52302 Uzma Mahmood

39. Metadata Management

Abstract

When we are making a database schema change, it is important to identify all the dependencies in the program code that might be affected by the change. Without doing a complete impact assessment, we run the risk of causing problems when implementing the schema change. If we need to make a change to an existing database object, then we need to determine what code fragments will be affected by our change. We also have to make sure that whether changes in the database objects can be handled by the application.

Dependency on the underlying database structure of an application is typically hardcoded. During upgrades when the database structure changes, the application needs to be recoded and recompiled i.e. dependency on the hardcoded database structure makes the up gradation of the application tedious and error prone process. The Metadata framework proposed in the thesis provides enough information to make the application modify itself to changes in the database at runtime. At runtime, Metadata helps us in finding out what is available in the database and then with the help of that information (called metadata); we can build proper SQL queries.

Typically software development is driven by the design of a database. Many of the tools especially Oracle Development Environment and Microsoft Integrated Development Environment's (IDE's) have their typical development methodologies and IDE's driven from the database structure. When we want to make a change in database, instead of directly making the changes, we will make the changes in the metadata. The framework will then automatically

Design the queries

Make the necessary changes to database

Metadata, which can be broadly defined as "data about data", refers to the searchable definitions used to locate information. On the other hand, database metadata, which can be broadly defined as "data about database data", refers to the searchable definitions used to locate database metadata. Imagine, at runtime, trying to execute a SQL query in a relational database without knowing the name of tables, columns or views. Metadata helps us in finding out what is available

in the database and then with the help of that information, we can build proper SQL queries at runtime. In a nutshell, database metadata enables dynamic database access.

I limited my scope of removing underlying database dependencies till combos i.e. dropdown list. The aim is to populate the combos on the basis of metadata at runtime. SQL parameterized queries are used for data integration. For instance, a simple join query between two relations has the generic structure

```
Select <attr-list>
From <rel1> r1, <rel2>r2
Where r1.<attr1>=r2.<attr2>
```

To construct the above query at runtime and populate combos, an algorithm is used which fetches the values from database based on<AttrID>. For this purpose, I worked on ERP based software ‘Business Pattern’ which is a complete information system. It takes organization to levels of higher efficiency and effectiveness. Business Patterns is a Windows based program that is built on state-of-the-art software technologies. This makes the program user friendly and easy to learn ensuring.

However, in this paper I have only focused to remove database dependencies from combos. This approach can be further extended and can be applied to the whole application, thus making the application totally dynamic and independent from hard coded values.