**Database Design & Administration - CST8250**

**Week 6 - February 15th**

Class Intro

* It's review week!
* Lab 5 is due at the end of the week

Midterm

* 45 Multiple choice
  + Questions are shuffled
* Closed book
* 1.5 hrs in class
* 20% of final grade
  + Please don't cheat :)

Topics

* ER diagrams
  + Entities, attributes, relationships
* Normalization
  + 1st, 2nd, 3rd and BCNF
* Database design
* Indexes and Views

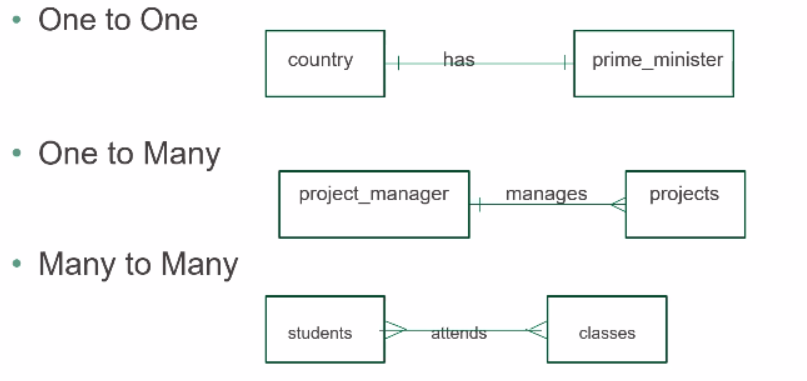
ER diagrams - Entities

* A person, place, object event or concept in the user environment about which the organization wishes to maintain data
  + Ex: book, customer etc.

ER diagrams - Attributes

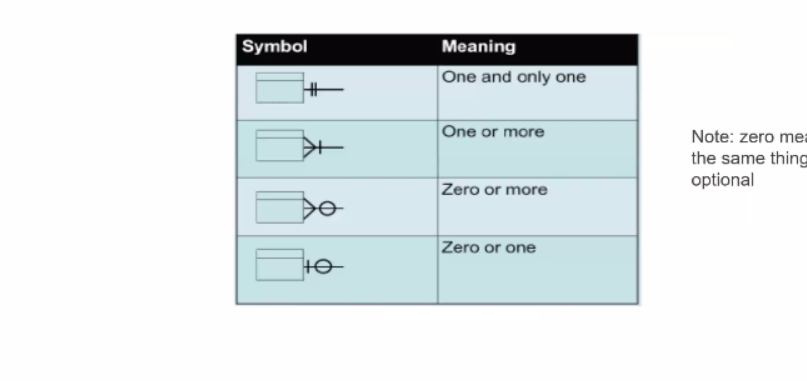
* Single valued attribute: only one value
  + Ex: SIN, birth month
* Multivalued attribute: hsa more than one value
  + Ex: phone number
* Composite attribute: has many parts, composed of many nub attributes
  + Ex: a name

Relationships



* Many to many = alabama relationship

Cardinality Notation



Cardinality

* Minimum cardinality: Minimum number of instances that can be associated with ONE instance of another entity
* Maximum cardinality: Maximum number of instances that can be associated with ONE instance of another entity

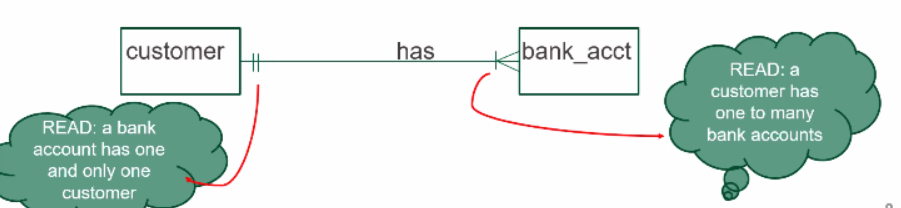
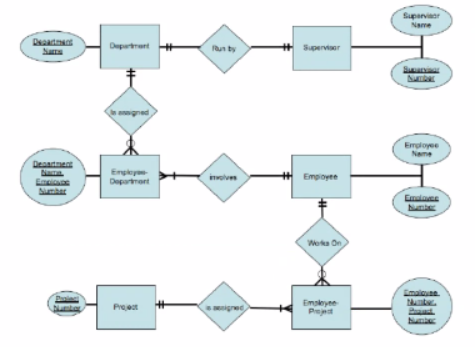


Diagram Example:

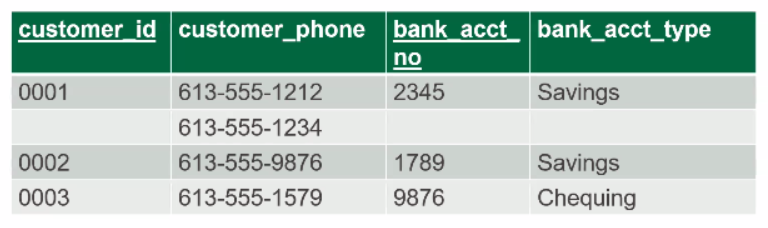
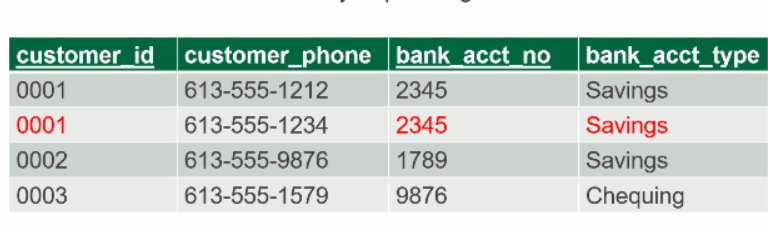


**Normalization Review**

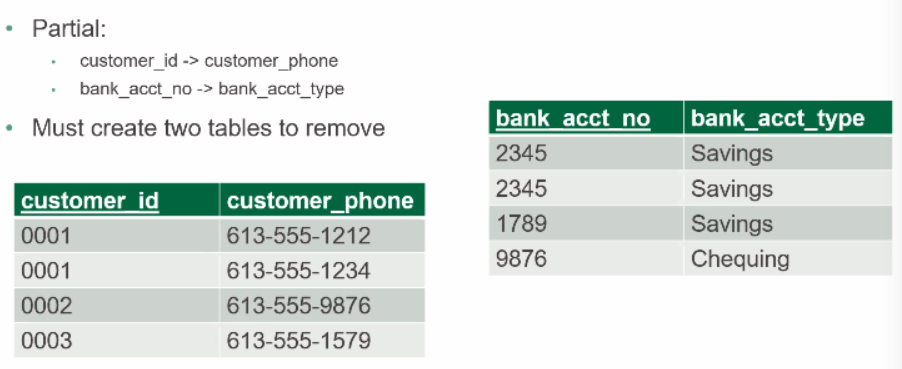
Normalization Goals

* Reduce redundancy
  + Makes databases more efficient, avoids duplication of data, uses less space
* Remove anomalies
  + Insertion anomaly: inserting data to force duplicate data
    - Placing an order and having to create duplicate customers for same customer
  + Deletion anomaly: removing forces user to lose data
    - When deleting a student leads results in the deletion of a course
  + Modification anomaly: updating data forces user to find and update additional row

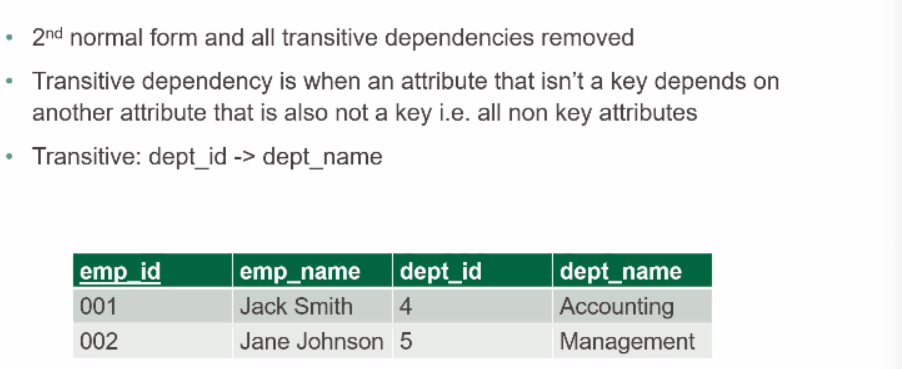
1st Form

* Need to identify and remove multi-valued attributes
* A table with multi-valued attributes will have rows with NULL
* Candidate keys have to be identified
* 
* Remove multivalued attributed by duplicating data
* 

2nd Normal Form

* Must be in first normal form
* AND partial dependencies must be removed
* Partial dependency is an attribute
* 

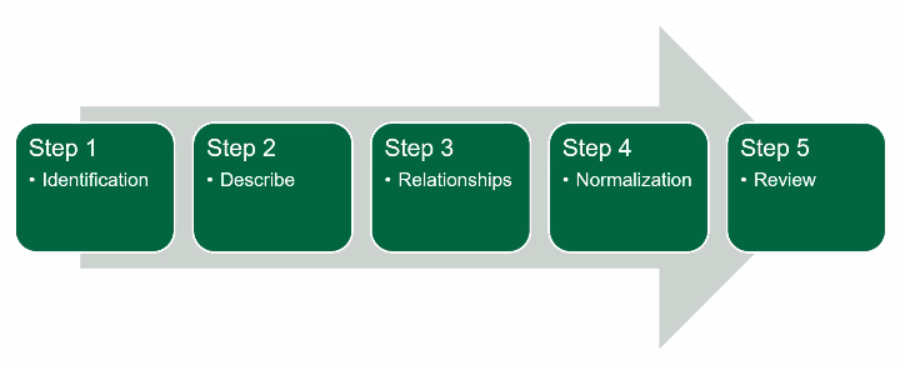
3rd Normal Form

* 
* Now both tables are 3rd normal form and dept\_id is no a foreign key in the new Employee table to preserve the relationship

Boyce Codd Normal Form

* This is more of a theoretical concept and is not widely used in practice
* 3rd normal form is generally sufficient

Design step process



**Indexes and Views**

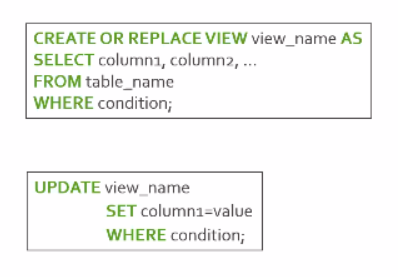
Definition and types

* Data structure that helps find location of data easily follow for effective query of data
* Overuse of indexes can slow database down
* Unique: all primary keys are indexed, can also create unique keys on other data
* Non: unique: typically information that is accessed frequently but not a primary key
  + Usually for information that is accessed frequently like a phone number
* A view allows information to be presented differently to different users ie groups of users may have a separate view
  + Useful for controlling access to data
* Virtual view: used in databases, computed on demand
* Materialized view: used in data warehousing to help speed up queries as it is precomputed offline

What happens when we query a view?

* A view is not a true table, it does not contain data
* When we query a view, it accesses the source table to capture the data
* This means that is data needs to be modified, it must be modified at the table level
  + Very much like a pizza pocket: pizza relationship

View Queries



Non Updatable Views

* A view is not always updatable – most of the time
* Specific criteria in a query can make a view non updatable
* Cannot apple UPDATE, INSERT, DELETE by the view
* GROUP BY
* Use of aggregate function such as COUNT()
* DISTINCT

Normalization Example

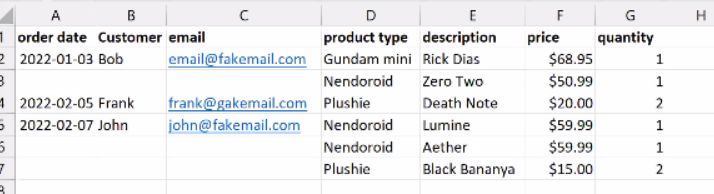
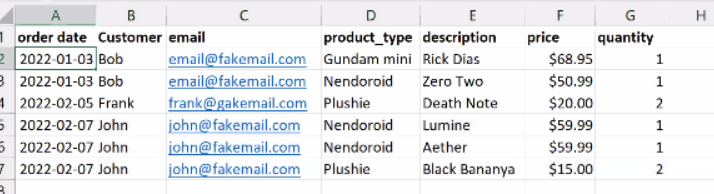
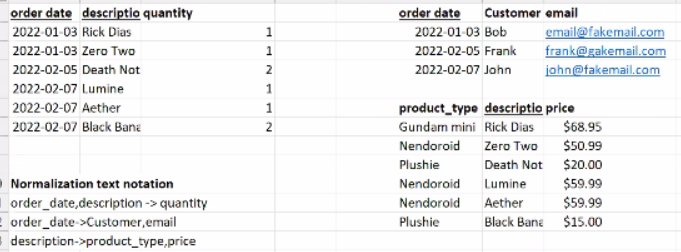
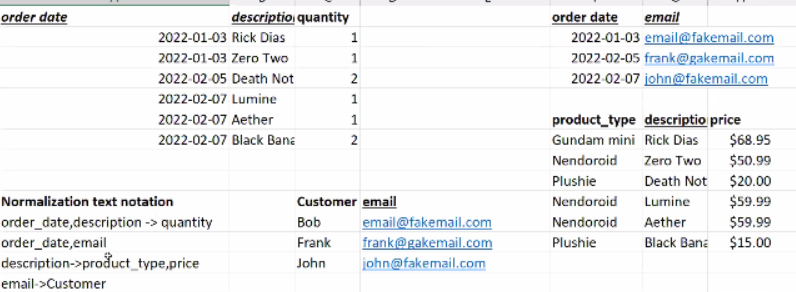
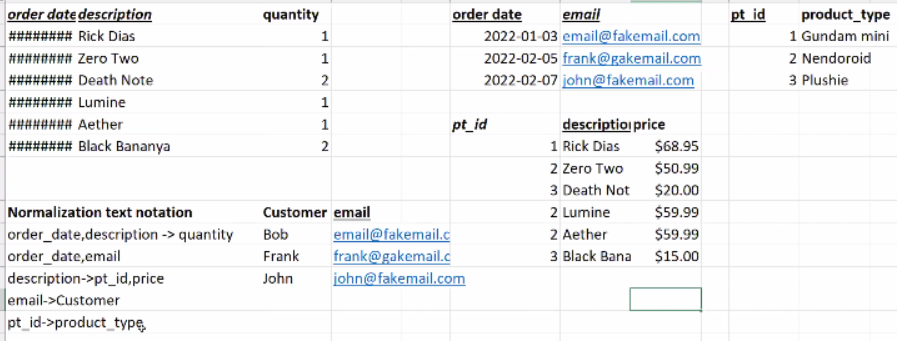
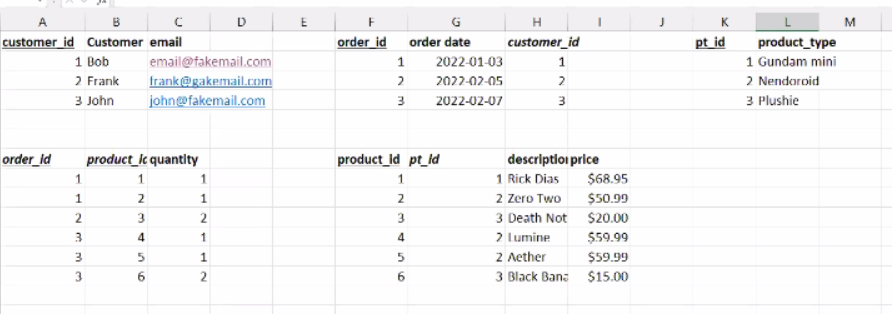
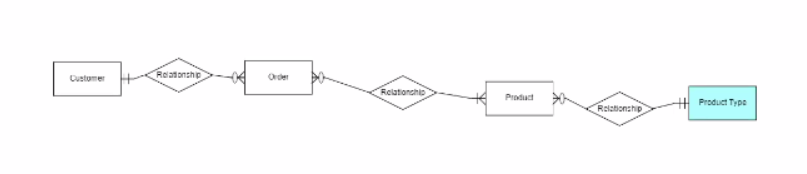
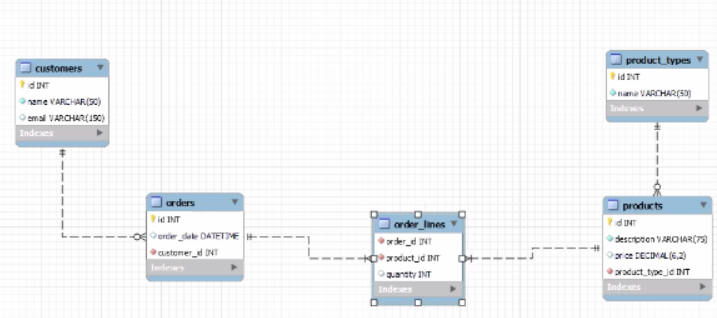
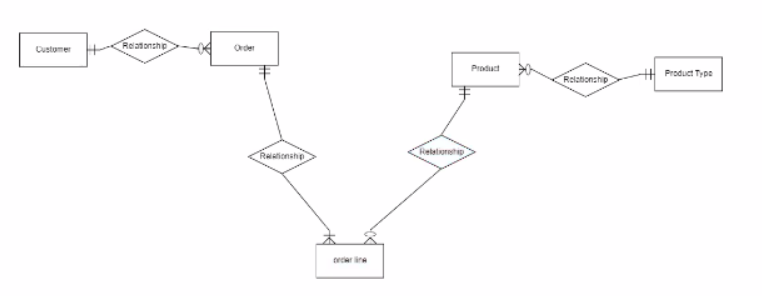
* This is an ordering system
* **First Normal Form:**
* Starting with un normalized data: order data and multivalued data
* 
* Populate the nulls with what is missing above it
* 
* You can uniquely identify each row by combining description + order date
* **Second Normal Form:**
* The customer email depends on the order, description depends on the product, and quantity depends on the description
* Create a table with the order, description and quantity. Create another table with an order date and customer email. Create another table with the product, description and price
* 
* **Third Normal Form:**
* We need to get rid of transitive dependencies and will not worry about duplicated data yet
* Create a customer/email table
* 
* This is a Boyce Codd situation: if you change the product type you have to update in multiple places
* Create another table for product type and give it a synthetic key
* 
* We still can’t place more than one order per customer per day
* You would also need to change the customer email in more than one place
* Get rid of the “email” because people change their emails frequently so replace with a key
* 

Diagram example:

Without associative entities:



With associative entities:



* Use a past tense verb for a date/time field name
* A deleted field can also be very helpful to keep record