**IS 6503 Term Project Deliverable 3**

Your Name: Bobb Shields

Your email: Robert.shields@my.utsa.edu

<https://github.com/xiebozhi/FA2024-IS6503-ArtDB>

1. **Description of the business context and related data management problem(s)**

The Nexus Art Collective is a group of artists who have built a suite of modern e-business tools, more than a e-sales platform but also a replacement for Etsy, CashApp, Quickbooks, TaxAct, payroll software, and project management tools. A part of this endeavor’s strategy is to use AI the right way, where artists will accept and adopt it. One of these ideas is to build an LLM that watches social media to analyze ongoing trends and build advertising copy on behalf of the artist members who sell through our platform.

The purpose of this data set is to train an AI to make it speak regarding art more fluently. We are considering this to be a curated set, full of high quality data that will help an AI understand how to talk about art that was popular enough to make it into the Met, the Modern Museum of Art, or the Carnegie Institute. The initial application for this project will be to reference the art in these museums and point patrons in the right direction, both providing URLs to online resources and helping the customer to narrow down where the art sits in real life. Furthermore, the data released by these various sources will be standardized to fit the most universal adaptation of a data dictionary as possible. This will enable data science operations on the combined set of art.

1. **The entities and the attributes (with description, constraint and data types)**

**Entity**: **Art work**(A piece of art and its metadata, the primary focus of the database)**:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Description:** the person who made the art and their available metadata. Could possibly have multiple art pieces in different museums. | | | |
| Variable Name | Definition | Data type | **Constraints** |
| artID | PK for Portfolio | int64 | PK |
| catalogue\_number | Unique identifier for the artwork | string | Index |
| City | City associated with the object | string |  |
| Classification | Classification of the object | string |  |
| Country | Country associated with the object | string |  |
| creation\_date | Date the artwork was created | string |  |
| Credit Line | Credit line for the object | string |  |
| Culture | Culture associated with the object | string |  |
| Department | Department responsible for the object | string |  |
| Dimensions | Dimensions of the object | string |  |
| Duration (s) | Duration of the artwork in seconds | string |  |
| Dynasty | Dynasty of the object | string |  |
| Excavation | Excavation site of the object | string |  |
| Geography Type | Type of geography associated with the object | string |  |
| image\_url | URL to the artwork’s image | string |  |
| Locale | Locale associated with the object | string |  |
| location | At which museum the piece sits | string |  |
| Locus | Locus associated with the object | string |  |
| Medium | Medium or materials used in the object | string |  |
| Object Begin Date | Beginning date of the object’s creation | string |  |
| Object Date | Date of the object | string |  |
| Object End Date | Ending date of the object’s creation | string |  |
| Object Name | Name of the object | string |  |
| Object Number | Unique object number | string |  |
| Period | Period of the object | string |  |
| Region | Region associated with the object | string |  |
| Reign | Reign during which the object was created | string |  |
| Repository | Repository of the object | string |  |
| Rights and Reproduction | Rights and reproduction information | string |  |
| River | River associated with the object | string |  |
| State | State associated with the object | string |  |
| Subregion | Subregion associated with the object | string |  |
| Title | Title of the object | string |  |
| Weight (kg) | Weight of the artwork in kilograms | string |  |
| web\_url | URL to the artwork’s webpage | string |  |
| source\_fk\_artID | ID for the object | string |  |
| Source\_identifyer | A description of this row’s source | string |  |

**Entity**: **Artist** (Describes a person or organization that produces Artworks)**:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Description:** the person who made the art and their available metadata. Could possibly have multiple art pieces in different museums. | | | |
| Variable Name | Definition | Data type | **Constraints** |
| artistID | PK for Portfolio | int64 | PK |
| Display Bio | Display biography of the artist | string |  |
| Display Name | Display name of the artist | string |  |
| Nationality | Nationality of the artist | string |  |
| Role | Role of the artist | string |  |
| Birth Date | Birth year of the artist | string |  |
| Death Date | Death year of the artist | string |  |
| Gender | Gender of the artist | string |  |
| Name | Name of the artist | string |  |
| Nationality | Nationality of the artist | string |  |
| Portfolio | Portfolio information | string |  |
| source\_fk\_ArtistID | Unique identifier for the artist, in the source | string | Not Null |
| Source |  |  |  |

**Entity**: **Portfolio** (A body of artworks):

|  |  |  |  |
| --- | --- | --- | --- |
| **Description:** A portfolio has many pieces of art. They may all be at one museum, in multiple, or exist outside of any museum. A portfolio typically has a theme and is decided by the originating artist, but also a portfolio could be an entire body of work given to a museum. | | | |
| Variable Name | Definition | Data type | **Constraints** |
| portfolioID | PK for Portfolio | int64 | PK |
| Title | Portfolio information | string | Null |
| Notes | Portfolio information | string | Null |

**Entity**: **Museum** (Associative Entity between Order and Product)**:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Description:** the person who made the art and their available metadata. Could possibly have multiple art pieces in different museums. | | | |
| Variable Name | Definition | Data type | **Constraints** |
| museumID | ID of the museum | Int | PK |
| City | City of the museum | string |  |
| State | State of the museum | string |  |
| Street | Street address of the museum | string |  |
| Zip | ZIP code of the museum | string |  |
| COMMONNAME | Common name of the museum | string |  |
| Country | Country where the museum is located | string |  |
| Discipline | Discipline of the museum | string |  |
| Gallery space in m2 (sq ft) | Gallery space in square meters | string |  |
| source\_fk\_MuseumID | Museum ID | int64 |  |
| Name | Name of the museum | string |  |
| PHONE | Phone number of the museum | string |  |
| WEBURL | Website URL of the museum | string |  |
| Year established | Year the museum was established | string |  |

1. **ERD and Relationships**

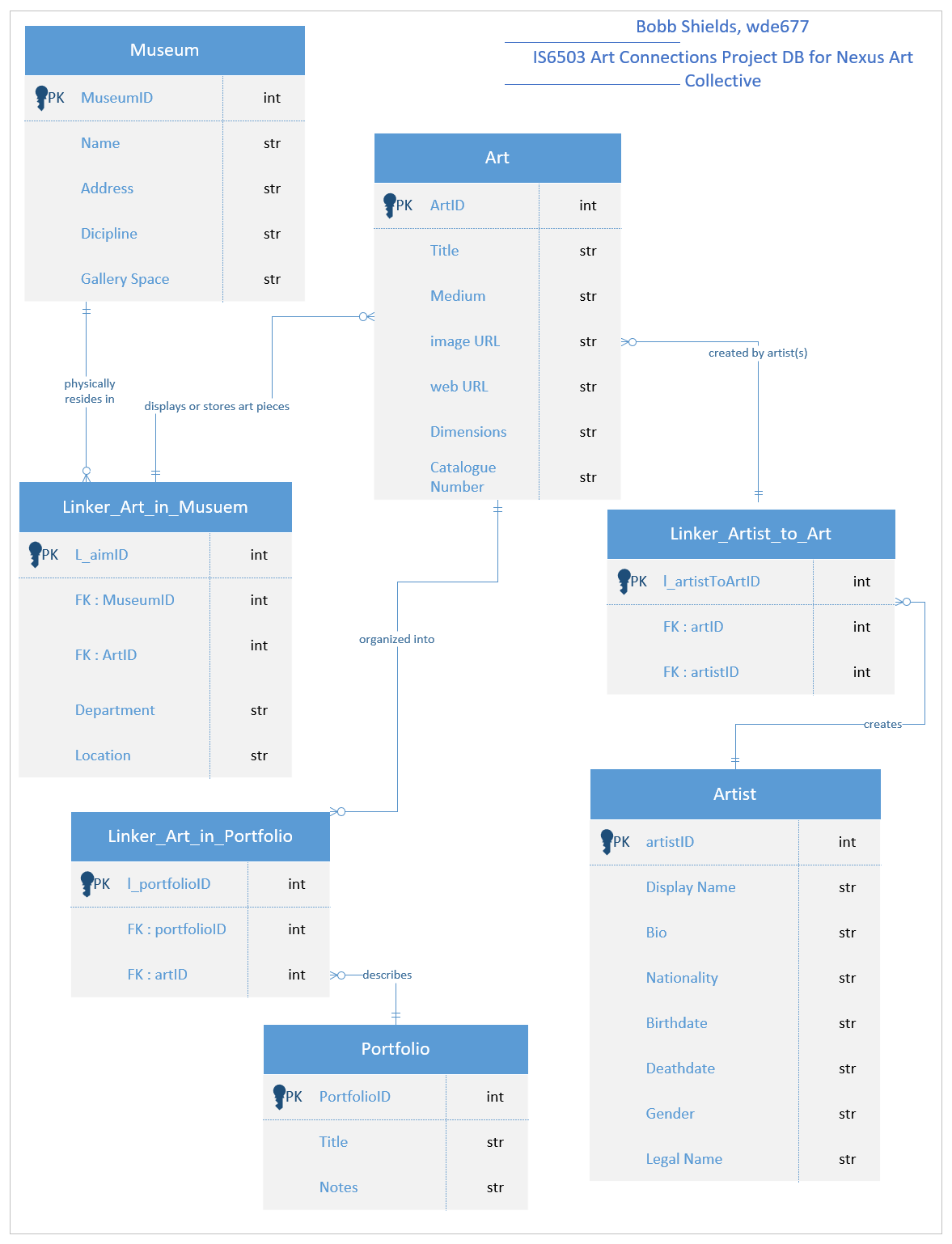


Figure 1 - v2\_final ERD

A screenshot of a computer

Description automatically generated

Figure 2 - Access DB Relationships on Trimmed Data

1. **Business rules that define all the relationships among entities, the constraints (if any).**

Relationships:

An artist can be unknown, or each artist can build one to many pieces of art, and multiple artists can collaborate on a single piece of art .: (0,M) Artists. The art must exist to be relevant and multiple pieces of art can be made by the same artist .: (1,M) Artworks. This M:N relationship is broken up by the Linker table, that assigns some to many artists to a single piece of art and any fields in this link would describe the individual artist’s contributions to the art. The linker has a 1:M relationship to art table, and again to the artist table.

A piece of art does not need to be part of a portfolio, or one to many pieces belong to a portfolio .: (0,M). Likely, a single piece of art would not belong to multiple portfolios. We will assume a piece of art belongs in only 0 or 1 portfolios, thus a 1:M Unary relationship. This is an M:M relationship broken up by the portfolio linker table, that has a 1:M from portfolio, and another 1:M relationship from art.

Each piece of art is located inside a museum in a 1:1 relationship, given that any official copies of art would likely have their own catalogue (unique) ID.

A museum will have at least one piece of art to display but likely many, so a 1:M. A piece of art is unique and exists in either 0 or 1 museums, so a (0,1) relationship. This too is broken up in a linker table, with the 1:M art assignments within the museum, and (0,1) for the art available for only one museum. Fields in this linker are useful for information about the art inside the museum – such as location.

No attempt will be made to identify full and complete portfolios by individual artists; what is made available per each museum’s public data repository is what is currently available to this database.

Atomicity – While the artist may be a person’s proper name, it could also be the name of a business entity, therefore it makes sense to not break down all the given text for artist name/title into legal first and last names. So true will this be for addresses and states, to keep it as text fields that will eventually allow for global entries.

Among the above relationships:

Strong Relationships

* Art – Artist: This relationship is strong because each piece of art must be associated with at least one artist, and an artist can create multiple pieces of art.
* Art – Portfolio: This relationship is strong if we assume that each piece of art must be part of a portfolio, even though a portfolio can contain multiple pieces of art.
* Art – Museum: This relationship is strong because each piece of art must be associated with a museum for it to be displayed, and a museum can display multiple pieces of art.

Weak Relationships

* N/A

Existence Dependency

* Art – Museum: This is an existence-dependent relationship, as the art needs to exist for it to be displayed in a museum. A piece of art being in a museum gives it relevance and context.
* Art – Portfolio: Similar to the museum relationship, a piece of art’s existence in a portfolio can be considered existence-dependent, as it adds value and context to the portfolio.

Mandatory Relationships

* Art – Artist: This requires a NOT NULL constraint on the ArtistID in the linker table because each piece of art must be associated with an artist.
* Art – Museum: If a piece of art must be displayed in a museum, then the MuseumID in the linker table should be NOT NULL.
* Portfolio – Art: If portfolio information is not available, the PortfolioID in the linker table should point to a dummy entry, so that generic text will display, requiring a NOT NULL constraint in the linker table.

Constraints

* NOT NULL Constraints on foreign keys where relationships are mandatory.
* UNIQUE Constraints where necessary, such as ensuring ArtID is unique.
* INDEX required on all source\_FK\* columns so they can easily be searched for the text FKs available in the source data.

1. **A summary of the functionality of the application (i.e. what the users can do with this application, future extension/integration with other systems etc.). These functionalities will be implemented in deliverable 2 and 3 and what you propose can be changed later.**

This new DBMS will allow users to keep track of art located in the three museums that have released public data. They will be able to search by discipline, medium, within a date range, by artist, to see listings of all matching art. If available, URLs will be called to display the art, and linked directly to its public webpage. Of course, showing the title, description of the art, and its associated meta data will be a primary focus.

ERD Update Notes

The ERD was updated for the following changes:

* V1
  + The verb on the relationships between entities was missing. Added
  + Added a third linker table to allow multiple artists to collaborate on the same piece of art
  + Changed portfolio to not hold artistID as an FK. Portfolio -> Art -> Artist exists as PKs so the artists that have work inside a portfolio can easily be queried as needed.
* V2 – merged into git, dropping filename version hereafter
  + Corrected duplicates and erroneous
  + Rearranged and re-worked relationships for aesthetic purposes only

Notes on Database Build

* Files and folders are self-descriptive. 10 use cases are defined, 6 & 8 need troubleshooting
* Imports done on .csv files done via flat file import functionality in SQL Server Manager
  + All columns allowed nulls and set to varchar(max) to ensure all rows were imported without error
* Creation script modifies the source tables to allow for indexes to be built, to enable the insert scripts to work faster
* Insertion script built for each museum.
  + Understanding of Teenie collection updated – is an entire archive received by CMOA so separate museum was deleted and made the Teenie archive into the first portfolio.
  + Processing times for museums have varied, most are at about 10 minutes. The Met took 6 hours for the last import
* Column analysis done by VS Code Copilot to ensure all columns were being captured where available, and fix discrepancies between museums. This corrected multiple issues, leading to a higher capture rate of data and assigning to the correct columns.
* The database seems to succeed in it biggest mission: normalizing the data from these various sources. Portfolio titles are no longer repeated in every row of the art table that belongs to the portfolio, and computations based on these are faster because of using PKs.

Issues

* Weight from all sources needs to be stated in units. Some sources marked as kg
* Dimensions need to be more explicit
* Portfolio management system needed to manually add portfolios to groups of art IDs
* Largest art museums and museum universe datasets were not used because they were out of scope, when a simple internet search built the three insert statements needed for the current version of the database where the primary focus is the art itself.

Summary of Use Cases

Functionality 1: Artist Across Museums

* Description: List artists and their artworks across different museums.
* Requirements Satisfied: #1, #2, #3, #5

Functionality 2: Specific Period

* Description: List artworks created organized by date.
* Requirements Satisfied: #1, #2, #3, #5

Functionality 3: Filter by Medium

* Description: Filter artworks by medium.
* Requirements Satisfied: #1, #2, #7

Functionality 4: Portfolio Management

* Description: Manage portfolios of artworks.
* Requirements Satisfied: #1, #2, #3, #5

Functionality 5: Artists with a Minimum Number of Artworks in Each Museum

* Description: List artists with a minimum number of artworks in each museum.
* Requirements Satisfied: #1, #2, #3, #4, #5, #6

Functionality 6: Museums with Average Number of Artworks per Artist Above a Certain Threshold

* Description: List museums with an average number of artworks per artist above a certain threshold.
* Requirements Satisfied: #1, #2, #3, #5, #6

Functionality 7: Artists with Artworks in Multiple Museums

* Description: List artists who have artworks in multiple museums.
* Requirements Satisfied: #1, #2, #8

Functionality 8: Artists with Artworks in the Same Medium

* Description: List pairs of artists who have created artworks in the same medium.
* Requirements Satisfied: #1, #2, #9

Functionality 9: Categorize Artworks into Subtypes Based on Medium

* Description: Categorize artworks into subtypes based on medium.
* Requirements Satisfied: #1, #2, #10

Functionality 10: Statistical Analysis of the Database

* Description: Perform statistical analysis on key tables and views.
* Requirements Satisfied: #1, #2, #3, #4, #5

A screenshot of a computer

Description automatically generated

Figure 3 - Use\_Case\_10 SQL Execution showing scope of DB

Summary of Requirements Satisfied

1. Involves multiple entities: Functionality 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
2. Involves more than one entity: Functionality 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
3. Uses mathematical functions: Functionality 1, 2, 4, 5, 6, 10
4. Uses the Count feature: Functionality 5, 10
5. Uses the Group By feature: Functionality 1, 2, 4, 5, 6, 10
6. Uses the Having clause: Functionality 5, 6
7. Uses the LIKE keyword: Functionality 3
8. Uses a correlated query: Functionality 7
9. Involves a unary relationship (self-join): Functionality 8
10. Involves both the supertype and one of the subtypes: Functionality 9

Future Use

* Portfolio management functionality meant for researching art outside of the available data and adding portfolio titles and notes to it since only one museum identified portfolio details.
* Computed columns are needed. As use Case 9 shows, it is relatively trivial to create sub-types to the art based on the medium descriptions, but that needed to use keywords. This should be researched and standardized in named procedures.
  + Artist Name – breaking out artists into two separate rows instead of collaborations being assigned uniquely
    - Artist ID in some data sets includes two integers and comma
  + Portfolio – Could assign a portfolio based on museum + artist
  + Medium – research the best keywords and keep searching the uncategorized for new medium categories to add.
  + Dates – cleanup is definitely needed. Some are noted as a range when it could have been, others are just 1800’s, some say or denote circa. Some logic to process this and standardize it across the different museums would be helpful
* With a standardized view of art that is meant to be generalized, this combined dataset can be used to train Large Language Models on how the largest museums describe art, so that can become more fluent in that style of language.
* Additional museums can be added by adapting the insert scripts to additional datasets.
* Eras or disciplines seem to be incomplete. Additional tables may be useful to define these eras and apply those rules to computed columns – such as reading out a silk painting from 1950 by this artist is likely a ‘Renaissance’ painting (or whatever is correct.)
* A thoroughly inspected database likely to be exported and released to Kaggle