ArchivesSpace Reporting with MySQL

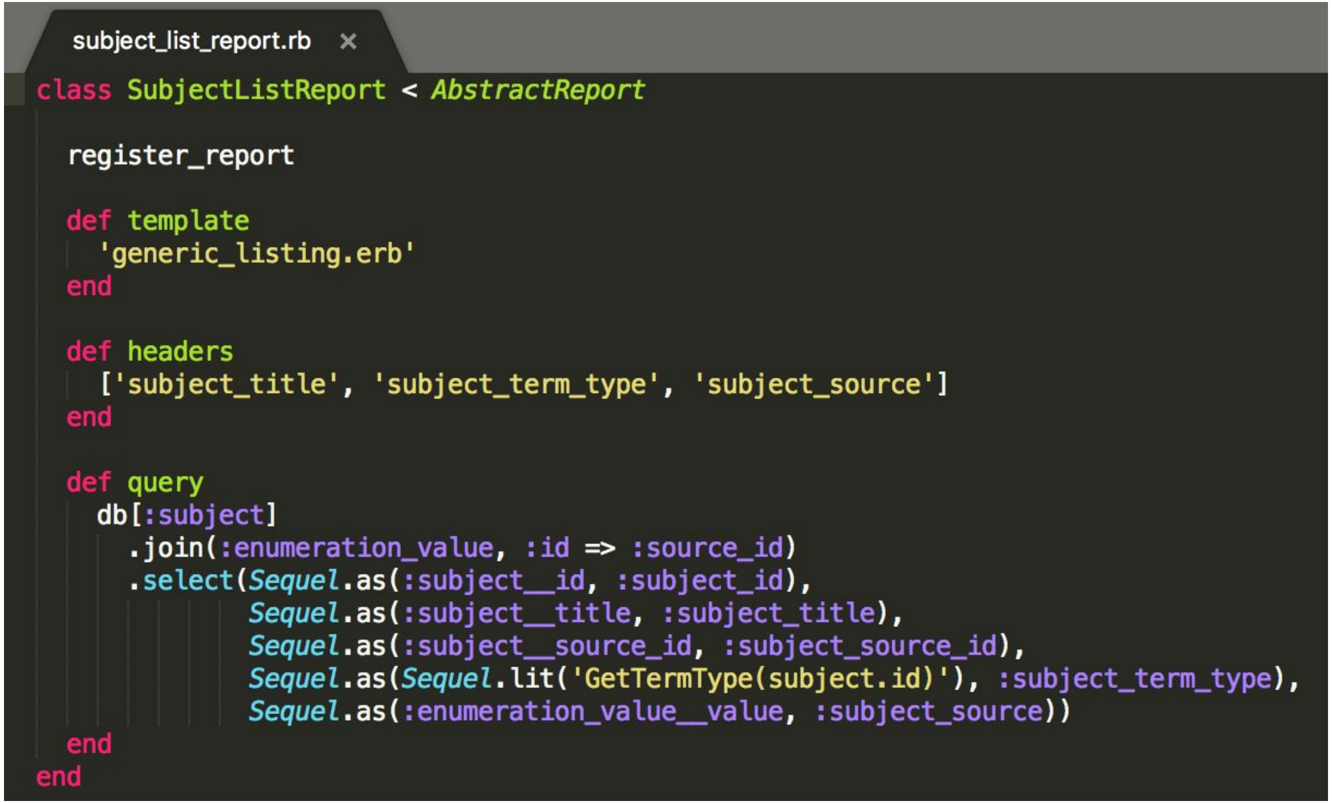
Alicia Detelich, Yale University

ArchivesSpace Member Forum 2018

Why MySQL?

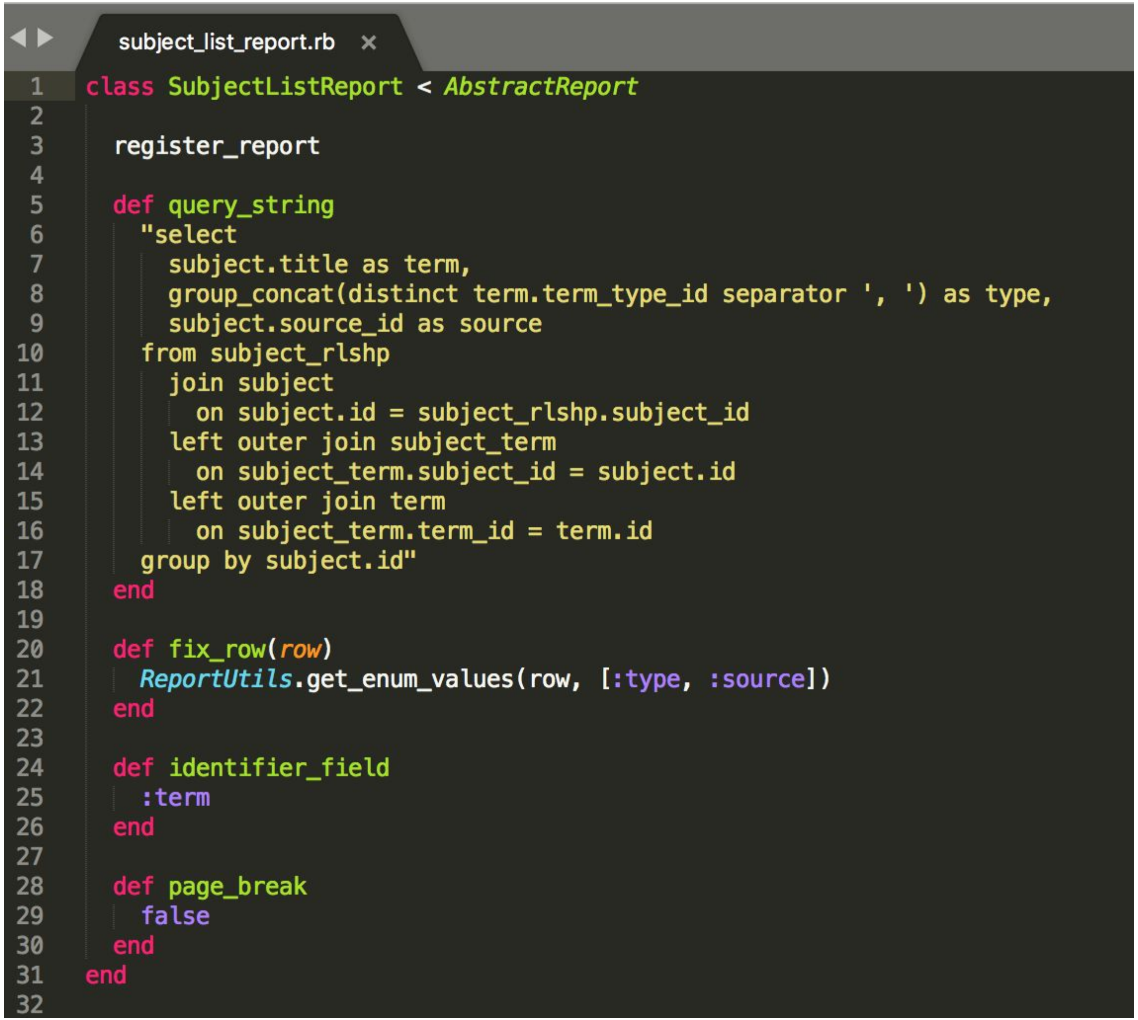
- With MySQL, you can:
  - Query across repositories (or limit by repo, if desired)
  - Filter results, apply conditions, group, sort, and more
  - Answer complex questions about your collections data
- Enhancements to reports module in v2.5.0-RC1 allow users to add their own raw SQL to ArchivesSpace
  - http://localhost:8080/jobs/new?job_type=report_job

Old subject list report



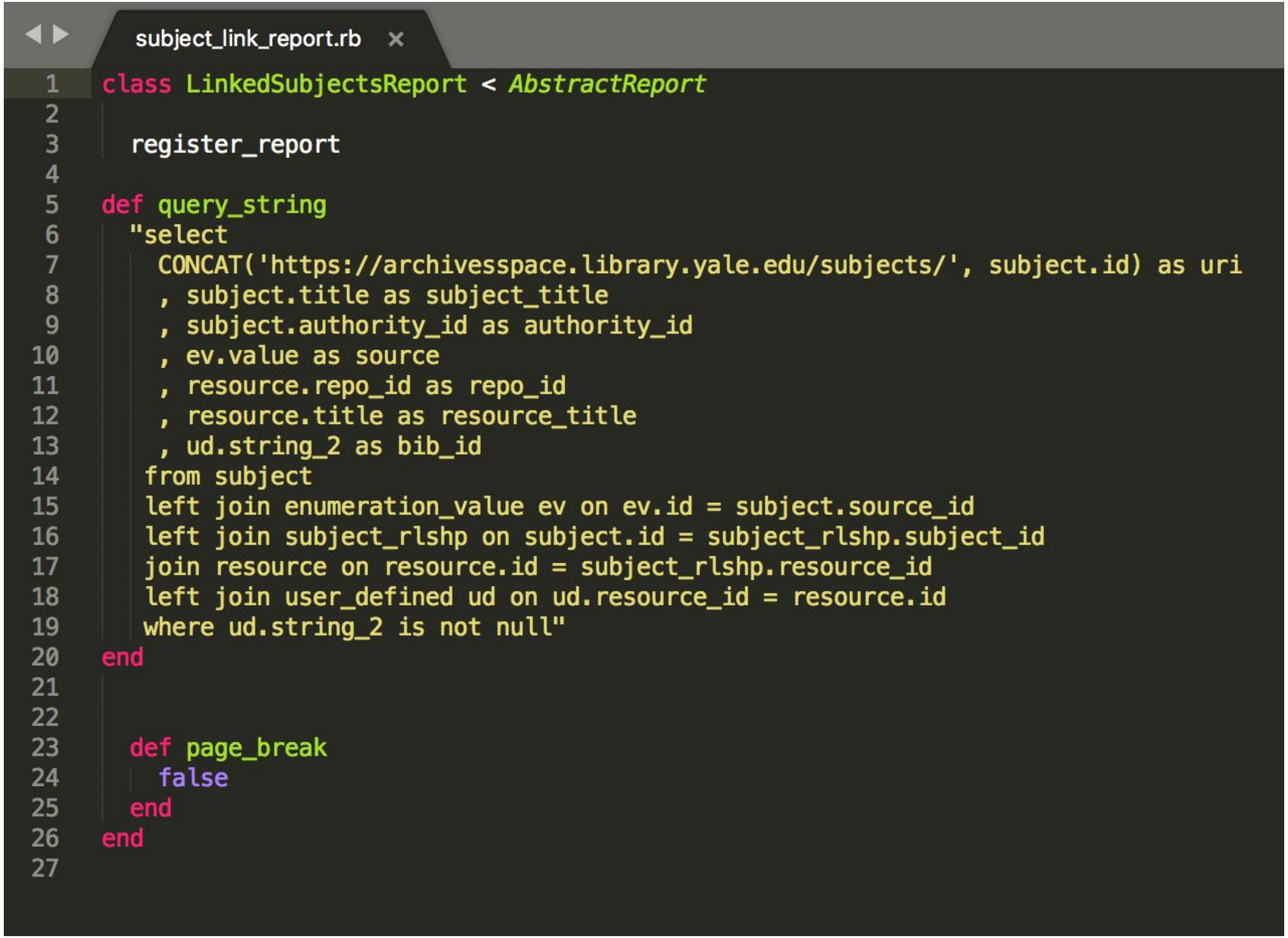
```ruby
class SubjectListReport < AbstractReport

  register_report

  def template
    'generic_listing.erb'
  end

  def headers
    ['subject_title', 'subject_term_type', 'subject_source']
  end

  def query
    db[:subject]
      .join(:enumeration_value, :id => :source_id)
      .select(Sequel.as(:subject__id, :subject_id),
              Sequel.as(:subject__title, :subject_title),
              Sequel.as(:subject__source_id, :subject_source_id),
              Sequel.as(Sequel.lit('GetTermType(subject.id)'), :subject_term_type),
              Sequel.as(:enumeration_value__value, :subject_source))
  end
end
```

New subject list report!



```ruby
class SubjectListReport < AbstractReport

  register_report

  def query_string
    "select
      subject.title as term,
      group_concat(distinct term.term_type_id separator ', ') as type,
      subject.source_id as source
    from subject_rlshp
      join subject
        on subject.id = subject_rlshp.subject_id
      left outer join subject_term
        on subject_term.subject_id = subject.id
      left outer join term
        on subject_term.term_id = term.id
    group by subject.id"
  end

  def fix_row(row)
    ReportUtils.get_enum_values(row, [:type, :source])
  end

  def identifier_field
    :term
  end

  def page_break
    false
  end
end

```

Custom report!



```ruby
class LinkedSubjectsReport < AbstractReport

  register_report

def query_string
  "select
    CONCAT('https://archivesspace.library.yale.edu/subjects/', subject.id) as uri
    , subject.title as subject_title
    , subject.authority_id as authority_id
    , ev.value as source
    , resource.repo_id as repo_id
    , resource.title as resource_title
    , ud.string_2 as bib_id
   from subject
   left join enumeration_value ev on ev.id = subject.source_id
   left join subject_rlshp on subject.id = subject_rlshp.subject_id
   join resource on resource.id = subject_rlshp.resource_id
   left join user_defined ud on ud.resource_id = resource.id
   where ud.string_2 is not null"
end


  def page_break
    false
  end
end

```

Why Not MySQL?

- IT Probs
  - Can’t get access to database
  - Can’t get MySQL clients installed/lack administrator permissions
- Requires some technical knowledge
  - Learning takes time
- Requires clean, complete, well-structured data
  - Data cleanup and enhancement takes time
- Cannot limit access by repository

Getting Started

- Read(-only!) access to ArchivesSpace database
  - Username
  - Password
  - Host name
  - Port (probably 3306)
  - Database name
- MySQL client
  - MySQL Workbench
  - Sequel Pro (Mac)
  - Heidi SQL (Windows)

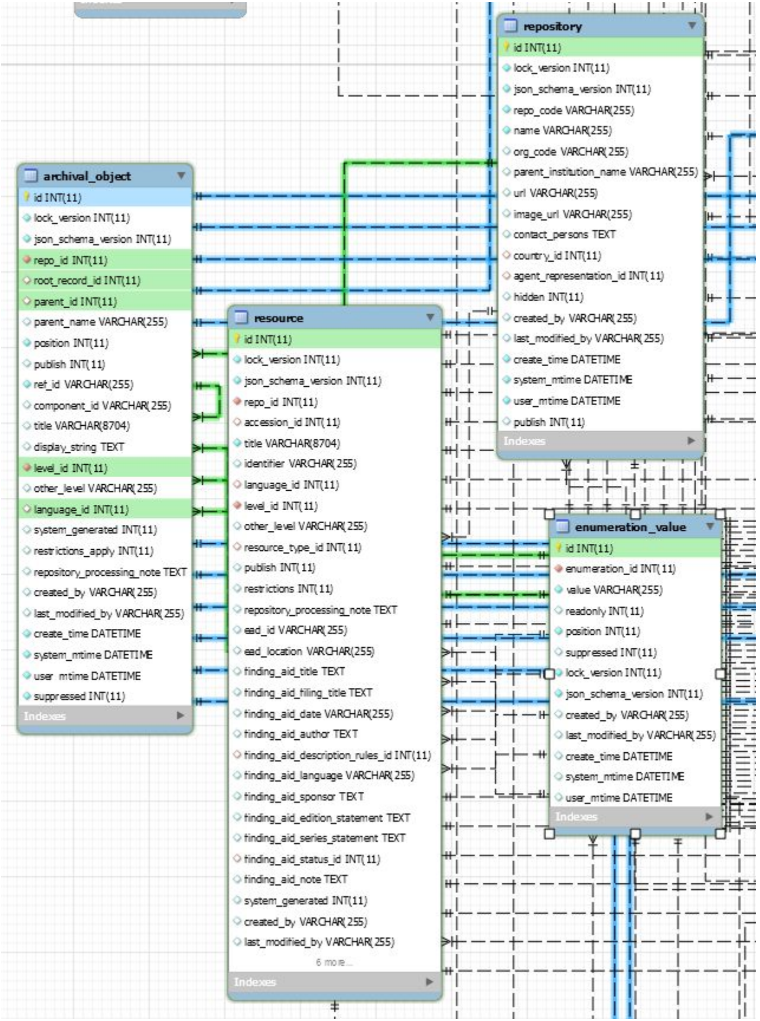
Getting Started

- Basic understanding of relational databases/SQL
  - W3Schools, Khan Academy, CodeAcademy, Lynda.com, and so on
  - StackOverflow
  - Books, etc. - many free and paid resources out there
- Basic knowledge of ArchivesSpace database table relationships
  - Documentation
  - MySQL client table explorers
  - Other people’s queries
  - Experimentation/trial and error (read-only!)

Relational Database Model

- Organizes data into one or more tables
- Each table represents a type of entity - i.e. archival object, resource, extent, instance, etc.
- A unique identifier - the *primary key* - is assigned to each row
- Tables are linked to one another by adding columns to a table which hold the primary keys other tables - *foreign keys*
- Queries are formulated by combining tables using these keys as match points. Can apply filters and conditions, and perform a variety of other operations.

Part of the ArchivesSpace entity-relationship diagram



archival_object

- id INT(11)
- lock_version INT(11)
- json_schema_version INT(11)
- repo_id INT(11)
- root_record_id INT(11)
- parent_id INT(11)
- parent_name VARCHAR(255)
- position INT(11)
- publish INT(11)
- ref_id VARCHAR(255)
- component_id VARCHAR(255)
- title VARCHAR(8704)
- display_string TEXT
- level_id INT(11)
- other_level VARCHAR(255)
- language_id INT(11)
- system_generated INT(11)
- restrictions_apply INT(11)
- repository_processing_note TEXT
- created_by VARCHAR(255)
- last_modified_by VARCHAR(255)
- create_time DATETIME
- system_mtime DATETIME
- user mtime DATETIME
- suppressed INT(11)

Indexes

resource

- id INT(11)
- lock_version INT(11)
- json_schema_version INT(11)
- repo_id INT(11)
- accession_id INT(11)
- title VARCHAR(8704)
- identifier VARCHAR(255)
- language_id INT(11)
- level_id INT(11)
- other_level VARCHAR(255)
- resource_type_id INT(11)
- publish INT(11)
- restrictions INT(11)
- repository_processing_note TEXT
- ead_id VARCHAR(255)
- ead_location VARCHAR(255)
- finding_aid_title TEXT
- finding_aid_filing_title TEXT
- finding_aid_date VARCHAR(255)
- finding_aid_author TEXT
- finding_aid_description_rules_id INT(11)
- finding_aid_language VARCHAR(255)
- finding_aid_sponsor TEXT
- finding_aid_edition_statement TEXT
- finding_aid_series_statement TEXT
- finding_aid_status_id INT(11)
- finding_aid_note TEXT
- system_generated INT(11)
- created_by VARCHAR(255)
- last_modified_by VARCHAR(255)

6 more...

Indexes

repository

- id INT(11)
- lock_version INT(11)
- json_schema_version INT(11)
- repo_code VARCHAR(255)
- name VARCHAR(255)
- org_code VARCHAR(255)
- parent_institution_name VARCHAR(255)
- url VARCHAR(255)
- image_url VARCHAR(255)
- contact_persons TEXT
- country_id INT(11)
- agent_representation_id INT(11)
- hidden INT(11)
- created_by VARCHAR(255)
- last_modified_by VARCHAR(255)
- create_time DATETIME
- system_mtime DATETIME
- user_mtime DATETIME
- publish INT(11)

Indexes

enumeration_value

- id INT(11)
- enumeration_id INT(11)
- value VARCHAR(255)
- readonly INT(11)
- position INT(11)
- suppressed INT(11)
- lock_version INT(11)
- json_schema_version INT(11)
- created_by VARCHAR(255)
- last_modified_by VARCHAR(255)
- create_time DATETIME
- system_mtime DATETIME
- user_mtime DATETIME

Indexes

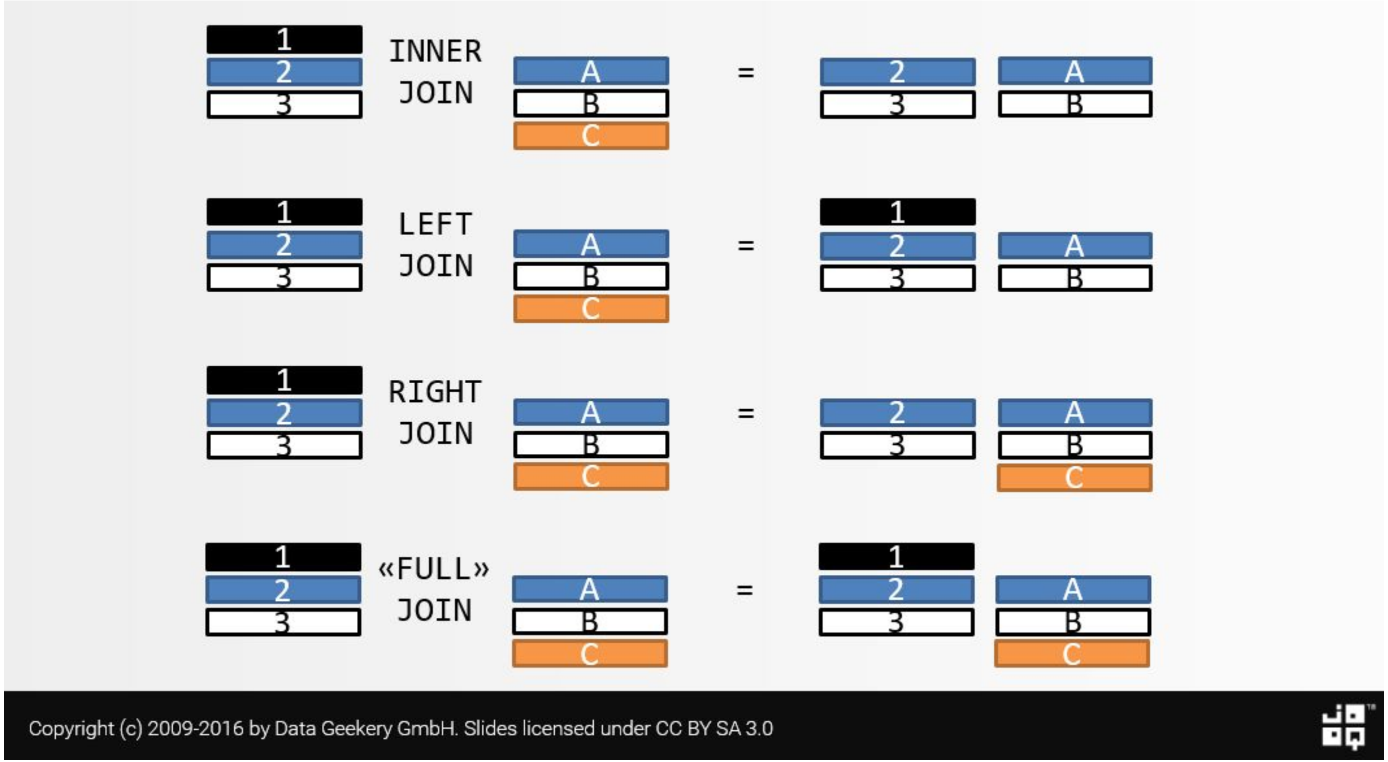
SQL Basics

- SQL is a robust query language, but it is possible to construct useful queries with relatively limited knowledge
- SELECT statement
  - Most basic query command
  - Retrieves data from one or more database tables
  - SELECT * FROM table will return all columns from a table
  - SELECT column_1, column_2 FROM table will return selected columns from a table

SQL Basics, continued

- JOINS - INNER, LEFT, RIGHT, OUTER
  - Most commonly use INNER, LEFT
  - Use LEFT JOINs when you might have missing values
  - JOIN TABLE_02 on TABLE_01.PID = TABLE_02.FID
- WHERE, HAVING, AND, OR, LIKE
  - Filtering, applying conditions
- GROUP BY
  - Used with COUNT, SUM, AVG functions
- ORDER BY
  - Define order of results

SQL Basics, JOINs illustrated



SQL Basics, continued

- Other Functions - CAST, CONCAT, etc.
  - CONCAT good for forming URIs to act upon later
  - CAST good for converting BLOBs into text - i.e. notes table
- UNION, UNION ALL
- More at: https://dev.mysql.com/doc/refman/8.0/en/sql-syntax.html

The ArchivesSpace Database

- Documentation
  - Data dictionary - https://desolate-tundra-60608.herokuapp.com/
  - Entity-relationship diagram for 2.1.0 (requires MySQL Workbench): https://bit.ly/2uVdL9J
  - db_info.sql
- Examine tables and relationships in SQL client
  - Sequel Pro ‘structure’ and ‘relations’ tabs

The ArchivesSpace Database - Entity Types/Tables

- Records
  - resources, archival objects, accessions, digital objects, collection management, classifications, locations, top containers, events etc.
- Subrecords
  - extents, instances, file versions, etc.
- Controlled value terms
  - enumerations, enumeration values, terms
- Relationship tables
  - contain foreign keys; link tables to one another
- Other tables
  - auth_db, active_edit

DEMO

All queries from this demo - and more! - are available on Github. Click on the “clone or download” button to download them all!

https://github.com/ucancallmealicia/mysql_demo

What’s the use?

- Advocacy, demonstrating value
    - Visualizations
- Collections management, physical control
- Data auditing
    - Manipulating query outputs
    - Using query output as API input

Thanks! Questions?

Alicia Detelich
Archivist
Manuscripts and Archives
Yale University Libraries
alicia.detelich@yale.edu
GH: ucancallmealicia