INST 733 Fall 2014 Final Project
Team: *Kathryn Miller*
Project: *Archive Accessions*
Date*: November 25, 2014*

A. Business Case

1. Overview

This project proposes to create a database for archives to track their accessions, i.e., all new archival resources received—similar to donations received by a non-profit. Types of resources include: paper records, artifacts, rare books, and digital records. Paper records and digital records are unique resources, while artifacts and rare books may have duplicates and are kept for preservation or posterity purposes. An accession can include one or more resources, and the resources in an accession can be a variety of types. Accessions can also include resources with different agreement terms, i.e., some resources in the accession may have public access or copyright restrictions while others may not. For example, Donor A donates everything in their grandfather's office to an archive at one time; this includes his entire filing cabinet of paper records, an 18th century writing desk, and a collection of first edition Lewis Carroll novels. This would be treated as one accession, with one agreement and overall value, but three different resources. As an alternative example, Donor A also includes their grandmother's filing cabinet of paper records in this same accession; the archive might choose to note the grandmother's paper records as a separate, fourth resource. The importance of the record creator is vital in an archive, and resource collections and groups are usually divided along creator lines. Therefore, a database is needed to track accessions, locations, and processing; the different types of resources in those accessions; and details about the individual resources and their creators. The database will only include the accession information and initial details about the resources so archives can more adequately manage inventories without having to wait for resources to be processed (i.e., arranged, described, labelled, and displayed for public use) by an archivist.

2. Client Need

Archives struggle with processing accessions in a timely manner, due to a lack of funds or personnel. If accessions are not properly tracked and given priority for processing, they will become lost on shelves, only to be found again when an archive has the resources and time to do a collection analysis (which can take more than a year to complete). Accessions are also complicated pieces, as an accession can contain numerous types of resources with complicated

restriction terms. How an archive divides the same types of resources varies from accession to accession, from archive to archive, so a database with that flexibility in mind is needed. Additionally, archives are in constant need of “big” accessions which will draw public attention in order to appease shareholders or administration. Therefore, archivists require a comprehensive list of donors, sellers, and transfer institutions to keep track of accession sources.

3. How Database will help

The database structure will meet the needs of an archive when it comes to managing accessions. Since there are so many facets to accessions, a complex design is needed. The database will (1) allow archivists to input details of the accession and the source; (2) access information about accession sources, even if the accession is deleted from the system; (3) input preliminary information about the resources in accessions; (4) access information about resource creators, even if the resource is deleted from the system; and (5) assign a processing status and assignment to each resource.

This structure will allow archivists to manage the location of accessions so they don’t become lost on the shelves, prioritize processing so the general public can access the resources more quickly, choose what constitutes as a resource, and build a comprehensive accession source database. While this database is only useful for managing accessions and processing priority, the unique identifiers assigned to each accession and resource can be linked to the final publicly accessible processing metadata (i.e., arrangement and description), creating a link between the internal and external catalogues.

4. Conceptual Model and Categories

The figure below is a high-level representation of the database organization. Brief descriptions of the categories is also included in this section.

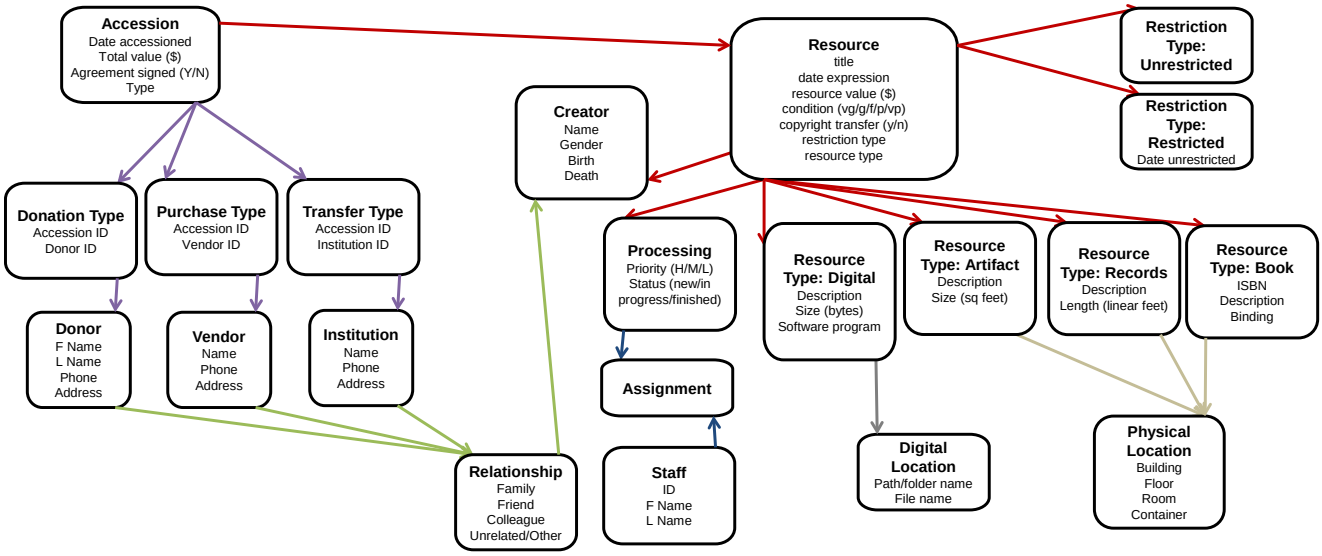


Figure 1 – High-Level Conceptual Model

Accession: A new addition to an archive's inventory, this can come through a donation from a donor, purchase from a vendor, or transfer from a collaborative institution. Data about accessions will include the date accessioned and if there was an agreement signed between the archive and the accession source.

Accession Source: Includes information about the person or organization that provided the new accession. Data about people will include first name, last name, phone number, and dress; data about organizations will include name, phone number, address, and tax/DUNS ID.

Creator: Biographical information about the creator of resources and their relationship with the accession source. Data about creators includes name, birth/death dates, birth/death places, and area of profession or expertise.

Donation: A type of accession in which permanent resources are received at no cost to the archive. Data about donations will include the overall value, if a donation receipt was given, and will be connected with the donor accession source.

Location: Either physical or digital location of resources. Data about locations includes exact location of where resource is being stored with regards to physical building, floor, room, and container or digital file path.

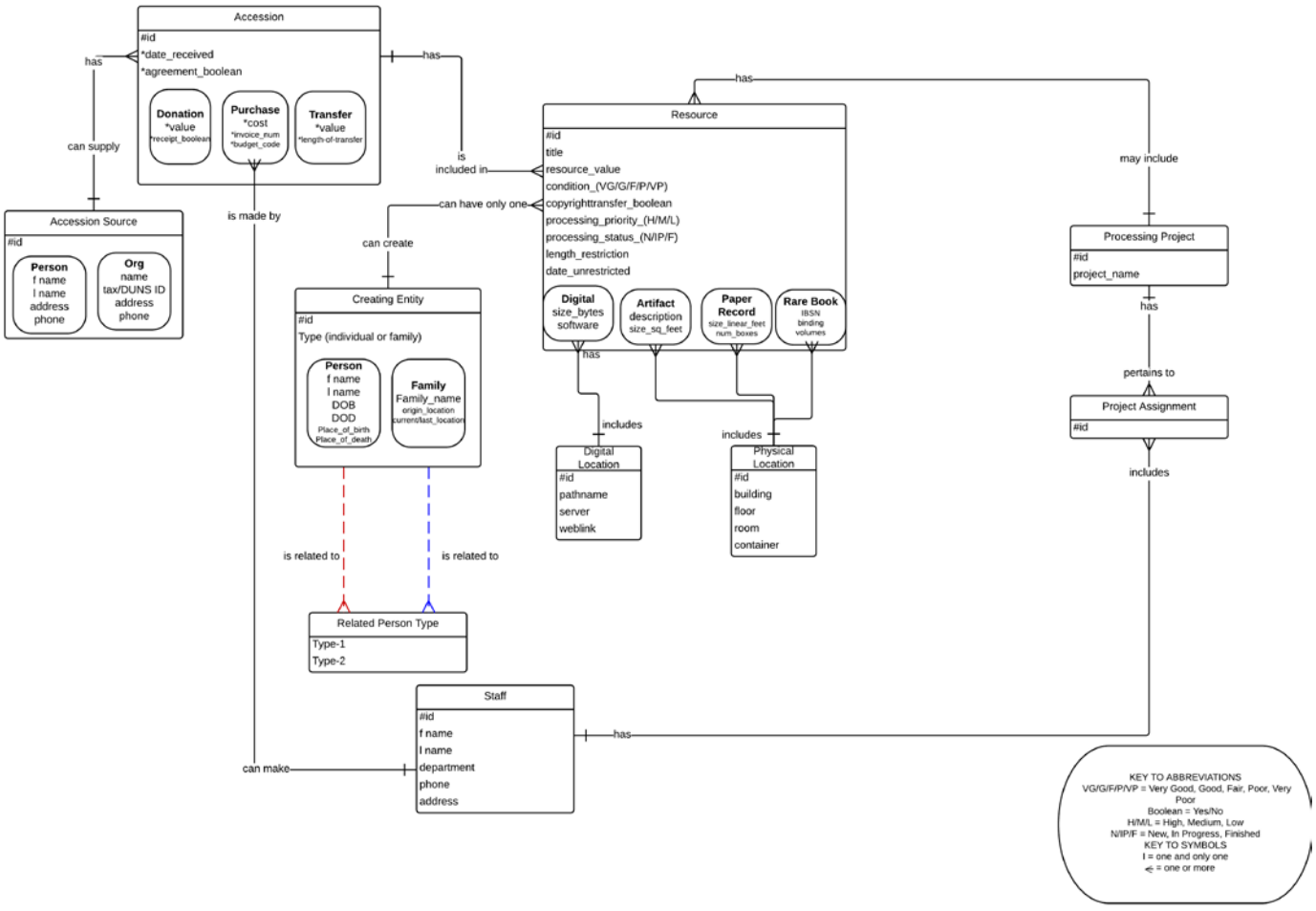
Processing: The status, priority, and assignment linked to accessioned resources. Data about processing includes a ranking of priority; status of new, in progress, or finished projects; and staff assignment.

Purchase: A type of accession in which an archive staff member purchases resources from a vendor at cost. Data about purchases will include the cost, invoice number, the internal budget code attached to the purchase, and the staff member purchaser.

Resource: The material items included in accessions, the archive has the flexibility to choose what constitutes a "resource," but resources can only have one type – digital, paper, book, or artifact. Data about the resource will include value, title, size, and copyright/access restrictions.

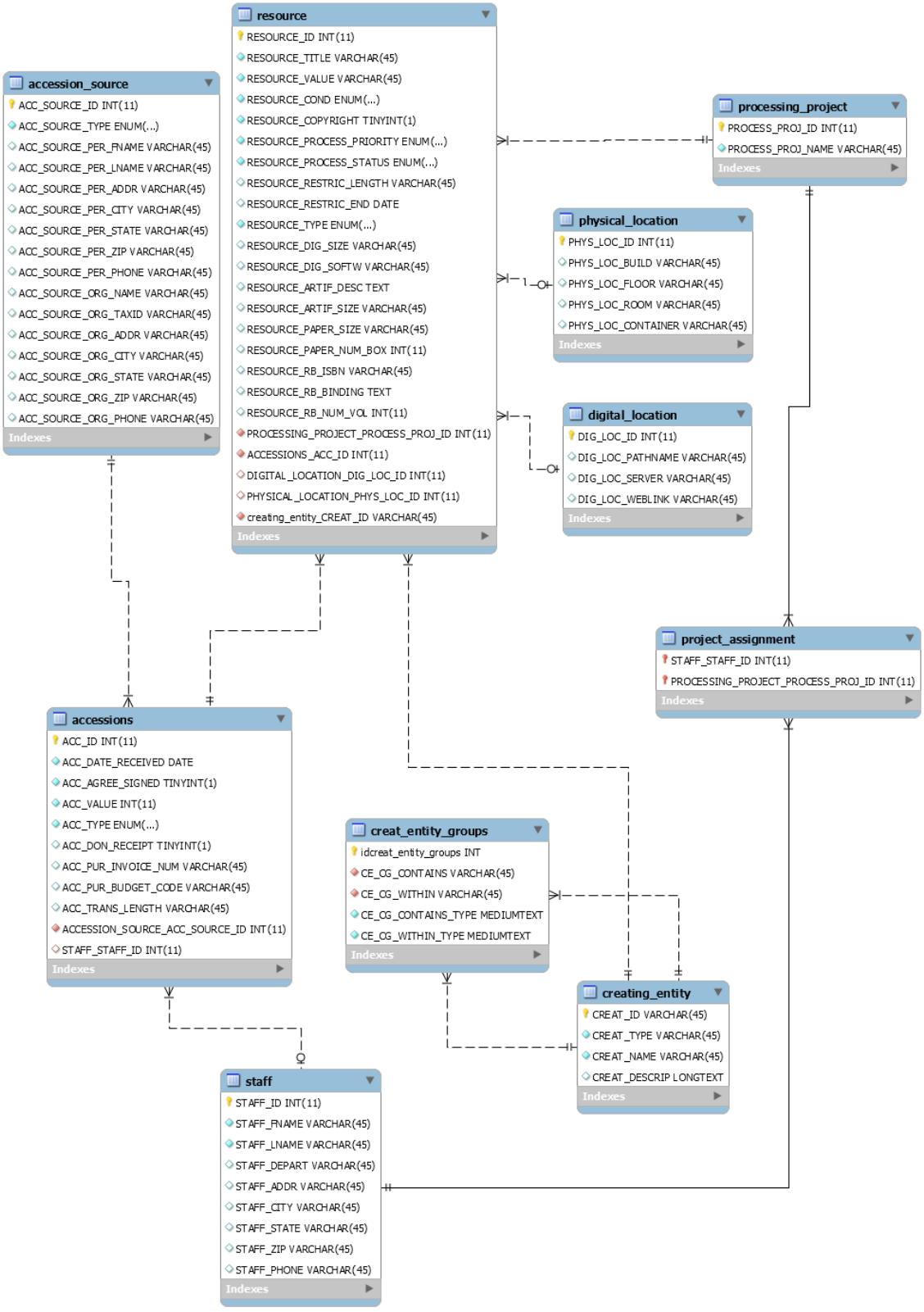
Transfer: A type of accession in which a collaborative institution transfers resources to the archive for a fixed or indeterminate amount of time. This is different from a donation because the resources could return to the collaborative institution at a future date and/or the institution could still retain copyright of the transferred resources. Data about transfers will include length of transfer and value. The copyright restrictions will be linked to individual resources.

B. Logical Design



C. Physical Design

1. Physical design diagram



2. Physical table build DDL

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
-- -----------------------------------------------------
-- Schema archives
-- -----------------------------------------------------
DROP SCHEMA IF EXISTS `archives` ;

-- -----------------------------------------------------
-- Schema archives
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `archives` DEFAULT CHARACTER SET utf8 ;
USE `archives` ;

-- -----------------------------------------------------
-- Table `archives`.`accession_source`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `archives`.`accession_source` ;

CREATE TABLE IF NOT EXISTS `archives`.`accession_source` (
  `ACC_SOURCE_ID` INT(11) NOT NULL,
  `ACC_SOURCE_TYPE` ENUM('person','organization') NOT NULL,
  `ACC_SOURCE_PER_FNAME` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_SOURCE_PER_LNAME` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_SOURCE_PER_ADDR` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_SOURCE_PER_CITY` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_SOURCE_PER_STATE` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_SOURCE_PER_ZIP` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_SOURCE_PER_PHONE` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_SOURCE_ORG_NAME` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_SOURCE_ORG_TAXID` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_SOURCE_ORG_ADDR` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_SOURCE_ORG_CITY` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_SOURCE_ORG_STATE` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_SOURCE_ORG_ZIP` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_SOURCE_ORG_PHONE` VARCHAR(45) NULL DEFAULT NULL,
  PRIMARY KEY (`ACC_SOURCE_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


-- -----------------------------------------------------
-- Table `archives`.`staff`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `archives`.`staff` ;

CREATE TABLE IF NOT EXISTS `archives`.`staff` (
  `STAFF_ID` INT(11) NOT NULL,
  `STAFF_FNAME` VARCHAR(45) NOT NULL,
  `STAFF_LNAME` VARCHAR(45) NOT NULL,
  `STAFF_DEPART` VARCHAR(45) NULL DEFAULT NULL,
  `STAFF_ADDR` VARCHAR(45) NULL DEFAULT NULL,
  `STAFF_CITY` VARCHAR(45) NULL DEFAULT NULL,
  `STAFF_STATE` VARCHAR(45) NULL DEFAULT NULL,
  `STAFF_ZIP` VARCHAR(45) NULL DEFAULT NULL,
  `STAFF_PHONE` VARCHAR(45) NULL DEFAULT NULL,
```

```
  PRIMARY KEY (`STAFF_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


-- -----------------------------------------------------
-- Table `archives`.`accessions`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `archives`.`accessions` ;

CREATE TABLE IF NOT EXISTS `archives`.`accessions` (
  `ACC_ID` INT(11) NOT NULL,
  `ACC_DATE_RECEIVED` DATE NOT NULL,
  `ACC_AGREE_SIGNED` TINYINT(1) NOT NULL,
  `ACC_VALUE` INT(11) NOT NULL,
  `ACC_TYPE` ENUM('donation','purchase','transfer') NOT NULL,
  `ACC_DON_RECEIPT` TINYINT(1) NULL DEFAULT NULL,
  `ACC_PUR_INVOICE_NUM` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_PUR_BUDGET_CODE` VARCHAR(45) NULL DEFAULT NULL,
  `ACC_TRANS_LENGTH` VARCHAR(45) NULL DEFAULT NULL,
  `ACCESSION_SOURCE_ACC_SOURCE_ID` INT(11) NOT NULL,
  `STAFF_STAFF_ID` INT(11) NULL DEFAULT NULL,
  PRIMARY KEY (`ACC_ID`),
  INDEX `fk_ACCESSIONS_ACCESSION_SOURCE_idx`
(`ACCESSION_SOURCE_ACC_SOURCE_ID` ASC),
  INDEX `fk_ACCESSIONS_STAFF1_idx` (`STAFF_STAFF_ID` ASC),
  CONSTRAINT `fk_ACCESSIONS_STAFF1`
    FOREIGN KEY (`STAFF_STAFF_ID`)
    REFERENCES `archives`.`staff` (`STAFF_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_ACCESSIONS_ACCESSION_SOURCE`
    FOREIGN KEY (`ACCESSION_SOURCE_ACC_SOURCE_ID`)
    REFERENCES `archives`.`accession_source` (`ACC_SOURCE_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


-- -----------------------------------------------------
-- Table `archives`.`creating_entity`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `archives`.`creating_entity` ;

CREATE TABLE IF NOT EXISTS `archives`.`creating_entity` (
  `CREAT_ID` VARCHAR(45) NOT NULL,
  `CREAT_TYPE` VARCHAR(45) NOT NULL,
  `CREAT_NAME` VARCHAR(45) NOT NULL,
  `CREAT_DESCRIP` LONGTEXT NULL DEFAULT NULL,
  PRIMARY KEY (`CREAT_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


-- -----------------------------------------------------
-- Table `archives`.`creat_entity_groups`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `archives`.`creat_entity_groups` ;

CREATE TABLE IF NOT EXISTS `archives`.`creat_entity_groups` (
  `idcreat_entity_groups` INT(11) NOT NULL,
  `CE_CG_CONTAINS` VARCHAR(45) NOT NULL,
  `CE_CG_WITHIN` VARCHAR(45) NOT NULL,
```

```
  `CE_CG_CONTAINS_TYPE` MEDIUMTEXT NOT NULL,
  `CE_CG_WITHIN_TYPE` MEDIUMTEXT NOT NULL,
  PRIMARY KEY (`idcreat_entity_groups`),
  INDEX `fk_creat_entity_groups_creating_entity1_idx` (`CE_CG_CONTAINS` ASC),
  INDEX `fk_creat_entity_groups_creating_entity2_idx` (`CE_CG_WITHIN` ASC),
  CONSTRAINT `fk_creat_entity_groups_creating_entity1`
    FOREIGN KEY (`CE_CG_CONTAINS`)
    REFERENCES `archives`.`creating_entity` (`CREAT_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_creat_entity_groups_creating_entity2`
    FOREIGN KEY (`CE_CG_WITHIN`)
    REFERENCES `archives`.`creating_entity` (`CREAT_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


-- -----------------------------------------------------
-- Table `archives`.`digital_location`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `archives`.`digital_location` ;

CREATE TABLE IF NOT EXISTS `archives`.`digital_location` (
  `DIG_LOC_ID` INT(11) NOT NULL,
  `DIG_LOC_PATHNAME` VARCHAR(45) NULL DEFAULT NULL,
  `DIG_LOC_SERVER` VARCHAR(45) NULL DEFAULT NULL,
  `DIG_LOC_WEBLINK` VARCHAR(45) NULL DEFAULT NULL,
  PRIMARY KEY (`DIG_LOC_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


-- -----------------------------------------------------
-- Table `archives`.`physical_location`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `archives`.`physical_location` ;

CREATE TABLE IF NOT EXISTS `archives`.`physical_location` (
  `PHYS_LOC_ID` INT(11) NOT NULL,
  `PHYS_LOC_BUILD` VARCHAR(45) NULL DEFAULT NULL,
  `PHYS_LOC_FLOOR` VARCHAR(45) NULL DEFAULT NULL,
  `PHYS_LOC_ROOM` VARCHAR(45) NULL DEFAULT NULL,
  `PHYS_LOC_CONTAINER` VARCHAR(45) NULL DEFAULT NULL,
  PRIMARY KEY (`PHYS_LOC_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


-- -----------------------------------------------------
-- Table `archives`.`processing_project`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `archives`.`processing_project` ;

CREATE TABLE IF NOT EXISTS `archives`.`processing_project` (
  `PROCESS_PROJ_ID` INT(11) NOT NULL,
  `PROCESS_PROJ_NAME` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`PROCESS_PROJ_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;
```

```
-- -----------------------------------------------------
-- Table `archives`.`project_assignment`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `archives`.`project_assignment` ;

CREATE TABLE IF NOT EXISTS `archives`.`project_assignment` (
  `STAFF_STAFF_ID` INT(11) NOT NULL,
  `PROCESSING_PROJECT_PROCESS_PROJ_ID` INT(11) NOT NULL,
  PRIMARY KEY (`STAFF_STAFF_ID`, `PROCESSING_PROJECT_PROCESS_PROJ_ID`),
  INDEX `fk_PROJECT_ASSIGNMENT_PROCESSING_PROJECT1_idx`
(`PROCESSING_PROJECT_PROCESS_PROJ_ID` ASC),
  CONSTRAINT `fk_PROJECT_ASSIGNMENT_STAFF1`
    FOREIGN KEY (`STAFF_STAFF_ID`)
    REFERENCES `archives`.`staff` (`STAFF_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_PROJECT_ASSIGNMENT_PROCESSING_PROJECT1`
    FOREIGN KEY (`PROCESSING_PROJECT_PROCESS_PROJ_ID`)
    REFERENCES `archives`.`processing_project` (`PROCESS_PROJ_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


-- -----------------------------------------------------
-- Table `archives`.`resource`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `archives`.`resource` ;

CREATE TABLE IF NOT EXISTS `archives`.`resource` (
  `RESOURCE_ID` INT(11) NOT NULL,
  `RESOURCE_TITLE` VARCHAR(45) NOT NULL,
  `RESOURCE_VALUE` VARCHAR(45) NOT NULL,
  `RESOURCE_COND` ENUM('VG','G','F','P','VP') NOT NULL,
  `RESOURCE_COPYRIGHT` TINYINT(1) NOT NULL,
  `RESOURCE_PROCESS_PRIORITY` ENUM('high','medium','low') NOT NULL,
  `RESOURCE_PROCESS_STATUS` ENUM('new','in progress','finished') NOT NULL,
  `RESOURCE_RESTRIC_LENGTH` VARCHAR(45) NULL DEFAULT NULL,
  `RESOURCE_RESTRIC_END` DATE NULL DEFAULT NULL,
  `RESOURCE_TYPE` ENUM('digital','artifact','paper','rare book') NOT NULL,
  `RESOURCE_DIG_SIZE` VARCHAR(45) NULL DEFAULT NULL,
  `RESOURCE_DIG_SOFTW` VARCHAR(45) NULL DEFAULT NULL,
  `RESOURCE_ARTIF_DESC` TEXT NULL DEFAULT NULL,
  `RESOURCE_ARTIF_SIZE` VARCHAR(45) NULL DEFAULT NULL,
  `RESOURCE_PAPER_SIZE` VARCHAR(45) NULL DEFAULT NULL,
  `RESOURCE_PAPER_NUM_BOX` INT(11) NULL DEFAULT NULL,
  `RESOURCE_RB_ISBN` VARCHAR(45) NULL DEFAULT NULL,
  `RESOURCE_RB_BINDING` TEXT NULL DEFAULT NULL,
  `RESOURCE_RB_NUM_VOL` INT(11) NULL DEFAULT NULL,
  `PROCESSING_PROJECT_PROCESS_PROJ_ID` INT(11) NOT NULL,
  `ACCESSIONS_ACC_ID` INT(11) NOT NULL,
  `DIGITAL_LOCATION_DIG_LOC_ID` INT(11) NULL DEFAULT NULL,
  `PHYSICAL_LOCATION_PHYS_LOC_ID` INT(11) NULL DEFAULT NULL,
  `creating_entity_CREAT_ID` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`RESOURCE_ID`),
  INDEX `fk_RESOURCE_PROCESSING_PROJECT1_idx`
(`PROCESSING_PROJECT_PROCESS_PROJ_ID` ASC),
  INDEX `fk_RESOURCE_ACCESSIONS1_idx` (`ACCESSIONS_ACC_ID` ASC),
  INDEX `fk_RESOURCE_DIGITAL_LOCATION1_idx` (`DIGITAL_LOCATION_DIG_LOC_ID`
ASC),
  INDEX `fk_RESOURCE_PHYSICAL_LOCATION1_idx` (`PHYSICAL_LOCATION_PHYS_LOC_ID`
ASC),
```

```
  INDEX `fk_resource_creating_entity1_idx` (`creating_entity_CREAT_ID` ASC),
  CONSTRAINT `fk_RESOURCE_ACCESSIONS1`
    FOREIGN KEY (`ACCESSIONS_ACC_ID`)
    REFERENCES `archives`.`accessions` (`ACC_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_resource_creating_entity1`
    FOREIGN KEY (`creating_entity_CREAT_ID`)
    REFERENCES `archives`.`creating_entity` (`CREAT_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_RESOURCE_DIGITAL_LOCATION1`
    FOREIGN KEY (`DIGITAL_LOCATION_DIG_LOC_ID`)
    REFERENCES `archives`.`digital_location` (`DIG_LOC_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_RESOURCE_PHYSICAL_LOCATION1`
    FOREIGN KEY (`PHYSICAL_LOCATION_PHYS_LOC_ID`)
    REFERENCES `archives`.`physical_location` (`PHYS_LOC_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_RESOURCE_PROCESSING_PROJECT1`
    FOREIGN KEY (`PROCESSING_PROJECT_PROCESS_PROJ_ID`)
    REFERENCES `archives`.`processing_project` (`PROCESS_PROJ_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;
SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

3. View DDL

```
-- -----------------------------------------------------
-- -----------------------------------------------------
-- View `archives`.`accession_source_names`
-- -----------------------------------------------------
DROP VIEW IF EXISTS `archives`.`accession_source_names` ;
DROP TABLE IF EXISTS ``archives`.`accession_source_names`;
USE `archives`;
CREATE  OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `archives`.`accession_source_names` AS select
`SourceName`(`archives`.`accession_source`.`ACC_SOURCE_PER_FNAME`,`archives`.
`accession_source`.`ACC_SOURCE_PER_LNAME`) AS
`Source_Name`,`archives`.`accession_source`.`ACC_SOURCE_TYPE` AS
`Source_Type` from `archives`.`accession_source` where
((`archives`.`accession_source`.`ACC_SOURCE_PER_FNAME` is not null) and
(`archives`.`accession_source`.`ACC_SOURCE_PER_LNAME` is not null)) union
select `archives`.`accession_source`.`ACC_SOURCE_ORG_NAME` AS
`Source_Name`,`archives`.`accession_source`.`ACC_SOURCE_TYPE` AS
`Source_Type` from `archives`.`accession_source` where
(`archives`.`accession_source`.`ACC_SOURCE_ORG_NAME` is not null);


-- -----------------------------------------------------
-- View `archives`.`accession_types_org_source`
-- -----------------------------------------------------
DROP VIEW IF EXISTS `archives`.`accession_types_org_source` ;
DROP TABLE IF EXISTS `archives`.`accession_types_org_source`;
USE `archives`;
CREATE  OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `archives`.`accession_types_org_source` AS select
```

```
`archives`.`accession_source`.`ACC_SOURCE_ORG_NAME` AS
`Source_Name`,`archives`.`accession_source`.`ACC_SOURCE_ORG_PHONE` AS
`Source_Phone`,`archives`.`accessions`.`ACC_TYPE` AS `Type_of_Accession` from
(`archives`.`accession_source` join `archives`.`accessions`) where
((`archives`.`accession_source`.`ACC_SOURCE_ID` =
`archives`.`accessions`.`ACCESSION_SOURCE_ACC_SOURCE_ID`) and
(`archives`.`accession_source`.`ACC_SOURCE_TYPE` = 'organization')) order by
`archives`.`accessions`.`ACC_TYPE`;


-- ----------------------------------------------------
-- View `archives`.`accession_types_person_source`
-- ----------------------------------------------------
DROP VIEW IF EXISTS `archives`.`accession_types_person_source` ;
DROP TABLE IF EXISTS `archives`.`accession_types_person_source`;
USE `archives`;
CREATE  OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `archives`.`accession_types_person_source` AS select
`archives`.`accession_source`.`ACC_SOURCE_PER_FNAME` AS
`Source_First_Name`,`archives`.`accession_source`.`ACC_SOURCE_PER_LNAME` AS
`Source_Last_Name`,`archives`.`accession_source`.`ACC_SOURCE_PER_PHONE` AS
`Source_Phone`,`archives`.`accessions`.`ACC_TYPE` AS `Type_of_Accession` from
(`archives`.`accession_source` join `archives`.`accessions`) where
((`archives`.`accession_source`.`ACC_SOURCE_ID` =
`archives`.`accessions`.`ACCESSION_SOURCE_ACC_SOURCE_ID`) and
(`archives`.`accession_source`.`ACC_SOURCE_TYPE` = 'person')) order by
`archives`.`accessions`.`ACC_TYPE`;


-- ----------------------------------------------------
-- View `archives`.`albert_moss_relations`
-- ----------------------------------------------------
DROP VIEW IF EXISTS `archives`.`albert_moss_relations` ;
DROP TABLE IF EXISTS `archives`.`albert_moss_relations`;
USE `archives`;
CREATE  OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `archives`.`albert_moss_relations` AS select
`archives`.`creating_entity`.`CREAT_NAME` AS
`Family_Member`,`archives`.`creat_entity_groups`.`CE_CG_WITHIN_TYPE` AS
`Relation_to_Albert_Moss` from (`archives`.`creat_entity_groups` join
`archives`.`creating_entity`
on((`archives`.`creat_entity_groups`.`CE_CG_WITHIN` =
`archives`.`creating_entity`.`CREAT_ID`))) where
(`archives`.`creat_entity_groups`.`CE_CG_CONTAINS` = 'AlMoss') union select
`archives`.`creating_entity`.`CREAT_NAME` AS `Family
Member`,`archives`.`creat_entity_groups`.`CE_CG_CONTAINS_TYPE` AS `Relation
to Albert Moss` from (`archives`.`creat_entity_groups` join
`archives`.`creating_entity`
on((`archives`.`creat_entity_groups`.`CE_CG_CONTAINS` =
`archives`.`creating_entity`.`CREAT_ID`))) where
(`archives`.`creat_entity_groups`.`CE_CG_WITHIN` = 'AlMoss');


-- ----------------------------------------------------
-- View `archives`.`moss_family_members`
-- ----------------------------------------------------
DROP VIEW IF EXISTS `archives`.`moss_family_members` ;
DROP TABLE IF EXISTS `archives`.`moss_family_members`;
USE `archives`;
CREATE  OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `archives`.`moss_family_members` AS select
`archives`.`creating_entity`.`CREAT_NAME` AS
`Members_of_Moss_Family`,`archives`.`creating_entity`.`CREAT_DESCRIP` AS
`Biography` from (`archives`.`creat_entity_groups` left join
```

```
`archives`.`creating_entity`
on((`archives`.`creat_entity_groups`.`CE_CG_WITHIN` =
`archives`.`creating_entity`.`CREAT_ID`))) where
(`archives`.`creat_entity_groups`.`CE_CG_CONTAINS` = 'Moss');


-- ----------------------------------------------------
-- View `archives`.`projects_with_more_than_one_resource`
-- ----------------------------------------------------
DROP VIEW IF EXISTS `archives`.`projects_with_more_than_one_resource` ;
DROP TABLE IF EXISTS `archives`.`projects_with_more_than_one_resource`;
USE `archives`;
CREATE  OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `archives`.`projects_with_more_than_one_resource` AS
select `resources_in_projects`.`Project_Name` AS
`Processing_Project_Name`,count(`resources_in_projects`.`Project_Name`) AS
`Number_of_Resources` from `archives`.`resources_in_projects` group by
`resources_in_projects`.`Project_Name` having
(count(`resources_in_projects`.`Project_Name`) > 1) order by
`Number_of_Resources` desc;


-- ----------------------------------------------------
-- View `archives`.`res_acc_values`
-- ----------------------------------------------------
DROP VIEW IF EXISTS `archives`.`res_acc_values` ;
DROP TABLE IF EXISTS `archives`.`res_acc_values`;
USE `archives`;
CREATE  OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `archives`.`res_acc_values` AS select
sum(`archives`.`resource`.`RESOURCE_VALUE`) AS `Total Resource
Value`,sum(`archives`.`accessions`.`ACC_VALUE`) AS `Total Accession Value`
from (`archives`.`resource` join `archives`.`accessions`);


-- ----------------------------------------------------
-- View `archives`.`resources_in_lerner_building`
-- ----------------------------------------------------
DROP VIEW IF EXISTS `archives`.`resources_in_lerner_building` ;
DROP TABLE IF EXISTS `archives`.`resources_in_lerner_building`;
USE `archives`;
CREATE  OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `archives`.`resources_in_lerner_building` AS select
`archives`.`resource`.`RESOURCE_TITLE` AS `Resources` from
`archives`.`resource` where
`archives`.`resource`.`PHYSICAL_LOCATION_PHYS_LOC_ID` in (select
`archives`.`physical_location`.`PHYS_LOC_ID` from
`archives`.`physical_location` where
(`archives`.`physical_location`.`PHYS_LOC_BUILD` = 'Lerner'));


-- ----------------------------------------------------
-- View `archives`.`resources_in_projects`
-- ----------------------------------------------------
DROP VIEW IF EXISTS `archives`.`resources_in_projects` ;
DROP TABLE IF EXISTS `archives`.`resources_in_projects`;
USE `archives`;
CREATE  OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `archives`.`resources_in_projects` AS select
`archives`.`processing_project`.`PROCESS_PROJ_NAME` AS
`Project_Name`,`archives`.`resource`.`RESOURCE_TITLE` AS `Resource_Title`
from (`archives`.`processing_project` join `archives`.`resource`) where
(`archives`.`processing_project`.`PROCESS_PROJ_ID` =
```

```
`archives`.`resource`.`PROCESSING_PROJECT_PROCESS_PROJ_ID`) order by
`archives`.`resource`.`RESOURCE_TITLE`;
-- ------------------------------------------------------
-- View `archives`.`staff_projects`
-- ------------------------------------------------------
DROP VIEW IF EXISTS `archives`.`staff_projects` ;
DROP TABLE IF EXISTS `archives`.`staff_projects`;
USE `archives`;
CREATE  OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `archives`.`staff_projects` AS select
`archives`.`staff`.`STAFF_ID` AS `Staff_ID`,concat_ws('
',`archives`.`staff`.`STAFF_FNAME`,`archives`.`staff`.`STAFF_LNAME`) AS
`Staff_Name`,`archives`.`processing_project`.`PROCESS_PROJ_NAME` AS
`Processing_Project_Name` from ((`archives`.`processing_project` join
`archives`.`project_assignment`) join `archives`.`staff`) where
((`archives`.`project_assignment`.`STAFF_STAFF_ID` =
`archives`.`staff`.`STAFF_ID`) and
(`archives`.`project_assignment`.`PROCESSING_PROJECT_PROCESS_PROJ_ID` =
`archives`.`processing_project`.`PROCESS_PROJ_ID`)) order by `Staff_Name`;
```

4. DDL for server logic

a. Function: Source Name

```
CREATE definer=`root`@`localhost`
  function `sourcename`(first_name text,
                        second_name text) returns text charset utf8
    READS SQL data
    DETERMINISTIC
  begin
    DECLARE fullname text;IF second_name IS NULL THEN
      SET fullname = first_name;END IF;SET fullname = concat(second_name,','
,first_name);RETURN fullname;END
```

D. Sample Data

1. Accession_Source

access-source.csv

2. Accessions

access.csv

3. Creat_Entity_Groups (Bill of Materials Assembly)

creat-entity-groups-final.csv

4. Creating_Entity (Bill of Materials Products)

creat-entity.csv

5. Digital_Location

digi-loc.csv

6. Physical_Location

phys-loc.csv

7. Processing_Project

process-project.csv

8. Project_Assignment

project-assign.csv

9. Resource

resource.csv

10. Staff

staff.csv

E. Sample Queries.

1. Query 1: View `archives`.`accession_source_names`

```
SELECT `SourceName`(`archives`.`accession_source`.`ACC_SOURCE_PER_FNAME`,
`archives`.`accession_source`.`ACC_SOURCE_PER_LNAME`) AS `Source_Name`,
       `archives`.`accession_source`.`ACC_SOURCE_TYPE` AS `Source_Type`
FROM `archives`.`accession_source`
WHERE ((`archives`.`accession_source`.`ACC_SOURCE_PER_FNAME` IS NOT NULL)
      AND (`archives`.`accession_source`.`ACC_SOURCE_PER_LNAME` IS NOT NULL))
UNION
SELECT `archives`.`accession_source`.`ACC_SOURCE_ORG_NAME` AS `Source_Name`,
       `archives`.`accession_source`.`ACC_SOURCE_TYPE` AS `Source_Type`
FROM `archives`.`accession_source`
WHERE (`archives`.`accession_source`.`ACC_SOURCE_ORG_NAME` IS NOT NULL);
```

Query 1 Results:

Source_Name	Source_Type
Wunderlich,Maggie	person
Page,Tim	person
Sonet,Sarah	person
Gonzalez,Edward	person
Reyes,Dennis	person
Shaw,Alice	person
Sanchez,Henry	person
Gardner,Mildred	person
Stone,Douglas	person
Anvil, Inc.	organization
Respotio, LLC	organization
Acme, Inc.	organization
Circle K	organization
Youopia	organization
Kamba	organization
Zoombox	organization

2. Query 2: View `archives`.`accession_types_person_source`

```
SELECT Sourcename(`accession_source`.`acc_source_per_fname`,
`accession_source`.`acc_source_per_lname`) AS `source_name`,
       `accession_source`.`acc_source_per_phone` AS `source_phone`,
       `accessions`.`acc_type` AS `type_of_accession`
FROM (`accession_source`
      JOIN `accessions`)
WHERE ((`accession_source`.`acc_source_id` =
       `accessions`.`accession_source_acc_source_id`)
       AND (`accession_source`.`acc_source_type` = 'person'))
ORDER BY `accessions`.`acc_type`;
```

Query 2 Results:

Type_of_Accession	Type_of_Accession	Type_of_Accession
Sonet,Sarah	301-568-4615	donation
Sanchez,Henry	286-617-7418	donation
Page,Tim	465-235-6418	donation
Wunderlich,Maggie	254-897-5621	purchase
Gonzalez,Edward	559-973-3996	purchase
Stone,Douglas	176-330-9993	purchase
Shaw,Alice	223-445-3651	transfer
Gardner,Mildred	167-947-5995	transfer

3. Query 3: View `archives`.`albert_moss_relations`

```
SELECT `creating_entity`.`creat_name` AS `Family_Member`,
       `creat_entity_groups`.`ce_cg_within_type` AS `Relation_to_Albert_Moss`
FROM (`creat_entity_groups`
      JOIN `creating_entity` ON ((`creat_entity_groups`.`ce_cg_within` =
       `creating_entity`.`creat_id`)))
WHERE (`creat_entity_groups`.`ce_cg_contains` = 'AlMoss')
UNION
SELECT `creating_entity`.`creat_name` AS `Family Member`,
       `creat_entity_groups`.`ce_cg_contains_type` AS `Relation to Albert Moss`
FROM (`creat_entity_groups`
      JOIN `creating_entity` ON ((`creat_entity_groups`.`ce_cg_contains` =
       `creating_entity`.`creat_id`)))
WHERE (`creat_entity_groups`.`ce_cg_within` = 'AlMoss');
```

Query 3 Results:

Family_Member	Relation_to_Albert_Moss
Mike Moss	Son
Patricia Moss	Wife
Moss Family	Includes
Melanie Moss	Mother
Mark Moss	Father

4. Query 4: View `archives`.`moss_family_members`

```
SELECT `creating_entity`.`creat_name` AS `Members_of_Moss_Family`,
       `creating_entity`.`creat_descrip` AS `Biography`
FROM (`creat_entity_groups`
      LEFT JOIN `creating_entity` ON ((`creat_entity_groups`.`ce_cg_within` =
      `creating_entity`.`creat_id`)))
WHERE (`creat_entity_groups`.`ce_cg_contains` = 'Moss');
```

Query 4 Results:

Members_of_Moss_Family	Biography
Mike Moss	Born 8-13-1987.
Patricia Moss	Born 5-25-1947. Died 6-21-2016.
Albert Moss	Born 4-24-1944. Died 9-21-2012.
Melanie Moss	Born 1-31-1910. Died 8-17-1998.
Mark Moss	Born 6-30-1909. Died 4-21-1944.

5. Query 5: View `archives`.`projects_with_more_than_one_resource`

```
SELECT `resources_in_projects`.`project_name` AS `Processing_Project_Name`,
       Count(`resources_in_projects`.`project_name`) AS `Number_of_Resources`
FROM `resources_in_projects`
GROUP BY `resources_in_projects`.`project_name` HAVING
(Count(`resources_in_projects`.`project_name`) > 1)
ORDER BY `number_of_resources` DESC;
```

Query 5 Results:

Processing_Project_Name	Number_of_Resources
Shaw Family Artifacts	2
Patricia Moss Papers	2

6. Query 6: View `archives`.`res_acc_values`

```
SELECT sum(`archives`.`resource`.`RESOURCE_VALUE`) AS `Total Resource Value`,
       sum(`archives`.`accessions`.`ACC_VALUE`) AS `Total Accession Value`
FROM (`archives`.`resource`
      JOIN `archives`.`accessions`);
```

Query 6 Results:

Total Resource Value	Total Accession Value
569535	10862456

7. Query 7: View `archives`.`resources_in_lerner_building`

```
SELECT `resource`.`resource_title` AS `Resources`
FROM `resource`
WHERE `resource`.`physical_location_phys_loc_id` IN
  (SELECT`physical_location`.`phys_loc_id`
   FROM `physical_location`
   WHERE (physical_location`.`phys_loc_build` = 'Lerner' ))
GROUP  BY `resource`.`resource_title`;
```

Query 7 Results:

Resources
Patricia Moss Papers (1960-1980)
Patricia Moss Papers (1981-2014)
Albert Moss Papers
Melanie Moss Papers (1925-1955)
1856 Writing Desk
Alice in Wonderland First Editions
Economics Rare Books
Marline Dieter Papers
Sarah Smith Papers

8. Query 8: View `archives`.`resources_in_projects`

```
SELECT `processing_project`.`process_proj_name` AS `project_name`,
       `resource`.`resource_title` AS `resource_title`
FROM (`processing_project`
      JOIN `resource`)
WHERE (`processing_project`.`process_proj_id` =
       `resource`.`processing_project_process_proj_id`)
ORDER BY `resource`.`resource_title`;
```

Query 8 Results:

Project_Name	Resource_Title
Moss Family Artifacts	1856 Writing Desk
Albert Moss Papers	Albert Moss Papers
Moss Family Rare Book Collection	Alice in Wonderland First Editions
Shaw Family Artifacts	Antique Inkwell (date unknown)
Donna Shaw Papers	Donna Shaw Papers (1980-2000)
Coyote, Inc. Rare Book Collection	Economics Rare Books
John Shaw Papers	John Shaw Papers (1985-2005)
Marline Dieter Papers	Marline Dieter Papers
Mary Shaw Digital Files	Mary Shaw Digital Files
Melanie Moss Papers	Melanie Moss Papers (1925-1955)
Melissa Shaw Digital Files	Melissa Shaw Digital Files (1995-2012)
Mike Moss Digital Files	Mike Moss Digital Files (2000-2014)

Project_Name	Resource_Title
Shaw Family Artifacts	Ming Dynasty Vase
Patricia Moss Papers	Patricia Moss Papers (1960-1980)
Patricia Moss Papers	Patricia Moss Papers (1981-2014)
Sarah Smith Papers	Sarah Smith Papers
Melissa Shaw Rare Book Collection	Wizard of Oz First Editions

9. Query 9: View `archives`.`staff_projects`

```
SELECT Sourcename(`staff`.`staff_fname`, `staff`.`staff_lname`) AS `staff_name`,
       `processing_project`.`process_proj_name` AS `processing_project_name`
FROM ((`processing_project`
       JOIN `project_assignment`)
      JOIN `staff`)
WHERE ((`project_assignment`.`staff_staff_id` = `staff`.`staff_id`)
       AND (`project_assignment`.`processing_project_process_proj_id` =
            `processing_project`.`process_proj_id`))
ORDER BY `staff_name`;
```

Query 9 Results:

Staff_Name	Processing_Project_Name
Camp,Brian	Coyote, Inc. Rare Book Collection
Camp,Brian	Donna Shaw Papers
O'Brien,Peggy	Melanie Moss Papers
O'Brien,Peggy	Melissa Shaw Rare Book Collection
O'Brien,Peggy	Melissa Shaw Digital Files
Potter,Melanie	Patricia Moss Papers
Potter,Melanie	Sarah Smith Papers
Smith,Mark	Mike Moss Digital Files
Smith,Mark	Marline Dieter Papers
Thomas,Albert	Albert Moss Papers
Thomas,Albert	Shaw Family Artifacts
Thomas,Nucky	Moss Family Artifacts
Thomas,Nucky	Mary Shaw Digital Files
Willams,Will	Moss Family Rare Book Collection
Willams,Will	John Shaw Papers

PROJECT REQUIREMENTS

1. Include at least seven SELECT queries as views in your database.
    - DONE
2. These queries (views) should not make use of the wildcard character (*) unless it is unavoidable given the nature of the query. If that is the case, and you use the wildcard character (*), you need to include in your report an explanation about the need for that use.
    - DONE
3. At least six of your queries (saved as views) should involve multiple (two or more) tables.
    - Query 1
    - Query 2
    - Query 3
    - Query 4
    - Query 5
    - Query 6
    - Query 7
    - Query 8
    - Query 9
4. At least five of your queries should involve some form of filtering (WHERE, etc.) and/or sorting (ORDER BY, etc).
    - Query 1 - Where
    - Query 2 – Order By & Where
    - Query 3 – Where
    - Query 4 – Where
    - Query 5 – Order By & Where
    - Query 7 – Where
    - Query 8 – Order By & Where
    - Query 9 – Order By & Where
5. At least three of your queries should involve some form of aggregation over records (SUM, COUNT, AVERAGE, GROUP BY, etc.)
    - Query 5 – Group By & Count
    - Query 6 – Sum
    - Query 7 – Group By

You can pick the fields to use as long as they are reasonable from a business case.

6. At least two of your queries should involve a join table and both of its source tables.
   Note: the Bill of Material if you have one can serve as a join table.
    - Query 3 – Bill of Materials
    - Query 4 – Bill of Materials
    - Query 9 – Project Assignment
7. At least one of your queries should use a subquery. (The subquery used in this main query will count as one of the seven queries, if it is saved as a view, and satisfies one or more of the other requirements as listed about.
    - Query 7