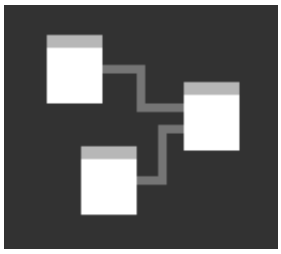
Project 01 ERM/ERD with MySQL Workbench

You will be installing and leaning to use MySQL Workbench for this assignment. MySQL Workbench is an open-source graphical tool for designing relational databases based on conceptual models.

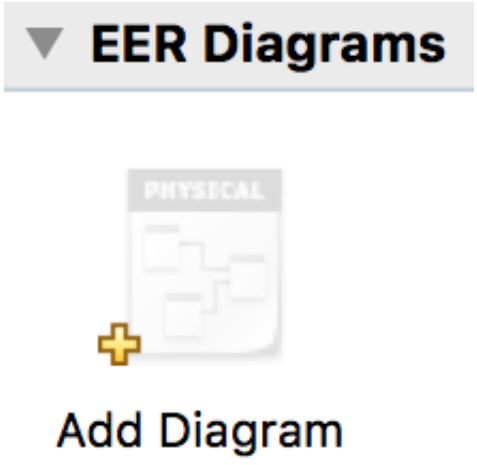
MySQL Workbench

Start by downloading MySQL Workbench and installing it. It works on nearly any system. You only need it for this project, so if you may delete it after the project is graded (though you may want to keep it around for reference and practice!)

Once the program is running, you should navigate your way to the Models menu and create a new model.



From here, add a new EER Diagram.



Here you can drag and drop elements from the palette to the diagram canvas. You can complete this project using only the **table** and **relationship** elements. You can modify element properties by double clicking them and navigating through the tabs in the detail area at the bottom.

Tables in this model correspond to **entities**, while **columns** correspond to **attributes**. **Relationships** are implemented using **primary keys** and **foreign keys**.

Business Rules

Our database will track well-structured data describing the configuration and activities of several auction houses in the U.S. The rules of this business are described below. You should infer appropriate data types from the descriptions and examples given. Units ($, %, feet) do not need to appear in the data.

Carefully note the cardinality of relationships. "*Every child has a home*" leaves you to infer that every child belongs to one home, while a home can house many children. The relationship may or may not be implemented in the entity where it is listed. Follow this simple rule: The foreign key goes on the *to-many* side of every relationship.

Hint: You may want to draw this on paper before trying to implement it in MySQL Workbench.

Entities:

- **Auction House** (implement as **AuctionHouse** - no space)
    - Each Auction House has a name(*id*), an ESTD (date when it opened), square footage, and is either an *indoor* or an *outdoor* facility
    - An **Auction House** belongs to a **Territory**. Many auction houses operate within any one territory.
- **Territory**
    - A Territory has a short name(*id*), an optional long name, a state (two letter abbreviated), and a five-digit zip-code with an optional four-digit extension.
- **Item**
    - An Item has a name and a date it was put up for auction.
    - Items are described simply by their names, which are not guaranteed to be unique in any way. Therefore it is appropriate to add an Integer ID column, with the AI (Autoincrement) option selected so that the database generates IDs automatically.
    - Each **Item** belongs to exactly one **Seller**.
- **Seller**
    - A Seller has a first name, a middle initial, a last name, and a mailing address (US: number and street (one text attribute), city, state (two letter abbreviated), and a five-digit zip-code with an optional four-digit extension).
    - Sellers are identified by their full names.
    - Each **Seller** sells goods in several **Territories**. *Hint: This implies a "sellsIn" many-to-many relationship between these two entities.*
- **Auctioneer**
    - An Auctioneer has a first name, a middle initial, a last name, a standard commission (a percentage, two digits), and a mailing address (US: number and street (one text attribute), city, state (two letter abbreviated), and a five-digit zip-code with an optional four-digit extension).
    - Auctioneers are identified by their full names.
    - Each **Auctioneer** works in several **Territories**. *Hint: This implies a "worksIn" many-to-many relationship between these two entities.*

- **Buyer**
    - A Buyer has a first name, a middle initial, a last name, a standard commission (a percentage, two digits), and a mailing address (US: number and street (one text attribute), city, state (two letter abbreviated), and a five-digit zip-code with an optional four-digit extension).
    - Buyers are also each given a unique Integer ID. Use the AI (Autoincrement) option so that the database generates IDs automatically.
- **Bid**
    - A bid contains a dollar amount, under *one million dollars*, and includes cents. The highest bid possible is $999 999.99.
    - Not all bids are winning bids, but those that are will have a time and date marking when they were accepted as winning bids.
    - A **bid** is placed at an **auction house**.
    - A **bid** connects is placed on an **item** by a potential **buyer**. Each buyer can make more than one bid on the same item, however, so *the bid **amount**, the item, and the buyer are all three needed to identify a unique bid*.
- **Account**
    - And account tracks the amount of money (limit *one million dollars*) that is owed to a seller from a buyer.
    - An account is unique to a given **buyer** and **seller** together.

Database Rules

You should use the data types listed here:

- **text(n)** or **varchar(n)** [n = maximum characters]
- **char (n)** [n = number of characters, exactly]
- **int**
- **decimal** (digits, decim) [also called **numeric**]
- **datetime**
- **date**
- **boolean** [MySQL uses **tinyint** for boolean]

Fixed-length strings should use char(n), while variable-length strings should use text. All non-optional attributes must be constrained to Not-Null (**NN**) in the database.

A *limited multivalued attribute*, where the number of values stored for one attribute is known exactly at design time, must be implemented by adding the corresponding number of attributes to that entity.

An *unlimited multivalued attribute*, where any number of values may be stored for the attribute, must be implemented using an additional table.

All relationships should be *many-to-one*. A *many-to-many* relationship must be broken up with an associative entity. *Hint:* MySQL Workbench has a many-to-many tool that does this for you.

Do *not* add any surrogate keys to the model unless specified to do so. Use simple primary keys and composite primary keys where appropriate.

All non-foreign attributes in any given entity should start with the same prefix named for that entity, such as "auc", "auc_" or "auctioneer" for the auctioneer entity. Camel case or underbars are fine.

Submission Instructions

Arrange your model elements so that they look neat on the diagram. Keep them fairly close together so that you should not need to scroll much to see the whole model.

Save your MySQL Workbench file, ***Auctions.mwb***, using the **File -> Save Model As...** menu command. Submit them through Scholar here.