**n this project deliverable, you should turn in one zip file that includes the following files:**

**a) A MS Word document that contains the updated proposal (include my feedback for deliverable 2 and highlight any changes made in response to the feedback, if any). A list of 10 specific functionalities that you will implement using queries in SQL (a functionality is a task that your application can accomplish, often through a query, e.g., search for a product, display the total sales volume, etc.), with the following specific requirements (Note that you may have the one query that satisfies multiple requirements below) Note: you are to write out the query as it would entered into SQL (These are not required to be executed in SQL):  
    1) The 10 functionalities should cover at least four entities in your ERD plus at least one of the subtypes (see requirement #9).   
    2) At least 3 functionalities should involve more than one entity (Example: show the total number of the products supplied by customer X) - this functionality involves at least the Product and Customer entities).  
    3) At least 3 functionalities should involve the use of mathematical functions such as Sum, AVG, Min, Max, or the ANY and ALL keywords.    
    4) At least 1 functionality should involve the use of the Count feature (Example: Count the number of vendors who have supplied more than one product).  
    5) At least 2 functionalities should involve the "Group By" feature (Example: Display the average prices of the products supplied by each vendor).  
    6) Among the above 2 Group By queries, at least one of them should have the "Having" clause.  
    7) At least one functionality that involves a query with the "LIKE" keyword.  
    8) At least 1 correlated query.  
    9) At least 1 functionality should involve the entity that exhibits the unary relationship (i.e. a self-join is needed).   
    10) At least 1 functionality should involve both the supertype and one of the subtypes.   
After you list the 10 functionalities, indicate which of the above requirements each functionality satisfies, e.g.: Functionality 1: Find out the average number of products purchased by customers from Florida (#2, #4, #5).**

**B)  An SQL file that contains all the create table commands and INSERT statements outlined in the ERD (or an Access file) with both entity integrity (through PK) and referential entity (through FK) properly enforced in the relationship diagram. Each table should have at least 4-5 records. The tables should be created in an order that allows the FK to properly reference the table that have been created.   
When you insert these test records, make sure they are consistent with the business rules and constraints you specified in the proposal. For example, if you specified the "Price" attribute to be numeric and not null, then the column should not contain either a string value or a null value. Similarly, if you specified that "Each vendor must supply at least one product", then you need to make sure that every vendor that appears in the Vendor table must show up at least once in the Product table.   
At the beginning of your code file, please insert "CREATE DATABASE [Name of the database on your SQL Server]".**

**C) The Visio file that contains the updated ERD (include the Visio file even if no changes are made).**

**IS 6503 Term Project Deliverable 2**

Your Name: Bobb Shields

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

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

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

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

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

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

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

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

|  |  |  |  |
| --- | --- | --- | --- |
| **Description:** the person who made the art and their available metadata. Could possibly have multiple art pieces in different museums. | | | |
| Variable Name | Definition | Data type | **Constraints** |
| portfolioID | PK for Portfolio | int64 | PK |
| artistID | FK for Artist, enables M2M | int64 | FK |
| artID | FK for Art, enables M2M | int64 | FK |
| Portfolio Notes | Portfolio information | string | Null |

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

|  |  |  |  |
| --- | --- | --- | --- |
| **Description:** the person who made the art and their available metadata. Could possibly have multiple art pieces in different museums. | | | |
| Variable Name | Definition | Data type | **Constraints** |
| artID | PK for Portfolio | int64 | PK |
| catalogue\_number | Unique identifier for the artwork | string | Index |
| City | City associated with the object | string |  |
| Classification | Classification of the object | string |  |
| Country | Country associated with the object | string |  |
| creation\_date | Date the artwork was created | string |  |
| Credit Line | Credit line for the object | string |  |
| Culture | Culture associated with the object | string |  |
| Department | Department responsible for the object | string |  |
| Dimensions | Dimensions of the object | string |  |
| Duration (s) | Duration of the artwork in seconds | float64 |  |
| Dynasty | Dynasty of the object | string |  |
| Excavation | Excavation site of the object | string |  |
| Geography Type | Type of geography associated with the object | string |  |
| image\_url | URL to the artwork’s image | string |  |
| Locale | Locale associated with the object | string |  |
| location | At which museum the piece sits | string |  |
| Locus | Locus associated with the object | string |  |
| Medium | Medium or materials used in the object | string |  |
| Object Begin Date | Beginning date of the object’s creation | int64 |  |
| Object Date | Date of the object | string |  |
| Object End Date | Ending date of the object’s creation | int64 |  |
| source\_fk\_Object\_ID | ID for the object | int64 |  |
| Object Name | Name of the object | string |  |
| Object Number | Unique object number | string |  |
| Period | Period of the object | string |  |
| Region | Region associated with the object | string |  |
| Reign | Reign during which the object was created | string |  |
| Repository | Repository of the object | string |  |
| Rights and Reproduction | Rights and reproduction information | string |  |
| River | River associated with the object | string |  |
| State | State associated with the object | string |  |
| Subregion | Subregion associated with the object | string |  |
| Title | Title of the object | string |  |
| web\_url | URL to the artwork’s webpage | string |  |
| Weight (kg) | Weight of the artwork in kilograms | float64 |  |

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

|  |  |  |  |
| --- | --- | --- | --- |
| **Description:** the person who made the art and their available metadata. Could possibly have multiple art pieces in different museums. | | | |
| Variable Name | Definition | Data type | **Constraints** |
| City | Museum | City of the museum | string |
| State | Museum | State of the museum | string |
| Street | Museum | Street address of the museum | string |
| Zip | Museum | ZIP code of the museum | string |
| Zip5 | Museum | 5-digit ZIP code of the museum | string |
| AKA\_DBA | Museum | Also known as/doing business as | string |
| ALT\_Name | Museum | Alternative name of the museum | string |
| COMMONNAME | Museum | Common name of the museum | string |
| Country | Museum | Country where the museum is located | string |
| Discipline | Museum | Discipline of the museum | string |
| Gallery space in m2 (sq ft) | Museum | Gallery space in square meters | string |
| source\_fk\_MuseumID | Museum | Museum ID | int64 |
| Name | Museum | Name of the museum | string |
| PHONE | Museum | Phone number of the museum | string |
| WEBURL | Museum | Website URL of the museum | string |
| Year established | Museum | Year the museum was established | string |

1. **ERD (there should be at least 6 entities plus at least two subtypes**, the ERD can be modified in the next deliverable). Your ERD must have all primary/foreign keys, relationships, minim and constraints correctly specified. The ERDs below are just example.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

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

Relationships:

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

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

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

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

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

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

Among the above relationships:

**Strong Relationships**

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

**Weak Relationships**

* **N/A**

**Existence Dependency**

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

**Mandatory Relationships**

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

**Constraints**

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

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

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

ERD Update Notes

The ERD was updated for the following changes:

* The verb on the relationships between entities was missing. Added
* Added a third linker table to allow multiple artists to collaborate on the same piece of art
* Changed portfolio to not hold artistID as an FK. Portfolio -> Art -> Artist exists as autonumber PKs so the artists that have work inside a portfolio can easily be queried as needed.

Access To-Do Notes

* “Untitled” and “Artist Unknown” should be entered in Art and Artist tables (and similar treatment for Porfolio) to allow for dummy data to be easily chosen when those data are unavailable upon import.
* Imports working to append for a subset of the data, but needing to work on logic to reference the source fk (text based) and find the PK integer, to enter as the valid FK in