Ryan Somma
StudenID: 9989060874
Ryeguy123@yahoo.com

Assignment #1

1) Explain why database design is important?

The primacy of database design is most keenly illustrated by looking at how data was managed with File systems before the advent of relational databases, when data was stored in solitary files without standards across departments within organizations, making it difficult to produce reports compiling data from disparate data sources, and requiring data processing specialists to write custom applications to generate reports. A poorly designed database can include all the problems a File System presents, were tables in the same RDBMS are not related properly, organized inefficiently, or organized in such a way as to allow for the same data to exist in multiple locations, providing an environment where the data may not get updated in one place after being updated in others.

Good database design must facilitate the production of information from data in a relational database. It will provide table structures related in such a way as to allow using Structured Query Language (SQL) and RDBMS interfaces, so that users may manipulate data, coax it into different structures and formats according to the organization's specific business needs. A DBA should be able to take an organization's requirements and properly interpret them into relational table structures that will ensure data integrity, prevent redundancy of data, and prevent islands of data, while providing a data environment that will inform the organization's decision-making. A well-designed database will perform efficiently in data retrieval and manipulation, and ensure data integrity through normalization and rules to enforce business logic.

2) Discuss the various database models?

File System: While not an actual database model, it is important to include File Systems as a predecessor to the more formal database models. In this system, data is stored in individual files that handle department-specific data needs. There are no relations between files, except where a data processing specialist has written applications to draw such associations.

Hierarchical: In this model, a series of top-down 1:M relationships are established to relate data into parent-child relationships, where each parent may have many children. While this model establishes relationships between tables, called "segments" in this model, it does not provide an easy way to establish more complex relationships.

Network: This data model was intended to overcome the lack of support for complex relationships provided in the Hierarchical model, allowing child records to have more than one parent. The Network Model also establishes standards for concepts like the schema, the organization of the database as viewed by the DBA, subschema, the database as viewed by applications, and DML and DDL languages to define the data management environment and the schema components respectively.

Relational: This model introduced the Relational Database Management System (RDBMS), which hides the complexity of the relational model from users, managing the physical details of storing data in relations/tables. The relational model introduced more complex relationship types among tables, with primary and secondary keys, foreign keys, and using relational algebra for creating new relationships between data at the time they are needed.

Entity-Relationship: While the relational database model was excellent for storing, retrieving, and describing data in relations, it was conceptually difficult to work with for database modeling. The ER model was intended to overcome this conceptual difficulty by describing data in terms of entities, which are anything about which data are to be collected and stored. Entities are related to other entities through connectivity labels. ER diagrams illustrate database entities and relationships in an easier-to-understand visual format.

Object Oriented: The OODM defines entities in a semantic format, providing, not only the meaning of relationships between entities or objects, but the meaning of attributes and relationships within the object itself. Objects extend the concept of entities by including operational procedures that may be performed on the object as well. As a result, objects become a basic building block within the model, encapsulating attributes, relationships, and functionality within its single unit.