Rad Chad’s Rad Bikes

True to his name, Rad Chad is, in fact, pretty rad. He became famous for taking his bike off some sweet jumps and is leveraging his fame to start a bicycle company. He has commissioned us to build the database for his company’s information system.

After extensive conversation, our requirements gathering team has developed the entity-relationship diagrams for three different subsystems: **Sales**, **Purchasing**, and **Materials Management**.

In groups, decompose these ERDs into the proper logical model diagrams. Because we won’t have access to Draw.io to collaborate, use the following notation to describe your tables (italics used for emphasis. No need to italicize your work):

*TableName*

*Columns:*

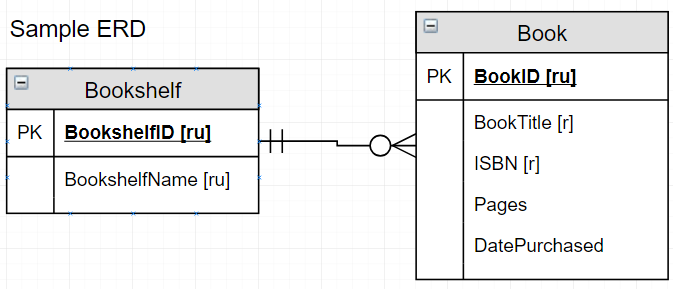
*columnName1 datatype properties,*

*columnName2 datatype properties,*

*…*

*columnNameN datatype properties*

For instance, consider the following ERD:



We will write this in our group notes as (again, italics for emphasis only):

*Bookshelf*

*Columns:*

*BookshelfID int PK,*

*BookshelfName varchar(50) required, unique*

*Book*

*Columns:*

*BookID int PK*

*BookTitle varchar(50) required,*

*ISBN varchar(50) required,*

*Pages int,*

*DatePurchased datetime*

*BookshelfID int required FK*

**Note:** We added a foreign key, BookshelfID, to the MANY side of the relationship (Book table). If it helps, simply copy the PK from the ONE side of the relationship and change PK to FK in the table on the MANY side of the relationship.

For brevity and expedience, take heed of the following notes:

* Make all alphanumeric columns varchar(50) data type.
* Addresses will decompose into **street address, city, state, zip code**
* For any currency, just use the **money** data type.
* Make all **PK** column data types **int**. We’ll see why shortly.
* Avoid adding unnecessary detail. As an academic exercise, there are some obvious things missing from this model. This is more about the systematic approach to logical modeling than it is manufacturing domain expertise.
* Choose a person to document the conversion in the notes provided in Connect. **If you’re not comfortable with the process**, **volunteer to be the typist**. You’ll find this helps with understanding of the concepts! (Feel free to take turns if you’d like)

## Here is a quick cheat sheet of the steps to do this conversion:

**1. Map Entities to Relations/Tables**

- basically, just create empty tables named for each existing entity from the ERD.

**2. Map Attributes to Columns**

- multivalued (m) attributes become new entities/tables

- decompose composite (c) attributes

- map rest of attributes

- add primary keys to tables

Recommended surrogate key template: “TablenameID int PK”

**3. Map Relationships**

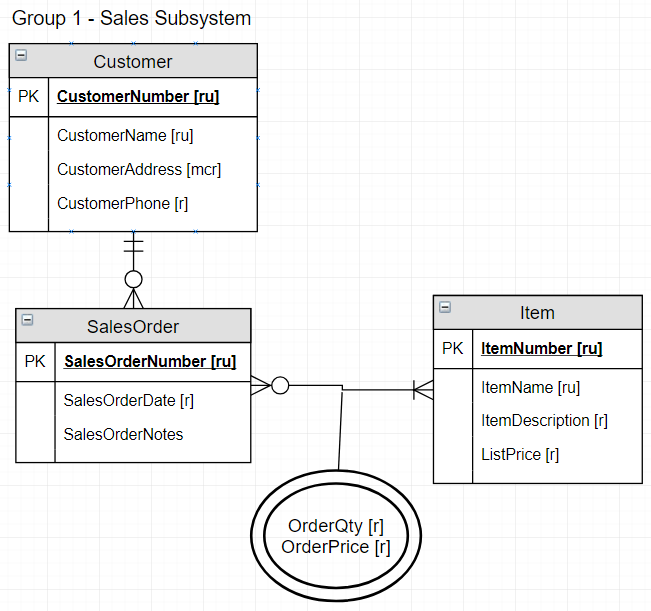
- 1 to Many maps perfectly, as is. **Don’t forget to add foreign keys to the Many side.**

- New tables from multivalued (m) attributes, are *1 to many Original table = 1 side, new table is the many side*

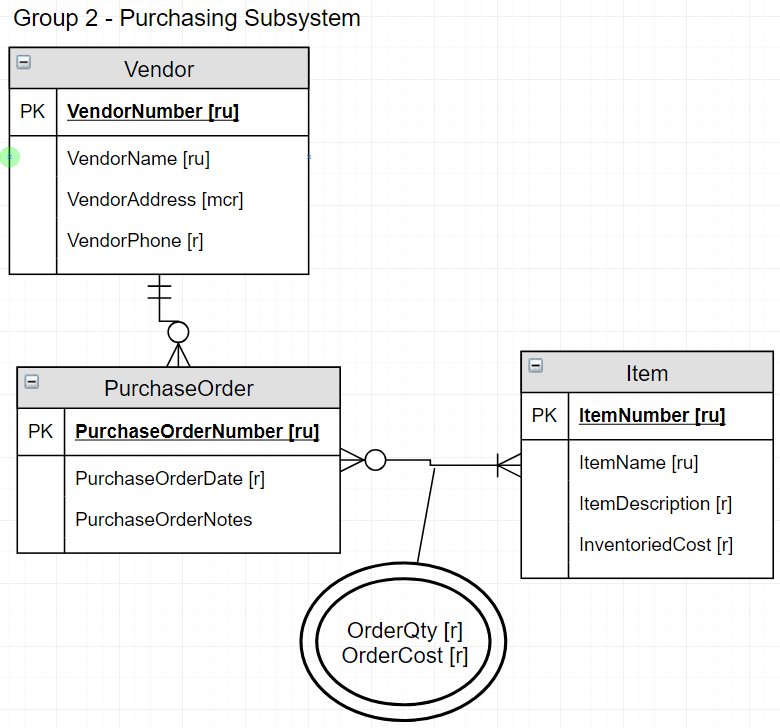
- Many to Many - Add a new associative table (aka Bridge table), *1 to many* from both of the other tables to that new table

Go to the next page for the the ERDs for each group! All of these diagrams are also here:

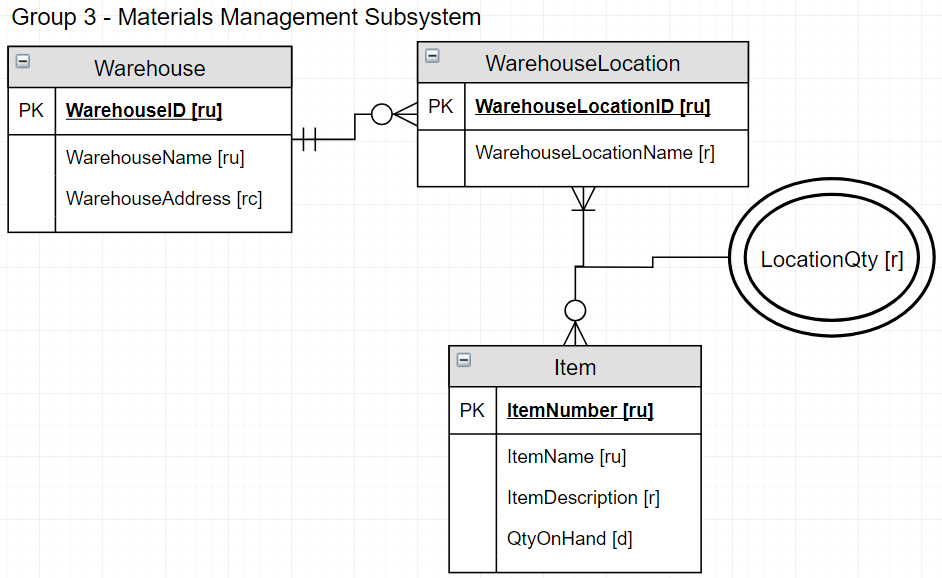
<https://drive.google.com/file/d/1dtj0U_tOcOIvHsYws00cfdbJYlSeOLOI/view?usp=sharing>



Note the attributes of the relationship between SalesOrder and Item. Add them to the new associative table!



Note the attributes of the relationship between PurchaseOrder and Item. Add them to the new associative table!



Note the attribute of the relationship between WarehouseLocation and Item. Add it to the new associative table!