Kevin Cozart
CS355
HW1 prob 1 – 7

1. Three characteristics of database approach are Self-Describing nature, Multiple Views of data and isolation of program and data. The self-describing nature is exactly what it seems like it is. The data item names and data values are stored together in such a way that makes it easy to distinguish what you are looking at. Whereas with a traditional file system, data definition is typically part of the application programs and not in the data. The multiple views of data makes the database available to anyone who needs it and is able to limit or expand permissions for the users on an individual basis. With a more traditional file system this needs to be built in and wouldn't always allow for multiple permissions or users at once. The isolation of program and data allows for a more robust processing of data without risking the corruption or removal of important data from your database. Traditionally you would end up having both the program and data in the same application which can cause issues if an unskilled user input improper commands. An example of a self-describing nature would be naming the employee's name field something like EmployeesFirstName so that without looking at the data you can know instantly what it is and how it relates to other tables and fields. With the multiple views of data you end up being able to allow many users with varying access. We have a water billing program that is written in Access with an SQL backend at one of my clients, we have two versions of it, Live and day old data, which allows those less savvy to not mess up the live database while still being able to pull the information. We also have different permission for each user so when they access either of these two databases through our GUI they are able to not mess up the data unless they try. Isolation of program and data example would be the web hooks we have setup on one of our client's websites that displays rainfall data over certain period that is pulled from a SQL database and securely posted online. Without the ability to separate the website data from the actual database and have them be secure we would never be able to get accurate data without people being able to edit our rainfall data.

2. Facebook would have a table for UserInformation (Name, age, and such personal data), Associate pages (Who are your friends you are connected too) and probably a psychological profile information tab similar to Cambridge Analytica used (likes, dislikes, political associations, etc.).

3. Data models is the structure of a database and defines how information is accessed, updated, removed and stored. CRUD and ACID. Examples: Hierarchical, Relational, Network, etc. We will be focusing on Relational database modeling.

4. DBMS allows us to implement in many layers. Physical data level which describes how the data is stored in the database, the logical level which describes what data is in the database and the view level which allows for users to interact with the database without knowing how it works. These levels of abstraction allow for separation between user and direct data access and significantly reduces the learning curve for usage of a database.

5. Fusion tables would be considered at database due to its flexibility in being able to collect, arrange and view data that is stored on multiple tables. Excel could be considered a database but in a loose sense. You are able to separate your data into different tables and access other tables through VBA scripts similar to a database, however it isn't strickly a database system as it doesn't allow for multiple users at a single time for an excel document. Also from an IT perspective, linking excel documents to other excel documents is a pain in the ass to manage cause everyone leaves the documents open on their machine, goes to lunch and locks the linked tables from that file for everyone else who needs to use them.

6. You can assume that, but you would be wrong, I would be more inclined to utilize the ID section as long as these entries are unique, mainly due to if there is a change of department chair you might end up with a duplicate name entry but you shouldn't ever get a duplicate ID scenario unless you or the administration did something really bad.

7. a. Strong Entities: BANK, ACCOUNT, LOANS, CUSTOMER

   b. Weak Entities: BANK_BRANCH, Partial Key: Branch no, Relationship: Only one bank, but many branches.

   c. The Partial Key shows there are many branches each with an individual number. The relationship shows there is ONLY one bank and MANY branches for that bank.

   d. Bank <-> Branches : 1 bank, many branches (1,N) – 1 to Many
   Bank_Branch< -> Account: bank has many accounts but only 1 bank is giving them - (1, N) - 1 to Many
   Account <-> Customer: Customer can have many accounts accounts can have many people on them - (M:N) – Many to Many (Joint Accounts)
   Bank_Branch< -> Loans: Bank has many loans but only one bank is giving them - (1, N) - 1 to Many
   Loan <-> Customer: Customer can have many loans and loans can have many people on them (M:N) – Many to Many (Cosigned loans)

   e. Bank has a Code (Primary), Name and Address
   Only One Bank but Many branches
   Branches have a Branch No(Primary) and Address.
   Bank Branches create Accounts and Give loans to Customer
   Bank can only Loan its money and no other banks money.
   Loans have Loan No(Primary), Amount and type
   Accounts have Acct No (Primary), Balance, and Type
   Each Customer has SSN(Primary), Phone, Name and Address
   Customers can have Many accounts or Loans