

Data management

Data relationships

Data relationships

- We have options
- Cardinality
- Embedding vs references, e.g.:
 - user-address, 1:1, embedded (address in user)
 - user-group, many:many, references

Modeling decisions

- Which way?
 - two-way: user and group each store list of refs
 - one-way: only store on one side
- How are you most likely to access data?

Data access scenarios

- Some questions to ask:
 - more likely to access users from groups, or the other way?
 - more users per group, or more groups per user?
- Our scenario:
 - Equally likely to access either way
 - More users per group than groups per user

Provisional decision

- Store user id's in group. (one way only)
- group_models.py:

```
class GroupIn(BaseModel):  
    ''' name is unique field  
    users is a list of unique user identifiers (e.g. id)'''  
  
    name:str  
    users:Optional[list[str]]=[]
```

GroupManager

- copy and modify from UserManager:
 - init / reset
 - CRUD operations
- We will add methods for group-user relationship

Adding users to group

- Our result models can be the same as before
- GroupManager.py:

```
def add_user(self,g_id,u_id) -> UpdateGroupResult:

    query = {'_id':g_id}
    y = self.col.update_one(query, {'$addToSet':{'users':u_id}})
    if y.modified_count>0:
        ''' return appropriate result '''
    else:
        ''' return appropriate result '''
```

Removing users

```
def remove_user(self,g_id,u_id) -> UpdateGroupResult:

    query = {'_id':g_id}
    y = self.col.update_one(query, {'$pull':{'users':u_id}})
    if y.modified_count>0:
        ''' return appropriate result '''
    else:
        ''' return appropriate result '''
```


Accessing user by group

- This is a simple read to get the list of user ids.
- Notice that we are using a 'projection' to get only the user list

```
def get_group_users(self, gid) -> list[str]:  
  
    doc = self.col.find_one({"_id":gid},{'users':True})  
    if doc:  
        return doc.get('users')  
    else:  
        return None
```

What about accessing groups by user?

- E.g. list of groups that user A is in
- If we only have one-way references, we will need to query all of the groups to see which group user A is in.

```
def get_user_groups(self, uid) -> GroupCollection:  
  
    query = {"users": {"$in": [uid]}}  
  
    docs = self.col.find(query)  
    return GroupCollection(groups=docs)
```

Storing some information with references

- Scenario: go to group and get a list of users
- If all we have is a unique id, we don't have anything to show the user
 - This might not be a problem if we use a unique username that is also presentable.
- Solution: store a display string along with the unique identifier.