DESIGN OF AN INTEGRATED DBMS TO SUPPORT ADVANCED APPLICATIONS

V. Lum, P. Dadam, R. Erbe, J. Guenauer, P. Pistor,
G. Walch, H. Werner, J. Woodfill
IBM Wissenschaftliches Zentrum
D6900 Heidelberg

Abstract

New applications of DBMS's in areas of sciences, engineering and offices have produced new requirements that are not satisfied in current DBMS's. Included among these requirements are support for both the normalized and non-normalized models directly at the system interface level, support for text processing, and support of the temporal domain. To provide these supports, one can try to build additional functions on top of or into an existing DBMS. This approach has been deemed to be inefficient. It is believed that much can be gained by designing a new system to satisfy the new requirements more directly.

This paper describes the specific design of a DBMS directed to satisfy the three requirements just cited. The authors first discuss the overall architecture before proceeding to discuss some aspects of its internal data management. Included here is the internal data structure and how they are used to provide the necessary supports. Summarized in the conclusion are the features of the system that are not available generally in current DBMS's. In addition, the status and planned enhancements are also outlined.

1. Introduction

In recent years the use of DBMS's has moved from the traditional area of commercial applications such as inventory control and banking to include many new areas such as engineering (e.g. CAD/CAM) and office applications (e.g. Di84, GP83, HL82, HR82, KL82, Lo82, LP83, Lue83, Lu84, SP82, St82, St83). As a result, new requirements for DBMS's have been discovered and current DBMS's have been found to be lacking in support of these new applications.

The Advanced Information Management (AIM) project in the IBM Heidelberg Scientific Center, established several years ago in anticipation of some of these advanced applications, originally saw the need to incorporate textual retrieval into the normal data management facility in a DBMS. The need for such a capability has not only been

A. Blaser et al. (eds.), *Datenbank-Systeme für Büro, Technik und Wissenschaft*

established in office document retrieval, but has also been found in engineering applications as well (IBM82). Since then, other basic facilities necessary to support the divergent applications have been uncovered.

Our initial thought has been to build a system on top of an existing DBMS. Our analysis soon showed that this is not a good approach as the constraints to fit into a given system are too restrictive, the modifications required to achieve our objectives are too much and that the resulting system will likely perform unsatisfactorily (see Lu84). These observations and analyses led us to the decision that a completely new design is needed, with all constraints removed. This paper describes the design of such a system to meet the objectives discussed below. Moreover, while some of the objectives is similar to those in the paper by Deppisch (De85), these two papers are different in in the following way: Our paper describes a system intended to have normal level DB interface, the other paper emphasizes on the construction of a low level kernel data management system that can be integrated into the file and data management portion of an operating system.

One objective in the design of our system is to support non-atomic data types and relations with complex - i.e. non-flat or non-normalized - tuples in addition to normalized relations and simple data types. Recent applications of DBMS have shown that a non-normalized form of relations (also referred to as hierarchical structure) can be used to advantage (e.g. GP83, HL82, KL82, KTT83, Lo82, Lue83, Lu84, SP82, St82, St83). Some researchers feel that a hierarchical structure built with normalized relations would be sufficient. However, we think there is considerable merit in implementing a system which directly supports normalized and non-normalized relations at the same time.

Part of the attractiveness of normalized relations lies in their mathematical soundness and tractability. The non-normalized form has similar appeal, as recent theoretical work (Ja84, JS82, SS84) has indicated that algebra for non-normalized relations, which include normalized relational algebra as a special case, is now available. This provides us with mathematical soundness as a foundation on which to construct such a system. Further, Shu and Housel (SHL75, HS76) and Pistor (PHH83) have demonstrated that high level languages similar to the relational languages for normalized forms can be defined. The question, then, is how to construct a system to have the flexibility and efficiency similar to relational systems based on the normalized form.

Another objective is to have a database system for supporting history data. Bubenko (Bu77), Clifford (CW83), among others (e.g. Gu83, Ki83, KL83, MSW83) have stated the desirability of having the temporal domain information. For example, temporal information has always been used for tracking and control purposes in office applications. One can, however, easily see applications beyond the office for such a facility. Thus, the second major objective is to incorporate time information or history data in the databases of the system.