Object-Relational Design Assignment

Neha Tarey

Grand Canyon University: MIS-610

May 1, 2019

**Object-Relational Design**

**Question 1**  
Describe simple and complex data types. Provide examples of each and describe their typical uses in a business environment.

**Solution**

According to Hoffer, Venkataraman and Topi (2017, p. 249), a data type is a detailed coding scheme to represent organizational data recognized by the software systems- DBMS in this case. A data type selected should be able to represent all possible values, improve data integrity, support all data manipulations and minimize storage space. The method of storing data types and speed of accessing those depend on the specific DBMS in use.

Simple data types are the ones that come built-in with the DBMS. For the simple types, DBMS provides basic functions (operations) like read, value comparison (=, >, <), Boolean (AN

D, OR, NOT), arithmetic (+, -, /, \*) and many other functions. SQL Server offers the data types: DECIMAL, NUMERIC, FLOAT, REAL, INT, SMALLINT, TINYINT, CHAR, VARCHAR. For example, in the Adventure Works database, Production.Product is composed of simple data types like smallint, money, nvarchar, etc. Most of the times the basic data types suffice the need in the relational data model.

SQL Server and other DBMS also support user-defined types (UDTs) that are built using basic data types. They help in modifying the definition of a data type; for example, the database should only accept EMPNo column values in 20-digit format, user-defined data types help in creating such business rules. According to SQL Server documentation(“Data types”, 2017), “User-defined types obtain their characteristics from the methods and operators of a class that you create by using one of the programming languages support by the .NET Framework.” The Adventure Works database has several UDTs: dbo.AccountNumber, dbo.Flag, dbo.Name etc.

The UDTs can be considered complex types, as all of their operators need to be defined. However, SQL server provides some system-defined complex data types as well (XML, Spatial Geometry types, Spatial Geography type, hierarchyid, etc.) and they are useful in the image, video, XML data processing. The spatial data types are purely implemented as .NET common language runtime (CLR) data types in SQL Server.

**Question 2**

Explain what an impedance mismatch is and why is this important in analytics. Discuss measures a business can institute to address this problem. Describe barriers that might exist for a business to institute such measures.

**Solution**

During the application design and development phase, while persisting the object, there is no direct, one-to-one mapping present between the class objects (considering only class attributes) and the table structure, this problem is called impedance mismatch. Both the relational and object-oriented paradigms follow a different approach for data modeling, also they are based on different principles. While the relational model uses ERDs and is based on mathematical principles, the object-oriented model uses UML class diagrams and is based on software engineering principles (Wambler, n.d.).

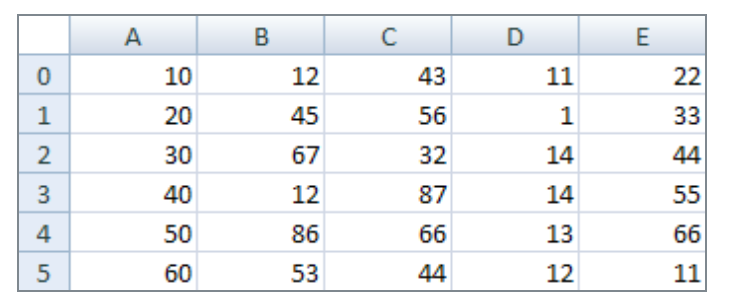
There can be the following solutions to address the problem of impedance mismatch. First, use Object wrappers to handle the differences between class objects and relational tables. Object wrappers are classes and methods providing the object-relational mapping between application objects and database tables; this mapping is performed in the database servers (“An ORDBMS that is truly relational”, n.d.). Another similar approach is to define a persistent layer as suggested by Oracle TopLink Developer website (2005), an in-memory XML schema transformation mapping between the Java objects and database element can be created and stored- this should help in effectively handling the persistence of the objects. Relational domains or UDTs are another solution as suggested by Firstsql website (n.d.), creating user-defined data types and then handling their behavior by writing methods in an object-oriented language. Using the UDTs, one can implement object-oriented features like inheritance, encapsulation, polymorphism, etc.

Implementing the above solutions may not be quite easy, as each of these solutions has their challenges. The in-memory XML schema transformation is a resource-intensive approach, every time an object needs to be persisted, the transformation must run- unless some caching mechanism has been implemented. Furthermore, implementing any other solution would face backlash due to the difference in the fundamental modeling and design principles. The staff and ownership of both the technologies in an organization are generally different, causing communication gaps.

**Question 3**

Explain an array and its pros and cons. Explain why this structure may be preferred for storing/referencing data.

**Solution**

An array is a data structure that can be used to store primitive data types, complex objects, and even references. The objects stored in an array can either be all of the same type or different types. Mostly, all the programming languages provide some support for arrays. Arrays are basically like indexed lists, each of the elements of an array can be accessed individually by their position in the list, and the index starts from zero. They can be one dimensional or multiple dimensional; two dimensional arrays can be used to store information looking like a basic table structure or a spreadsheet like following: 

Arrays are stored as a single block of memory, and the data elements are stored contiguously in that designated block (Brind, 2018). They are one of the most efficient data structures as only data is stored, and random access allows for faster access to individual elements. Elements can be added or removed from specific locations using the index and the pre-defined methods for those operations.

References

An ORDBMS that is Truly Relational. (2009). Retrieved from http://www.firstsql.com/ordbms.html

Brind, S. (2018, May 15). What Are Data Structures? Arrays. Retrieved from https://owlcation.com/stem/Arrays

Hoffer, J. A., Ramesh, V., & Topi, H. (2017). *Modern database management*(12th ed.). Retrieved from http://www.gcumedia.com/digital-resources/pearson/2015/modern-database-management\_ebook\_12e.php

Ray, M., & Guyer, C. (2017, September 12). Data types (Transact-SQL) - SQL Server. Retrieved from https://docs.microsoft.com/en-us/sql/t-sql/data-types/data-types-transact-sql?view=sql-server-2017

Singh, S. K. (2011). *Database systems: Concepts, design and applications*(2nd ed.). Retrieved from https://learning.oreilly.com/library/view/database-systems-concepts/9788131760925/30\_chapter016.html

Solving the Object-Persistence Impedance Mismatch. (2005, July 10). Retrieved from http://otndnld.oracle.co.jp/document/products/as10g/101300/B25221\_03/web.1013/b13593/undtl002.htm

Wambler, S. (n.d.). The Object-Relational Impedance Mismatch. Retrieved from http://www.agiledata.org/essays/impedanceMismatch.html