**JEREMY PETERSON DUTOYA**

**DR. MOHAMMAD AWWAL**

**PMBA-8317-RHA APPLIED DATA MGT FOR BUSINESS USERS**

1. Describe the characteristics of a table that make it a relation. Define the term domain and domain integrity constraint to a relation.

To be characterized as a relation, a table has to respect certain characteristics. First, the table has to be a two-dimensional table composed of rows and columns. Second, data regarding an entity is contained in rows, and information regarding the entities' attributes is contained in columns. Third, each column of the table must have a distinct and unique name. Fourth, a column contains only entries of the same data type (number, char, varchar..). This indicates, in relational words, that all entries in a column originate from the same domain.

A domain is a collection of data that adheres to a particular type of definition. In other words, it is a grouping of possible and accepted values for a column. For instance, a student id column will only include a particular student id number. Never does it include extra details like age, date of birth, or name. For this reason, the term "domain integrity constraint" refers to the demand that each value in a column is of the same type. Because users and developers may be certain of the type of data present in a specific column, this characteristic makes data access simpler. It also makes data validation simpler. Given that all values come from the same domain, the database software's Data Definition Language (DDL) can be used to define and enforce the domain.

Fifth, the Cells of the table must hold a single value. Sixth, it doesn't matter how the columns and rows are arranged. Columns must be referred to by name rather than position since the order in which they appear is unimportant. Seventh, the table can not contains duplicate rows. Last, Each row in a table needs to be uniquely identified by one or more attributes, which is known as the primary key.

2. What is a primary key? Explain the difference between a candidate key and a primary key.

SQL databases place a great deal of emphasis on primary keys and there is so much to say about them as they provide each row in a database table with a special ID. You can use a column to generate a primary key once you've found one that differs in value for every row in the database and that can serve to identify each row of the table. In short, the Primary key is the "main identifier" for each row in the database. When using primary keys, a technical check is made to see if the primary key column in each row contains a UNIQUE and NOT NULL value. For instance, the primary key constraint will reject a new row if you attempt to insert it with a duplicate value in the primary key column. Primary keys are very important to ensure the concept of data redundancy and integrity.

To explain the difference between a candidate key and a primary key, let’s go straight to the point. A candidate key is a key (one or more columns that are used to identify particular rows in a relation) that determines (or has functional dependencies with) all of the other columns in a relation. A Primary key has this same characteristic, so a primary key is a candidate key selected as the primary and only means of identifying rows in a relation. Let's see it this way, a primary key is always a candidate key but a candidate key will not be a primary key before it is chosen as the only mean of identifying rows in the relation. For example, if a database is not in 3NF, the candidate key will not be the primary key because it contains a key that will not be there anymore after normalization.

3. What is the significance of normalization in database design? What conditions are required for a relationship to be in 3NF?

In relational database design, normalization is a step-by-step method for analyzing and reducing a relational database to its most streamlined form. In other words, this technique facilitates the proper organization of data tables. The procedure seeks to produce an optimal database system that accurately reflects data and relationships by evaluating and correcting table structures to ensure minimum data redundancy or data loss and maximum data integrity. When a table's properties depend solely on the primary key, this is referred to as normalized data.

To be in 3NF, a table has to be a relation: that means it has to respect the characteristics explained in question 1: Atomic values in columns, same domain, columns have unique names, and row uniqueness ensured by primary key. But in addition, it has to present no partial dependencies (some columns of the table only depend on a part of the primary key - this can not happen if the primary key is not a composite key) and no transitive dependencies (a column depends on a key that is not the primary key).

4. Discuss Entity Relationship Diagram (ERD). How does it help developers and business users to understand database design?

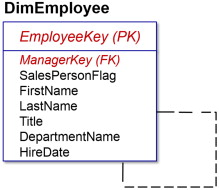
An entity Relationship Diagram also referred to as ER Diagram, is a structural diagram used in database architecture and design. The various symbols and connectors seen in an ERD, as its name suggests, aid in the visualization of two key pieces of information when creating a database system: the principal entities included and represented in the system's context and their interrelationships. When we discuss entities in ERD, we frequently refer to business objects like persons and their roles (for example, students, employees, customers) and tangible business items (products, orders). In the context of the database system, "relationship" refers to how various elements connect with one another. An ERD is also an efficient way to give more information about the cardinality and optionality of the relationship as well.

The entity relationship diagram is very useful for developers and business users to understand the database design because it gives a graphical representation of the way the database is structured and how the entities are linked to each other. It is important during the creation or alteration of the database because at this very moment, by creating ER diagrams to represent database design concepts, the team can come together and see errors (entities or relationships or attributes missing or not defined correctly) and design flaws (problems in the cardinality or optionality of the ERD relationships) and make the appropriate adjustments before implementing the modifications to the database by explaining the goals for the database system to achieve. Also, when a database has several tables and you need to write complex SQL queries in order to retrieve the information you need, it might be difficult to have a correct view of the design of the database. An ERD gives you a complete view of the entire database schema. It is now simple to locate entities, evaluate their attributes' names and domains, and determine the connections they have with other entities. All of these make it easier to assess and query an existing database by explicitly representing each entity's attribute name and make it easier to formulate certain queries such as a JOIN for example as we have seen in class that it is hard to retrieve the common attributes that serve as the foreign key.

5. What is a recursive relationship? Explain it with examples using ERD.

Before we get into too much detailed and complicated information, let's give a brief and simple definition of a recursive relationship: put simply, when an entity type has a relationship to itself, this is known as a recursive relationship, also known as a unary relationship. The very same entity is both the parent and the child in this nonidentifying, optional special type of relationship.

I took the same example we had in class for this question as the relationship between employees and managers is a perfect example of this type of relationship. The manager who oversees multiple other employees falls under the category of employees himself, so he has all the attributes of the employees he oversees, this means that his employeekey is the managerkey for all the employees he is supervising. In other terms, his primary key is the foreign key for his employees which will help identify him as their manager in the database system. There are many attributes in the employee table, notably ManagerKey, which is really just another EmployeeKey. That manager is an employee, and employees often advance through the organizational hierarchy from one manager to the next. “The parent entity primary key has shifted to the non-key area of the child entity instance and became a foreign key.”

****