Database Design Document

version 1.1

prepared by: Team 8

# Outline

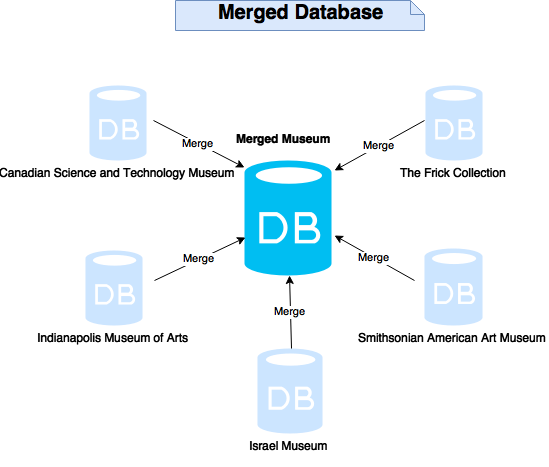
1. Introduction
   1. Scope
   2. Purpose
   3. Brief overview of merged museum
   4. Requirement of Environment
2. Domain
3. Simplified Diagram
   * 1. Core Diagram
     2. Traveling Centered Diagram
     3. Transaction Centered Diagram
4. Tables
5. Temporal Data
6. Views
7. Future Consideration

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**

CMPT355GeneralDiagram(3).png

**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.