Transactions for inserting, updating, and deleting data

This section of the report is focused on the set of transactions developed for our database. They are divided into two sections, transactions that were used in our own previous databases, and new ones that were written for our new database design. They are now outlined below:

**Part 1: Transactions that were used in prior museum databases**

**Insertion Transactions:**

**Insert into doors:** This is used to insert a new entry into the doors table. This form allows entry into each field of the table manually, to insert new paths, such as when a new gallery added, so a new path would possibly be needed. This transaction then inserts the user inputted data into the appropriate table in the database.

Tables Affected: Doors

Testing: This table was tested by attempting insertion with valid and invalid values to ensure that the transaction/script worked properly.

**Insert into ExhibitionLocations:** This is used to insert a new entry into the exhibitionlocations table. This form allows the assignment of an exhibition to a specific location or locations, so it is useful for inserting into the database where an exhibition will be held. The form allows for the input of information into the database in a friendly user interface.

Tables Affected: ExhibitionLocations

Testing: Testing was conducted by inserting into the database some test data, and viewed in the database to ensure that it works properly.

**Insert into Exhibitions:** This is used to insert a new exhibition into the database system. This can specify the name and details of an exhibition.

Tables Affected: Exhibitions

Testing: Testing for this table was accomplished by submitting test data and querying the table to ensure that everything was inserted into the database correctly.

**Insert into exhibitionworks:** This transaction script is used input a work that will be a part of a certain exhibition. It allows every field to be inputted for more meticulous editing.

Tables Affected: exhibitionworks.

**Insert into Locations:** This transaction script allows the entry of new locations into the database system, whether they may be for new museum locations, or places that a travelling exhibit may visit. Every field is accessible for insertion for more meticulous editing.

Tables Affected: Locations

Testing: Testing for this was accomplished by using test data, in association with other scripts to produce data that was queried to ensure that it was input properly.

**Insert into owners:** This transaction script allows the entry of new owners into the new database. This is done through a graphical form that allows the customization of information for every field. This can be used for situations such as transactions that involve someone new coming into the museum’s database.

Tables Affected: Owners

Testing: This was tested like the other scripts by inserting test data and querying the applicable table to ensure that the data was submitted correctly.

**Insert into sponsorexhibitions:** This script allows for the insertion into the database of a sponsor of an exhibition, including the entry of how much money that the sponsor may have donated. The script displays a form which allows for database entry, and once filled, the new information is inserted into the sponsorexhibitions table.

Tables Affected: sponsorexhibitions

Testing: To test this script, a test sponsor was added for one of our exhibitions, which when queried, did properly link up to the information that we had inputted.

**Insert into themes:** This script allows for the insertion of new themes into the database and which works relate to them. This allows the input of new themes which aids in the sorting of works. Due to the database constraints, nothing will be submitted if the work doesn’t exist.

Tables affected: Themes

Testing: New themes were inserted for certain works, and then were queried and displayed correctly, signifying that the inputted information had been input into the database correctly.

**Insert into worklocations:** This allows the manual insertion of the information of where a work was into the database. It allows entry for all fields, and is more administrative in nature, allowing for the meticulous input of new worklocation data.

Tables Affected: worklocations

Testing: Testing was accomplished by adding test data and querying the worklocations table to ensure that the new data was properly added.

**Insert into workowners:** This allows for the manual insertion of new data into the workowners table. This can be used to insert new data about the owner of a work, when a work is inserted, or when a work changes owners.

Tables Affected: workowners

Testing: Testing was done by inserting new data into the workowners table using test data and querying the respective table to ensure that it was properly put into the database.

**Insert into works:** This allows the input of new museum works to the database, it allows for all the fields to be filled in. This allows for insertion of new works with all applicable data fields available to be filled in.

Tables Affected: works

Testing: This was tested normally, by inserting test data into the works table using the transaction, and querying to ensure that the data was correct.

**Insert into worksinsurance:** This allows for the entry of data into the worksinsurance table. Every field is editable for meticulous insertion if necessary. This generally won't happen all that often, but it is handy to have in emergencies.

Tables Affected: worksinsurance

Testing: Data was inserted and the subsequently queried from the worksinsurance table to ensure that data was inserted correctly.

**Insert into worksmedium:** This allows for the insertion of new medium types for works in the database. This allows the museum to insert new mediums for the sorting of works.

Tables affected: worksmedium

Testing: Testing for this was accomplished by inserting test data into the database and querying the data to ensure that it was input correctly.

**Insert into worktransaction:** This allows the insertion of manual transactions into the table. Traditionally it would not happen all that often, but this is more intended to add transactions manually in the possible case of a power outage, etc.

Tables Affected: worktransaction

Testing: Testing was accomplished by submitting a test transaction and ensuring that all the data was submitted into the museum’s database correctly.

**Deletion Transactions:**

**Delete from doors:** This allows a user of the system to delete from the doors table. This can be useful if the structure of a building changes. It may not be used often, but it is useful to have.

Tables Affected: doors

Testing: This was tested in conjunction with its corresponding insert function. Data was put into the table and this was used to remove said data, with a query then being done to confirm that it was removed.

**Delete from exhibitionlocations:** This allows a user of the system to remove an entry from the exhibitionlocations table. This may be done if an exhibition is suddenly forced to change locations, or if incorrect data is entered into the system.

Tables Affected: exhibitionlocations

Testing: This was tested in conjunction with its corresponding insert function. Data was put into the table and this was used to remove said data, with a query then being done to confirm that it was removed. Originally noticed a slight issue where a certain item would not be deleted, it was corrected.

**Delete from exhibitions:** This allows a user to remove an exhibition from the system. A user may wish to do this if an exhibition is suddenly cancelled before taking place, or if incorrect information is entered into the system.

Tables Affected: Exhibitions

Testing: This was tested in conjunction with its corresponding insert function. Data was put into the table and this was used to remove said data, with a query then being done to confirm that it was removed.

**Delete from exhibitionworks:** This allows the removal of data from the exhibitionworks table. A user may wish to do this while planning the museum works that may go into a certain exhibition that is being planned for the future.

Tables Affected: exhibitionworks

Testing: This was tested in conjunction with its corresponding insert function. Data was put into the table and this was used to remove said data, with a query then being done to confirm that it was removed. Once this was achieved we knew that the function worked.

**Delete from locations:** This allows the removal of data from the locations table. This commonly would only be used by the user if a location is entered incorrectly into the system.

Tables Affected: Locations

Testing: This was tested in conjunction with its corresponding insert function. Data was put into the table and this was used to remove said data, with a query then being done to confirm that it was removed.

**Delete from owners:** This allows for the deletion of an owner from the owners table. A user of the database may wish to do this if data was entered incorrectly into the database.

Tables Affected: owners

Testing: This was tested by deleting some test data that was inserted, and querying the owners table to ensure that the data was properly removed.

**Delete from sponsorexhibitions:**  This allows the removal of data from sponsorexhibitions. This may be done for various rare cases, but is still necessary to have for the odd case that it may be needed.

Tables Affected: sponsorexhibitions

Testing: This was tested in conjunction with its corresponding insert function. Data was put into the table and this was used to remove said data, with a query then being done to confirm that it was removed.

**Delete from themes:** This allows for data to be removed from the themes table. This allows a user to remove themes that may no longer apply to works or were perhaps inserted incorrectly.

Tables affected: themes

Testing: This was tested in conjunction with its corresponding insert function. Data was put into the table and this was used to remove said data, with a query then being done to confirm that it was removed.

**Delete from worklocations:** This allows a user to remove data from the worklocations table. This will rarely be used, except in the rare cases that past false data may need to be removed.

Tables Affected: worklocations

Testing: This was tested in conjunction with its corresponding insert function. Data was put into the table and this was used to remove said data, with a query then being done to confirm that it was removed.

**Delete from workowners:** This allows data about who a specific work belonged to to be removed from the database. This will rarely be done except in the odd case of false historical data.

Tables Affected: workowners

Testing: An owner was inserted into the table using the counterpart transaction, and this transaction was used to delete it, with a query confirming its deletion.

**Delete from works:** This allows a work to be deleted from the database. This allows a user to find a specific work, and perhaps because of incorrect data, remove that work from the database completely provided that the work is not referenced in another table.

Tables Affected: works

Testing: This was tested in conjunction with its corresponding insert function. Data was put into the table and this was used to remove said data, with a query then being done to confirm that it was removed. This test was proved successful, so the data was properly removed as expected.

**Delete from worksinsurance:** This allows for the deletion of the insurance history on a specific work in the database. This will rarely be used, perhaps in the odd case that information that is historically incorrect needing to be deleted

Tables Affected: worksinsurance

Testing: This was tested in conjunction with its corresponding insert function. Data was put into the table and this was used to remove said data, with a query then being done to confirm that it was removed.

**Delete from worksmedium:** This allows for the removal of medium data on a specific work. A user may delete one entry from the worksmedium table using this method. A user may wish to do this to remove incorrect data from the database, but due to the fact that museum works traditionally don’t change materials, it likely won’t be used all that often.

Tables Affected: worksmedium

Testing: Testing for this was accomplished by manually inserting a new medium to an existing work, and then using this method to delete it, finally querying the specific table to ensure the data was removed.

**Delete from worktransaction:** This allows the deletion of specific transactions from the museums database. Generally this will only be used in the rarest of cases, and by a system administrator as transactions should usually be kept on record.

Tables Affected: worktransaction

Testing: This was tested in conjunction with its corresponding insert function. Data was put into the table and this was used to remove said data, with a query then being done to confirm that it was removed.

**Update Functions:**

**Update exhibitionlocations:** This allows for the updating of data on where an exhibition is held, in the case that an exhibition may change locations or if data was input into the system incorrectly and needed to be changed. It also allows all data of it to be edited if need be

Tables Affected: exhibitionlocations

Testing: Testing was done by editing all the fields of test data and querying the table to ensure that the information had been changed correctly.

**Update exhibitions:** This allows all the fields of data related to an exhibition to be changed by a user. Useful in the case in which more information may be added, or incorrect information needed to be changed.

Tables Affected: exhibitions

Testing: Testing was done by editing test data that was entered into the system, and querying the new data to ensure that it was changed correctly.

**Update exhibitionworks:** This allows the editing of data of a specific entry in the exhibitionworks table. This is useful if data is incorrect and needs to be changed, or possible new information needs to be added.

Tables Affected: exhibitionworks

Testing: Testing was accomplished by changing data that had been inserted for testing purposes, and then ensuring through a report that the data had been changed appropriately.

**Update locations:** This allows for the updating of information on a location. Each field is editable for various purposes, such as when more data is obtained or if a location gets a new owner.

Tables Affected: locations

Testing: Testing for this was done by creating test date, and editing it, finally doing a report on the test data to ensure that it was changed correctly.

**Update owners**: This allows for the editing of data within a specific record of the owners table. A user may want to do this if more information is added, or if data needs to be changed for whatever reason.

Tables Affected: owners

Testing: Testing was accomplished by making edits to test data, and then querying the owners table to make sure that the new data that was updated into the database was there correctly.

**Update sponsorexhibitions:** This allows the updating of information in the sponsorexhibitions table. A user may wish to do this if a sponsor perhaps donates more money to a specific exhibition and the data needs to be changed.

Tables Affected: sponsorexhibitions

Testing: Testing on this table was accomplished by editing the amount of money that a specific sponsor had donated to an exhibition, and then querying to ensure that the update had occurred properly.

**Update themes:** This allows the name of specific themes that relate to a work to be updated. A user may wish to do this if new themes replace old themes from within the system.

Tables Affected: themes

Testing: A theme for a specific work was changed in the themes table, and the data was queried to ensure data had updated properly.

**Update worklocations:** This allows a user to manually update or change data in the worklocations table. They may wish to do this to update data in the database in the case that something changes. This was primarily used to add an end date

Tables Affected: worklocations

Testing: Testing was accomplished by editing the end date of a worklocation record and ensuring through a query that the data was changed correctly.

**Update workowners:** This allows a user to manually change data related to the owner of a work. This temporal data likely wont be changed often, but can be used to update the final date field to keep relevant temporal information.

Tables Affected: workowners

Testing: Data was entered and edited with this function, and queried to ensure that data was changed as intended.

**Update works:** This allows a user to add or change information about a specific museum work. They may wish to do this if more data about the work was obtained or if incorrect data was added and needed to be changed.

Tables affected: works

Testing: New data was inserted for a work that was obtained, and the work was queried from the table to ensure that the new data was entered correctly

**Update worksinsurance:** This allows a user to update data in a specific record in the worksinsurance table. A user will primarily use this to add an end date when a works insurance value changes but it may be used for other purposes as well

Tables Affected: worksinsurance

Testing: Test data was updated using this method and subsequently queried to ensure that the information changed correctly.

**Update worksmedium:** This allows a user to update data in a specific record in the worksmedium table. This commonly wont be used but it may be used if more data is obtained, such as if a metal is discovered to be a more specific element.

Tables Affected: worksmedium

Testing: This testing was accomplished by editing test data and querying that corresponding data to ensure that it was changed correctly.

**Update worktransaction:** This allows the updating of specific transactions on the odd cases that something needs to be changed.

Tables Affected: worktransaction

Testing: A test transaction record was created and edited using this function, and then queried to ensure proper data editing.

**Part 2: New Transaction Structures**

**Museum Purchasing a work**: This is a multi-table transaction that is higher level, allowing the database to process the purchase of a new work for the museum. In a single form, a user can enter the information of a work, and it will automatically be entered into the works table, recorded as a new insurance record in the workinsurance table, and be logged as a purchase transaction in the worktransaction table.

Tables Affected: Works, worksinsurance, worktransaction

Testing: Testing this required using the form to input data into the database, and then querying the three respective tables to ensure that data was inserted into the table properly.

**Museum selling a work:** This is a multi-table transaction that is meant to be user friendly to process selling a work that belongs to a museum. Through filling out the form, the data entered by the user automatically logs the work into the worktransaction table as being sold, and creates a new record of ownership and sets and end date of the previous record in the workowners table.

Tables Affected: workowners, worktransaction

Testing: This was tested by inputting a test work labeled AAAA-4444 into the database system, and using this function to sell the work to another owner. The appropriate tables were then queried to ensure that the data had been added and updated appropriately.

**Loan out a work:** This is a multi-table transaction that is meant to be user friendly to allow a user to process the loaning of a museum work to another museum. Through filling out the form, a transaction is created, and the location of the work is updated in the worklocations table by inserting the new location and setting the end date for the old location.

Tables Affected: worktransaction, worklocations

Testing: This was tested by inputting a test work into the system, and then loaning it to another owner and location. The data was then queried to ensure that it was logged as a transaction and that the work changed locations.

**Borrow A work:** This is a multi-table transaction that is meant to be user friendly. It allows a user to create a log in the system of borrowing a work. By filling out the form, the work being borrowed is recorded as a transaction and its location is updated and previous location ended, similiar to the loaning of a work.

Tables Affected: worktransaction, worklocations

Testing: This was tested by inputting a test work into the system, and then borrowing it from another owner and location. The data was then queried to ensure that it was logged as a transaction and that the work changed locations.