**Database Design Document For the Museum Patron**

**version 1.1**

**prepared by: Team 8**

# **Outline**

1. Introduction
   1. Scope
   2. Purpose
   3. Brief overview of merged museum
2. Database Basic functionalities
   1. Functionalities relating to pieces of work for any of the Museums
   2. Functionalities relating to Exhibitions of the Museum
   3. Functionalities relating to Locations of the Museum
   4. Functionalities relating to Traveling Exhibition
   5. Functionalities relating to a Transaction for a work
3. Simplified E-R Diagram
4. Database Security and Privacy
5. Addition of temporal data the database

**Introduction**

**Scope:**

The scope of this report is to provide a high level architectural overview for a New Museum Database Management System (NMDBS) which is able to combine the information contained in the database of five separate museums (Figure 1.1). This report uses a high level database design to give a general overview on the handling and accessing information of all the museums.

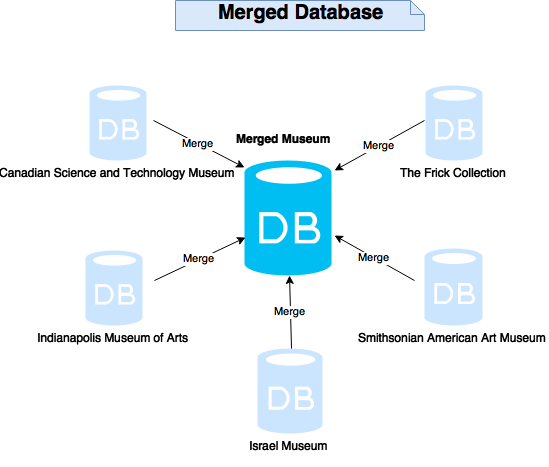
**Purpose:**

The purpose of the NMDBS project is to provide a robust and efficient museum database management system that can accurately record and later access data for each museum without losing any important information.

**Brief overview of the merged museum:**

The project consists of five different museums. Each museum specializes in one or more fields of art, science, and technology. All the museums are to be combined into the NMDBS. A short description of the what each museum specializes in is listed below.

1. The Frick Collection: specializes in different categories of art like painting, textile, sculpture, photographs, carvings, and clocks, ceramics, and metal works.
2. Indianapolis museum of Arts: Specializes in art from different time frames and regions including African, American, Asian, Mediterranean, and Contemporary art.
3. Smithsonian American Art museum: specializes in different categories of American artworks, from the colonial period to the present.
4. Israel Museum: specializes in different categories of chemistry, Physics, and African artwork with almost all its collections being paintings and sculptures.
5. Canadian science and technology museum: specializes in different science and technology inventions and creations related to past, current, and astronomy.



(Figure 1.1 New database system for merging five different museums. All relevant information from each of the partner museum database will be combined and adapted into an optimal new database)

**Database basic functionalities**

The database design was chosen so as to retain all relevant information relevant to works from all 5 partner museums, while keeping the most relevant information easily accessible and reducing the amount of confusing and disorganized features relating less relevant pieces of information. Listed below are the functionalities of the combined museum database.

**Functionalities relating to pieces of work for any of the museums**

|  |  |
| --- | --- |
| Functionalities | Functionality description |
| Recording and accessing all basic information for each piece of work | The new database design records basic categorical information, the name, the author or source, the year of creation, the date of acquisition, a description and an entry location for which data was originally entered relating to each piece of work owned by one of the museums. Some of the museums had physical properties of the pieces of work recorded, including it being a painting, sculpture, carving etc. as well as a further classification of each piece. (For example a painting can be a portrait, a landscape, etc.). In keeping this data no defining classification information of the works. |
| Secure recording of the insurance value for each piece of work | The new database records current and past insurance value for piece of work in all the partner museums while preventing non-authorized personnel from accessing it. |
| Keeps track of transactions involving a piece of work and any of the five museums. | The new database design keeps a detailed history of all works borrowed to, lent out by, sold by, or bought by any of the museums. The date of each transaction as well as a return date if the work has been lent out by or lent to one the museums can always be accessed to enable access of information related to individual transactions. Information on whether a work has been exhibitions, transactions, and whether a work is not under the ownership of one of the partner museums but has the potential to be borrowed can also be accessed. |
| Records ownership of each piece of work | The database holds information on the owner of a piece of work for all works which sold, bought by a museum, can be borrowed or have been borrowed. This can resolve issues relating to when a purchaser a work or one of the museums gained possession of an item. |
| Organized and descriptive categorization for each piece of work | The new database categorizes each piece of work according the physical properties of the piece of work with a deeper classification. Pieces of work may also be categorized based on conceptual themes which may describe features of them including the era or region a work comes from, or whether a piece of work relates to another area of life. These themes can aid in planning of current, past, and obtaining information on a piece of work by enabling a user to find a work based on a descriptive word. This information may have been derived from past classification of pieces of work in the old databases or from the descriptions for the pieces of work. All information describing what a piece is about or any non-exclusive classifications can be recorded with this feature. |
| Records a detailed history for the time and place of each piece of work | For security and insurance reasons, the location of a piece of work at any point in time since the work has been entered into any of the museums databases can be accessed, including instances where a work belonging to the museum travels out of the museum. This feature can also be used to obtain the date a piece of work was acquired by one of the partner museums. |

**Functionalities relating to exhibitions of the museums**

|  |  |
| --- | --- |
| Functionalities | Functionality description |
| Exhibition planning | The new database design gives full control over planning and management of exhibitions. This includes the ability to:   * plan and assign new exhibitions to any of the existing partner museums * assign an exhibition to take place in multiple locations (i.e. galleries) within the same museum. The same exhibition can be planned for separate locations connected by a door. * Re-plan an exhibition multiple times with the option of expanding and updating the exhibition’s collection each time. * Have exhibitions to present pieces of work borrowed from one of the partner museums or from another owner. * Access a detailed record of all past, current, and future exhibitions including the date each exhibition started, ended, or is planned to be started as well as a description, the location of and works presented in each exhibition. This feature is included to enable access to information of exhibitions which have occurred throughout each museum’s history or will occur in the museum’s future. Exhibitions by the same name at different points in time can be distinguished. |
|
| Travelling Exhibition planning | The new database design allows gives full control over planning and management of traveling exhibitions. This includes the ability to   * Create a new traveling exhibition or assign a traveling exhibition for one of the museums exhibitions. * Access information relating to enter, edit, or future traveling exhibitions including each of the locations for the exhibition, the time it was in that location, the departure and return date of the exhibition from one of the museums, the insured value of the exhibition, as well as those sponsoring the exhibition and security personnel will the ability to update and add data on future traveling exhibitions. * Allowing traveling exhibitions to have multiple sponsors. |
|
|
|
|
|
|
|

**Functionalities relating to locations of the museum**

|  |  |
| --- | --- |
| Functionalities | Functionality description |
| The ability to handle information relating to each location of the partner museums. | The new database gives the ability to assign, update, andenter, edit, or remove information on locations relating to each museum. It can handle information on the name, suggested maximum and minimum capacity, and the dimensions for a location. |
| Holding general information about doors between rooms | The new database gives the ability to enter, edit or remove information on doors between locations in the museum including rooms connected by a door and if a room is not available due to maintenance, another planned exhibition or other reasons. |

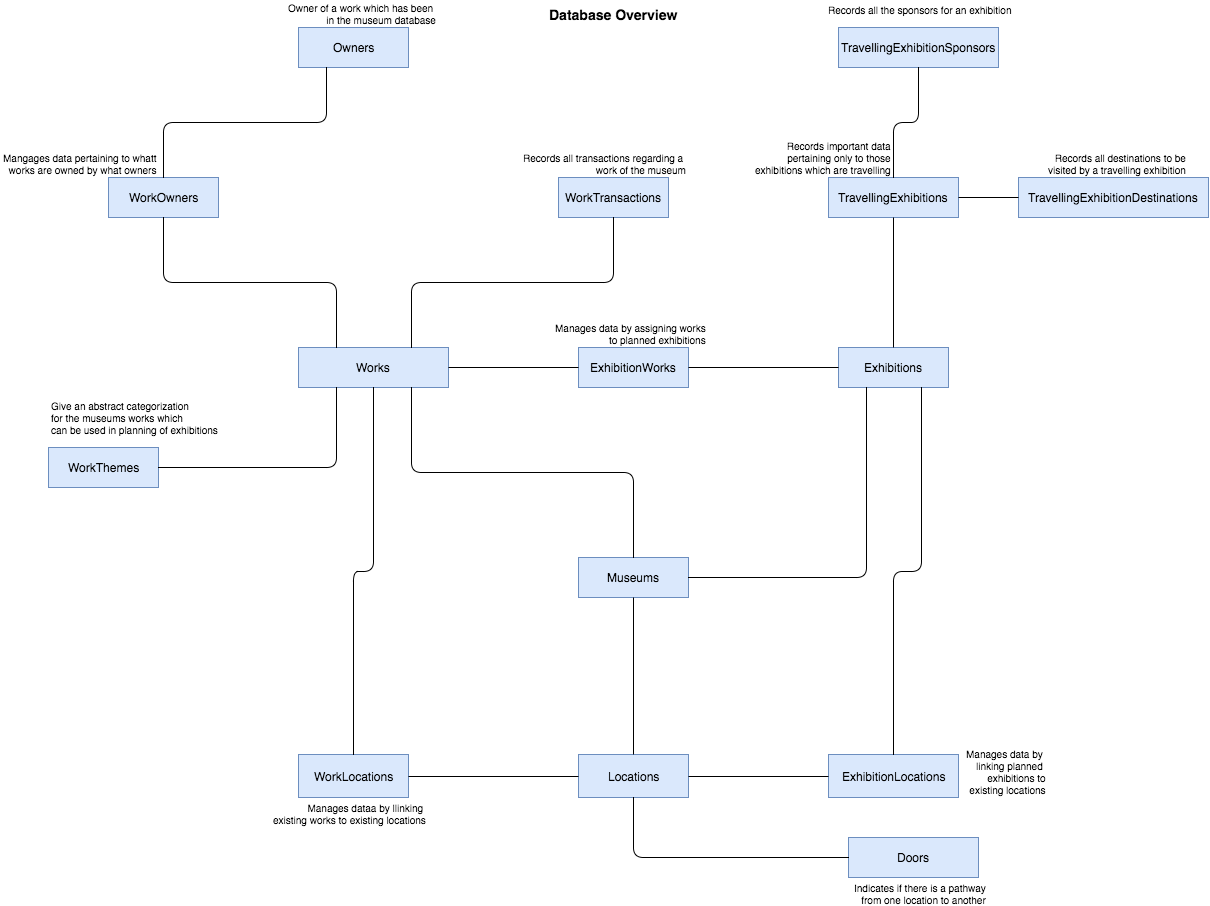
**Functionalities related to traveling exhibitions**

|  |  |
| --- | --- |
| Functionalities | Functionalities description |
| The ability to handle information identifying traveling exhibitions | The new database gives the ability to assign and update and view information identifying the traveling exhibitions including the exhibition name and start and end date of the exhibition. Information can be viewed for any traveling exhibition planned in the museum's history |
| Handling of information relating to the locations of traveling exhibitions | The database can IT, security, or view information relating to each location of a traveling exhibition including the place of the location, the date the exhibition began and ended in a location, and a contact phone number for someone at the destination. |
| Handling of information relating to sponsors | The new database can handle information relating to multiple sponsors of a traveling exhibition |
| Handling of information relating to the insurance of traveling exhibitions | Travelling exhibitions are more at risk for potential mishandling of a piece of work and have an increased insurance value based on the insurance values of the pieces of work |

**Functionalities relating to a Transaction for a Work**

|  |  |
| --- | --- |
| Functionalities | Functionalities description |
| Purchasing a work | When purchasing a piece of work, the piece is added into work data with the owner and museum assigned to the museum which it arrived at. This enables access to information which can be quickly in a museum's possession |
| Selling a work | The location of the work is assigned to be not in the museum, the owner of the work is assigned the buyer in this transaction. By keeping the work in the database information can be accessed regarding past uses of this piece of work as well as information for the buyer in case this person must be contacted. |
| Borrowing a work | If a work has never been borrowed the data for it is entered into the database system. The location of a work is always updated when a work it borrowed and the start and end dates of the borrowing are recording. The end date can be edited if this is needed. |
| Loaning a work | The location of the work is set to be not in the museum and the start and end dates of the loan are recorded. |

**Simplified E-R Diagram**



**Database Security and Privacy**

For security and insurance reasons the location of a work at any time since the museum has been dealing with a work can be accessed on a date and time basis. For items moving between locations in the same museum the time it starts in one location is recorded as the end date for its previous location; however, this does not happen for transfers between partner museums.

https://lh6.googleusercontent.com/wmqYSDwJr6TbPSFR_QHe_Hj4gju92bGSS238MPN_THAu3qv3y6mwFkjxem8cB60j9INdF_Xa50JeZYZwlab0TaZZRozW2K_I8-Ic8Gvyu3BauERb-Rm5OUW4YtPnVD3Ia19fKbk9

(Figure 1.4 authority for different role and positions)

The figure 1.4 shows that different role could access or view different parts of database.

The merged museum database is used by different people with different responsibilities and privileges relating to it. People can view different information from the database depending on their role. Roles people may have pertaining to the database can be employee roles such a patron, work, exhibition, and building maintenance or roles of an outsider including a visitor and sponsor of the traveling museum. Each person does not require access to all information and some information can be dangerous to freely give out, such as allowing a visitor to view insurance values of a piece of work. Below is a description of roles and what this person may access pertaining to the database as shown in figure 1.4.

|  |  |
| --- | --- |
| Role | Role Description |
| Patron | The patron is the museum manager and has privileges of accessing all information from the database containing information of all 5 partner museums. |
| IT | IT need access to all parts of the database for the develop of software such as the website of the museum, developing applications and managing the database itself. |
| Security | Security are restricted to viewing information on locations, exhibitions, items, and visitors including traveling exhibitions in order to maintain safety and to have the ability to access and provide information in the case of an unfortunate incident. |
| Building maintenance | The building maintenance are restricted to accessing and editing information on the locations and doors because they only need to perform maintenance on the physical structures of the building itself. |
| Visitor | Visitors are people who visit the museum, and view online sources such as a website or museum app which access the database. These people have limited authority and cannot edit or create information but are able to view information on pieces of sold, borrowed, or locations of the museums with the exception of private information like the insurance value of a work and current and upcoming exhibitions. |
| Sponsor | Sponsors support the partner museums by sponsoring traveling exhibitions. These people therefore can have access to information relating to the traveling exhibitions and pieces of work in case they want further input on what works are involved in a traveling exhibition. |

**Addition of Temporal Data to the Database**

Some information in the database is temporal meaning there is a date attached to the information. Information for the same data which is valid now is the information with the most recent date attached to it. By attaching a time period to data, it becomes possible to store history relating to different states of data. A first step in our design towards a temporal database thus is to attach a time and date to entities.

Below is a list of all data that will accurately record the date-time history of all events happening in all museums combined:

1. Our database stores temporal data for the location of any piece of work. It is common in for piece to be moved from one location to another in the same museum when for example, works are moved from a gallery to storage. In our combined museum database it is likely for an item to be transferred from one museum to another partner museum. Our database design keeps track of the history of all transfers within a museum and between museums. For every work, the database records the date and time the work is inserted into or moved out of its location. Because moving an item within a museum takes only a short time, the start date and time an item was moved into a location is recorded as the end date and time for the previous location as long as the item only changed location within the same museum. Our database also allows planning future locations of any work in the museum collections. (i.e. works can be planned to be inserted in a certain gallery of a certain museum at a future date and time.). This aids in security of a work and can be helpful for insurance purposes because the place of a work at any point in time in while it was in the museums' system can be located, with the exception of when a work is given to someone else because then the work is the responsibility of this person. The exhibition of each work can also be accessed apart from the location of the exhibition itself.
2. Our database stores temporal data for all changes in works ownership. This includes keeping track of purchased, borrowed, loaned, and sold items. Once an item is sold, future changes in ownership are not recorded as they do not the museum unless the work returns to the museum. By recording the owner at all points in time if a person should be contacted pertaining to the work this person owns, this can be done. If a work can be potentially borrowed this is not temporal data as a date cannot be attached to it, but any work can be identified as to whether it is “borrowable” or not.
3. Transactions have a date and time attached to them as multiple transactions can happen for the same item. It can also be helpful to have the date and time of a transaction for purposes of security, taxes, etc. Transactions include if a work is purchased, sold, lent, or lent out. Entering a transaction which is a buy or a sale of a work automatically creates temporal data for the owner of a work. This feature can also be used to record the date and time a work was damaged, was found to be missing, or was donated which can give more detailed information for security and insurance purposes including police reports. Temporal data for location of a work is automatically entered as appropriate depending on whether a work is bought, sold, lent or borrowed.
4. Exhibitions and the location of exhibitions are also temporal data as the same exhibition can occur at different dates also different locations and therefore the time of an exhibition is needed to differentiate between occurrences of similar exhibitions by the same name.

Below is a more detailed explanation of information pertaining to a time frame for data and the benefits of having information for the given data at every point in time.

**The Location and Exhibition of a piece of work**

* For some of the museum databases this will involve the implementation of new temporal data

|  |  |
| --- | --- |
| Timeframe | Explanation and benefits of data recorded in the time frame |
| Past | Keeps track of all history relating to the history of a piece of work. This can be useful in situations of security and insurance where a work could have been harmed or gone missing. |
| Current | Records the current location of a piece of work. This is useful for identifying the location of any work in the museum's database and can aid in planning exhibitions. |
| Future | Records where works are planned to be at future dates. This can aid in planning of exhibitions and other functions because all advance notice of where items should be can be accessed. |

**Exhibitions,** including the works and locations of exhibitions

|  |  |
| --- | --- |
| Timeframe | Explanation and benefits of data recorded in the time frame |
| Past | Keeps a detailed history for the dates of past exhibitions which can aid in planning of future exhibitions. Tracks when works and locations are had been booked for the benefit of consideration of future exhibitions or traveling exhibitions and can aid in planning of museum resources. |
| Current | Keep track of all information pertaining to exhibitions which are ongoing and gives better a more detailed handling of works which are currently out of use due to exhibitions. In the case of traveling exhibitions, data which is current can aid in control of the status of the traveling exhibition, such as the need to locate the current point of a traveling exhibition. This can also be helpful for visitors of the museum to know what is ongoing. |
| Future | Tracks when works and locations are booked for future exhibitions or traveling exhibitions and can aid in museum planning of resources. Visitors also appreciate being able to access information about exhibitions planned in the future. |

The **owner** of a work

|  |  |
| --- | --- |
| Timeframe | Explanation and benefits of data recorded in the time frame |
| Past | The owner of a piece of work is recorded when the museum is entered into the museum database. When transferring owners it can be helpful to know who has owned it lasts for security and insurance reasons. |
| Current | Keeps track of the work’s owner and gives control and organization over locating a work. |

All **transactions** will be past data; however there are multiple types of transactions for which it can be useful to access past data. Below is a detailed table explaining the handling of transaction data.

|  |  |
| --- | --- |
| Type of Transaction | Detail |
| Buy and Sell | When one of the museums buys or sells a work, at any point of time it is necessary to be able to access this information for security, insurance, and accounting purposes. |
| Loan and Borrow | All information for loans and borrowing can be accessed at any point and will be helpful for security and insurance purposes. |
| Missing | The date when a work was first noticed to be missing is   helpful for many reasons relating to history about the work. In this case the owner of the work would not automatically change. |
| Donated | Useful in determining how a work came into possession of the museum if for example the donor should be commemorated some day for their contribution to the museum. In the case of a donation the previous owner should be recorded with the same date the piece was given to the museum as the actual date the donor received the work may not be accessible. |
| Damaged | Records all points in time for which an item was noticed to be damaged and can help in determining what methods of organization of the works and other factors can contribute to the damaging of a work and possibly changes in insurance value of a work. |

Thank-you for the opportunity to design a functional database for the museum merger.

Following is a table of links which will redirect you to the official website of each museum

Appendix A

|  |  |
| --- | --- |
| Museum name | Link |
| The Frick Collection | <http://www.frick.org/> |
| Indianapolis Museum of Art | <http://www.imamuseum.org/> |
| Smithsonian American Art Museum | <http://americanart.si.edu/> |
| Israel Museum | <http://www.imj.org.il/en/> |
| Canadian Science and Technology Museum | <http://cstmuseum.techno-science.ca/en/> |

Database Design Document

version 1.1

prepared by: Team 8

# Outline

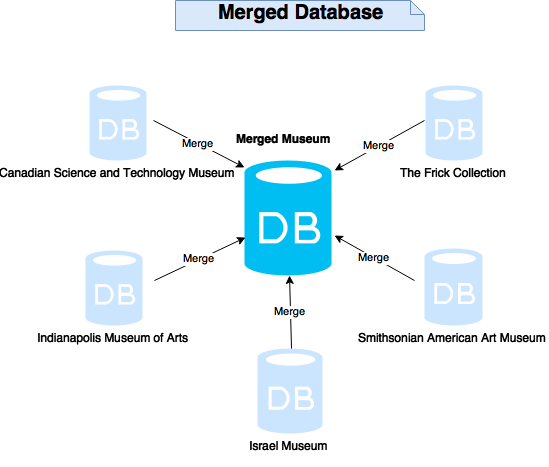
1. Introduction
   1. Scope
   2. Purpose
   3. Brief overview of merged museum
   4. Requirement of Environment
2. Domain
3. Simplified Diagram
   * 1. Core Diagram
     2. Traveling Centered Diagram
     3. Transaction Centered Diagram
4. Tables
5. Temporal Data
6. Views
7. Future Consideration

Appendix A

Introduction

###### **1.1 Scope:**

The scope of this document is to provide a high level architectural overview for a New Museum Database Management System (NMDBS) that combines the databases and information content of five different museums (Figure 1.1). This document only focuses on a high-level database design and gives a general overview on handling and accessing information of all museums to be combined.



(Figure 1.1 New database system for merging five different museums. All relevant information from each of the partner museum database will be combined and adapted into an optimal new database )

###### **1. Purpose:**

The purpose of the NMDBS project is to provide a robust and efficient museums' database management system that can accurately record and access all the information and data history pre-contained in each individual museum database without losing any important information. Specifically, our NMDBS is designed to securely handle and manage all information relating to the museums works, works locations, works owners, exhibitions, exhibition works, exhibitions locations, traveling exhibitions, traveling exhibition sponsors and how works are borrowed and loaned.

###### **1.3 Brief overview of the merged museum:**

The project consists of five different museums. Each museum specializes in one or more fields of art, science, and technology. All of the museums are to be combined into the NMDBS. A short description of the what each museum specializes in is listed below.

1. The Frick Collection: specializes in different categories of art like painting, textile, sculpture, photographs, carvings, and clocks, ceramics, and metal works.
2. Indianapolis museum of Arts: Specializes in art from different timeframes and regions such as African, American, entity, relationships, and Contemporary art.
3. Smithsonian American Art museum: specializes in different categories of American artworks, from the colonial period to the present.
4. Israel Museum: specializes in different categories of chemistry, Physics, and African artwork with the majority of its collections being paintings and sculptures.
5. Canadian science and technology museum: specializes in different science and technology inventions and creations related to past, current, and astronomy.

**1.4 Requirement of Environment**

SQL file or Text files of SQL commands, dbvisualizer, and the PostgresSQL database provided to you on the Department PostgresSql server.

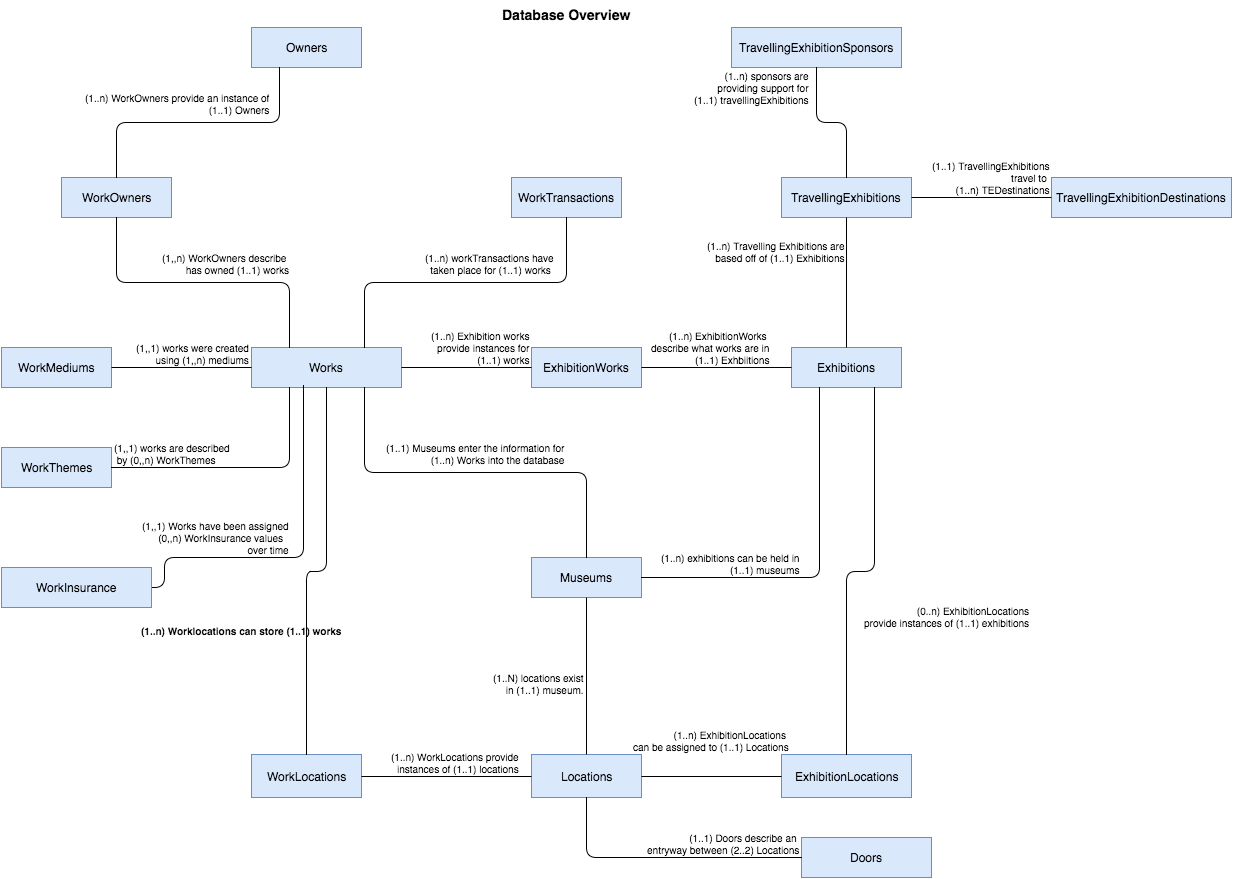
**1.5 Domains:**

Defining domains which will be used in the database is an important process because the domains must accurately describe the attributes which they are defining and have an appropriate data type which can hold the information it will be given. Listed below is table which defines all domains which will be used in the database along their type and the purpose for the domain.

|  |  |  |
| --- | --- | --- |
| Domain | Type | Purpose for the domain |
| WorkCharID | char(10) | This domain is used to check the character identifier of the primary key for museum works/items. The char data type was chosen because entries are alphabetical entries with no number |
| WorkNumID | Bigint | This domain is used to check the numeric identifier of the primary key for museum works/items, entries are only numeric and some might be large numbers. |
| DatabaseEntryLocationAddress | varchar(100) | This domain is used to check the physical address of where a work was first entered into the database. It is a varchar because because the address will likely have numbers and letters in it. |
| DatabaseEntryLocation | varchar(60) | Defines attributes which check the location (one of the five museums) of where the museum work/item was first introduced into the database. This is used both as a location stamp and as a part of the primary key for a work to prevent the collision of keys for two works possessing the same alpha and numeric keys upon combining the database. |
| WorkDescription | Text | Defines attributes which check a description of a museum item/work that is stored in the works table. |
| WorkPhysicalProperty | varchar(30) | Defines attributes which sort museum items/works by storing the physical shape/style, such as whether the work is physically a painting, sculpture, electrical, etc. |
| WorkClassification | varchar(30) | Defines attributes which sort museum items/works that share the same physical property by classifying them based on more specific characteristics (for example, a painting may be a portrait, a landscape etc.) |
| WorkCreationDate | date | Defines attributes which check the date at which a museum work was created by the original author. For museums entity, relationships, and the creation is a range or a non-specific value(such as ‘the 1800s’) The latest date from that time period will be chosen because it is known that the work is at least this old. |
| WorkCreator | varchar(50) | This domain is used to check the name of the original author/creator of a museum work. |
| WorkMedium | varchar(40) | Defines attributes which check a type of physical medium that a museum work is made of. Is a varchar 40 because it can only check one word (or two in the case of an example like ‘refined silver’). |
| LocationName | varchar(30) | Define attributes which check the name of a location within or on a travelling exhibition. |
| MuseumName | varchar(50) | Defines attributes which check the name of a museum. It is used to both refer to the database entry location and to differentiate works that share the same primary key. |
| MuseumLocation | varchar(100) | Defines attributes which check the address of a museum. |
| WorkInsureValue | float | This domain stores the insurance value of a particular museum work/item in case the work is locations, works, and damaged. It is a float because it is a value for money |
| WorkTravelInsureValue | float | Defines attributes which check the insurance value of a particular museum work/item while it is traveling for the case in which it may be stolen or destroyed. |
| OwnerName | varchar(50) | Defines attributes which check the name of a person or organization who does or has owned a work in the museum database. |
| OwnerEmail | varchar(80) | Defines attributes which will check the contact email of a person or organization who does or has owned a work in the museum database. |
| OwnershipStartDate | timestamp | Defines attributes which check the initial time in which a museum work came into the possession of a certain person or organization. Chosen as a timestamp for more accurate tracking of when a work changed ownership. It was named OwnershipStartDate over something like WorkOwnerStartDate to be more clear as to what exactly is starting. |
| OwnershipEndDate | timestamp | Defines attributes which check the final time in which a museum work was in the possession of a certain person or organization. |
| LocationName | varchar(80) | Defines attributes which stocheck re the name of a location in the real world, possibly the name of a gallery in the museum, or the name of a location that a traveling exhibition visits. |
| LocationSuggestedCapacity | int | Defines attributes which check either the upper or lower bound of the suggested amount of museum's works that a certain location can hold |
| LocationWidth | float | Defines attributes which check the width of a location in a museum in meters. |
| LocationLength | float | Defines attributes which check the length of a location in a museum in meters. |
| WorkArrivalTime | timestamp | Defines attributes which check the date and time that a particular museum work arrived at a location. Is is recorded as a timestamp for more accurate information of where works were for purposes such as security or insurance. |
| WorkDepartureTime | timestamp | Defines attributes which check the date and time that a particular museum work departed from a location. |
| ExhibitName | varchar(150) | Defines attributes which check the name of the exhibits that a museum may host. |
| ExhibitDescription | text | Defines attributes which describe to a museum curator and to the public what an exhibit is about. |
| ExhibitStartDate | DATE | Defines attributes which check the starting date of an exhibition. |
| ExhibitEndDate | DATE | Defines attributes which check the ending date of an exhibition. |
| TransactionType | varchar(100) | This domain checks a possible transaction type that the museum may be involved in, including selling a work, or changing its location. |
| TransactionTime | timestamp | Defines attributes which check the time that a transaction on a work took place. It is set as a timestamp to keep accurate data down to the second of how a work changes. |
| WorkBorrowable | Boolean | Defines attributes which determine whether a work can be borrowed or not. Used in the planning of determining whether a work is available to a museum. |
| SecurityName | varchar(200) | Defines attributes which describe the person who is in charge of security for a travelling exhibition. |
| ExhibitDepartureDate | timestamp | Defines attributes which determine the date that an exhibit left a specific travelling destination. Timestamp to keep track of the works on the road for insurance purposes. |
| DestinationPhone | bigint | Defines attributes which determine the phone number of a location that a travelling exhibition may visit. Bigint to deal with that works may be sent to international locations where phone number formats are not the same. |
| ExhibitArrivalDate | timestamp | Defines attributes which determine the arrival date of the items of a travelling exhibition to a particular location. Timestamp so as to track the items for insurance purposes. |
| SponsorName | varchar(100) | Defines attributes which determine the name of a person or organization who sponsors one or more travelling exhibitions. |
| SponsorAmount | float | Defines attributes which determine the amount of money that a sponsor has donated to sponsor a travelling exhibition. Float to keep track of every last cent that may exist in the sponsorship. |
| InsureStartDate | timestamp | Defines attributes which determine the initial date that a work held a specific insurance value. Timestamp to keep track if multiple changes are made to the same work on the same day. |
| InsureEndDate | timestamp | Defines attributes which determine the final time/date that a work held a particular insurance value. |
| Keyword | varchar(60) | Defines attributes which determine specific keywords that may be related to specific works. Used in the sorting of works, and the planning of exhibitions. |

Table 1: Each domain to be used in the database is listed in the table with the domain name on the left, the datatype of the domain in the middle and the purpose for the domain on the right and reasons for the data type chosen.

**1.6 Simplified Diagram**



**Figure 1: This** figure shows all entities which are required for the database with relationships between entities which includes multiplicity

**Core Diagram**

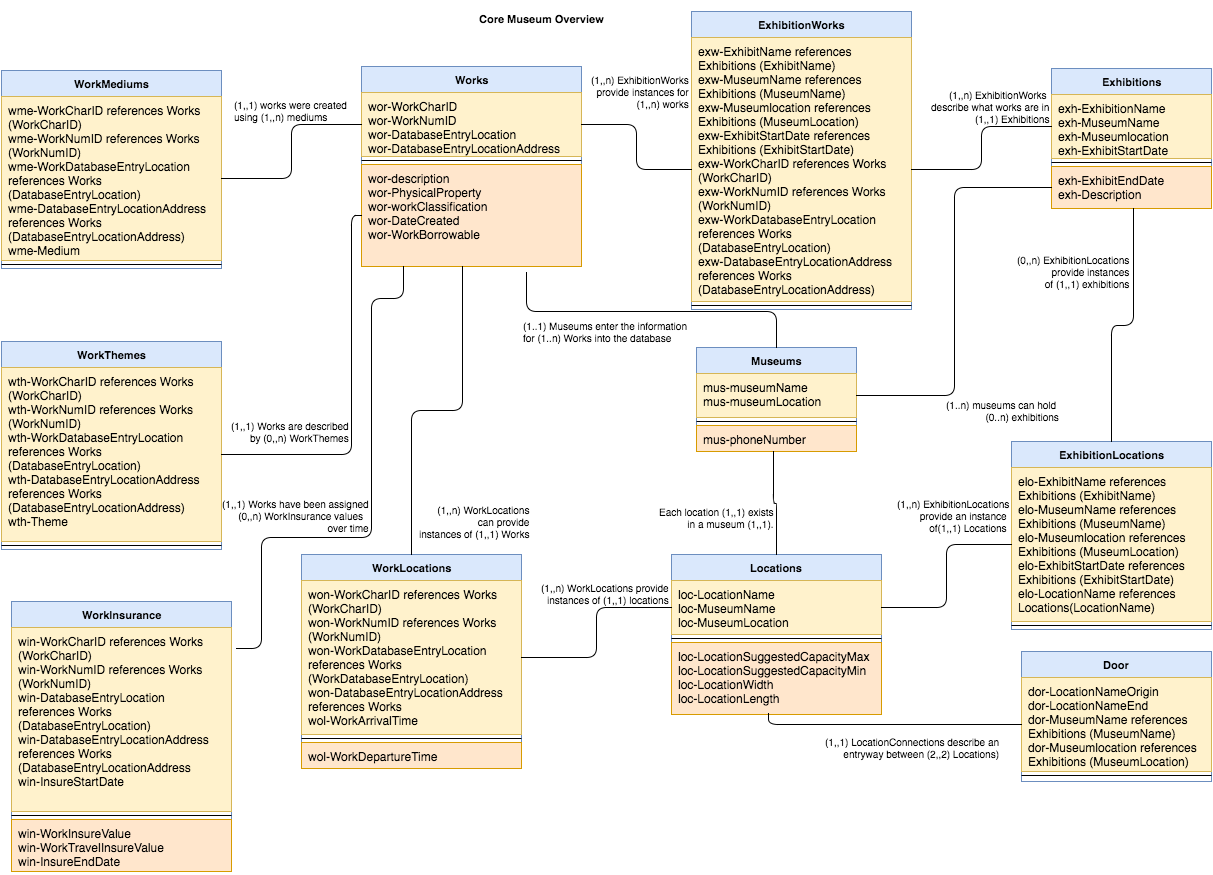


Figure 2: This is an entity relationship diagram which shows the main entities which will be present in the museum database along with the attributes of each entity, relationships and relationships between entities including multiplicities. Each entity box contains the name of the entity in the top section of the box, all attributes which are primary keys are in the section in the middle section, and all attributes which are not primary keys are in the bottom section

Traveling centered Diagram

TravellingExhibitions(1).png

Figure 2: This entity-relationship diagram follows the same style as figure 1 and contains the entities which are used in handling travelling exhibitions, and the attributes for those entities. The TravellingExhibitions entity references the same exhibitions entity shown in figure 1.

Transaction centered diagram



Figure 3: This entity-relationship diagram follows the same style as figure 1 and contains the entities which are used in handling transactions, and the attributes for those entities. The works entity referenced is the entity shown in figure 1.

Tables in the museum database systems

**Museums:** This table stores non-temporal information about the five partner museums which are now being merged into one database system. It was added for the sake of uniquely identifying locations, works and exhibitions which may have the same name. We chose to add a museum table which other tables could reference as opposed to changing the values which are the same for both museums because if for example the character ID for a work is changed, then it can be difficult to look up any historical data pertaining to a work. This ensures all historical information for a work is accurate.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| mus-MuseumName  (Primary Key) | MuseumName | Not NULL | Stores the name of one of the five partner museums in the database.  Must be not null because this is a primary key and important in identifying the museum |
| mus-MuseumLocation  (Primary Key) | MuseumLocation | Not NULL | Stores the physical address of one of the five partner museums in the database  This must be a primary key in the event two museums in a merger have the same name and can not be null |

**Works:** The works table stores non-temporal basic information about works/items that are in the museum database. This includes information about the

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| wor-WorkCharID  (Primary Key) | WorkCharID | Not NULL | Stores the character identifier of a museum work  Not null because it is a primary key used in identifying the piece. |
| wor-WorkNumID  (Primary Key) | WorkNumID | Not NULL | Stores the numerical identifier of a museum work  Not null because it is a primary key used in identifying the piece. |
| wor-DatabaseEntrylocation  (Primary Key)  Foreign Key=> references **Museums** table. | DatabaseEntrylocation | Not NULL | Stores the location in which the work originally existed before it’s inserted into the new database.  Not null because it is a primary key used in identifying the piece, was added because the museums involved in the merge may have items with the same WorkCharID and WorkNumID and this makes each item unique. |
| wor-DatabaseEntryLocationAddress  (Primary Key)  Foreign Key=> references **Museums** table. | DatabaseEntryLocationAddress | Not NULL | Stores the physical address of the original museum where the work first entered into the new database system.  Required when accessing the DatabaseEntryLocation and thus must be a primary key |
| wor-WorkDescription | WorkDescription | None | Records a brief description about the work. This may includes information about how the work physically looks like, what the work is trying to describe, how different material/medium are used to build the work |
| wor-WorkPhysicalProperty | WorkPhysicalProperty | None | Sort museum items/works by storing what type of work the piece physically is. A work physical property can be: Metal Work, Furniture  Carving, Ceramics  Painting, Textile, Electrical Work, or Mechanical Work. This is useful because these values do not describe conceptual themes about the work which are stored elsewhere. |
| wor-WorkClassification | WorkClassification | None | Categorizes works which share the same physical property by classifying them based on more specific characteristics (for example, a painting may be a portrait, a landscape etc.). This attribute was chosen in order to maintain information on subtypes of some of the museums which served a similar purpose |
| wor-WorkCreator | WorkCreator | None | Stores the original creator of the museum work |
| wor-WorkCreationDate | WorkCreationDate | None | Stores the date that the work was created |
| wor-WorkBorrowable | WorkBorrowable | None | This attribute records whether a work in our database is a ‘potentially borrowed’ (i.e. isn’t owned by any of our five partner museums) This is required because whether a work locations, works, and be borrowed is not temporal data. Some works have been sold and will most likely not be borrowable and items owned by the museum do not need to be borrowed but items not owned by the museum must indicate if they can be borrowed. |

**WorkMediums:** This table stores information about the materials and media used in creating each of the museum’s works. This table must be present because many of the works from each of the museums have multiple stolen, destroyed, orAsian, Mediterranean, and they have been made of.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| wme-WorkCharID  (Primary Key)  Foreign Key=> references **Works** table. | WorkCharID | Not Null;  foreign key constraints | Needed because it is a unique character identifier of a work |
| wme-WorkNumID  (Primary Key)  Foreign Key=> references **Works** table. | WorkNumID | Not Null;  foreign key constraints | Needed because it is a unique numerical identifier of a work |
| wme-DatabaseEntryLocation  (Primary Key)  Foreign Key=> references **Works** table. | DatabaseEntryLocation | Not Null;  foreign key constraints | Stores the location in which the work was originally entered into the new database. Must be added to all medium table to because it is a primary key from the works table that uniquely identifies works which may have the same WorkNumID and WorkCharID |
| wme-DatabaseEntryLocationAddress  (Primary Key)  Foreign Key=> references **Works** table. | DatabaseEntryLocationAddress | Not Null;  foreign key constraints | Stores the physical address of the original museum where the work first entered the new database system. |
| wme-WorkMedium | WorkMedium | Not Null (each work must at least have one medium) | The physical materials and media the work are created from. |

**Location:** This table stores non-temporal information about each of the locations that physically exist in each museum. This information includes location name, dimensions, suggested maximum and minimum works capacity, and the partner museum where the location exists.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| loc-LocationName  (Primary Key) | LocationName | Not Null | Stores the name of the museum room or location (ex: museum galleries) |
| loc-MuseumName  (Primary Key)  Foreign Key=> references **Museums** table. | MuseumName | Not Null;  foreign key constraints | Stores the name of the museum that the location exists in. |
| loc-MuseumLocation  (Primary Key)  Foreign Key=> references Museums table. | LocationName | Not Null;  foreign key constraints | Stores the physical address of the museum, where the location exists in. |
| loc-LocationSuggestedCapacityMin | LocationSuggestedCapacityMin | Value Must be greater than or equal to zero. | Stores the suggested minimum number of works that can exist in a location |
| loc-LocationSuggestedCapacityMax | LocationSuggestedCapacityMax | Value Must be greater than or equal to zero. | Stores the suggested maximum number of works that can fit in a location |
| loc-LocationWidth | LocationWidth | Value Must be greater than or equal to zero. | Stores the width of the location measured in meters. |
| loc-LocationLength | LocationLength | Value Must be greater than or equal to zero. | Stores the length of the location measured in meters |

**WorkLocations:** This is table records temporal information regarding the works’ location. This includes recording past, current, and future locations of a work. It provides a full location history for all works in the five partner museums. The table is necessary in order to obtain information on where a work was located at different times and that is why the location of a work must be temporal data. The table mainly has attributes for foreign keys which reference a location and a work and also possesses the time of arrival and time of departure for a work.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| wol-WorkCharID  (Primary Key)  Foreign Key=> references **Works** tabl | WorkCharID | Not Null;  foreign key constraints | Stores the character identifier of a museum work |
| wol-WorkNumID  (Primary Key)  Foreign Key=> references **Works** tabl | WorkNumID | Not Null;  foreign key constraints | Stores the numerical identifier of a museum work |
| wol-DatabaseEntrylocation  (Primary Key)  Foreign Key=> references **Works** table. | DatabaseEntrylocation | Not Null;  foreign key constraints | Stores the location from where the work originally entered into the new database. This was added because a DatabaseEntryLocation is primary key in the works table. |
| wol-DatabaseEntryLocationAddress  (Primary Key)  Foreign Key=> references **Works** table. | DatabaseEntryLocationAddress | Not Null;  foreign key constraints | Stores the physical address of the museum where the work first entered the system |
| wol-LocationName  (Primary Key)  Foreign Key=> references **Locations** table. | LocationName | Not NULL | Stores the name of the museum room or location (ex: museum galleries) that the work is stored or was stored in |
| wol-MuseumName  (Primary Key)  Foreign Key=> references **Locations** table. | MuseumName | Not Null;  foreign key constraints | Name of the museum that has the location where the work was or is being stored in |
| wol-MuseumLocation  (Primary Key)  Foreign Key=> references **Locations** table. | MuseumLocation | Not Null;  foreign key constraints | Stores the physical address of the museum, where the work’s location exists in |
| wol-WorkArrivalTime  (Primary Key) | WorkArrivalTime | Not Null | Stores the time and date that a museum work arrived at a specific location, required as a primary key to differentiate occasions of the same work showing up in the same location. |
| wol-WorkDepartureTime | WorkDepartureTime | None | Stores the time and date that a museum work departed from a specific location in order to know the extent that an item was in a location. |

**Doors:** This table stores non-temporal information about the pathways between rooms and/r gallery in each of the five partner museums. This table allows a location (i.e. room or gallery) to be connected to one or more other locations in the same museum. This table helps in planning exhibitions that need to use more than one location (i.e. gallery) in a museum. For every door physically in the museum between two rooms, room a and room b for this example, 2 entries are made into the database showing that there is a door leading from room a to room b and a door leading from room b to room a excluding cases of a one way door. This decision was made so that it could be specified if a door is only way because there would only be one record for the door in the database and the original location would not be reachable by the end location.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| dor-MuseumName  (Primary Key)  Foreign Key=> references **Locations** table. | MuseumName | Not Null;  foreign key constraints | Stores the name of one of the five partner museums in the database. This attribute is not need for both locations because locations in separate museums would not be connected by a door. |
| dor-MuseumLocation  (Primary Key)  Foreign Key=> references **Locations** table. | MuseumLocation | Not Null;  foreign key constraints | Stores the physical address of one of the five partner museums in the database |
| dor-LocationNameOrigin  (Primary Key)  Foreign Key=> references **Locations** table. | LocationName | Not Null;  foreign key constraints | A museum’ location name that represents the origin or start of the connection (pathway). |
| dor-LocationNameEnd  (Primary Key)  Foreign Key=> references **Locations** table. | LocationName | Not Null;  foreign key constraints | A museum’s location name that represents the end or final point of the connection (pathway). |

**Exhibitions:** This table stores basic and temporal information about all current and past exhibitions that has or will be displayed in any of the five partner museums. This information include the name of exhibitions, a brief description of the exhibition and its collection, the museum in which the exhibition has occurred, currently occurring or will occur in the future and most important the date and time in which each exhibition starts and ends. It was chosen to be temporal data because exhibitions with the same name may happen more than once so a startdate is needed to differentiate them

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| exh-ExhibitName  (Primary Key) | ExhibitName | Not Null; | Stores the name of an exhibition that is either held in one of the five museums or as a traveling exhibition abroad. |
| exh-ExhibitDescription | ExhibitDescription | None | Stores a brief description of the exhibition and the exhibition works’ collection. |
| exh-ExhibitStartDate  (Primary Key) | ExhibitStartDate | Not Null; | The start date that an exhibition is displayed to the public. |
| exh-ExhibitEndDate | ExhibitEndDate | Not Null;  Value > StartDate | The end date that an exhibition is displayed to the public. |
| exh-MuseumName  (Primary Key)  Foreign Key=> references Museums table. | MuseumName | Not Null;  foreign key constraints | Stores the name of one of the five partner museums in which the exhibition is or will be held. Was added as a primary key in order the differentiate exhibitions which may occur or have occurred at separate museums on the same date. |
| exh-MuseumLocation  (Primary Key)  Foreign Key=> references **Museums** table. | MuseumLocation | Not Null;  foreign key constraints | Stores the physical address of one of the five partner museums in which the exhibition was, currently, or will be held. |

**ExhibitionLocations:** This table stores temporal information about the locations that exhibits are, have, or will be occupying. It is temporal because it is accessing temporal data from the exhibitions table which is needed to differentiate exhibitions by the same name in the same museum. All attributes are needed to reference the exhibition and location for which an Exhibition\_Location record will refer to.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| elo-ExhibitName  (Primary Key)  Foreign Key=> references Exhibitions table. | ExhibitName | Not Null; | Stores the name of an exhibit that is either displayed in one of the five museums or as a traveling exhibition abroad. |
| elo-ExhibitStartDate  (Primary Key)  Foreign Key=> references **Exhibitions** table. | ExhibitStartDate | Not Null;  foreign key constraints | The start date that the exhibit was displayed at a certain location. Sets apart exhibitions at the same museum by the same name and also gives a timeline for when an exhibition will be run. |
| elo-LocationName  (Primary Key)  Foreign Key=> references **Locations** table. | LocationName | Not Null;  foreign key constraints | Name of a location within one of the five partner museums that an exhibit is displayed. |
| elo-MuseumName  (Primary Key)  Foreign Key=> references **Locations** table. | MuseumName | Not Null;  foreign key constraints | Stores the name of one of the five partner museums in which the exhibition will be held.. |
| elo-MuseumLocation  (Primary Key)  Foreign Key=> references **Locations** table. | MuseumLocation | Not Null;  foreign key constraints | Stores the physical address of one of the five partner museums in which the exhibition was, will be held. |

**ExhibitionWorks**: This table stores non-temporal information about the works contained in each Exhibition. Note that: Exhibitions are identified by their name, starting date and the museum they are located at. The date an item entered the exhibition is not needed because only the location is needed for security and insurance purposes so it was decided a start date for the item would not be included. All attributes are needed as they are needed to uniquely identify exhibitions and works for which the table refers to.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| elo-ExhibitName  (Primary Key)  Foreign Key=> references **Exhibitions** table. | ExhibitName | Not Null; | Stores the name of an exhibit that is either displayed in one of the five museums or as a traveling exhibition abroad. |
| elo-ExhibitStartDate  (Primary Key)  Foreign Key=> references **Exhibitions** table. | ExhibitStartDate | Not Null;  foreign key constraints | The start date that the exhibit was displayed at a certain location. |
| exw-MuseumName  (Primary Key)  Foreign Key=> references **Exhibitions** table. | MuseumName | Not Null;  foreign key constraints | Name of the museum that the exhibit was contained in. |
| exw-MuseumLocation  Foreign Key=> references **Exhibitions** table. | MuseumLocation | Part of the primary key so it cannot be null. | Physical address of the museum that the exhibition was contained in. |
| exw-WorkCharID  (Primary Key)  Foreign Key=> references **Works** table. | WorkCharID | Not Null;  foreign key constraints | Stores the character identifier of a museum work |
| exw-WorkNumID  (Primary Key)  Foreign Key=> references **Works** table. | WorkNumID | Not Null;  foreign key constraints | Stores the numerical identifier of a museum work |
| exw-DatabaseEntrylocation  (Primary Key)  Foreign Key=> references **Works** table. | DatabaseEntrylocation | Not Null;  foreign key constraints | Stores the location in which the work originally existed before it’s inserted into the new database. |

**Owners:** This table stores non-temporal information about the organizations or people who currently possess or have owned a work/item in the museum’s database. This was chosen to be one of the tables for the sake of a lack of redundancy of information. A work can have multiple owners over its existence so it is not sufficient to enter an owner attribute into the works table. An owner can also have multiple works so if there is an owner's table that a Works\_Owners table must reference then this will make finding information owned by an owner easier. If, for example, a Works\_Owners table did not have to reference a record from an owner’s table then the owner may be misspelled sometimes when being entered into the database and this would create problems when trying to access all works by the same owner.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| own-OwnerName  (Primary Key) | OwnerName | Not Null | Stores the name of a person or organization who owns or has owned any of the works in the five museums collection. Is a primary key to differentiate owners using the same email. |
| own-OwnerEmail  (Primary Key) | OwnerEmail | Not Null | Stores the email address of a person or organization who owns or has owned any of the works in the five museums collection. Included to contact the owner if needed and is primary to differentiate owners by the same name. |

**WorkOwners:** This table stores temporal information about the organizations or people who have owned or currently own a specific museum work/item in the museum database. All attributes except for OwnershipStartDate and OwnershipEndDate are required because they reference the primary keys of other tables. OwnershipStartDate is still required as a primary key however in the event that an owner purchases and sells a work multiple times.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| won-WorkCharID  (Primary Key)  Foreign Key=> references **Works** table. | WorkCharID | Not Null;  foreign key constraints | References the character identifier of a museum work. |
| won-WorkNumID  (Primary Key)  Foreign Key=> references **Works** table. | WorkNumID | Not Null;  foreign key constraints | References the numerical identifier of a museum work. |
| won-DatabaseEntryLocation  (Primary Key)  Foreign Key=> references **Works** table. | DatabaseEntryLocation | Not Null;  foreign key constraints | stores the name of the museum that first introduced the work into the system |
| won-DatabaseEntryLocationAddress  (Primary Key)  Foreign Key=> references **Works** table. | DatabaseEntryLocationAddress | Not Null;  foreign key constraints | references the physical address of the museum where the work first entered the system. |
| won-OwnerName  (primary key)  Foreign Key=> references **Owners** table. | OwnerName | Not Null;  foreign key constraints | The name of the person or organization who possesses the work at a given point in time. |
| won-OwnerEmail  (primary key)  Foreign Key=> references **Owners** table. | OwnerEmail | Not Null;  foreign key constraints | The email of the person or organization who possesses the work at a given point in time. |
| won-OwnershipStartDate  (Primary Key) | OwnershipStartDate | Not Null | The initial date that a work came into a person or organization’s possession. |
| won-OwnershipEndDate | OwnershipEndDate | None | The last date that a work was in a people possession. Helpful in determing how long a museum owned a work. Will not have values for works sold from the museum most of the time because it is not the museums business to know how long an outside owner has owned a work. It can therefore be null. |

**WorkTransactions:** This table stores temporal information about transactions involving museums’ works. Works’ transaction include purchasing a new work, loaning, borrowing, and selling a work as well as incidence of a work being donated, damaged, or when a work has gone missing.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| wtr-WorkCharID  (Primary Key)  Foreign Key=> references **Works** table. | WorkCharID | Not Null;  foreign key constraints | References the character identifier of a museum work which was involved in the transaction. |
| wtr-WorkNumID  (Primary Key)  Foreign Key=> references **Works** table. | WorkNumID | Not Null;  foreign key constraints | References the numberical identifier of a museum work which was involved in the transaction. |
| wtr-DatabaseEntryLocation  (Primary Key)  Foreign Key=> references **Works** table | DatabaseEntryLocation | Not Null;  foreign key constraints | references the location that the museum work which was involved in the transaction initially entered the database. |
| wtr-DatabaseEntryLocationAddress  (Primary Key)  Foreign Key=> references **Works** table | DatabaseEntryLocationAddress | Not Null;  foreign key constraints | references the physical address of the museum where the work first entered the system. |
| wtr-TransactionType  (Primary Key) | TransactionType | Not Null | Stores the type of transaction happened on the work. A transaction can be either purchased, loaned, borrowed, sold, gone missing, donated, or damaged |
| wtr-TransactionTime  (Primary Key) | TransactionTime | Not Null | Stores the date and time that the transaction took place. Required as a primary key in order to differentiate items which have the same kind of transaction more than once. |

**TravellingExhibitions:** This table stores temporal information about exhibitions that are set to be travelling. Each travelling exhibition must originally belong to one of our five partner museums. It was chosen to be it’s own table apart from the exhibitions table because there are attributes for the travelling exhibitions which are not required by exhibitions. A travelling exhibition is still an exhibition however, so it refers to the exhibitions table.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| tre-ExhibitName  (Primary Key)  Foreign Key=> references **Exhibitions** table | ExhibitName | Not Null;  foreign key constraints | Name of an exhibition that is sent to various locations (traveling). |
| tre-MuseumName  (Primary Key)  Foreign Key=> references **Exhibitions** table | MuseumName | Not Null;  foreign key constraints | Name of one of the five museums that is in charge of a travelling exhibition. |
| tre-MuseumLocation  (Primary Key)  Foreign Key=> references **Exhibitions** table | MuseumLocation | Not Null;  foreign key constraints | The address of one of the five museums that is in charge of a travelling exhibition. |
| tre-ExhibitStartDateOriginal  (Primary Key)  Foreign Key=> references **Exhibitions** table | ExhibitStartDate | Not Null;  foreign key constraints | The date the exhibition originally ran in any of our five partner museums. |
| tre-ExhibitDepartureDate | ExhibitDepartureDate | Not Null;  Value > than the value of ExhibitStartDateOriginal | The date the works departed from the original museum and were sent out to be displayed in a traveling exhibition. |
| tre-Security | SecurityName | None | Name of the person in charge of security for the exhibit |

**TravellingExhibitionDestinations:** This table stores temporal information about the specific locations abroad that a traveling exhibition has visited. It is included because each travelling exhibition has multiple locations for which it visits, which all have unique information. It is also temporal because a travelling exhibition can visit the same location more than once; therefore, along with the start date for the whole travelling exhibition the time spent in the destination must also be recorded and must be part of the primary key.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| ted-ExhibitName  (Primary Key)  Foreign Key=> references **TravellingExhibitions** table | ExhibitName | Not Null;  foreign key constraints | The name of the travelling exhibit that is being displayed at a certain destination |
| ted-MuseumName  (Primary Key)  Foreign Key=> references **TravellingExhibitions** table | MuseumName | Not Null;  foreign key constraints | Name of the original museum that is in charge of the exhibition. |
| ted-MuseumLocation  (Primary Key)  Foreign Key=> references **TravellingExhibitions** table | MuseumLocation | Not Null;  foreign key constraints | Physical address of the museum that is in charge of the exhibition. |
| ted-ExhibitStartDateOriginal  (Primary Key)  Foreign Key=> references **TravellingExhibitions** table | ExhibitStartDate | Not Null;  foreign key constraints | The initial date in which the traveling exhibition as a whole began being displayed to the public. |
| ted-DestinationPhone | DestinationPhone | None | The phone number of the destination that the travelling exhibition is being displayed at was included in this table as opposed to having another table with destination phone numbers for each location because a different phone number may need to be accessed every time a location is visited, even if the location is visited more than once on the same tour. |
| ted-LocationName  (Primary Key) | LocationName | Not Null | The name of the location that the travelling exhibition is being displayed at. |
| ted-LocationAddress  (Primary Key) | LocationAddress | Not Null | The physical address of the location that the travelling exhibition is being displayed at.  Note: two locations can have the same name, but they physically exists in two different addresses. |
| ted-ExhibitArrivalDate  (Primary Key) | ExhibitArrivalDate | Not Null  Value > than the value of ted-ExhibitStartDateOriginal and tre-ExhibitDepartureDate | stores the initial date that the traveling exhibit was at that location. |
| ted-ExhibitDepartureDate  (Primary Key)  Foreign Key=> references **TravellingExhibitions** table | ExhibitDepartureDate | Not Null;  foreign key constraints  Value > than the value of ted-ExhibitArrivalDate | The last date that the travelling exhibition was at a specific location. |

**TravellingExhibitionSponsors:** This table stores temporal information about the people or organizations who sponsor specific traveling exhibits. It has it’s own table because there can be multiple sponsors for a travelling exhibition so it is not appropriate to have a sponsor attribute in the travelling exhibition table. All attributes except sponsorAmount are required because they are either foreign keys which refer to other tables primary keys for the travelling exhibition or in the case of sponsor name, it is a unique identifier for

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| spo-SponsorName  (Primary Key) | SponsorName | Not Null | Holds the name of a person or organization sponsoring a travelling exhibition. |
| spo-ExhibitName  (Primary Key)  Foreign Key=> references **TravellingExhibitions** table | ExhibitName | Not Null;  foreign key constraints | The name of the travelling exhibit the sponsor is sponsoring. |
| spo-MuseumName  (Primary Key)  Foreign Key=> references **TravellingExhibitions** table | MuseumName | Not Null;  foreign key constraints | The name of the museum that is responsible for the traveling exhibition that the person or organization is sponsoring. |
| spo-MuseumLocation  (Primary Key)  Foreign Key=> references **TravellingExhibitions** table | MuseumLocation | Not Null;  foreign key constraints | The physical address of the museum that is responsible for the traveling that the person or organization is sponsoring. |
| spo-ExhibitStartDateOriginal  (Primary Key)  Foreign Key=> references **TravellingExhibitions** table | ExhibitStartDate | Not Null;  foreign key constraints | The original starting date of the traveling exhibit that the person or organization is sponsoring. |
| spo-SponsorAmount | SponsorAmount | None | The amount of money that the sponsor paid to sponsor the exhibit. Included because it is helpful to know some of the financial benefits of the travelling exhibitions. |

**WorkThemes:**  This table stores non-temporal information about keywords and concepts related to museum works/items that helps in the organization and planning of future exhibitions. A work may fit into multiple themes so this table was created and the themes are based on what conceptual qualities a piece may have such as it being “fine brush be commemoratedwork”, a theme may describe a region or era a piece comes from. Themes do not fall under the same category as Physical property or classification because they are describing things about the work which are more relative terms and do not describe anything about what type of work the piece is.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| wth-WorkCharID  (Primary Key)  Foreign Key=> references **Works** table | WorkCharID | Not Null;  foreign key constraints | stores the reference to the character identifier of a museum work. |
| wth-WorkNumID  (Primary Key)  Foreign Key=> references **Works** table | WorkNumID | Not Null;  foreign key constraints | stores the reference to the numerical identifier |
| wth-DatabaseEntryLocation  (Primary Key)  Foreign Key=> references **Works** table | DatabaseEntryLocation | Not Null;  foreign key constraints | stores the reference to the location where the work initially entered the database. |
| wth-DatabaseEntryLocationAddress  (Primary Key)  Foreign Key=> references **Works** table | DatabaseEntryLocationAddress | Not Null;  foreign key constraints | references the physical address of the museum where the work first entered the system. |
| wth-Theme  (Primary Key) | Theme | None | stores the theme that is linked to a particular work |

**WorksInsurance:** This table stores temporal information about the insurance value of a particular museum work/item, including its insurance value when traveling abroad. It shows the changes the item may experience over time and this reason it is stored as temporal data.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain Name | Attribute constraints | Attribute description |
| win-WorkCharID  (Primary Key)  Foreign Key=> references **Works** table | WorkCharID | Not Null;  foreign key constraints | references the character identifier of a museum work. |
| win-WorkNumID  (Primary Key)  Foreign Key=> references **Works** table | WorkNumID | Not Null;  foreign key constraints | References the numerical identifier of a museum work. |
| win-DatabaseEntryLocation  (Primary Key)  Foreign Key=> references **Works** table | DatabaseEntryLocation | Not Null;  foreign key constraints | references the museum which entered the work into the database originally. |
| win-DatabaseEntryLocationAddress  (Primary Key)  Foreign Key=> references **Works** table | DatabaseEntryLocationAddress | Not Null;  foreign key constraints | references the physical address of the museum which entered the work into the database originally |
| win-WorkInsureValue | WorkInsureValue | Monetary insurance based so it must be greater than or equal to 0. | The insurance value that a work held at a particular point in time. |
| win-WorkTravelInsureValue | WorkTravelInsureValue | Based on the assignment. If no value is entered the default value is 110% of the original value of the work but this value may be higher. | The insurance value that a work held at a particular point in time if it was traveling. |
| win-InsureStartDate  (Primary Key) | InsureStartDate | Part of the primary key so it cannot be null. | The initial date that a work held a particular insurance value. |
| win-InsureEndDate | InsureEndDate | NONE | The last date that a work held a particular insurance value. |

Primary Key win-WorkCharID, win-WorkNumID, win-DatabaseEntryLocation, win-DatabaseEntryLocationAddress, win-InsureStartDate

Foreign key win-WorkCharID, win-WorkNumID, win-DatabaseEntryLocation, win-DatabaseEntryLocationAddress references Works.

###### **Temporal Data**

Below is a detailed explanation of information pertaining to a timeframe for data and the benefits of having information for the given data at every point in time organized by tables requiring temporal data.

**Work location**

Keeps track the work location including the history location, current location and future planned location. A datetime attribute is necessary because the exact time of an item needs to be known. When works are moved between locations in the same museum a trigger will change the end date of the last location to the start date of the new location because moves within a museum are short. The same trigger will not apply for moves between museums.

|  |  |
| --- | --- |
| Past | Keeps track of all history relating to the history of a piece of work. This can be useful in situations of security and insurance where a work could have been harmed or gone missing. |
| Current | Records the current location of a piece of work. This is useful for identifying the location of any work in the museum's database and can aid in planning exhibitions. |
| Future | Records where works are planned to be at future dates. This can aid in planning of exhibitions and other functions because all advance notice of where items should be can be accessed. |

**Exhibitions**

An exhibition is temporal data because exhibitions that have the same name. The start date of an exhibition is therefore recorded as a primary key to differentiate between these exhibitions.

|  |  |
| --- | --- |
| Past | Keeps a detailed history for the dates of past exhibitions which can aid in planning of future exhibitions. |
| Current | Keep track of all information pertaining to exhibitions which are ongoing. In the case of travelling exhibitions, data which is current can aid in control of the status of the travelling exhibition, such as the need to locate the current point of a travelling exhibition. This can also be helpful for visitors of the museum to know what is ongoing. |
| Future | Visitors appreciate being able to access information about exhibitions planned in the future and museum staff can benefit from knowledge of future exhibitions. |

**Exhibition locations** (references exhibition and location)

This data is temporal by extension of it referencing the exhibition table

|  |  |
| --- | --- |
| Past | Can answer questions of interest for which exhibition was occurring in which location. |
| Current | Gives better a more detailed handling of locations which are currently out of use due to exhibitions |
| Future | Tracks when locations are booked for future exhibitions or travelling exhibitions and can aid in museum planning of resources. |

**Exhibition works**

|  |  |
| --- | --- |
| Past | Tracks when works have been booked for the benefit of consideration of future exhibitions or travelling exhibitions and can aid in planning of museum resources. |
| Current | Gives better a more detailed handling of works which are currently out of use due to exhibitions |
| Future | Tracks when works are booked for future exhibitions or travelling exhibitions and can aid in museum planning of resources. |

**Works owner**

|  |  |
| --- | --- |
| Past | The owner of a piece of work is recorded when the museum is entered into the museum database. When transferring owners it can be helpful to know who has owned it last for security and insurance reasons. |
| Current | Keeps track of the work’s owner and gives control and organization over locating a work. |

**Work transactions**

|  |  |
| --- | --- |
| Type of Transaction | Detail |
| Buy and Sell | When one of the museums buys or sells a work, at any point of time it is necessary to be able to access this information for security, insurance, and accounting purposes. For this reason it is necessary to be able to query all instances of purchases or sales. |
| Loan and Borrow | All information for loans and borrowing can be accessed at any point and will be helpful for security and insurance purposes. All loans and works borrowed should be recorded to be able to easily query for how many loans are occurring. |
| Missing | The date when a work was first noticed to be missing is very helpful for many reasons relating to history about the work. In this case the owner of the work would not automatically change. It can be helpful to query a history of when items have gone missing for security purposes. |
| Donated | Useful in determining how a work came into possession of the museum if for example the donor should whether some day for their contribution to the museum. In the case of a donation the previous owner should be recorded with the same date the piece was given to the museum as the actual date the donor received the work may not be accessible. |
| Damaged | Records all points in time for which an item was noticed to be damaged and can help in determining what methods of organization of the works and other factors can contribute to the damaging of a work and possibly changes in insurance value of a work. It can be helpful to be able to query when a work was damaged for security purposes |

**Traveling Exhibits**

Travelling exhibitions are temporal data for the same reason that exhibitions are temporal data, because a travelling exhibition by the same name can occur more than once and the dates attached to the primary key help to differentiate the travelling exhibitions.

**Travelling destinations**

Destinations are temporal data because the same destination can be travelled to within the same travelling exhibition, even on the same tour and the dates of arrival at these destinations need to differentiate them.

Which tables are needed to add new temporal (improve design), why

1. For the WorkLocations table
   1. We need to add a timestamp (i.e. time) to the moving in date and moving out dates. This will allow our database not to not only record a more accurate time for when an item is inserted in a location or moved out of it.
   2. A detailed date-time works location history is important for museum administrators and insurance companies for the purpose of tracking the location of an item minute by minute
2. For the Exhibitions table we need to add temporal information about the time an exhibition starts and time it ends. This temporal data is helpful for the museum visitors to know what times they can expect to visit the exhibitions and what times it closes.
3. For the ExhibitionWorks table we need to to add time to the attributes containing date.

###### **Views**

There are a number of benefits to implementing views in this database

**A view could hide complexity**

Which means if we need to query multiple tables or perform complex logic or calculations to get a result we can simply code the logic into a view. Views can also improve the security of a database by limiting access to certain information depending on the responsibilities and privileges each person has pertaining to the database.

**Views can perform a task**

Listed below are a number of task views of database can have.

**Task views**:

1. Listing of all publicly available data on all works in each exhibition sorted by the exhibition and by the name of the work
2. Listing of works sorted by when they are available for use in a new exhibition, by classification, and by the name of the work
3. Listing of the amount of additional works that could be added to each exhibition based on the unused capacity of the galleries that they are currently in. This listing should just have the amount of additional works that could be added and the name of the exhibit,
4. Listing all current and future exhibitions including the name of the exhibition, the dates it is happening (starting and ending dates), the maximum recommended capacity of the locations it is using, and amounts of works planned for it
5. Listing the different locations that a given work was/is/will be in between two dates
6. Listing all the exhibitions that make use of a location between two dates

And save those into view for further usage

Below is a list of views which can be used by different roles of people accessing the database which can create better security for the database.

**View could be used as security mechanism**

by allowing users to access data through the view, without granting the user permission to directly access the underlying tables or information the user does not need



( Figure 4 )

The merged museum database is used by different people with different responsibilities and privileges relating to it. People are able to view different information from the database depending on their role. Roles people may have pertaining to the database can be employee roles such a patron, IT, security, or building maintenance or roles of an outsider such as a visitor and sponsor of the travelling museum. Each person does not require access to all information and some information can be dangerous to freely give out, such as allowing a visitor to view insurance values of a piece of work. Below is a description of roles and what this person may access pertaining to the database as shown in figure 4..

|  |  |  |  |
| --- | --- | --- | --- |
| Role | Role Description | View Name | View Detail |
| Patron | Museum manager | PatronView | The patron is the museum manager and has privileges of accessing all information from the database containing information of all 5 partner museums. The patron will not want to however, deal with the underlying tables. |
| IT | It a representative of the IT Department which maintains things like the museum website or app, | ITView | IT need access to all parts of the database for the develop of software such as the website of the museum, developing applications and managing the database itself. |
| Security/Guard | Is a representative of the security department who are responsible for safety in the museum and items which belong to the museum. | SecurityView | Security are restricted to viewing information on locations, exhibitions, items, and visitors including travelling exhibitions in order to maintain safety and to have the ability to access and provide information in the case of an unfortunate incident. |
| Building maintenance | Is representative of the maintenance department and may not be an employee of the museum, but should still have access to information about the location and doors to perform maintenance | BuildingMaintenance | The building maintenance are restricted to accessing and editing information on the locations and doors because they only need to perform maintenance on the physical structures of the building itself. |
| Visitor | Are representatives of a group people who visit the museum or should not be able to edit or create information but should be able to view it. | VisitorView | Visitors are people who visit the museum, and view online sources such as a website or museum app which access the database. These people have limited authority and cannot edit or create information but are able to view information on pieces of work, exhibition, and locations of the museums with the exception of private information like the insurance value of a work and current and upcoming exhibitions. |
| Sponsor | It a representative of a group people who sponsor or support our museum by sponsoring one or more traveling exhibition (s) | SponsorView | Sponsors support the partner museums by sponsoring travelling exhibitions. These people therefore can have access to information relating to the travelling exhibitions and also pieces of work in case they want further input on what works are involved in a travelling exhibition. |

###### **Future Consideration**

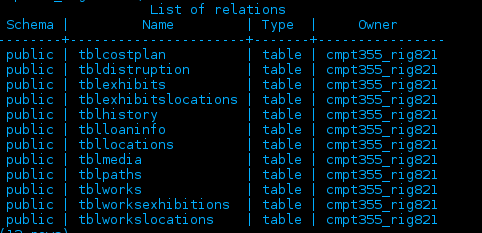
Thinking of database design can be a heavy subject, and the goal of the team database is for the database to be efficient and scalable.

Indexes can be created for frequently used queries on big tables, which will improve the speed of data retrieval operations.

**Individual Reports**

**Database Migration Report- Richard Granger: Canadian Museum of Science And Technology**

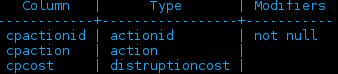
The following is a report on how I plan to adjust my database to fit the new amalgamated museum database design that our group has developed. As my database stood prior to the group project, it contained 12 tables, as evidenced here:



In this report I will examine the structure of each of the tables in my current database, and will explain how I intend to migrate the data into the new database design. In the last section I will also explain my tests table by table.

**Table 1: tblcostplan**

**The following was the original structure of this table:**



**Purpose:**

This table was designed for storing alternate options for calculating distruption costs in regard to planning exhibits.

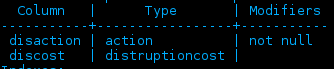
**Future Plans:**

I designed this table for the bonus part of assignment 3, and since it does not contain any useful data, the table will remain part of the legacy database and no information will be carried forward to the new system.

Testing: No testing is required since no data is being moved forward.

**Table 2: tbldistruption**

**The following was the original structure of this table:**



**Purpose:**

This table was designed for storing the cost of each action with regard to stopping or postponing exhibitions with regards to the planning of disruption costs of new exhibitions.

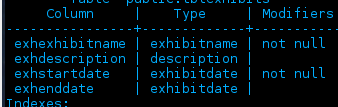
**Future Plans:**

This table was designed for the bonus section of assignment 3, and since it does not contain any useful data, will not be moved forward to the new database. All data in this table will remain in the old database exclusively.

Testing: No testing is required since no data is being moved forward.

**Table 3: tblExhibits**

**The following was the original structure for this table:**



**Purpose:** This table was designed to hold the information related to the exhibitions that my museum had, whether they were showcased within the museum or abroad as a travelling exhibition.

**This table will primarily be moved to the exhibitions table:**

Exhexhibitname- this attribute is the name of the exhibit, and so we can simply move over this field into the new exh-ExhibitName attribute of the unified database.

Exhdescription- this attribute is the description of the exhibit and can be simply moved over to exh-ExhibitDescription. They are the same datatype and should have no trouble moving with the names.

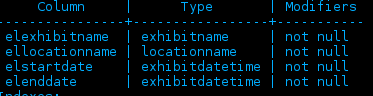
exhStartDate- this attribute is the starting date of the exhibit and can also be moved forward to exh-ExhibitStartDate. They are similiar datatypes so transferring them should be easy.

Exhenddate- this attribute is the end date of an exhibit, and it shares the similar data type, so it can easily be transferred to the exh-ExhibitEndDate field in the new database structure.

**Testing**: To test this migration, we will produce a report of the old tblExhibits and the new Exhibits table. Their structures are almost identical so an examination of the two tables will show whether the data transferred successfully. We simply create a report of both tables and the data of the two tables will be exactly the same.

**Table 4: tblExhibitsLocations**

The following was the original structure for this table:



Purpose: This table stored the locations of the exhibits that my museum displayed, including the locations of travelling exhibitions.

This table will be moved to the Exhibitions\_Locations table in the unified database, with the elo-MuseumLocation, and elo-MuseumName which did not exist in my own database, will be added to the new table manually by referencing the new Museums table. As for the attributes in my current table:

elexhibitname-This is the name of the exhibition and can simply be moved over to the new table easily

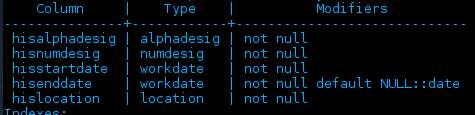
ellocationname-This is the name of the location that the exhibit was at that location. This can simply moved manually to the new table, as the names of the locations are still the same.

Elstartdate, elenddate- This two attributes can simply be moved to the new table, but since it is a timestamp instead of a date, the time may need to be adjusted to ensure that no data is simply created.

Testing: To ensure that this data has been moved forward successfully, we can create a report of the table from the old database, and a table from the new database with specific examples of the locations. Since the data is the same, the extra data about the museums will have to be included in the report due to the new attributes in the unified database.

**Table 5: tblHistory**

**The following was the design of this table in my database:**



**Purpose:** This table was intended to store the historical locations of the works.

This table will primarily be moved to the Works\_Locations of the unified database. The DatabaseEntryLocation and DatabaseEntryAddress will have to be added manually here since they did not exist in our database.

Hisalphadesig, hisnumdesig- these two identifiers of the primary key of a work can simply transferred over to the Works\_Locations table of the new database

hisStartDate, hisEndDate- These can be moved over to the wol-WorkArrivalTime, and wol-WorkDepartureTime, but like the others, the new attribute is a timestamp so the times may need to be adjusted as necessary.

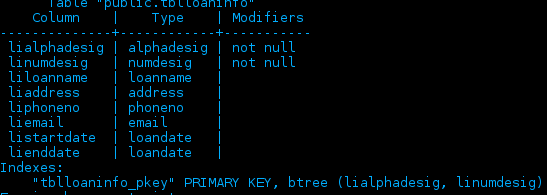
hisLocation- Our museum used a character identifier for locations, so this data will have to be substituted by hand with the actual location name from the locations table.

All other attributes that exist in the new database design will be added from their foreign tables since they are data that are added elsewhere.

**Testing:** To test this we will create a report that will generate the information of specific works locations in both the original and new tables. The transferred data must match.

**Table 6: tblLoanInfo**

**The following was the design of this table in my database:**

**Purpose**: This table in my original database stored the loan info of works and when and who they were loaned to.

**The following changes will be made:**

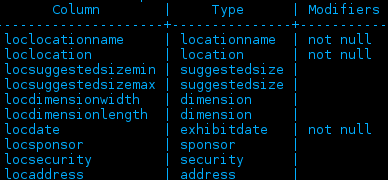
Lialphadesig, linumdesig-these will already be in the works table of the new database, so they will not be moved forward.

Liaddress, liphoneno- this are not used as part of the new database, so they will remain in the original database but they will not move forward.

The other data will remain in the old database, but will be moved to other tables as necessary.

**Tests**: We will produce a report that queries the proper tables to get the information on works where the wor-DatabaseEntryLocation is our museum, and compare the results of that report to the results of a report of the loans from the old database.

**Table 7: tblLocation**

This was the structure of the locations table within the database of our museum:

**Purpose:** This table in our database stored the information on locations both within our museum and locations that were visited by.

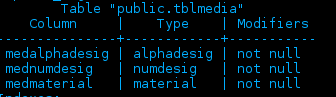
**The following changes will be made:**

Loclocationname, locsuggestedsizemin, locsuggestedsizeman, locdimensionwidth, and locdimension length can easily be moved to the new Locations table, with additional manual input of data where necessary, such as for the museum name.

The locations within this table that are not in our museum, we have agreed to move all appropriate data to the Travelling\_Destinations table of the new database. This will have to be done by manually going over the table and moving the proper location to the proper table

**Testing**: Testing this migration will be a tad more complex, but it will consist of two different tests, we will produce reports of the appropriate locations from both the communal database Locations table, and the Travelling\_Destinations table. We will then compare the transferred data to the data in the original database.

**Table 8: tblmedia**

The following were the attributes contained in this table in our museum’s database:

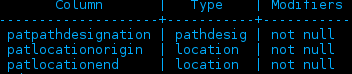
**Purpose:** This table stored the medium that the works in the museum were made out of, basically the same function as the Work\_Mediums table in the database.

**The Following changes will be made**:

Medalphadesig, mednumdesig, medmaterial all correspond to wme-WorkCharID, wme-WorkNumID, and wme-WorkMedium in the Work\_Medium table. The additional fields in the new table will be filled in manually with the proper data as necessary, such as museum location and museum address.

**Testing:** This will consist of creating a report that will pull the identifier of a work and the name of the material that the work is made of. We can then compare the two reports and if they are the same we know that the transfer was conducted successfully.

**Table 9: tblpaths**

The following were the attributes contained in this table of our museum’s database:

**Purpose**: This table was responsible for storing the information about the connections that existed between rooms in our own museum.

**The following changes will be made:**

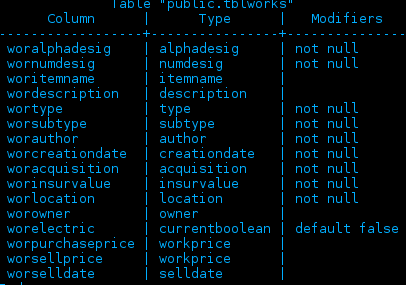
Patpathdesignation was a numerary indicator of path, it was redundant data and since it is not used and also redundant there is no need for it to be carried forward.

Patlocationorigin and patlocationend correspond to lcn-LocationNameOrigin and lcn-LocationNameEnd of the Doors table and thus can be moved forward by manually referring to the actual names of the locations.

**Testing**: To test the successful transfer of data, we will produce a report by querying the new database for the doors that correspond to our museum. We will then cross-reference that information with our original data to ensure that everything transferred successfully.

**Table 10: tblworks**

The following was the design for the works table in our museum:



**Purpose:** This table contained the data on the museums works, as well as other various data about them.

**The following changes will be made:**

Woralphadesig, wornumdesig- these attributes will be the wor-CharID, and wor-NumID in the new works table in the unified database. Same data type so simply moving them over should not be a problem.

Wor-itemname, wordescription- these can be moved as well to the Works table as wor-WorkName and wor-WorkDescription, a simple copy-over and they will move without issue.

Wortype, worsubtype- these classified the types and subtypes of my works, but the unified database has a new unified system of categories, so these attributes will not be moved forward and instead I will have to reclassify every work manually.

Worauthor, worcreationdate-these can be moved to the Works table as the attributes wor-WorkCreator and wor-CreationDate. The datatype of the date is different so the dates may need to be adjusted manually to ensure that there is no loss of data

Worinsurevalue-this will be moved manually to the Works\_Insurance table and placed with the appropriate dates into the table.

worLocation-This was old data that was made redundant by the history table in our database, so there is no need to move it forward.

worowner-This data will have to be manually moved to the Owners table and the dates looked up and put in the right place.

worelectric-This attribute was the custom one produced for assignment 3, and it has been amalgamated into the physical properties of the new database structure. The works that had it will have to be manually adjusted, but none of this data needs to be carried forward.

Worpurchaseprice, worselldate, worsellprice-The worselldate will be added as a change in ownership in the Works\_Owners table. Worselldate will be used as a basis, and I will adjust any data as necessary. Worsellprice is not included in the new database and will not be moved forward.

**Testing**: To test the successful transfer of data the following must be done:

-Create a report on the three works that my museum has sold from the WorksOwners table and compare that to worselldate from the old database.

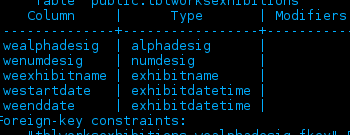
-Create a report on the works that feature electric in their property and make sure that they line up with the works that have the electric attribute in the old database.

-Query the Works\_Owners table and the works table to ensure that the works and their owners current and previous line up properly and the data is the same.

-Query the works\_insurance table and the Works table and do the same to the old tblworks and compare the two results to ensure that they are the same

**Table 11: tblworksexhibitions**

The following is the structure of the table in our database:



**Purpose:** This table was responsible for storing the information on the works that belonged to each exhibit.

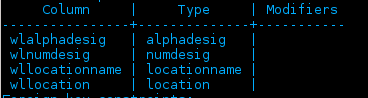
**The following changes will be made:**

Wealphadesig, wenumdesig-This data will already exist in the works table so there is no need to retransfer it again as it would just cause errors.

Weexhibitname, weexhibitstartdate, weexhibitenddate- These attributes will have already been transferred (already exist in the exhibits table) so I can rebuild the data here in the new database within the references by using the proper manual insert statements.

**Testing**: Once we have rebuilt the exhibits and works together in the new table, we will create a report and compare the items of the first exhibition that our museum had, and create the same report in the original database and compare the two reports.

**Table 12: tblworkslocations**

The design of this table in my database was as followed

**Purpose:** This table was originally designed to replace the tblhistory table, but was not fully implemented, and thus contains no useful data of any kind

**The following changes will be made**:

No changes need to be made as the table is functionally similar to tblhistory and it contains no data. Nothing needs to be carried forward.

**Testing**: Nothing was moved so nothing needs to be tested.

**Additional Notes:**

-In the museums table I will manually add the data for our museums.

-Each change to a work, throughout all the tables will have to be added manually to the works\_transactions table.

Team 8 Individual Report

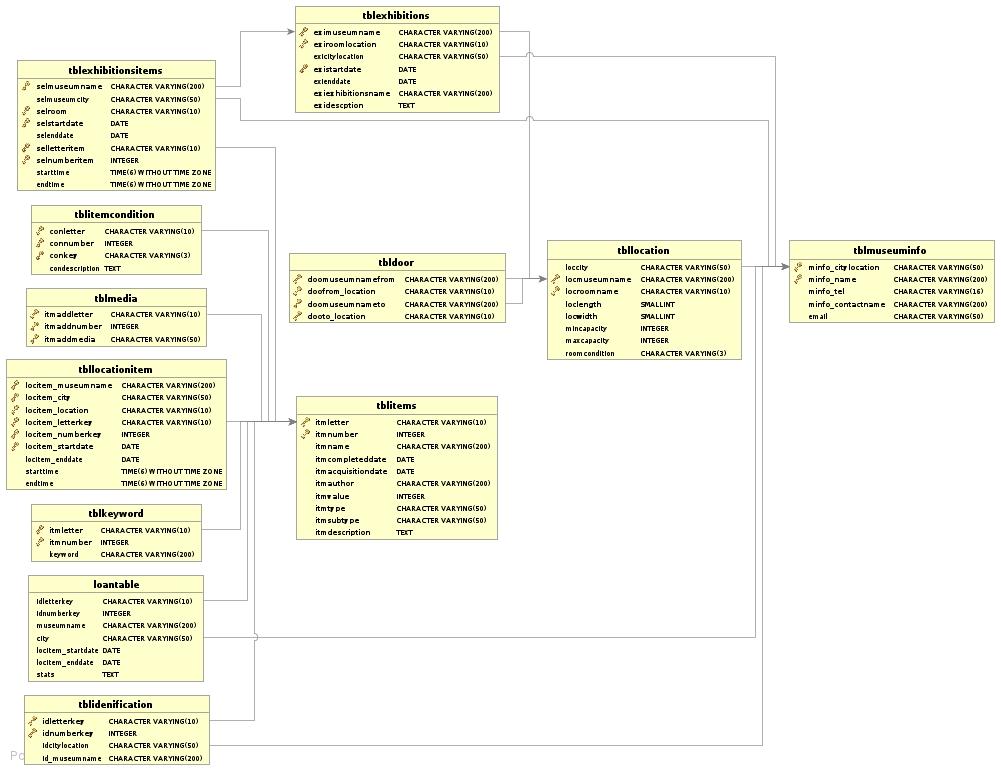
## Name: Zang, JiaWei

## NSID: 11148860

Team Database Structure



My Database Structure



### Difference between database

Core functionalities

Works

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| wor-WorkCharID | My database has those but in different name | NONE | NONE |
| wor-WorkNumID | My database has those but in different name | NONE | NONE |
| wor-DatabaseEntryLocation | My database Does not have it | When I insert works information into team database on this table from query museum information, I need to added the location information | Query this data from my information and check does it have database entry location which is same with my museum name. |
| wor-DatabaseEntryLocationAddress | My database Does not have it | When I insert works information into team database on this table, I need to added the location address information | Query this data from this table and check does it have database entry location which is same with my museum address. |
| wor-description | My database has those but in different name | NONE | NONE |
| wor-PhysicalProperty | My database has those but in different name.  However, the standard is different | Need to change my type fit into the physical property standard | Query this column and check this result are all in the standard.  It should be painting or sculpture. |
| wor-workClassification | My database has those but in different name.  However, the standard is different | Need to change my type fit into the Classification  standard | Query this column and check this result are all in the classifications standard |
| wor-DateCreated | My database has those but in different name | NONE | NONE |
| wor-InsuranceValue | My database has those but in different name | NONE | NONE |
| wor-WorkBorrowable | Does not have this attributes. | When I insert my data, I need to check the loanwork table which the table contain all the loan works | Check this out the loanable items are all in the loanwork table in my original database |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing or wrong value in the query result, make sure they are all there, and make sure they are all valid by evaluating the result. | | |

WorkMediums

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| wme-WorkCharID references Works (WorkCharID) | My database has those but in different name | NONE | NONE |
| wme-WorkNumID references Works (WorkNumID) | My database has those but in different name | NONE | NONE |
| wme-WorkDatabaseEntryLocation references Works (DatabaseEntryLocation) | Does not have this attributes. | When I insert works information into team database on this table, I need to added the location information | Query this table to check this table have item with char and number have same name with my museum name |
| wme-DatabaseEntryLocationAddress  references Works (DatabaseEntryLocationAddress)  wme-Medium | Does not have this attributes. | When I insert works information into team database on this table, I need to added the location address information | Query this table to check this table have item with char and number have save with my museum address |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

WorkThemes

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| wth-WorkCharID references Works (WorkCharID) | My database has those but in different name | NONE | NONE |
| wth-WorkNumID references Works (WorkNumID) | My database has those but in different name | NONE | NONE |
| wth-WorkDatabaseEntryLocation references Works (DatabaseEntryLocation) | Does not have this attributes. | When I insert works information into team database on this table, I need to added the location information | Query this date to check this table have item which are belong to me with char and number have museum name |
| wth-DatabaseEntryLocationAddress  references Works (DatabaseEntryLocationAddress) | Does not have this attributes. | When I insert works information into team database on this table, I need to added the location address information | Query this data to check this table have item which belong to me with char and number have museum address |
| wth-Theme | Does not have this attributes. | Need to copy what I have in works table and insert into this table. | Query the item belonged to me, and make sure that my original themes are changed into standard theme. |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

WorkLocations

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| won-WorkCharID references Works (WorkCharID) | My database has those but in different name | NONE | NONE |
| wol-WorkDepartureTime | My database has those but in different name | NONE | NONE |
| won-WorkNumID references Works (WorkNumID) | My database has those but in different name | NONE | NONE |
| won-WorkDatabaseEntryLocation references Works (WorkDatabaseEntryLocation) | Does not have this attributes. | When I insert works information into team database on this table, I need to added the location information | Query this data to check this table have item which are belong to me with char and number have museum name |
| won-DatabaseEntryLocationAddress  references Works | Does not have this attributes. | When I insert works information into team database on this table, I need to added the location address information | Query this data to check this table have item which belong to me with char and number have museum address |
| wol-WorkArrivalTime | My database has those but in different name | NONE | NONE |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

ExhibitionWorks

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| exw-ExhibitName references Exhibitions (ExhibitName) | My database has those but in different name | NONE | NONE |
| exw-MuseumName references Exhibitions (MuseumName) | My database has those but in different name | NONE | NONE |
| exw-Museumlocation references  Exhibitions (MuseumLocation) | My database has those but in different name | NONE | NONE |
| exw-ExhibitStartDate references  Exhibitions (ExhibitStartDate) | My database has those but in different name | NONE | NONE |
| exw-WorkCharID references Works (WorkCharID) | My database has those but in different name | NONE | NONE |
| exw-WorkNumID references Works (WorkNumID) | My database has those but in different name | NONE | NONE |
| exw-WorkDatabaseEntryLocation references Works (DatabaseEntryLocation) | Does not have this attributes. | When I insert works information into team database on this table, I need to added the location information | Query this data to check this table have item which are belong to me with char and number have museum name |
| exw-DatabaseEntryLocationAddress  references Works (DatabaseEntryLocationAddress) | Does not have this attributes. | When I insert works information into team database on this table, I need to added the location address information | Query this data to check this table have item which belong to me with char and number have museum address |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

Museums

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| mus-museumName | My database has those but in different name | NONE | NONE |
| mus-museumLocation | My database has those but in different name | NONE | NONE |
| mus-phoneNumber | My database has those but in different name | NONE | NONE |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

Locations

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| loc-LocationName | My database has those but in different name | NONE | NONE |
| loc-MuseumName | My database has those but in different name | NONE | NONE |
| loc-MuseumLocation | My database has those but in different name | NONE | NONE |
| loc-LocationSuggestedCapacityMin | My database has those but in different name | NONE | NONE |
| loc-LocationWidth | My database has those but in different name | NONE | NONE |
| loc-LocationLength | My database has those but in different name | NONE | NONE |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

Door

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| dor-LocationNameOrigin | My database has those but in different name | NONE | NONE |
| dor-LocationNameEnd | My database has those but in different name | NONE | NONE |
| dor-MuseumName references Exhibitions (MuseumName) | My database has those but in different name | NONE | NONE |
| dor-Museumlocation references  Exhibitions (MuseumLocation) | My database has those but in different name | NONE | NONE |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

ExhibitionLocations

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| elo-ExhibitName references Exhibitions (ExhibitName) | Does not have this attributes.  Because I stored the information about exhibition location in exhibition table | My exhibition location information is stored in exhibition table, mean I need move the exhibition name information into this table, when I insert my information in team database | Query from my original database, and check the exhibition table check they are same in exhibition location table |
| elo-MuseumName references Exhibitions (MuseumName) | Does not have this attributes.  Because I stored the information about exhibition location in exhibition table | My exhibition location information is stored in exhibition table, mean I need move the museum name information into this table, when I insert my information in team database | Query form the museum name table to check this is same with the new team database.  And query this and check this with query from my exhibition table to get the name and make sure they are same |
| elo-Museumlocation references  Exhibitions (MuseumLocation) | Does not have this attributes.  Because I stored the information about exhibition location in exhibition table | My exhibition location information is stored in exhibition table, which is mean I need move the museum location information into this table, when I insert my information in team database | Query form the museum location table to check this is same with the new team database in museumlocation |
| elo-ExhibitStartDate references  Exhibitions (ExhibitStartDate) | Does not have this attributes.  Because I stored the information about exhibition location in exhibition table | My exhibition location information is stored in exhibition table, which is mean I need move the start date information into this table, when I insert my information in team database | Query from the exhibition table with the same format in new team database, and check this is same with the query result from start date exhibition table |
| elo-LocationName references Locations(LocationName) | Does not have this attributes.  Because I stored the information about exhibition location in exhibition table | My exhibition location information is stored in exhibition table, which is mean I need move the location name information into this table, when I insert my information in team database | Query the data which what are mine, and check this is same with the query the data which are in exhibition table. |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

Exhibitions

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| exh-ExhibitionName | My database has those but in different name | NONE | NONE |
| exh-MuseumName | My database has those but in different name | NONE | NONE |
| exh-Museumlocation | My database has those but in different name | NONE | NONE |
| exh-ExhibitStartDate | My database has those but in different name | NONE | NONE |
| exh-ExhibitEndDate | My database has those but in different name | NONE | NONE |
| exh-Description | My database has those but in different name | NONE | NONE |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

Transactions



Works

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| wor-WorkCharID | During database has those but in different name | NONE | NONE |
| wor-WorkNumID | My database has those but in different name | NONE | NONE |
| wor-DatabaseEntryLocation | My database Does not have it | When I insert works information into team database on this table, I need to added the location information | Query this information from this information and check does it have database entry location which is my museum name. |
| wor-DatabaseEntryLocationAddress | My database Does not have it | When I insert works information into team database on this table, I need to added the location address information | Query this information from this information and check does it have database entry location which is my museum address. |
| wor-description | My database has those but in different name | NONE | NONE |
| wor-PhysicalProperty | My database has those but in different name.  However the standard is different | I need change my type fit into the physical property standard | Query this column and check this result are all in the standard.  It should be painting or sculpture. |
| wor-workClassification | My database has those but in different name.  However the standard is different | I need change my type fit into the Classification  standard | Query this column and check this result are all in the classifications standard |
| wor-DateCreated | My database has those but in different name | NONE | NONE |
| wor-InsuranceValue | My database has those but in different name | NONE | NONE |
| wor-WorkBorrowable | Does not have this attributes. | When I insert my data, I need to check the loanwork table which the table contain all the loan works | Check this out the loanable items are all in the loanwork table in my original database |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

WorkOwners

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| won-WorkCharID references Works (WorkCharID) | My database has those but in different name. | NONE | NONE |
| won-WorkNumID references Works (WorkNumID) | My database has those but in different name. | NONE | NONE |
| won-WorkDatabaseEntryLocation references Works (WorkDatabaseEntryLocation) | I don’t have this attribute, therefore, I need to create this attribute | When I insert works information into team database on this table, I need to added the location information | Query this table, and check out the museum name information about the item, check out the is this my museum name |
| won-DatabaseEntryLocationAddress  references Works (DatabaseEntryLocationAddress) | I don’t have this attribute, therefore, I need to create this attribute | When I insert works information into team database on this table, I need to added the location address information | Query this table, and check out the museum address information about the item, check out the is this my museum address |
| won-ownerName references Owners (ownerName) | I don’t have this attribute, therefore, I need to create this attribute | Add new attribute change the institution name into this attribute | Query this modification attributes and check out this is valid name. |
| won-ownerEmail references Owners (ownerEmail) | I don’t have this attribute, therefore, I need to create this attribute | Create new data information about the email | Query this data which is new data that I originally don’t have this information, and make sure those are email address |
| won-OwnershipStartDate | I don’t have this attribute, therefore, I need to create this attribute | Create new date into new table when I insert into this team database | Query this data which is new data, that I originally don’t have this information, and make sure those are start date |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

Owner

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| own-ownerName | I don’t have this table, but I save this owner information as institutions | Needed to change to valid name when we insert into team database | Query this data, and compare with the owner result in works owner table. |
| own-ownerEmail | I don’t have this table, I save this owner information as institutions with name and address | Needed to create email information when I insert this data into team database | Query this data, and compare with the owner result in works owner table. |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

WorkTransactions

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| wtr-WorkCharID references Works (WorkCharID) | My table to contain this information is in item condition table | Needed to select this information from condition table and insert into transactions table. | Query this form my item table and make sure those are my work |
| wtr-WorkNumID references Works (WorkNumID) | My table to contain this information is in item condition table | Needed to select this information from condition table and insert into transactions table. | Query this form my item table and make sure those are my work |
| wtr-WorkDatabaseEntryLocation references Works (WorkDatabaseEntryLocation) | This table does not have this attribute | When I insert works information into team database on this table, I need to added the location information | Query this from mine in museum information table museum name is same value, and make sure thosesure those are same |
| wtr-DatabaseEntryLocationAddress references Works (DatabaseEntryLocationAddress) | This table does not have this attribute | When I insert works information into team database on this table, I need to added the location address information | Query this from my museum table which called museum address is same value, and make sure those are same |
| wtr-TransactionType | My database has those but in different name. | NONE | NONE |
| wtr-TransactionTime | My database has those but in different name. And this is in loan table | Need to query this from loantable to get the start date into this table. | Query this data and make sure they are all same date with loan table in my original table |
| wtr-Enddate | My has those but it attribute in item table, and item condition table | Need to query this from location item table and item condition which is borrowed items to get the end date of the borrow items end date | Query this data and query the tblitemcondition table to find the borrowed items and find those item in location item table and find the end date to make sure those are the same. |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

## 

## Traveling Exhibition

TravellingExhibitions

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| tre-ExhibitName references Exhibitions (ExhibitName) | I don’ t have this table, because I stored this traveling exhibition information into the exhibition table | Query this from exhibition table which is traveling exhibition and insert into team database | Query this data and compare the traveling exhibition which in the exhibition table, and make sure they are same |
| tre-Security | I am missing this information, therefore, I need to add security name.  Because I save this information in museum information take | Add security name into the team database by make the name up. | Query this information and compare with the name in the table of museum information table, and make sure they don’t exist in the table, because my original database does not have those name information about security |
| tre-MuseumName references Exhibitions(MuseumName) | I don’t have this in the table but in different name in different table. | NONE | NONE |
| tre-MuseumLocation references  Exhibitions(MuseumLocation) | I don’t have this in the table but in different name in different table. | Query this from exhibition table which is traveling exhibition and insert into team database | Query this data and compare with the exhibition table which exhibition are traveling exhibition, and make sure they are same. |
| tre-ExhibitStartDateOriginal references Exhibitions(ExhibitStartDate) | I don’t have this in the table but in different name in different table. | Query this from exhibition table which is traveling exhibition and insert into team database | Query this data and compare with the start date in the exhibition table (which are traveling exhibition), and make sure they are same |
| tre-ExhibitDepartureDate | I don’t have this in the table but in different name in different table. | Query this from exhibition table which is traveling exhibition and insert into team database | Query this data and compare with the end date in the exhibition table (which are traveling exhibition), and make sure they are same |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

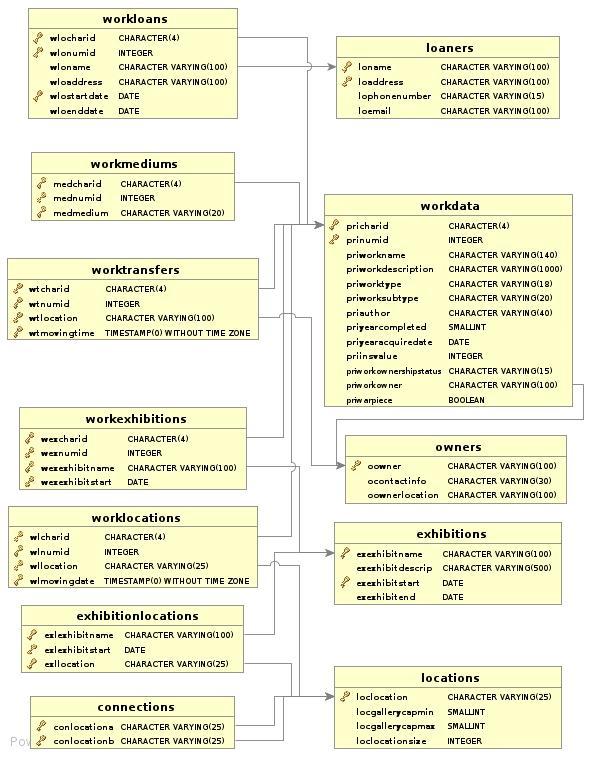
TravellingExhibitionSponsors

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| spo-ExhibitName references TravellingExhibitions(ExhibitName) | I don’t have this table in my database design, but this attribute is in exhibition table | Needed to query from exhibition table which is traveling, and insert this into team database. | Query this data and compare this with exhibition table which are traveling exhibition and make sure they are all same |
| spo-MuseumName references TravellingExhibitions (MuseumName) | I don’t have this table in my database design, but this attribute is in exhibition table | Needed to query from exhibition table which is traveling, and insert this into team database. | Query this data and compare this with exhibition table which are traveling exhibition and make sure they are all same |
| spo-MuseumLocation references TravellingExhibitions (MuseumLocation) | I don’t have this table in my database design, but this attribute is in exhibition table | Needed to query from exhibition table which is traveling, and insert this into team database. | Query this data and compare this with exhibition table which are traveling exhibition and make sure they are all same |
| spo-ExhibitStartDateOriginal references TravellingExhibitions (ExhibitStartDateOriginal) | I don’t have this table in my database design, but this attribute is in exhibition table | Needed to query from exhibition table which is traveling, and insert this into team database. | Query this data and compare this with exhibition table which are traveling exhibition and make sure they are all same |
| spo-ExhibitDepartureDate references TravellingExhibitions (ExhibitDepartureDate) | I don’t have this table in my database design, but this attribute is in exhibition table | Needed to query from exhibition table which is traveling, and insert this into team database. | Query this data and compare this with exhibition table which are traveling exhibition and make sure they are all same |
| spo-sponsorName | I don’t have this attribute | Needed to create a value for information about sponsor name. | Query this data and make sure those are new data, because I don’t have this.  Or compare with museum information table, because this table contain contractor information. |
| spo-SponsorAmount | I don’t have this attribute | Needed to calculate this value when I insert the value. | Query this data and make sure those are match the insurance values are greater than the total value of insurance |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

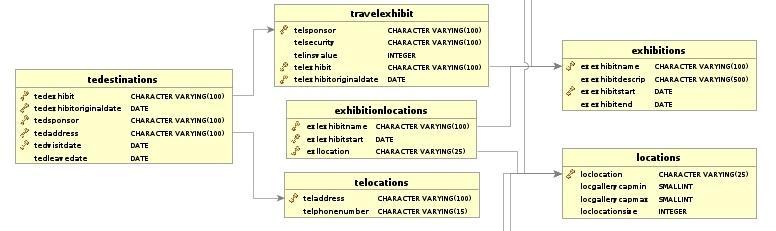
TravellingExhibitionDestinations

|  |  |  |  |
| --- | --- | --- | --- |
| Team Database | My Database | Modifications | Modifications Evaluation |
| spo-ExhibitName references TravellingExhibitions(ExhibitName) | I don’t have this table because I’m using exhibition table because exhibition table contain the information about exhibition information including traveling | Query the exhibition name information from exhibition traveling, and insert into the team database | Query this data and compare with the exhibition table which are traveling. And make sure they are same |
| spo-MuseumName references TravellingExhibitions (MuseumName) | I don’t have this table because I’m using exhibition table because exhibition table contain the information about exhibition information including traveling | Query the museum name information from exhibition which is traveling, and insert into the team database | Query the museum name and compare with query the museum name which are traveling exhibition, and make sure they are same. |
| spo-MuseumLocation references TravellingExhibitions (MuseumLocation) | I don’t have this table because I’m using exhibition table because exhibition table contain the information about exhibition information including traveling | Query the museum location information from exhibition which is traveling, and insert into the team database | Query this data and compare this information with museum information table and |
| spo-ExhibitStartDateOriginal references TravellingExhibitions (ExhibitStartDateOriginal) | I don’t have this table because I’m using exhibition table because exhibition table contain the information about exhibition information including traveling | Query the exhibition end date information from exhibition which is traveling, and insert into the team database | Query this data and compare with the exhibition table which are traveling exhibition and make sure they are same |
| spo-ExhibitDepartureDate references TravellingExhibitions (ExhibitDepartureDate) | I don’t have this table because I’m using exhibition table because exhibition table contain the information about exhibition information including traveling | Query the exhibition end date information from exhibition which is traveling, and insert into the team database | Query this data and compare with the exhibition table which are traveling exhibition and make sure they are same |
| ted-LocationName | I don’t have this table because I’m using exhibition table because exhibition table contain the information about exhibition information including traveling | Query the exhibition location information from exhibition which is traveling, and insert into the team database | Query this data and query the original data and check the location to be same. |
| ted-ExhibitArrivalDate | I don’t have this table because I’m using exhibition table because exhibition table contain the information about exhibition information including traveling.  And I separate the one exhibition into different exhibition and stored as different exhibition. | Query the exhibition start in exhibition table which is traveling information and insert into the team database | Query this value and compare the exhibitions are exist in the exhibition table which are traveling exhibitions |
| ted-DestinationPhoneNumber | I don’t have this think that exhibition table, because I stored this information in the museum information table, which I museum information could also contain the information about the museum contractor of traveling | Query the information from the information in table museum information table | Query this data and compare the query result in museum information table, and make sure they are same |
| ted-ExhibitDepartureDateLocation | I don’t have this table because I’m using exhibition table because exhibition table contain the information about exhibition information including traveling.  And I separate the one exhibition into different exhibition and stored as different exhibition. | Query the exhibition end date in exhibition table which is traveling information and insert into the team database | Query this value and compare the exhibitions are exist in the exhibition table which are traveling exhibitions |
| How to transformations | During modification of the attributes, and evaluate the result of modification, then Select all the information which I need and save as view, and insert into the team database. | | |
| Check the new database is correct | Query all attributes in this table, and check the order and there is no missing value in the query result, make sure they are all there | | |

**Database Conversion Report – Warren Fehr**

A chart showing my db design as of the end of Assignment 3 for reference

Another chart, showing how I handled TravellingExhibitions:



**Overall Changes**

We have made a lot of tiny changes to prefixes and names to standardize the database structure between groups. For example, I used “pri” as the prefix for my WorksData table, but our group decided on the prefix “wor” for the equivalent table in the merged database. All of these will be changed to the standards found the previous report unless stated in the individual table report.

**New Table (Museums)**

I added the table Museums to my database. It will have all the attributes as stated in the general report: the name of the museum, the location of the museum (which are both primary keys), and the phone number for the museum. I will also need to add the relevant domains to make the table as necessary. This will only have one entry, for my own museum, which will be a primary key for all works,exhibitions, and locations.

**Changes to WorksData Table (Now called Works), & creation of WorksInsurers**

Beyond renaming this table to Works, I will need to make changes to several attributes. First, I will remove “priwarpiece” attribute, because its information will be moved to the WorkThemes table (as told in the WorkThemes section below). I will also remove the “priownership” and “priowner” attributes, and move all that data to the Owners, Work Owners and and WorkTransactions tables (which will be described in greater depth in that part of the report). We will add the “wor-classification” and “wor-physicalproperty” attributes as well. All works in my museum are paintings, so all the PhysicalProperty enteries will be “Painting”. The classifications will be set to whatever the type attribute was originally(Such as “Landscape” or “Portrait”). After that, we can remove the unnecessary “priworktype” and “prisubtype” attributes. We will need to add the foreign/primary keys “DatabaseEntryLocation” and “DatabaseEntryLocationAddress” for all entries, which will be the ones we entered into the Museum table. The insurance values can be removed from here and put into the new WorksInsurance table.  All other attributes just need to be renamed to the ones we have specified in the Museums table for our museum.

**Changes to WorkMediums**

All I need to do are add the foreign keys from the Museums table and touch up attribute names as needed.

**Changes to Exhibitions Table**

All I need to do are add the foreign keys from the Museums table and touch up attribute names as needed.

**Changes to Locations Table**

After adding the foreign keys from Museums, I also will have to change the “loclocationsize”, which represents the total square meters of the locations into the two attributes representing length and width and create domains for them. After that, all I need to do is to just touch up attribute names.

**Changes to Connections Table (Now called Doors)**

All I need to do are add the new foreign keys from the Locations(Museums) table and touch up attribute names as needed.

**Changes to ExhibitionLocations and WorkExhibitions (Now called ExhibitionWorks)**

After changing WorkExhibitions to ExhibitionWorks and adding all the new foreignkeys in their parent tables, all I need to do to these two tables is touch up names as needed.

**Changes to WorkLocations**

In the original database, I made the assumption that because moving works between locations in the same museum is basically instantaneous, I only needed an arrival timestamp for works moved between locations. I'll need to add an attribute for departure for that in my table. The values for it will be the time it arrived at the other location. For example, if I moved an item from Location A to Location B, the departure time of the entry for A would be the arrival time for B. This will need to be touched up for all WorkLocation transactions. After that, all I need to do is add the new foreign keys in the parent tables and touch up names.

**New Table (WorkThemes)**

I need to add to add a table for WorkThemes to my museum. Considering my museum specializes in American art, I can put an entry in here for all works with theme “American”. I can also put an entry for all my pieces that are marked as war pieces by putting an entry in here for all of them with theme “War”. I will also add a few themes for each piece based on what they are based on their description to provide the data needed for themes to be useful in the new database.

**How to handle work transactions**

A lot of the information I had for handling transactions with works was scattered throughout the database in several tables (Owners, Loaners, WorkTransfers, WorkLoans). We can compile all these into the new Owners, WorkTransactions, and WorkOwners without too much trouble. All data from my Owners and Loaners tables can be compiled into the new Owners table. WorkLoans data can be transferred over to the new standard by making an entry in the WorkTransactions table (an entry showing a tranasction with the trasaction type  “LOANED” with start and end dates) and a entry into the WorkOwners table (showing the ownership transfer to the new party, as well as start and end dates of it). WorkTransfers, holding data about sold and bought works, can have its data transferred in the same manner. This would be by making an entry in the works WorkTransactionsTable with “BOUGHT” or “SOLD” as the transactiontype and appropraite start and end dates, and an entry in the workOwners table as well. This also helps to handle the inital acquire date of the work and the issue of owners, which was handled by the Works table before this. After this, all that really needs to be done is changing attribute names and adding the museum foreign keys.

**Changes to Travelling Exhibitions & Adding TravellingExhibitSponsors**

I can offload all of the sponsor data I have in this table to the new TravellingExhibitionSponsors table. I have no idea what the sponsors donated before this point in time for my data (and I shouldn't be making this amount up), so all these new sponsor entries will have the SponsorAmount be NULL. After that all I need to do are add the new foreign keys from the Exhibitions table and touch up attribute names as needed.

**Changes to TEDestinations (Now called TravellingExhibitionDestinations) &**

**TELocations**

The new database schema doesn't have a tedestinations table, but instead covers all that information in the TravellingExhibitionDestinations instead. I can transfer over the data from TELocations over to TravellingExhibitionDestinations without any problem. Other then that, just routine renaming, as well as carrying over the new foreign keys from TravellingExhibitions.

**Testing & Verification**

Beyond the data simply fitting into the new database schema, I'll need to run some tests and checks to verify it will work properly in the merged database. I've devised a major and a minor test to help verify the data is up to standard.

**Full Transactional Integrity**

I will devise a query to check if each work has full transactional integrity. What this means is that there should be an unbroken record of what the location and who the owner are for each work up until the most recent recorded activity. This is very important for our client because if there is missing information, we may have troubles if something happens to a work and we need to collect insurance on it. For a quick example, here's the sequence of events that should happen for a typical work we've bought:

* Work is bought (Entry in WorkTransactions for Sale). The item is also recorded into the Works Table at this time under the Museum that bought it, as well as an entry in the WorksOwners table (the EndTime should be NULL, because we don't know when we will stop owning the work). The WorkOwners and WorkTransactions times should match.
* An entry is made in the WorksLocation table is made when it arrives at the Museum and is placed into Storage.
* At some point, it will be entered into an Exhibition. When it is moved, the workDeparture time from the Storage entry should match the arrival time at the exhibit location. If we decide that moving times aren't instantaneous for the museum, we would decide on how long a move could be at most, and if we have a period longer then that, it would fail.
* After the Exhibition is done, it is moved back to storage. We would apply the same checks here.

We would continue like this until we hit a discrepency or we determine there is a clear chain of events. There are a lot of possibilities to cover (Loans, Sales, Donated Works, Damaged Works), but in all those cases, the new schema is able to describe the clean chain of ownership and location, and I should be able to confirm that all my data has no holes once I have placed it into it. If there is any works that fail the test, I will have to take a look and either check if it is a mere transcribing error because of moving the data, or a logic error (which may indicate a problem with the schema) and fix it accordingly. For transcribing errors it is simple, but logic errors can be used to point out if we haven't covered our cases properly.

**Themes Checking**

For themes to be useful for clients and visitors, we need to have quite a few themes for each work. All works will have the “American” theme, as my museum is an American Art, and a few works were identified as “war” pieces in the process of doing Assignment 3. I'll need to check that each piece has at least 3 themes associated with it, and I'll need to devise new ones to describe the work based on what information I have.

Samuel Germain

11131725

sjg970

CMPT 355

Dr. James Carter

Project 1 Individual report

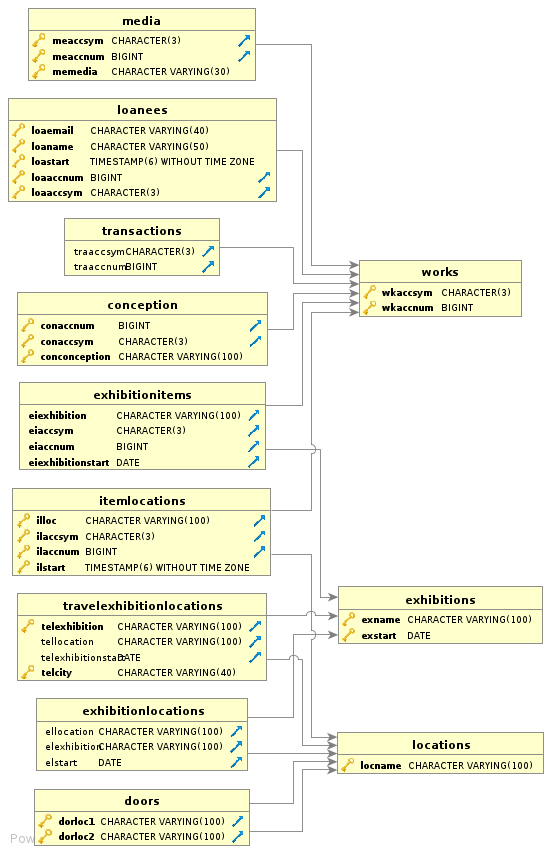


Figure 1: A dbvisualizer created entity relationship diagram showing the current nature of my database

I will be adding a **museum** table to the database, which holds primary keys of the name of the museum and the address of the museum. I will test for these changes entering the museum into the database and checking for the existence of the museum records with a query.

For my **works** table the alpha ID, numerical ID, work name and description will not need to be edited as they easily fit into the new works table under attributes of the same data type. My subtypes mostly describe an era or a region works came from and will need to be entered as theme records for the works. I will test for my subtypes converting to themes by entering all the subtypes into my themes table and then testing for the existence of all subtypes within the themes table as well as check for if a few work is matched to the particular subtype they had previously. My types fit under the category of physical property because they describe that a piece of work is something physical like a painting, a sculpture etc. This data can be easily added as an attribute to the physical property. I will create classification for my works because any of my works can be further classified on attributes pertaining to their physical property (for example many of my paintings will be able to be classified as portraits, landscapes etc.) I will test for this by searching for the existence of all classifications for which I have entered into the database and change those which are misspelled words and search fro resons for why some may not have ben entered in. Because we are going with a date data type for the creation date of a work I will need to change many of my date attributes because they are unspecific (for example some creation dates are recorded as something such as the 18th century) I will test for this by querying for all attributes in this field which do not fit the proper date format, such as being too short or containing a character. For my creation dates I will enter dates in for them that coincide to the last possible date for a period of time (if the date is the 18th century then I will enter a date of 1799/12/31. All of my values for artists will fall under the creator attribute in the new database. For my dates of acquisition I will need to create owner records for when the museum first came into possession of a work I will test for this by making sure the number of transactions that share the same acquisition date and same primary keys for a work is the number. Ideally there should be one transaction like this for each one of my works. For all my insurance values I will need to create insurance records because insurance values are becoming temporal data and I will enter the date as the current date because I know that work had this insurance value on this day. I will again test for this by checking that my records for insurance values are equal in number to the amount of works I have. I will need to move all my values for owners into owner records because this is now temporal data. I will choose the start dates of the owner records to be today because I know that the owner of that work owns the work on this date. I will test for this by making sure the number of owner records which I have is equal to the distinct attributes of owners in my work table. For my borrow status attribute I will make the records which record borrowed and create borrowed transaction records with the date it was borrowed as the current date because I know that it was being borrowed on this date. For those works recorded as potentially borrowed I will enter these into the new database works table under the borrowable attribute. I will also need to add the data entry museum name and data entry museum address as attributes and primary keys to my works table, which will reference the Museums table. I will test for this by checking that the same amount of records that I have for borrowed works matches those works which have borrowed recorded as an attribute.

For my **media** table I can easily put all these records into the new WorksMediums table because the attributes in my media table match the attributes in the mediums table and both sets of attributes use the same data types with the exceptions of adding attributes of the museum name and museum address to account for the new primary keys pertaining to the works table

For my **locations** table I will not need to change anything because my attributes and datatypes for my attributes sync with the attributes and domains for the attributes decided for the new museum with the exceptions of adding the museum name and museum address for all the locations

For my **doors** I will need to double the amount of records because my museum only has one entry for each door referencing the two locations the door connects but our collective database design had decided to have two entries for each door where each entry designates that there is a path from one location to the next. I will test for this by running queries which check that for every door leading from one room to another there will be a door leading from the other room to the first room.

For my **WorkLocations** table I will need to add in the data entry museum name and data entry museum location for each WorkLocation record because they are primary keys for the work that workLocations must reference. As well I will need to add the attributes that hold the value for the museum that is the primary key of the location.

For my **exhibitions** table I will need to add an attribute for all the records that says what museum the exhibition originated out of. I will give the value of ‘Indianpolis Museum of Art’ to all my exhibition records. I will need to add the corresponding address as well. To my ExhibitionLocations table I will need to add the same attributes, which will reference the museum attributes from the exhibitions table for the corresponding record.

My **ExhibitionWorks** table will need to have attributes added indicating the data entry museum of the exhibition and the work each record references. The values of my museum will be given for both.

My **transactions** table can retain what it has because it’s attributes and data types sync with the design of the new database. There are no records indicating a transaction was a borrow and a loan but the table already specifies that the transaction was a sale or a buy so the new values or borrow and loan will not cause any problems.

For all my tables where the museum name and museum address are added I will test to see if the number of records returning for those works matching my museum is the same as the number of records in the table total.

For my **loanees** table I will move all of the data referring to individuals into the owners’ table. All information referring to the loaning of a work will be recorded into the transactions table.. I will check for this by querying to see if all the owners listed in the loans table are all in the owners table.

I will need to add attributes for the data entry museum and the data entry museum address to my **travelling exhibitions** table and set the attributes to be my museum for every record. I will also need to change my travelling exhibitions so that they reference the exhibition, which they were based on of because now they reference a record that is, and exhibition record for the travelling museum. I will need to split information in my travelling exhibitions table. I will need to take all information relating to a location of a travelling exhibition and put it into a travelling locations table which I will need to create. The remainder of the travelling exhibition information can stay as it is because it will be able to fit into the new database travelling exhibitions table. I will test for this by querying for data on the original exhibition of my travelling exhibition and by querying to check that my travellingExhibitionLocations has the same amount of entries as my travelling exhibitions table which basically describes the locations my travelling exhibition will visit.

My table called **conception**, which I created a table to describe a more abstract way of categorizing works, I will not need to alter because it will fall perfectly under the themes table.

These changes will prepare my database to have all of its information transferred to the new database.