recalculate real time
- We calculate all in batch at night to sync up discrepancies. This takes several hours. We only update records which need updating.

Configuration subject.properties to the grouper database

```
# Copyright 2016 Internet2
#
```

```
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#   http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

#
# Subject configuration
#

# The subject properties uses Grouper Configuration Overlays (documented on wiki)
# By default the configuration is read from subject.base.properties
# (which should not be edited), and the subject.properties overlays
# the base settings.  See the subject.base.properties for the possible
# settings that can be applied to the subject.properties

# enter the location of the sources.xml.  Must start with classpath: or file:
# blank means dont use sources.xml, use subject.properties
# default is: classpath:sources.xml
# e.g. file:/dir1/dir2/sources.xml
subject.sources.xml.location =


#########################################
## Configuration for source id: pennperson
## Source configName: pennperson
#########################################
subjectApi.source.pennperson.id = pennperson

# this is a friendly name for the source
subjectApi.source.pennperson.name = Penn person

# type is not used all that much.  Can have multiple types, comma separate.  Can be person, group, application
subjectApi.source.pennperson.types = person

# the adapter class implements the interface: edu.internet2.middleware.subject.Source
# adapter class must extend: edu.internet2.middleware.subject.provider.BaseSourceAdapter
# edu.internet2.middleware.grouper.subj.GrouperJdbcSourceAdapter2  :  if doing JDBC this should be used if
possible.  All subject data in one table/view.
# edu.internet2.middleware.grouper.subj.GrouperJdbcSourceAdapter   :  oldest JDBC source.  Put freeform queries
in here
# edu.internet2.middleware.grouper.subj.GrouperJndiSourceAdapter   :  used for LDAP
subjectApi.source.pennperson.adapterClass = edu.internet2.middleware.subject.provider.JDBCSourceAdapter2

subjectApi.source.pennperson.param.maxPageSize.value = 100

# edu.internet2.middleware.subject.provider.C3p0JdbcConnectionProvider (default)
# edu.internet2.middleware.subject.provider.DbcpJdbcConnectionProvider (legacy)
# edu.internet2.middleware.grouper.subj.GrouperJdbcConnectionProvider
# (same settings as grouper.hibernate.properties, the driver, url, pass, maxActive, maxIdle, maxWait are
forbidden
subjectApi.source.pennperson.param.jdbcConnectionProvider.value = edu.internet2.middleware.grouper.subj.
GrouperJdbcConnectionProvider

# the table or view to query results from.  Note, could prefix with a schema name
subjectApi.source.pennperson.param.dbTableOrView.value = person_source_v

# the column name to get the subjectId from
subjectApi.source.pennperson.param.subjectIdCol.value = penn_id

# the column name to get the name from
subjectApi.source.pennperson.param.nameCol.value = name

subjectApi.source.pennperson.param.descriptionCol.value = description
```

```
# search col where general searches take place, lower case
subjectApi.source.pennperson.param.lowerSearchCol.value = description_lower

# optional col if you want the search results sorted in the API (note, UI might override)
subjectApi.source.pennperson.param.defaultSortCol.value = description

# you can count up from 0 to N of columns to search by identifier (which might also include by id)
subjectApi.source.pennperson.param.subjectIdentifierCol0.value = pennname

# you can count up from 0 to N of columns to search by identifier (which might also include by id)
subjectApi.source.pennperson.param.subjectIdentifierCol1.value = penn_id

# you can count up from 0 to N of columns to search by identifier (which might also include by id)
subjectApi.source.pennperson.param.subjectIdentifierCol2.value = eppn

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol0.value = pennname

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName0.value = PENNNAME

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol1.value = email

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName1.value = EMAIL

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol2.value = eppn

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName2.value = EPPN

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol3.value = first_name

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName3.value = FIRST_NAME

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol4.value = last_name

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName4.value = LAST_NAME

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol5.value = email_public

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName5.value = EMAIL_PUBLIC

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol6.value = name_first_public

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName6.value = NAME_FIRST_PUBLIC
```

```
# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol7.value = name_last_public

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName7.value = NAME_LAST_PUBLIC

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol8.value = name_public

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName8.value = NAME_PUBLIC

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol9.value = preferred_first_name

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName9.value = PREFERRED_FIRST_NAME

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol10.value = description_lower

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName10.value = DESCRIPTION_LOWER

# the 1st sort attribute for lists on screen that are derived from member table (e.g. search for member in
group)
# you can have up to 5 sort attributes
subjectApi.source.pennperson.param.sortAttribute0.value = DESCRIPTION_LOWER

# the 2nd sort attribute for lists on screen that are derived from member table (e.g. search for member in
group)
# you can have up to 5 sort attributes
subjectApi.source.pennperson.param.sortAttribute1.value = LAST_NAME

# the 1st search attribute for lists on screen that are derived from member table (e.g. search for member in
group)
# you can have up to 5 search attributes
subjectApi.source.pennperson.param.searchAttribute0.value = DESCRIPTION_LOWER

# list your identifiers here by attribute
subjectApi.source.pennperson.param.subjectIdentifierAttribute0.value=PENNNAME
subjectApi.source.pennperson.param.subjectIdentifierAttribute1.value=EPPN
```

Configuration subject.properties to an external database

```
# Copyright 2016 Internet2
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#   http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

#
# Subject configuration
#

# The subject properties uses Grouper Configuration Overlays (documented on wiki)
# By default the configuration is read from subject.base.properties
# (which should not be edited), and the subject.properties overlays
# the base settings.  See the subject.base.properties for the possible
# settings that can be applied to the subject.properties

# enter the location of the sources.xml.  Must start with classpath: or file:
# blank means dont use sources.xml, use subject.properties
# default is: classpath:sources.xml
# e.g. file:/dir1/dir2/sources.xml
subject.sources.xml.location =


#########################################
## Configuration for source id: pennperson
## Source configName: pennperson
#########################################
subjectApi.source.pennperson.id = pennperson

# this is a friendly name for the source
subjectApi.source.pennperson.name = Penn person

# type is not used all that much.  Can have multiple types, comma separate.  Can be person, group, application
subjectApi.source.pennperson.types = person

# the adapter class implements the interface: edu.internet2.middleware.subject.Source
# adapter class must extend: edu.internet2.middleware.subject.provider.BaseSourceAdapter
# edu.internet2.middleware.grouper.subj.GrouperJdbcSourceAdapter2  :  if doing JDBC this should be used if
possible.  All subject data in one table/view.
# edu.internet2.middleware.grouper.subj.GrouperJdbcSourceAdapter   :  oldest JDBC source.  Put freeform queries
in here
# edu.internet2.middleware.grouper.subj.GrouperJndiSourceAdapter   :  used for LDAP
subjectApi.source.pennperson.adapterClass = edu.internet2.middleware.subject.provider.JDBCSourceAdapter2

subjectApi.source.pennperson.param.maxPageSize.value = 100

# edu.internet2.middleware.subject.provider.C3p0JdbcConnectionProvider (default)
# edu.internet2.middleware.subject.provider.DbcpJdbcConnectionProvider (legacy)
# edu.internet2.middleware.grouper.subj.GrouperJdbcConnectionProvider
# (same settings as grouper.hibernate.properties, the driver, url, pass, maxActive, maxIdle, maxWait are
forbidden
subjectApi.source.pennperson.param.jdbcConnectionProvider.value = edu.internet2.middleware.subject.provider.
C3p0JdbcConnectionProvider

#       e.g. mysql:           jdbc:mysql://localhost:3306/grouper
#       e.g. p6spy (log sql): [use the URL that your DB requires]
#       e.g. oracle:          jdbc:oracle:thin:@server.school.edu:1521:sid
#       e.g. hsqldb (a):      jdbc:hsqldb:dist/run/grouper;create=true
#       e.g. hsqldb (b):      jdbc:hsqldb:hsql://localhost:9001
#       e.g. postgres:        jdbc:postgresql:grouper
subjectApi.source.pennperson.param.dbUrl.value = jdbc:mysql://localhost:3306/grouper

# username when connecting to the database
subjectApi.source.pennperson.param.dbUser.value = sa

# password when connecting to the database (or file with encrypted password inside)
subjectApi.source.pennperson.param.dbPwd.value = whatever

# the table or view to query results from.  Note, could prefix with a schema name
subjectApi.source.pennperson.param.dbTableOrView.value = person_source_v

# the column name to get the subjectId from
subjectApi.source.pennperson.param.subjectIdCol.value = penn_id

# the column name to get the name from
```

```
subjectApi.source.pennperson.param.nameCol.value = name

subjectApi.source.pennperson.param.descriptionCol.value = description

# search col where general searches take place, lower case
subjectApi.source.pennperson.param.lowerSearchCol.value = description_lower

# optional col if you want the search results sorted in the API (note, UI might override)
subjectApi.source.pennperson.param.defaultSortCol.value = description

# you can count up from 0 to N of columns to search by identifier (which might also include by id)
subjectApi.source.pennperson.param.subjectIdentifierCol0.value = pennname

# you can count up from 0 to N of columns to search by identifier (which might also include by id)
subjectApi.source.pennperson.param.subjectIdentifierCol1.value = penn_id

# you can count up from 0 to N of columns to search by identifier (which might also include by id)
subjectApi.source.pennperson.param.subjectIdentifierCol2.value = eppn

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol0.value = pennname

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName0.value = PENNNAME

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol1.value = email

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName1.value = EMAIL

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol2.value = eppn

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName2.value = EPPN

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol3.value = first_name

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName3.value = FIRST_NAME

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol4.value = last_name

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName4.value = LAST_NAME

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol5.value = email_public

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName5.value = EMAIL_PUBLIC

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol6.value = name_first_public
```

```
# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName6.value = NAME_FIRST_PUBLIC

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol7.value = name_last_public

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName7.value = NAME_LAST_PUBLIC

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol8.value = name_public

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName8.value = NAME_PUBLIC

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol9.value = preferred_first_name

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName9.value = PREFERRED_FIRST_NAME

# now you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeCol10.value = description_lower

# you can count up from 0 to N of attributes for various cols.  The name is how to reference in subject.
getAttribute()
subjectApi.source.pennperson.param.subjectAttributeName10.value = DESCRIPTION_LOWER

# the 1st sort attribute for lists on screen that are derived from member table (e.g. search for member in
group)
# you can have up to 5 sort attributes
subjectApi.source.pennperson.param.sortAttribute0.value = DESCRIPTION_LOWER

# the 2nd sort attribute for lists on screen that are derived from member table (e.g. search for member in
group)
# you can have up to 5 sort attributes
subjectApi.source.pennperson.param.sortAttribute1.value = LAST_NAME

# the 1st search attribute for lists on screen that are derived from member table (e.g. search for member in
group)
# you can have up to 5 search attributes
subjectApi.source.pennperson.param.searchAttribute0.value = DESCRIPTION_LOWER
```