CECS 323 LAB DRAW.IO TIPS AND TRICKS

OBJECTIVE: Give you experience in doing a basic UML and relation scheme diagram in draw.io so that you can compare/contrast that to DIA.

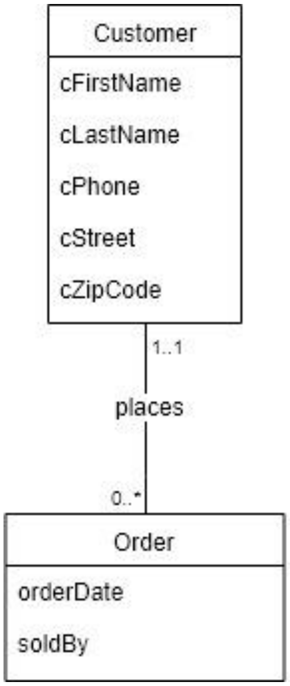
INTRODUCTION: It will quickly become evident to you that draw.io is not enforcing any of the business rules of the UML modeling languages when you build these diagrams. That must be up to you.

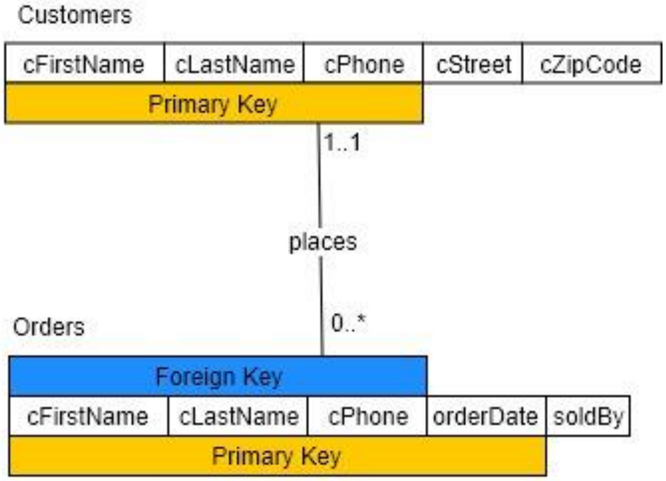
PROCEDURE:

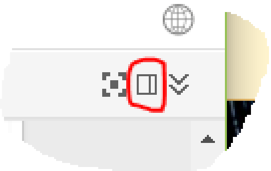
1. <Step by step instructions>

At this point in the semester, don't get too worried about the concepts behind the diagrams, we will be introducing you to those along the way. But I had to do some samples that were complex enough that you could see all of the ingredients to the models that you're likely to be doing in this class. I **will**, however, be going over a few modeling guidelines and standards because I have found that I need to repeat those before they sink in.

We are going to walk you through the steps necessary to build this UML class diagram and relation scheme diagram. The UML class diagram is the one to the left.

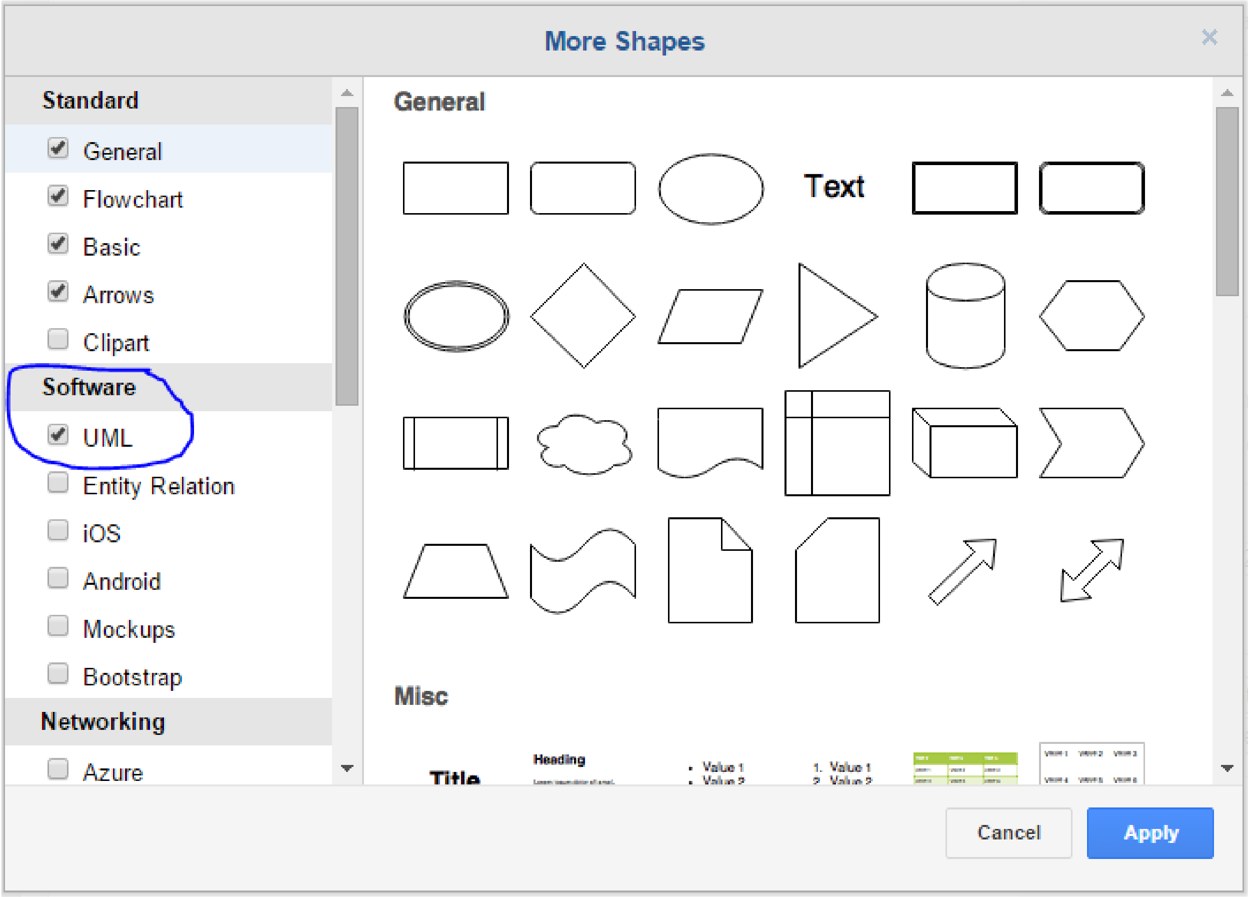




1. For starters, you can access Draw.io from: https://www.draw.io.
2. Draw.io has a format panel that you can hide/expose by using a button in the upper right-hand corner of the display: . Clicking on that toggles the format panel. Often, you will want to hide that because real estate on the screen is going to be precious. In the

instructions to follow, I will mention the format panel from time to time as a way for you to get access to configuration changes that you can make to a given object in your model.

3. We are going to start with the UML diagrams, since typically you build those first anyway.
    a. Select the UML stencil from the pulldown
        i. Select the stencil pulldown
        ii. "More Shapes" to get the following:



More Shapes

Standard

☑ General
☑ Flowchart
☑ Basic
☑ Arrows
☐ Clipart

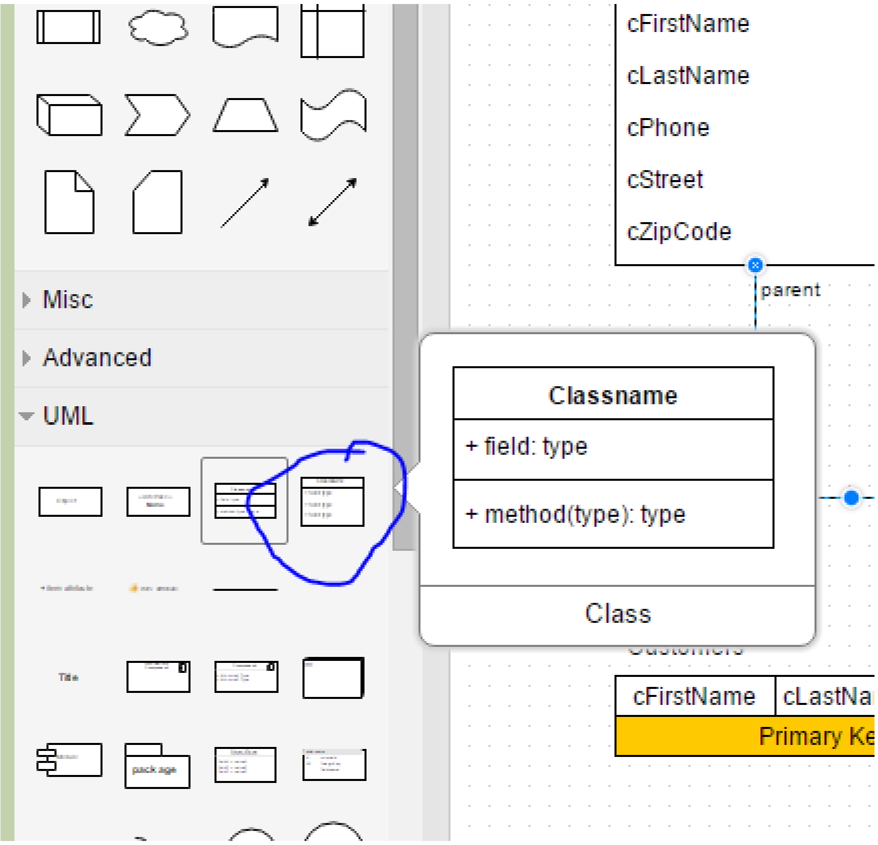
Software

☑ UML
☐ Entity Relation
☐ iOS
☐ Android
☐ Mockups
☐ Bootstrap

Networking

☐ Azure

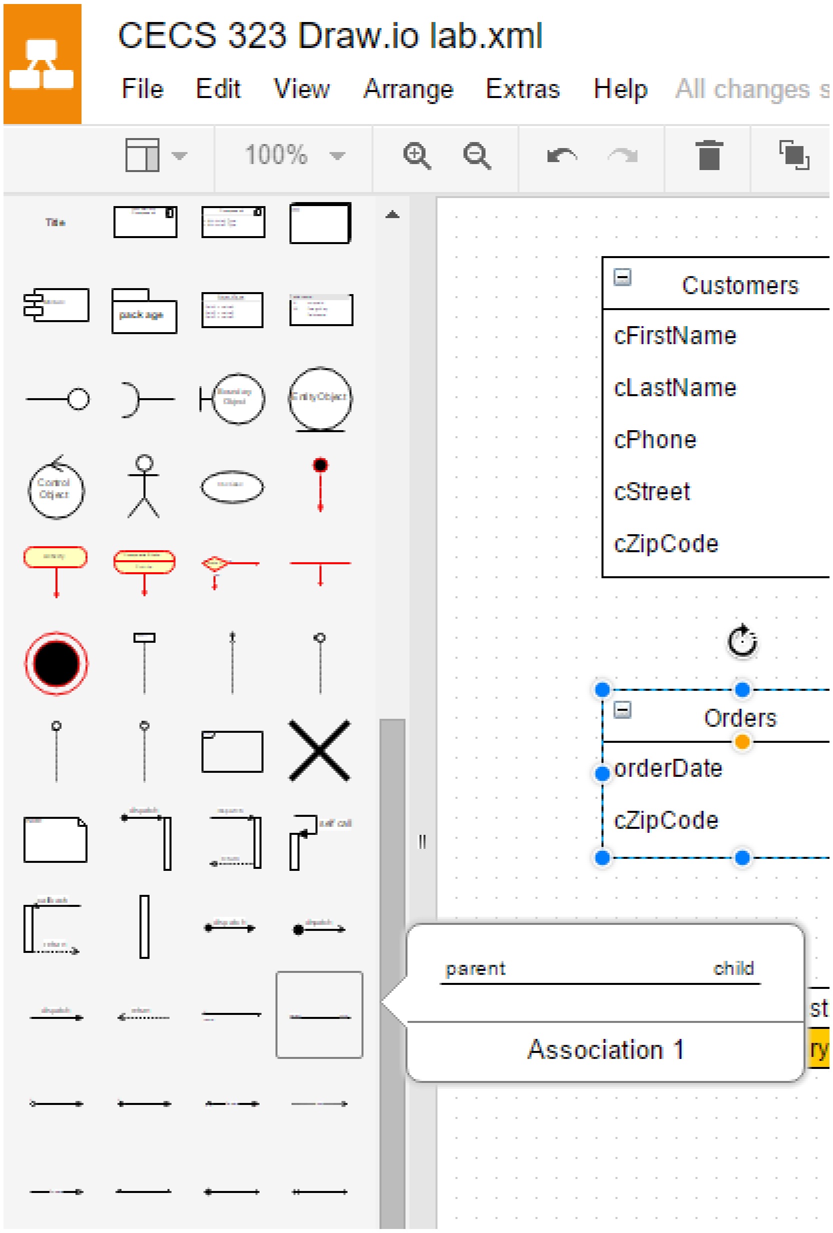
    b. The class object is the top row of this stencil.



i. Draw.io will put the class in the upper left hand corner of the canvas, so you'll have to drag it over to where you want it.

ii. This gives you three attributes by default. To get more attributes:
   1. Select the entire class (not just one of its attributes).
   2. Make the class box longer (from top to bottom).
   3. Select one of the attributes, and then cut and paste it into the same class. It will snap to a position left aligned with the rest of the attributes, and at the position that you give it.
   4. This is roughly how you rearrange the attributes of a class, just drag and drop them to their new location.

c. The association between two classes is:



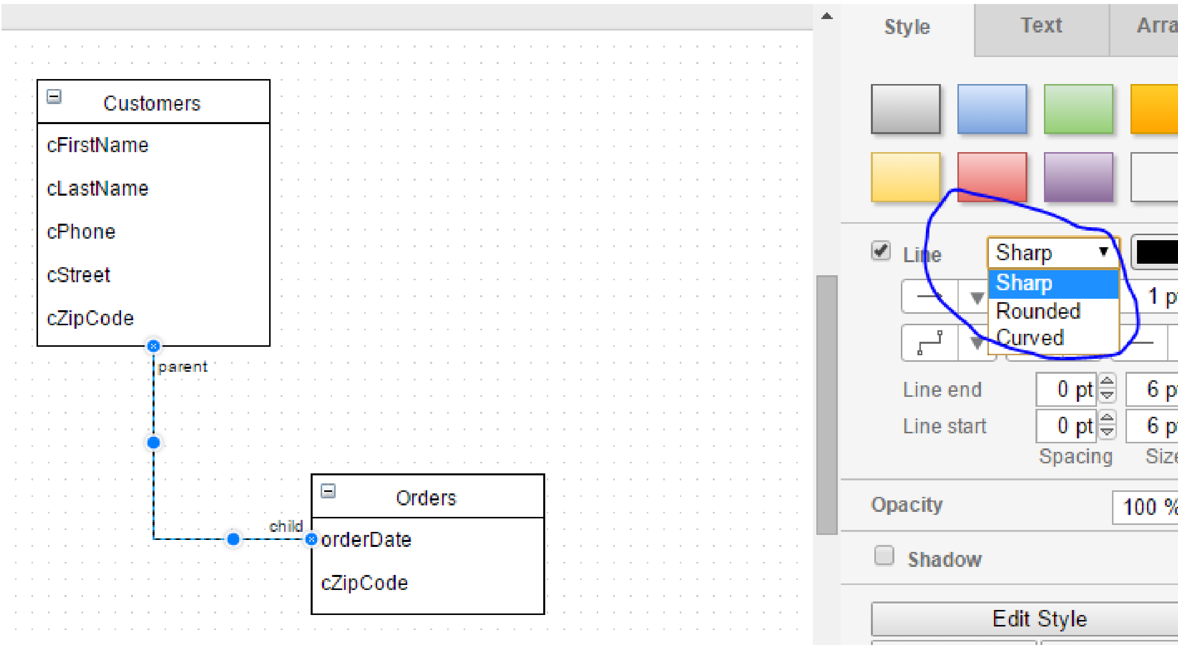
i. Just like the class, this will create an association object in the upper left-hand corner of your diagram. Drag the parent end of that over to one of the attachment points on the parent class. That will yield an association that points to nowhere.

   1. Drag that child end of the association over to the child class and attach it at an attachment point.
   2. If the association is not attached, when you move the class, the association terminus will not follow. Rather irritating at best.

ii. Then fill in the metadata for the association:

   1. The text in the middle of the association is the verb phrase, or name of the association. Update that.

        2. At either end of the association will be a text string: “parent” or “child”. Change those text strings to be the cardinality of the association. You will probably need to make the font a bit larger.
    iii. The routing of the relationship lines is a bit hard to get used to. One thing that makes it a bit easier is to override the curved default to tell it to make the relationship lines sharp:
        1. Select the relationship line, which makes a dialog box show up at the right of the canvas:



- d. Quality checks
    - i. Never forget the multiplicity of the associations between your classes.
    - ii. Never leave the associations unlabeled.
    - iii. Class names are singular, relation names are plural.
    - iv. Remember, UML diagrams are conceptual:
        1. Never put migrated foreign keys into your UML diagrams.
        2. Never put surrogate keys into your UML diagrams.

4. For a **relation scheme** diagram:
    - a. Making an attribute in the relation scheme:
        - i. Under the “General” stencil, select the rectangle and drag it to the diagram.
        - ii. In general, the shapes will have four blue dots outside of the shape. One each at the North, South, West and East sides of the shape. Left click any one of those to change the dimensions of the shape from that side.
        - iii. Click the shape in the middle and start typing to put text into the shape.
            1. You can also use the text tool (see below) for adding text to an attribute “brick”/box in a relation scheme, but that text is actually a separate object in the diagram, so it’s harder to manage should you need to push things around on your canvas.

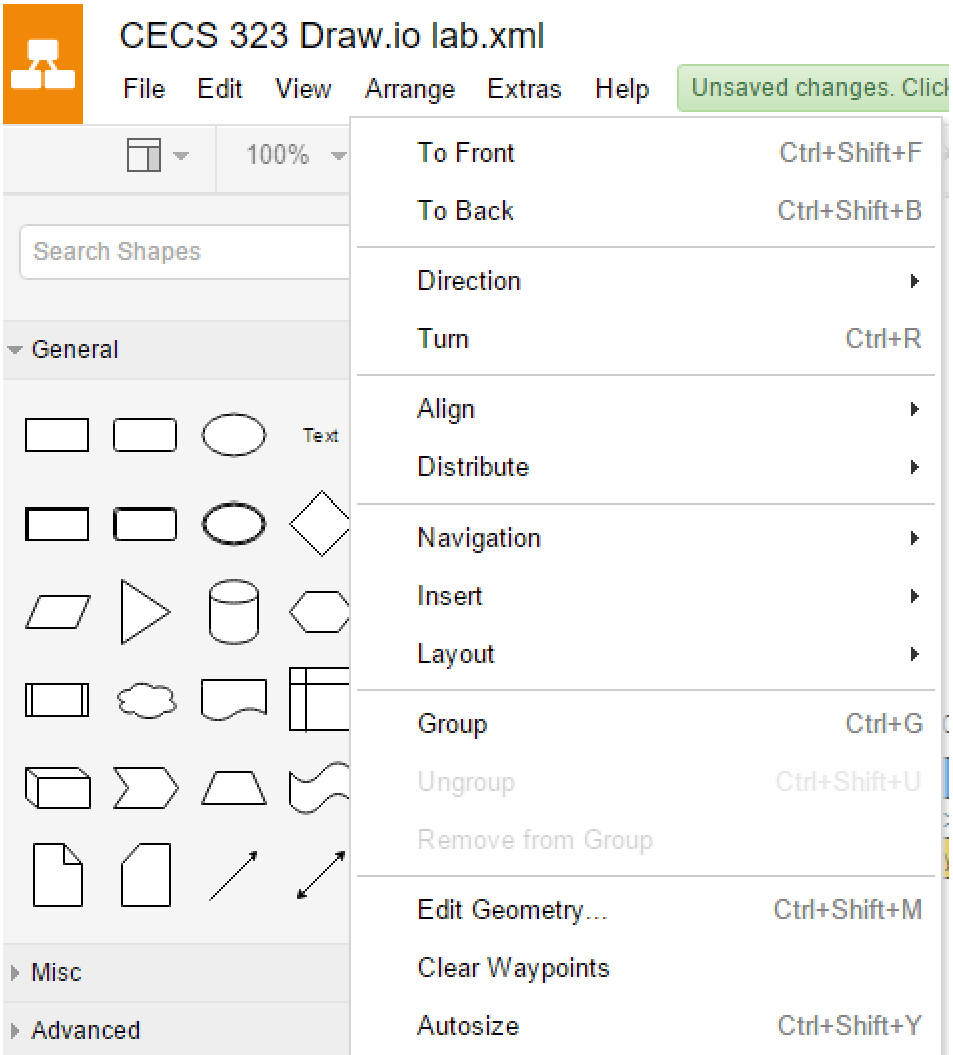
iv. If there is already text in the shape, then double click in the middle and it will select all the text. You can then use your cursor or left click once to move the current insertion point.

1. The default is for the text to be centered in the shape.
2. Up at the top, just below the menu bar, there is an icon for the text formatting that has a drop down next to it for you to select the formatting that you want the text to take on.

b. Multiple columns in your relation scheme.

i. You will nearly always have multiple columns in your relation scheme, so create additional boxes for the other attributes as needed, get them positioned, and then group them into a single object.

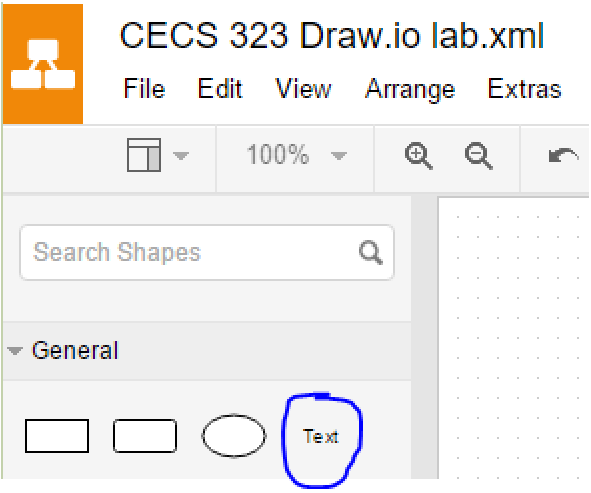
1. Be sure that you have the text (column names) that you need in each of the column boxes of your relation scheme.
2. Select all of the boxes of your relation scheme.
3. Then group them into a single object.



4. If you need to change the text, you will need to **un**group the boxes.

c. Labeling the relation will be done with a text object:

   i. Select the text object tool and drag it to where you want to place the object.



ii. The text can then be updated the same way that you do the text in the boxes…

iii. A shortcut for adorning the model with text is to simply double click a given spot on the canvas, and you will get a small text box that you can start typing into. Regardless of how you add the text, it is not associated in the model with the object nearby. Therefore, remember to select it if/when you should need to move the object that the text applies to.

d. Designating the primary key

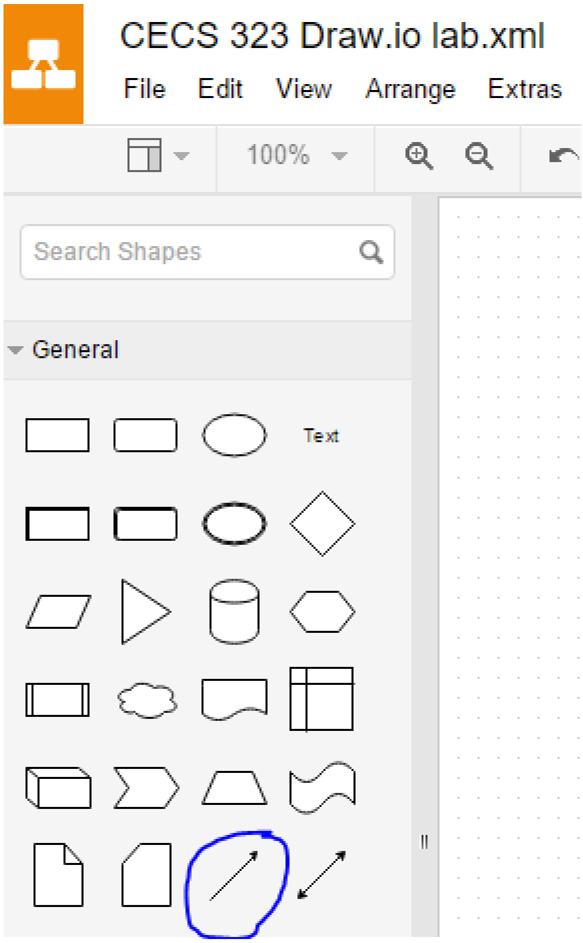
i. Create a primary key rectangle directly above the columns of the primary key. Note this may have to be broken up if the members of your primary key are not adjacent one to the other.

ii. Double click the primary key box to get into its properties dialog.

iii. Make the fill color = #FFC900.

e. Drawing the relationship line from the parent to the child:

i. Just use the line tool:

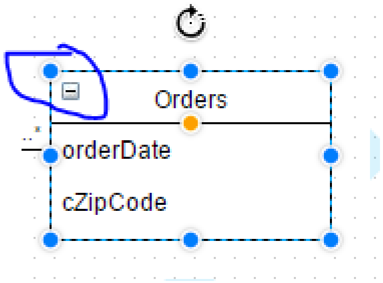


ii. The arrow should point to the child table in the relationship.
    1. This is not strictly provided for in the relation scheme diagram syntax, but I'll not worry about that if you use draw.io.
    2. The relationship gets a text label for it automatically, so be sure to initialize that text to the name of the relationship.
    3. Then use additional text elements for the parent cardinality and the child cardinality.

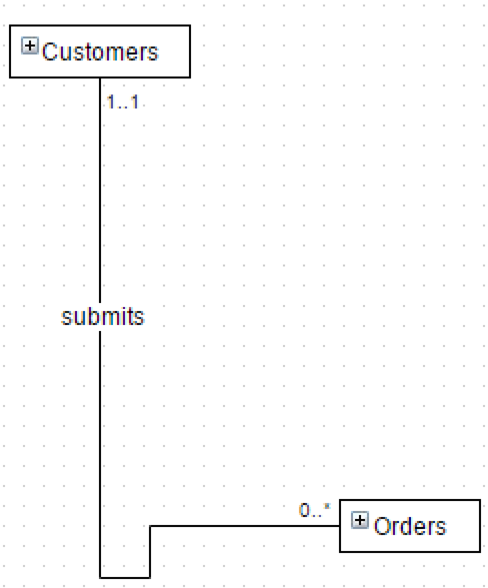
iii. Be sure to use attach points at both ends.

iv. Be sure that the Primary Key of the parent is attached to the foreign key of the child.

v. The diagrams will probably get rather busy over time, and there is a simple way to reduce that business if you need to. In the upper left hand corner of the class, there is a small minus sign:



1. If you click that, the box shrinks down to just enough room for the classname. In this example, I did that for both of the classes:



2. Just note that it messes with the association routing, and if you fix the association routing in this mode, when you expose the attributes of the class, it's going to mess with you association routing again.

f. Designating foreign key

   i. There should be one foreign key shown on the diagram for each migrated foreign key. Remember that a given relationship can migrate multiple foreign keys.

   ii. If those keys are not adjacent to each other in the relation scheme, you will have to make a box "layer" for them.

   iii. Make the fill color = #1E8EFF to get about the right shade.

g. Cut and paste –

   i. Remember that the first box is the hardest. The first relation is the hardest. After that, it's all about cut and paste to make things easier.

   ii. You might want to take an entire relation in your model and group it once you're happy with the way that it looks. This makes it easier to move all of the pieces together.

h. Modeling standards:

        i. Every relation needs a primary key, regardless whether or not that relation is a parent or not.
        ii. Draw the relationship line from the Primary Key of the parent to the migrated **Foreign** Key of the child.
        iii. Never forget the cardinality of the relationships.

5. Generally speaking
    a. Have the parent above the child.
    b. Be sure to label everything.

WHAT TO TURN IN:

- Your draw.io model (which will be an .xml file) for the Customers/Orders tables. It should look as much like the example up above as possible.