Lecture 4 Notes - Monday 09/12/16

Reading Quiz:
Question 1: Ans = A
Question 2: Ans = A
Question 3: Ans = A
Question 4: Ans = B
Question 5: Ans = C

Notes:

We use arrows to show subtype/supertype relationships in conceptual models, but this is just a design concept, note some sort of data type / data structure -- the DBMS doesn't know what a "subtype" is; the DBMS sees a subtype as just another table. The subtype/supertype structure is something we impose on the database

Concept Question 1: A. all these answers "make sense" in a way, but the best answer is **A**; all relational tables must have a primary key, and having a primary key would have the effect of requiring customer_id to be both unique and not null. This would also keep us from having multiple records for the same customer (unless, of course, one customer had more than one id somehow). So **A** covers all of these issues

Concept Question 2: B. Note in the diagram that both *Commercial* and *Non_Profit* have relationships with only the *Organization* table, so this must be the table to which their foreign keys are referring. Note this is consistent because we see that *customer_id* is in the *Organization* create table statement, and will uniquely identify a row.

Concept Question 3: B. Note that queries (1) and (2) only make you look up two tables: query (1) makes you look up info in the *Individual* table and "connect" it to some other info in the *Customer table* via a foreign key; similarly, query (2) makes you look up info in the *Organization* table and "connect" it to some other info in the *Customer table* via a foreign key. But if you want to look up all customers based in Austin, you're going to have to look in all three tables, because we could have both individual and organization customers in Austin. This way, you have to look through all the customers in the *Customer* table and check both the *Individual* or the *Organization* table for further information (because the *Customer* table doesn't tell you what type of customer it is) if the customer is in Austin. So this will take the longest.

Note we could add a *customer_type* identifier with a check constraint in the *Customer* table to improve performance on this type of problem. (see slides) This column will tell us whether a given customer is an individual or an organization, so we know which table to check for further info. But you should be careful about using this type of solution. It's only a good idea when the

column values are static rather than dynamic. If there's a possibility that we add other customer subtype or change the type of a customer, this could be bad.

Concept Question 4: E. This table has some bad redundancy -- consider the case where a person has both a non-commercial and a commercial driver's license. This person will have to have two rows in the *Driver* table to account for both license types. We tend to think the data would be better organized if each person only had one row in the *Driver* table. The child tables' foreign keys are also problematic because they only refer to *part* of the primary key in *Driver* (*ssn*). If a person has both types of license, how do we know which record the foreign key is pointing to?? Our solution would be to add an additional value to the *driver_type* check restraint, NC, which symbolizes that the driver has both types of license. This way we can reduce the primary key in *Driver* to *ssn,* and the child tables can have clean foreign keys pointing to *ssn*.

Concept Question 5: A. Recall a database schema is in 1NF if and only if all attributes have scalar values (this means no "complex" data types -- no lists, arrays, dictionaries, etc.). We can see that this is true of the table shown, so it must be in 1NF. (Note that *drug_nbr* and *start_date* are underlined on this table. That means that these rows form the primary key, which is composite in this case.)