Abstract—Today, information technology faces many challenges, and experiences exponential growth that is manifested in the great demand for machine accessible data. End-users are neither opt to learn programming language nor to express their queries in a language that they don’t comprehend. Clearly, software tools are designed and developed to assist end-users without being fluent in computer programming languages or non-native languages. End-users can simply issue instructions to the computer, or surf a GUI wizard program in their own mother tongue. In this paper, we describe a tool designed and implemented for database applications such as: a pay-roll system and medical applications. The tool works in two modes: (i) the set-up mode where it accepts the application schema and data dictionary, translates it into end-user native language, (ii) the operational mode where end-users may issue queries and create reports in their own native language (i.e. Arabic). The tool is implemented in VB and it may be adapted to any application providing the intervening of database developer of any specific application.

Keywords—Metadata, Metabase, MS SQL allocations Querying tool.

I. INTRODUCTION

Today information technology faces many challenges, and experiences exponential growth that is manifested in the great demand for Software tools. Computer-Aided Software Engineering (CASE) tools are defined by [1] as: "... the scientific application of a set of tools and methods to a software system which is meant to result in high-quality, defect-free, and maintainable Software products.". Historically, the term CASE was originally coined by Software development companies. They were aimed at (i) the harboring of computer assistance in Software development or Software maintenance processes, and (ii) an Engineering approach to the Software development and or maintenance. CASE tools are diversified, and they are used in a wide range of applications. Alfonso Fuggetta [2] gave 3 categories of CASE tools namely: (i) tools that support only specific tasks in the Software process, (ii) tools that work as workbenches, and are limited to only one or a few activities, and finally (iii) environmental tools that support(a large part of ) the Software process.

Our tool focuses on the logical model (the actual implementation of the conceptual data model) where entities/subtypes, attributes, relationships, and integrity rules are defined. Hence, the database schemas and their definitions (data dictionary) are targeted as the principle inputs for the proposed tool.

The rest of this paper is organized as follows: - we first describe the tight relation between database applications and its metadata resources in section 2. Then, we present some of the related research of querying tool in section 3. In section 4, we briefly describe the querying tool architecture and specifications. In section 5, we discuss tool implementation and its evaluation. Section 6, concludes the paper, and outlines trends of our future research work.

II. DATA BASE APPLICATIONS AND METADATA

Metadata is very crucial issue in today's information explosion. It is used to facilitate discovery of relevant information with less searching time, easy to use information, interoperable and publishable data. Database Schemas are holding a great amount of metadata elements, namely database relations, attributes, and data constraints. Names of metadata elements are important for database applications. Metadata elements can, however, be given terms such as synonyms, hyponyms, antonyms, etc., that will help end-users to comprehend, interpret, and easily express in their own language. Also, procedures contain so much of operations, transactions and business rules of specific application but they are beyond the scope of this paper.

Data Base and its Meta Data are tightly coupled for any Data Base Management System (DBMS) and MS SQL Server has no exception. For any given data base application, a schema and a data dictionary (system tables) can be shown and analyzed similar to data tables used in the application. In general, given the following database sub-schema can be expressed algebraically as:-

\[ \text{Database} = \langle T_1, T_2, T_3, \ldots, T_n \rangle \]

1 Ahmed.Arara is with the Tripoli University, Libya (+218 925181938, ahmedarara@hotmail.com).

2 Salma.Ali is a graduate student at the Academy of Graduate Studies-Janzour, Libya.
Where \( T_1, T_2, T_3 \ldots, T_n \) are a set of tables (relations) of the database.

Each table \( T_i \) or view in the database will have its own schema as:

\[ T_i < f_1, f_2, f_3, \ldots f_n > \]

Note that all fields (attributes) will be described by the system (system tables) with respect to their constraints, data types...etc. In specific, if we consider MS SQL server as Data Base Management Systems(DBMS), then the following command will offer complete metadata information about the application:

\[
\text{SELECT* FROM INFORMATION_SCHEMA.TABLES}
\]

In MS SQL server, there are two sources of metadata: (i) INFORMATION_SCHEMA views and (ii) SQL Server Management Objects (SMO) which is a collection of objects linked to programming and managing SQL server.

### III. Querying Tools-Related Work

Software tools have been around since the inception of computer programming. Querying tools are strongly needed to hide the complexity of database systems specifications from users. Metadata driven query tools have become essential with the wide spread of data base systems and database applications. In this respect, there were many contributions based on metadata repositories to assist end-users to apply GUI technology namely screens, menus, electronic forms, etc. Tool for code generation were developed based on metadata. Microsoft .NET [7] Framework and DB schema were exploited and used as a rich set of languages extensions to generate code [3][4]. A metabase repository is used to extract dynamically the SQL codes [6]. The metabase [6] shields users from having to keep up with details of the data model schema and to analyze data without any knowledge of SQL programming.

In data integration, several researches have been suggested where metadata is utilized across the data integration process [5]. The case study presented in [5], is used to collect and integrate historical data from various universities into the Data Warehouse system to develop student services through data mining.

Our research objectives are different in the sense that we focus on the semantics extracted from metadata repository. End-users are permitted to retrieve information but they may select their own interpretation of the data model. DB application developers play a major role in supporting end-user needs in the sense of removing ambiguities of terms used data elements.

### IV. Tool Development

#### A. Tool Architecture

The rapid prototyping model of SDLC was selected due to the fact that full requirements were not foreseeable in the early stages of tool development. Moreover, end-users and DB are strongly encouraged to participate using such software engineering model. The tool consists of 3 main elements: - the GUI element, metadata DB, and tool processor. The tool architecture is illustrated in figure 1 below:

![Figure 1 metadata-driven query tool architecture](image)

Capturing metadata from DB applications require the intervene of DB developers during the set-up mode of the tool. As a matter of fact, developer's contribution will enhance the quality, integrity and completeness of the metadata repository (metadata DB).

As a functional requirement constraint, we impose that the query tool operates in two modes:

1. **The set-up or tool configuration mode:** - As it was mentioned above, the DB developer (preferably DB designer) participates in building metadata repository of a specific application.

2. **The end-user or operational mode:** - This mode will operate in two steps:

   a. Once the tool is configured by DB specialists during the capturing mode, then it becomes available to end-user who is allowed to select their own terms of vocabulary. Only knowledgeable End-users are encouraged to use by suggesting appropriate synonymies, antonyms of data elements.

   b. Once the query tool is approved by DB developer, and by a knowledgeable end-user, then the tool enters the normal operation mode

#### B. Tool Requirements Specification

The tool main components (figure 2) are:
End-user interface: screens and message boxes and menus permit the end-user to navigate and prepare his queries and reports.

Developer interface: such user enhances the tool operations by widely contributing to the metadata DB extraction. In fact, his knowledge of a specific application will resolve many semantics conflicts that may face regular end-users.

Command processor: end users and developers issue a command to the tool, and the command processor will respond accordingly.

RDBM system: the system is needed for metadata extraction of both system files and data files (the application).

Metadata extractor: this module is dedicated to collect metadata elements and send it to the metadata DB (metadata repository).

Metadata DB: this is a relation database that supports end-users to query the applications and developers to add semantic aspects to the application. The metadata repository may be exploited in data mining.

Metadata editor: during the set-up mode, Metadata elements may be edited and appropriate terms are chosen. A thesaurus of terms and/or a dictionary will support the metadata editor.

Tool implementation and evaluation

The tool is implemented in visual basic under the visual studio framework. SQL server DBMS is used to store the metadata repository. The wizard program permits the end-user to navigate through the database application (see figure 3 a & b) in his/her own language without any comprehension of DB theory and or SQL commands in English language.

Figure 2 metadata driven query tool modules

Figure 3-a screen shot of wizard program

To evaluate the tool, we use the ISO standards [8] related to evaluation and selection of CASE tools as well as comparative analysis of quantitative measures. The technical evaluation objectives using ISO/IEC 14102 are to show how well a CASE tool meets its user’s stated requirements. Some characteristics and subcharacteristics are defined based on ISO/IEC 14102 and they cross-checked for our querying tool. The second technique of evaluation is based on quantitative analysis of characteristics related to CASE tool usage and general quality functionalities. A comparative analysis of the tool with its peers is performed, and some essential criteria are used namely query response time, Meta DB access time, and completeness.

V. Conclusion and Future Work

Metadata is very crucial issue in today’s information explosion. It used to facilitate discovery of relevant information with less searching time, easy to use information, interoperable and publishable data. Every good Database Management System (DBMS) has some sort of data dictionary or metadata and SQL Server is no exception. SQL Server has two sources for us to view the metadata namely INFORMATION_SCHEMA views and SQL Server Management Objects (SMO). The current research concentrates more on the objects related to tables, queries. Other objects are essential resources of metadata but it is
beyond the scope of our work. As future outlook, we intend to consider ontologies of several applications where end-user will be able to perform data mining as well as getting results for their queries from various applications that can be part of ERP (Enterprise Resource Planning).

ACKNOWLEDGMENT

We would like to thank the head of computer science department at the Academy of Graduate Studies in Tripoli Professor Abdulhafeed Eshweihedi for his comments and review of thesis related to this work.

REFERENCES