Research Paper: Advanced Database Technologies- A Future of High Performance Database Processing

Prof. Ravish Kumar Mishra*

Abstract

The need for recording time varying information databases has been recognized for quite some time. There have been significant research activities in formulating semantics of time at the conceptual level, developing a model for time varying databases analogous to the relational model for static databases and the design of temporal query languages. It has been argued that a single time attribute is insufficient, and that two time attributes are necessary to fully capture time-varying information. Unfortunately, there has been some confusion concerning terminology and the definition of these time attributes.

In this world, a single program performing global query optimization using a cost-based optimizer will not work well. Cost-based optimization does not respond well to site specific type extension, access constraints, charging algorithms, and time-of-day constraints. Since traditional distributed DBMSs have all used cost-based optimizers, they are not appropriate in a WAN environment, and a new architecture is required.

Traditional distributed relational database systems that offer location-transparent query languages, such as Distributed INGRES and SDD-1all make a collection of underlying assumptions.

Traditional distributed DBMSs do not meet these requirements. Use of an authoritarian, centralized query optimizer does not scale well; the high cost of moving an object between sites restricts data mobility, schema changes typically require global synchronization, and centralized management designs inhibit local autonomy and flexible policy configuration.

UML has intentionally been developed as a langue for modeling object-oriented systems. Its use has however widely been spread out. This standardization makes it possible that different vendors produce products that comply with the standard specification.

This paper explores a mechanism to support user-defined data types for columns in a relational data base system. We suggested to how to support new operators and new data types. The contribution of this work is to suggest ways to allow query optimization on commands which include new data types and operators and ways to allow access methods to be used for new data types.

1. Introduction

1.1 Database

It computer science a Database Management System (DBMS) is a set of computer program that controls the creation, maintenance, and the use of a database. It allows organizations to place control of database development in the hands of database administrators (DBAs) and other specialists. A DBMS is a system software package that helps the use of integrated collection of data records and files known as databases. It allows different user application programs to easily access the same database. DBMSs may use any of a variety of database models, such as the network model or relational model.

Database servers are dedicated computers that hold the actual databases and run only the DBMS and related software. Database servers are usually multiprocessor computers, with generous memory and RAID disk arrays used for stable storage. DBMSs are found at the heart of most database applications. DBMSs may be built around a custom multitasking kernel with built-in networking support, but modern DBMS typically rely on a standard operating system to provide these functions.

1.2 Distributed Database Management System

A distributed database management system is software for

Poona School of Business, Pune, India.

managing databases stored on multiple computers in a network. A distributed database is a set of databases stored on multiple computers that typically appears to applications on a single database. Consequently, an application can simultaneously access and modify the data in several databases in a network. DDBMS is specially developed for heterogeneous database platforms, focusing mainly on Heterogeneous Database Management Systems (HDBMS).

1.3 Relational Database Management System

The Relational Database Management System is a type of database management system (DBMS) that stores data in the form of related table(Rows and Columns). Relational databases are powerful because they require few assumptions about how data is related or how it will be extracted from the database. As a result, the same database can be viewed in many different ways. An important feature of relational systems is that a single database can be spread across several tables. This differs from flat-file databases, in which each database is self-contained in a single table.

1.4 Object-Relational Database

An Object-Relational Database (ORD), or Object-Relational Database Management System (ORDBMS), is a database management system (DBMS) similar to a relational database, but with an object-oriented database model, such as objects, classes and inheritance are directly supported in database schemas and in the query language. In addition, it supports extension of the data model with custom data-types and methods.

1.5 Unified Modeling Language (UML)

UML is a result of the evolution of object-oriented modeling languages. It was developed by Rational Software Company by unifying some of the leading object-oriented modeling methods,

- Booch by Grady Booch,
- OMT (Object Modeling Technique), by Jim Raumbaugh and
- OOSE (Object-Oriented Software Engineering), by Ivar Jacobson.

UML is used for modeling software systems; such modeling includes analysis and design. By an analysis the system is first described by a set of requirements, and then by identification of system parts on a high level.

The basic building blocks in UML are things and relationships; these are combined in different ways following different rules to create different types of diagrams. In UML there are nine types of diagrams, below is a list and brief description of them.

1. Use case diagrams; shows a set of use cases, and how actors can use them
2. Class diagrams; describes the structure of the system, divided in classes with different connections and relationships
3. Sequence diagrams; shows the interaction between a set of objects, through the messages that may be dispatched between them
4. State chart diagrams; state machines, consisting of states, transitions, events and activities
5. Activity diagrams; shows the flow through a program from an defined start point to an end point
6. Object diagrams; a set of objects and their relationships, this is a snapshot of instances of the things found in the class diagrams
7. Collaboration diagrams; collaboration diagram emphasize structural ordering of objects that send and receive messages.
8. Component diagrams; shows organizations and dependencies among a set of components. These diagrams address static implementation view of the system.
9. Deployment diagrams; show the configuration of run-time processing nodes and components that live on them.

2. Database Model

Database systems can be based on different data models or database models respectively. A data model is a collection of concepts and rules for the description of the structure of the database. Structure of the database means the data types, the constraints and the relationships for the description or storage of data respectively.

2.1 Network Model and Hierarchical Model

The network model and the hierarchical model are the predecessors of the relational model. They build upon individual data sets and are able to express hierarchical or network like structures of the real world.

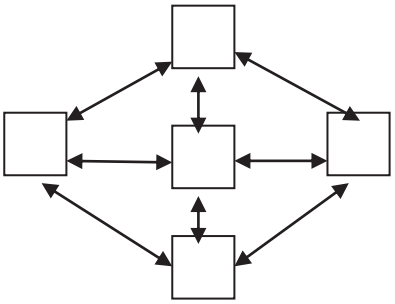


Fig1.1 Network Model

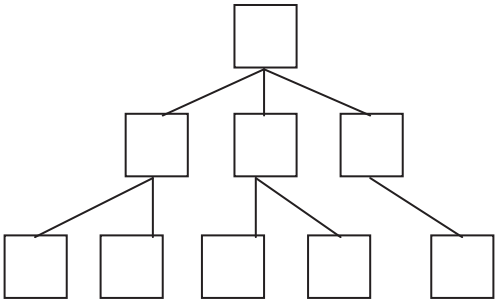


Fig 1.2 Hierarchical Models

P-ID	Name	Sur name	City
1	Sana	Mishra	Pune
2	Anita	Mishra	Pune
3	Prachi	Kulkarni	Latur
---	---	---	---

Fig 1.3 Relational Database Model

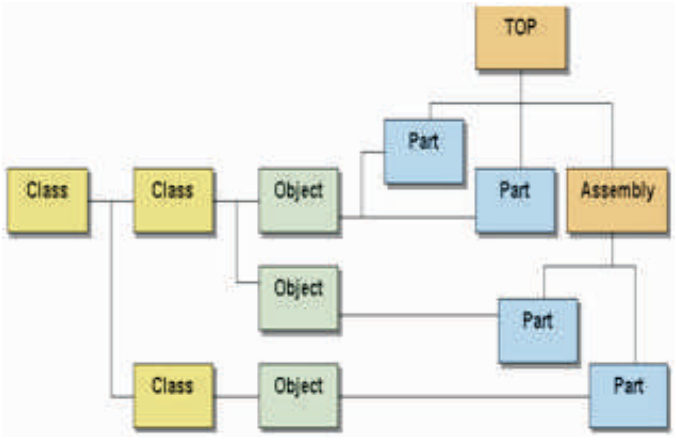
City-ID	Area	Type	---
2547	8974	Town	---
8547	5478	Town	---
8745	1547	Village	--
---	---	---	---

Fig: 1.4 Relational Models

2.2 Relational Model

The relational model is the best known and in today's DBMS most often implemented database model. It defines a database as a collection of tables (relations) which contain all data.

This module deals predominantly with the relational database model and the database systems based on it.



1.4 Object-Oriented Database Model

2.3 Object-Oriented Model

Object-oriented models define a database as a collection of objects with features and methods. A detailed discussion of object-oriented databases follows in an advanced module.

2.4 Object-Relational Model

Object-oriented models are very powerful but also quite complex. With the relatively new object-relational database model is the wide spread and simple relational database model extended by some basic object-oriented concepts. These allow us to work with the widely know relational database model but also have some advantages of the object-oriented model without its complexity.

ID	XY	DF	ER
56	Class	XXX	Assembly
45	Object	YYY	Object
---	---	---	---

Fig. 1.5 Object-Relational Database Model

3. The Entity-relationship Model

The Entity-Relationship Model is a data model or diagram for high-level descriptions of conceptual schemata. It provides a graphical notation for representing such schemata in the form of entity-relationship diagrams. It is developed by P. P. Chen in 1976 which is based on the Data Structure Model for conceptual design of Network Model-based database schemata.

3.1 Entities

Entities – or more precise Entity Types – represent sets of objects in a given universe of discourse sharing the same properties and relationships as other instances of this particular Entity Type. Their graphical notations are rectangular boxes.

3.2 Relationship

Relationships – or Relationship Types – represent how instances of Entity Types can be related in the universe of discourse. Their graphical notations are diamond-shaped boxes.

3.3 Attributes

Attributes represent properties of Entity or Relationship Types. Their graphical notations are rounded boxes or ellipses.

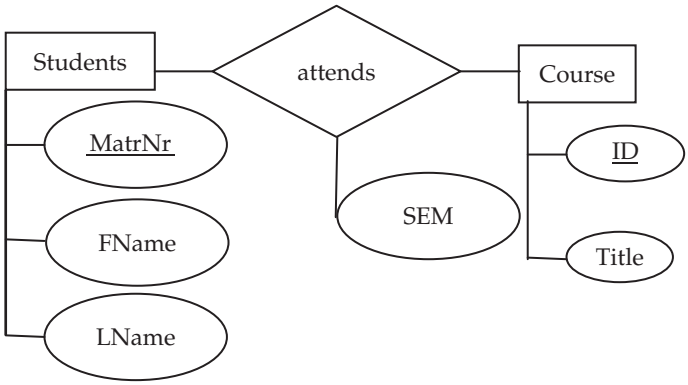


Fig1.6 Basic Construct of ER Model

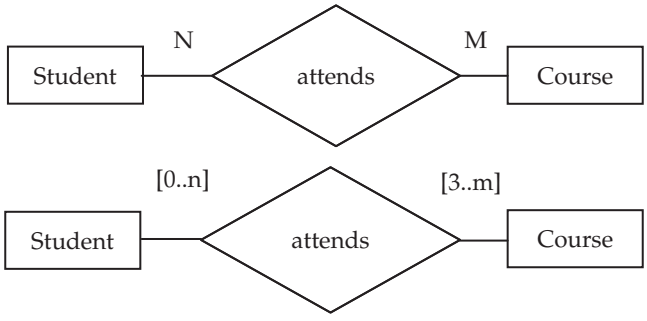


Fig 1.7 Cardinalities of ER Model

4. Comparison Between Relational and Object-relational Database

- Advantages inherited from relational SQL database model
  - Variety of integrity constraints
  - Concept of data dictionary
  - Descriptive Query language
  - Views
- Introduction of non-orthogonal features
  - Triggers versus Methods
  - Generic DML operations versus Methods
  - Primary/foreign keys versus object references
- ORDBMS still have impedance mismatch

5. Oo Features Being Added Include In Database

- User-extensible types
- Encapsulation,
- Inheritance,
- Polymorphism,
- Dynamic binding of methods,
- Complex objects including non-1NF objects,
- Object identity,
- Subtypes and sub tables.

6. The Third-generation Database Manifestos

- The first way: the "3-G Database System Manifesto" (to extend SQL and RDM):
- Selected features proposed by CADF:
  - A 3rd generation DBMS must have a rich type system.
  - Inheritance is a good idea.
  - Functions, including database procedures, methods… are a good idea.
  - DBMS assigns unique identifiers for records only if no user-defined PK
  - Rules (triggers, constraints) will become a major feature in future. They should not be associated with a specific function or collection.
  - All programmatic access to a database should be through a non-procedural, high-level access language.
  - Should be at least two ways to specify collections, one using enumeration of members and one using query language.
  - Updateable views are essential.
  - Performance indicators should not appear in data models
  - For better or worse, SQL is "intergalactic data speak".
- The second way: the "Third Manifesto"(to preserve RDM and replace SQL) :
  - Proposed by Darwin and Date (1995,2000), it attempts to defend the relational data model:
  - Acknowledged that certain OO features desirable, but believe features are orthogonal to RDM.
  - Thus, RDM needs 'no extension, no correction, no subsumption, and, above all, no perversion'.
  - However, SQL is unequivocally rejected as a perversion of model.
  - Instead a language called D is proposed.
  - Primary object is domain - a named set of encapsulated values, of arbitrary complexity, equivalent to data type or object class.
  - Domain values referred to as scalars, manipulated only by means of operators defined for domain.
  - Both single and multiple inheritances on domains proposed.
  - Nested transactions should be supported.

7. Mega Trends of Advance Database

- Commodization of servers
- Packaged Applications
- Data Warehouses
- Object-Relational Databases
- OO DBs
- Metadata Repositories

7.1 Commoditization:
- Microsoft SQL Server and Access are the drivers
  - All low end TPC numbers use MS SQL Server
- Ease of use is critical (e.g. Access)
- Automated storage mgmt, DB design, …as easy as files

7.2 Packaged Applications:
- Packaged applications drive a lot of DB business
  - Little custom application development at F500
- Commercial applications - finance, manufacturing, distribution, human resources, order processing ...
  - SAP ($3.8B), People soft ($700M), Baan ($500M), Oracle apps (~ $500M)
  - Growing at 40-60% … huge piece of the s/w business

7.3 Data Warehousing:
- Business goal - decision support for everyone
- Problem - Ad hoc query on production DBs doesn't work

7.4 Object-Relational Databases:
- Goal - Capture more of the world's data
  - SQL DBs are designed for simple scalar types
- More data types - text, video, image, audio, geographical data, time series, graphs, ...

7.5 Object-Relational Products:
- All vendors are working on OR DBMSs
  - Leaders - Informix, IBM DB2, Oracle 8
  - 3rd parties will supply data blades / extenders / cartridges
  - Significant features are available today
  - But systems are still incomplete and hard to extend

7.6 Object-Oriented Databases:
- New commercial applications are object-oriented
  - Impedance mismatch to store data in relations
  - Especially true for design apps: CAD, CASE, CAM
- Stores objects with behavior, not just records
- But OR is competing in this space

7.7 Metadata Repositories:
- Scientific data management
- Computer-aided design
- Discrete manufacturing
- Process industries

8. OODBMS vs. ORDBMS:

- Use an OODBMS:
  - When your application retrieves relatively few (large) complex objects and works on them for a long period before moving to the next object
  - When you are happy programming in an OO language
- Use an ORDBMS:
  - When your application processes a large number of short-lived transactions (e.g. ad-hoc queries) on complex data items.

9. Current Trends of Object-database Management System

In 1998, database management was in need of a new style of databases to solve current database management problems. Researchers realized that the old trends of database management were becoming too complex and there was a need for automated configuration and management. Since this new development process of database management there are more possibilities. Database management is no longer limited to "monolithic entities". Many solutions have been developed to satisfy the individual needs of users. The development of numerous database options has created flexibility in database management.

There are several ways database management has affected the field of technology. Because organizations' demand for directory services has grown as they expand in size, businesses use directory services that provide prompted searches for company information. Search engine queries are able to locate data within the World Wide Web. Retailers have also benefited from the developments with data warehousing, recording customer transactions. Online transactions have become tremendously popular for e-business. Consumers and businesses are able to make payments securely through some company websites. These current developments would not have been possible without the evolution of database management. Even with the progress of database management, there is a demonstrated need for new development as specifications and needs change.

Object-relational database management systems grew out of research that occurred in the early 1990s. That research extended existing relational database concepts by adding object concepts. The researchers aimed to retain a declarative query-language based on predicate calculus as a central component of the architecture. In the mid-1990s, early commercial products appeared. These included Illustra (Illustra Information Systems, which was in turn acquired by IBM) Omniscience (Omniscience Corporation. By the next decade, PostgreSQL had become a commercially viable database and is the basis for several products today which maintain its ORDBMS features.

10. Upcoming and Future Trends of Advance Database

We saw a variety of features of the extended data definition and query language, as well as the query language, and in particular support for collection-valued attributes, inheritance, and tuple reference. Such extensions attempt to preserve the relational foundations-in particular, the declarative access to data-while extending the modeling power.

Object-relational database systems provide a convenient migration path for users of relational databases who wish to use object-oriented features.

The database design mainly involves the design of the database schema. The entity-relationship (E-R) data model is a widely used data model for database design. It provides a convenient graphical representation to view data, relationships, and constraints. The ODMG standard defines classes and other constructs for creating and accessing persistent objects from C++, while the JDO standard provides equivalent functionality for Java.

The Unified Modeling Language (UML) provides a graphical means of modeling various components of a software system. The class diagram component of UML is based on E-R diagrams.

We can create the notation of multivalve dependencies, which specify constrains that cannot be specified with functional dependencies alone. We can also define fourth normal form (4NF) with multivalve dependencies. We can develop a relational database design from an E-R design, when schema may be combined safely, and when a schema should be decomposed. All valid decompositions must be lossless.

Apart from basic operation of relational algebra we can also implement different operations such as additional operations that can be expressed in terms of the basic operations and extended operations, some of which add further expressive power to relational algebra.

11. Conclusion

In reviewing the issues in the articles, note that the reason we could define rigorous approaches to relational database design is that the relational data model rests on a firm mathematical foundation. That is one of the primary advantages of the relational model compared with the other data models.

The Web browser has emerged as the most widely used user interface to database. The HTML provides the ability to define interfaces that combine hyper links with different user interface such as form etc. Web browsers communicate with Web servers by HTTP protocol. Web servers can pass on requests to application programs, and return the results to the browser.

The relational algebra is a terse, formal language that is inappropriate for casual users of a database system. Commercial database systems, therefore, use languages with more "Syntactic Sugar".

DB research has been enormously successful:
- It drove relational, OO, and OR technology
- Also transactions, distribution, replication

DB research is a mature field:
- Most current work is incremental on long-standing topics
- But there's much to do, due to changing requirements

12. Reference

12.1 Book Reference:

1. *Introduction of Database System, Pearson Education, Elmasri Navathe.*
2. *PL/SQL, Techmedia, Tom Luevs, Timothy, Atwood and Jonathan Gennick.*
3. *Database System Concepts, McGraw-Hill International Edition, Abraham Silberschatz, Henry F. Korth, S. Sudarshan.*
4. *Oracle, Vision Publication Pune, Ravish Kumar Mishra.*
5. *DBMS & Oracle, Vision Publication Pune, Ravish Kumar Mishra.*

12.2 Web Reference:

1. *www.computer.org/bookshelf*
2. *www.mysql.com*
3. *www.postgresql.com*
4. *www.uml.org*
5. *www.objectstore.com*