In-Class Activity #2: Database Schema

Submission Instructions

Submit your solutions as two *.mwb files through **Canvas>Assignments>To-Do Submissions.**

You must create your database schemas electronically, and they cannot be hand-drawn.

For both parts, use MySQL Workbench to create your schemas. When you finish creating your database schemas, save your file in *.mwb format (File->Save Model As…), e.g., "ica2_part1.mwb" and "ica2_part2.mwb".

How to access MySQL Workbench?
1. Thru the MIS community site, kindly refer to the download link/instructions to install MySQL Workbench. Once you successfully install and run it, your landing page should look something like what is shown below.

    You may not have defined any connections yet, and that's OK for this activity. MySQL Workbench has many features, and today we will only be using the EER Diagram features. EER is short for "Enhanced Entity Relationship" and it is what the diagram utility is called in MySQL Workbench.

Part 1.

Understanding database relationships and cardinality is the primary learning objective of this In-Class Activity.

Use the information from the following tables to create a database schema diagram for a **Housing Authority** database. You must specify the cardinality of the relationships when you create the schema.

Maximum cardinality is given; however, you will have to identify the minimum cardinality if necessary.

The following describes the schema diagram you are expected to created in Part 1.

Entities	Attributes
Development	DevelopmentID, Name, Number of units
Unit	UnitID, Unit Number, Bedrooms, Bathrooms, Square footage
Household	HouseholdID, Description
Resident	ResidentID, First name, Last name, Birthdate, Head of household?

| Relationships	Relationship Attributes
Development-Unit: One to Many	
Unit - Household: Many to Many	Move in date, Move out date
Household- Resident: One to many	

Step 1: open MySQL Workbench, select 'Models' on the left side, and then press the '+' at the top. That would create a new model.
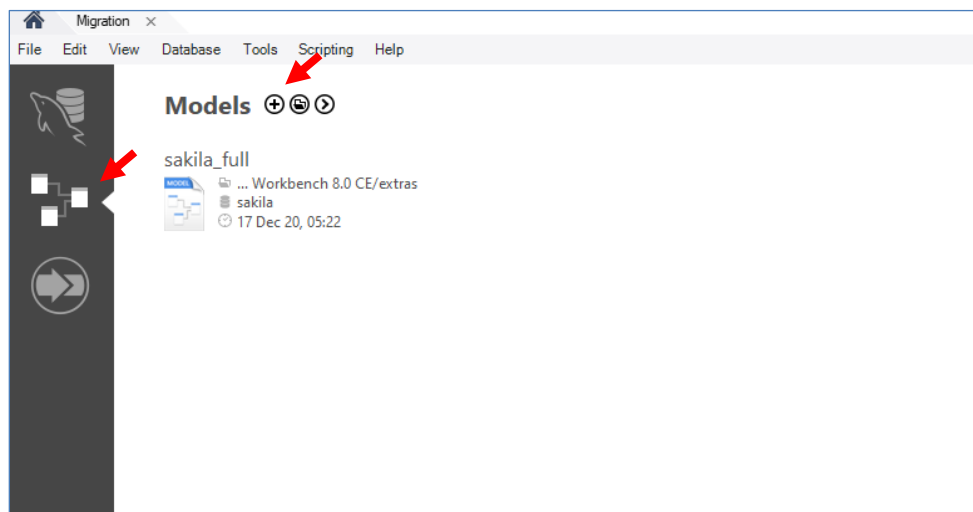
Step 2: Add a Diagram for your database. You do this by double-clicking on "Add Diagram".
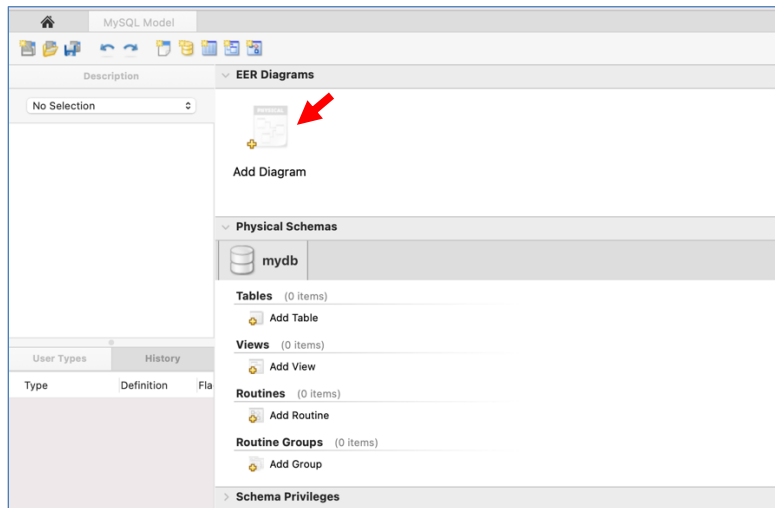

The center of your window will be dominated by a space called the diagram canvas. On the left edge of the canvas, you will see a palette of buttons.
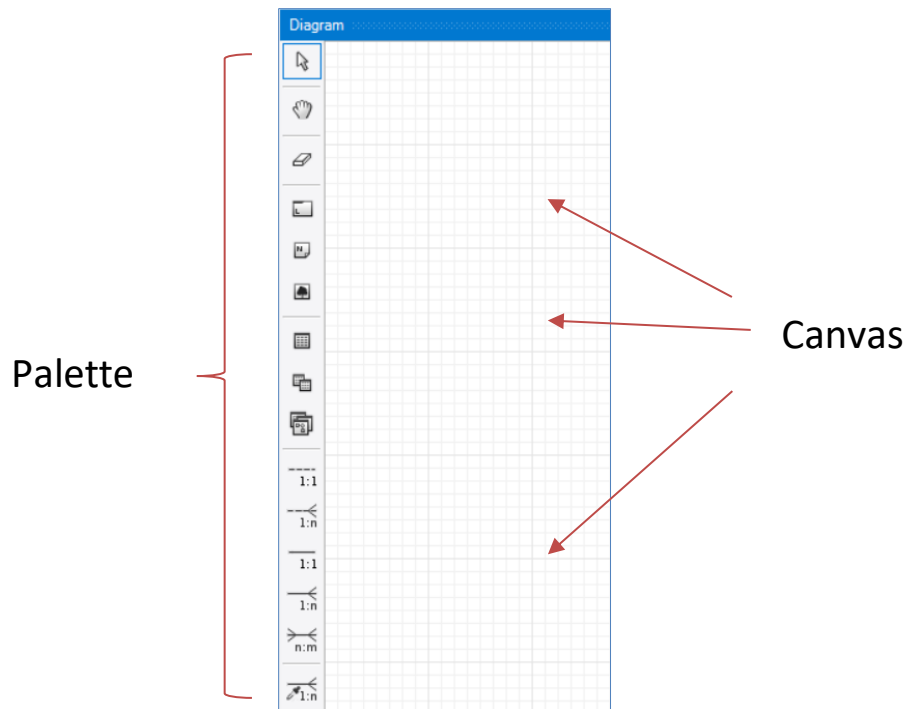
Palette

Canvas

Step 3: Look in the palette. Find a button that adds a new table to the diagram. Create your first table. You do this by clicking on the new table button, and then clicking again on the canvas.
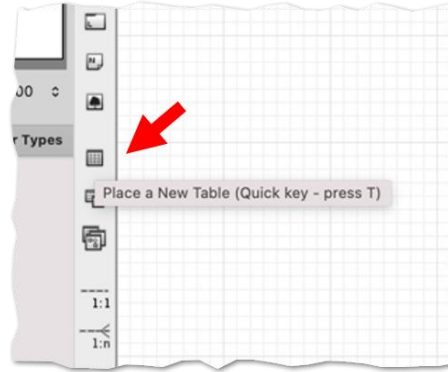
Step 4: Double-click on your table to open its properties.
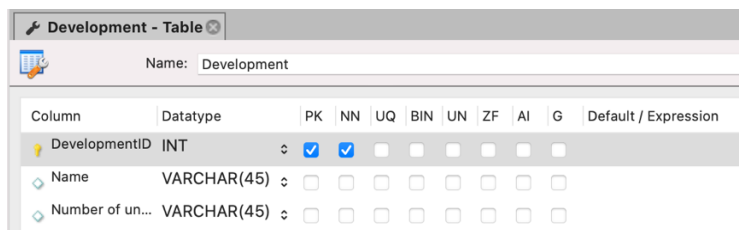
Change its name and add the list of columns. Select *Primary Key* by checking the PK box. For each column, you should specify its data type.

What is a data type?

In relational database,
- Each field should contain only one type of data
- That must be specified when the table is created
- There are many data types; we're only going to cover the most important ones

| Data type	Description	Examples
INT	Integer	3, -10
DECIMAL(p,s)	Decimal. Example: decimal(5,2) is a number that has 3 digits before decimal and 2 digits after decimal (like 123.45)	3.23, 3.14159
VARCHAR(n)	String (numbers and letters) with maximum length n	'Hello', 'I like pizza', 'MySQL!'
DATETIME, DATE	Date/Time, or just Date	'2011-09-01 17:35:00', '2011-04-12'
BOOLEAN	Boolean value	0 or 1

BOOLEAN is sometimes replaced with the data type **TINYINT**. A tiny integer has a value between 0 and 255.

Discuss: Looking at the examples above, which data type would be best suited to storing a person's name? A person's age? A person's phone number?

Step 5: Follow the same process to add all four tables in the *Housing Authority* database. When you are done, your EER canvas should look something like this:
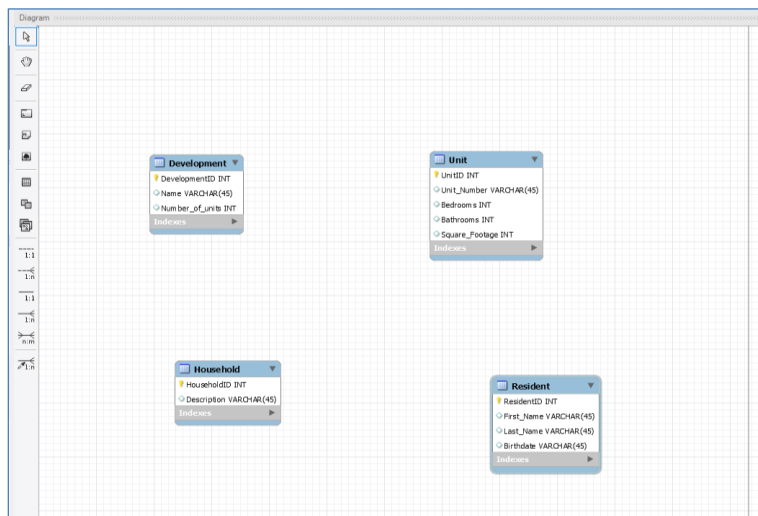
Step 6: Define the relationships between the tables and their cardinalities. *Getting the cardinality right is the main point of this ICA.*
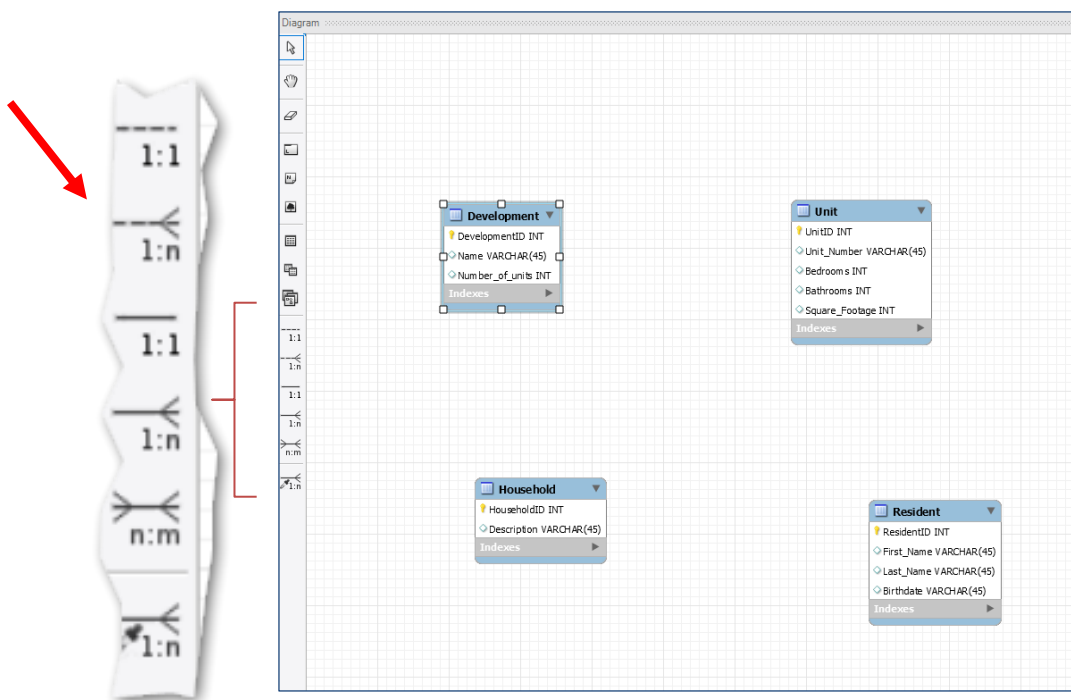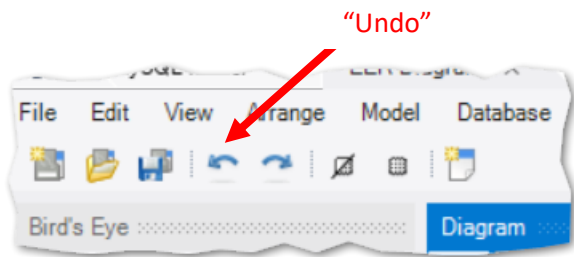
For each relationship,

(1) Click on the button with the corresponding cardinality on the left side
(2) Click on the table on the "Many" side of your relationship (in case of one-to-one, click on the table where you would like to put a foreign key).
(3) Click on the second table.

MySQL Workbench will create the connection and place a *Foreign* key on the Many side of your relationship. In case of many-to-many cardinality, Workbench will automatically create a "bridge" table to define it.

Think about it … in a relational database each column in a row should hold **one** piece of data (one number, one piece of text, etc.). Consequently, it would be impossible to represent a many-to-many relationship in such a database without a bridge table.

You might also notice that MySQL Workbench is making a distinction between what's called an "identifying relationship" and a "non-identifying relationship". Rest assured that you can safely ignore this. The term "identifying relationship" will not be important in future exams or lectures.

If you make a mistake, you can use the "Undo" and "Redo" options as needed.


Step 7: Refine your connections.

(1) Set their minimal cardinality using NN flag for the foreign key and "Mandatory" flag in the properties of the relationship's foreign key.
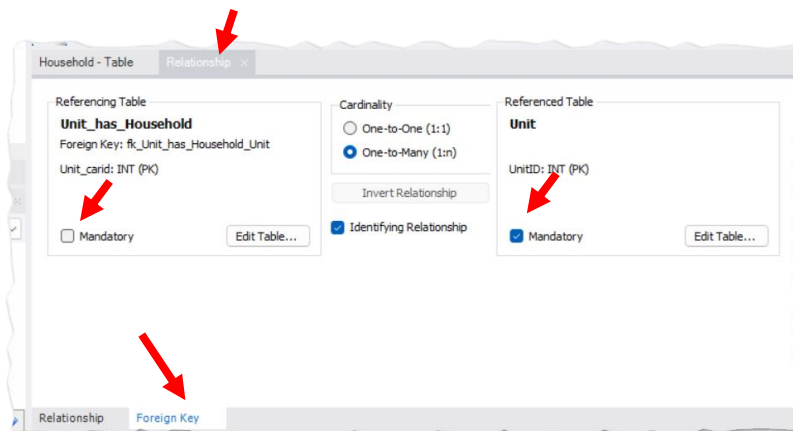
(2) Add properties of the connection between the Unit and Household into the bridge table.

TAKE NOTE! A bridge table can have attributes of its own. Sometime the is data that we want to track that is not clearly an attribute of one single *table*, but is an attribute of the *relationship* between two tables. We capture such a piece of data by adding a property to the to the bridge table and storing it there.

(3) Consider the diagram below. The cardinalities are correct. But some of the data type choices are not. **DISCUSS:** Which data types need to be corrected? **Which table is the bridge table?**
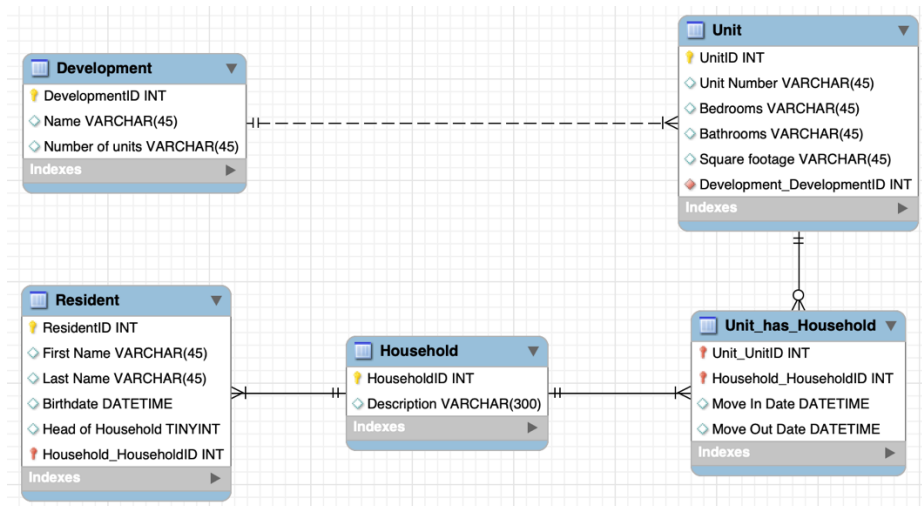
Step 8: Save your work. Click "File" and "Save As" to save your work as "ica1_part1.mwb"

Part 2.

In this part, you will work on the ***Safety Incident Database for Amusement Park*** database and specify the *maximum and minimum cardinality* of certain relations. If maximum or minimum cardinality is not specified in the description, define them based on your assumption.

Step 1: Read the description. (See the next page!)

A medium-sized amusement park in suburban Philadelphia is required to keep a record of all safety-related inspections and incidents (such as accidents) for each ride.

A ride is described by a name and type. An operator is described by first name, last name and social security number. Each time an operator runs a ride at the park, the date and shift are recorded (there is only one operator each time a ride is operated). (1) **Each ride has at least one operator, and different operators may run the same ride over time. Operators may be included in the database even if they haven't operated any ride (i.e., new operators with no ride), and an operator may run different rides.**

An incident involves both a ride and an operator. All operators and rides are included in the database whether or not they've been involved in a safety incident. (Hence it is possible that there are operators who have no incident; similarly, it is possible that there are rides without incident). When an incident occurs, its date and time are recorded along with a description of the incident.

Inspection records are also recorded in the database. An inspection occurs at a particular date and time, with a numeric score from 1 to 5, and location where the inspection happened. (2) **There is only one ride per inspection, but a ride can have multiple inspections over time. A ride may be included in the database even if no inspection has been performed yet**.

Step 2: Follow the same process as in part 1

(1) Create a new model.
(2) Create a new Diagram.
(3) Create tables for the Ride, Operator, and Inspection. Create Primary keys in them (no need to define all the columns and data types of these tables in this exercise).
(4) Create the following connections and use MySQL Workbench tools to define their maximum and minimum cardinalities:
   - Ride-Operator
   - Ride-Inspection
(5) Save your diagram as "ica1_part2.mwb"

Upload both "ica2_part1.mwb" and "ica2_part2.mwb" to the corresponding assignment on Canvas.