Unix RDBMS: The next generation

What are the Unix relational-database vendors doing to survive in the next generation of client/server environments?

Bill Rosenblatt *

1 Part One

The Unix relational-database vendors are scrambling, not only to compete with each other, but to provide new products and services that might better satisfy the demands of the next generation of client/server, distributed computing systems, and their users. The first generation of databases that have come to dominate the Unix market run on highly localized, mostly homogeneous client/server networks. The next generation's architectural requirements aren't nearly as lenient. To satisfy that next generation's demands, relational databases (RDB) will have to distribute a wider variety of datatypes to a wider variety of machines in enterprisewide, heterogeneous networks. And vendors will have to provide the tools and services that will satisfy the ever-skyrocketing demands by developers and users for faster and better access to their data. Will RDB vendors be able to survive by focusing on their traditional strengths alone, or will they need to dominate markets now being served by other vendors? Will the core relational-server technology be powerful enough to survive through 1995 and beyond? We can take a pretty good snapshot of the future by thoroughly examining the main technological requirements for next-generation Unix database systems and how three of the leading Unix database vendors – Oracle, Informix, and Sybase – along with the dark-horse innovator Borland, can and are addressing those issues. In this first of a two-part series, we examine how the selected Unix RDB vendors are improving the performance of their respective databases. We also examine how they are providing for wider data distribution over wide-area networks of heterogeneous, interoperating clients and servers. Next month, we'll see how those same vendors can and are providing for larger, more complex types of data and databases, and what they are doing – and must do – to provide developers with tools to write better database applications and users with tools to better access data.

*Bill Rosenblatt is manager of Information Resources at Moody's Investors Service in New York City. He can be reached at billr@panix.com. Reprinted from the February and March 1994 issues of Advanced Systems Magazine,

1.1 Improving performance

Performance nearly always leads the market. Older database performance-enhancing techniques, like buffering and query optimization, have reached the point where the improvements they offer can no longer keep pace with the increased transactional demands of the next generation of database systems. Two new techniques promise significant breakthroughs. One new way to improve database performance is to reduce contention at the server when several processes are trying to access the same data. The traditional method of "locking" forces transactions to queue up and wait for one-after-another access. Oracle7's multiversion read consistency and Interbase's multigenerational architecture drastically reduce locking by creating several versions of data on the servers. With versioning, each transaction can have its own copy of the data, so no waiting is necessary. Interbase also uses versioning for another clever purpose: dynamic schema maintenance – the ability to change data table definitions while the system is running. This is very convenient in large-scale heterogeneous systems, whose data schemas are bound to change regularly over time. Interbase is the only system that allows arbitrary on-the-fly changes.

1.2 Parallel transfer processing

The other new way to increase database performance is through various forms of parallel processing, including multithreading and parallel processing of queries. Vendors are developing these two techniques for tightly-coupled, multiple-CPU servers and loosely-coupled server "clusters" which communicate over a high-speed network and share common disk storage. The multithreaded architecture that Sybase first implemented and now all four vendors' systems use (it's new for Informix 6.0) avoids process idle states (during I/O, for example) by deploying threads, which are special "lightweight" processes that use less of the server's resources. The database server assigns queries to threads and can dynamically change the thread on which a query runs, much like Unix switches between processes. If a query goes idle, the server will swap it out of its thread, and swap in another query while the first one waits (see diagram Multithreading queries). Multithreading saves time and improves performance because switching between threads is considerably simpler than switching between processes. The result is that processes remain idle less often and queries finish faster. Sybase has its own custom threading mechanism, while other server products use the lightweight-process or parallel-process handing mechanisms of the host operating system. This gives Sybase an edge (all else being equal) on platforms that have neither. Servers can also improve performance from parallel processing: decomposing individual queries into parts that can be run independently, possibly on separate servers – a technology known generically as parallel data queries (PDQ – get it?). The general problem of how ideally to break a program into parts that can be executed simultaneously (parallelization) is an open issue in computer science and is likely to remain so for some time. Database vendors have not been able to significantly master PDQ – yet. Informix and Sequent Computer Systems did show one working PDQ demo last July (1993) on an eight-processor system that had a linear speedup factor of just over seven. That's close to a perfect eight – though under presumably optimal conditions for a demo. They expect to put this PDQ solution into production during the first half of 1994. Oracle expects to deliver PDQ technology with the next release of Oracle7. Sybase's Navigation Server is expected to have PDQ available for NCR's 3500/3600 series of computers in the first half of 1994. Meanwhile, Oracle has gone a step beyond balancing transaction-process loads on multiple-CPU machines: Oracle is implementing its Parallel Server technology on loosely-coupled (clustered) CPUs. A port of Oracle's Parallel Server to Digital's VAXClusters running VAX/VMS has been complete since 1991, and a port to Sequent's ptx/CLUSTERS network went into production this past June (1993). Oracle also has announced ports to Pyramid Technology Corp. and NCR clustered systems. Loosely-coupled parallelism (LCP) is usually slower than symmetric multiprocessing (SMP) because of the difference in data-transmission speed between an external network and an internal bus. However, LCP has two advantages. First, the separate power supplies and other hardware of clustered machines provide high availability and fault tolerance for the database. Second, with fiber-optic networks, it's possible to have a "parallel machine" consisting of CPUs that are separated by up to several miles. This adds an extra degree of availability that overcomes site-specific disasters.

1.3 Data distribution

The next generation of client/server systems will grow from mostly homogeneous, workgroup-level installations into heterogeneous, enterprisewide, mission-critical systems. The vendor who comes up with the right set of solutions to integrate the database with these advanced data-distribution technologies almost assuredly will own the market. The catch is that the "right set of solutions" is far from understood. Optimal distribution of data is technologically difficult and highly dependent on a given organization's requirements for response time, data integrity, availability, interoperability, and various other factors, among which there are always trade-offs. It's not even clear that the "right" solution falls within the purview of the database server or even the database vendor. Emerging technologies, such as distributed-object management systems, transaction monitors, and some kinds of so-called "middleware," address many of the same data-distribution problems. The technologies that server and database vendors do offer include remote procedure calls (RPCs), stored procedures, and triggers, which together serve as a sort of "assembly language" for data distribution in a network. The problem is that these features force developers to do ad hoc programming, particularly to do anything fancy like support certain types of integrity constraints or exchange data with another type of database system. Most technology analysts agree the proper way vendors should provide for the next generation of data distribution over enterprisewide systems is through features that let programmers use higher-level abstractions to describe and execute business tasks rather than database operations; just as higher-level languages offer data and control abstractions above those of assembly language. The database world already uses a few abstractions to accomplish database-application tasks. A transaction, for example, is a collection of operations that takes the database from one logically consistent state to another. Transactions normally are handled atomically: All changes that make up a transaction to the database are either committed or none are made. The transaction abstraction also can be used at the data-distribution level to ensure related groups of data and operations are moved from client to server or from one server to another in discrete logical units. Transaction processing (TP) monitors are tools, mostly from third-party vendors, that control distributed transactions across heterogeneous networks. The accepted, standard method for communication between TP monitors and database systems is the standards group X/Open's X/A protocol. Sybase System 10 and Oracle7 support X/A; Informix's OnLine supports it through the add-on product, Informix TP/XA. Database server vendors also have extended the two-phase commit (2PC) technique for facilitating transaction processing and ensuring atomicity to data distribution. In the first phase, all (two or more) servers are checked for readiness to commit a transaction to their respective databases. In the second phase, the servers commit only if all servers are ready. However, 2PC has serious drawbacks when applied to data distribution. One is performance: The network traffic can be enormous, data must be locked during the checking phase, making it unavailable for potentially long periods, and recovery from failures is a complex, time-consuming process. More importantly, 2PC is not appropriate in many situations, especially with complex, globally dispersed networks in which the probability of failure is relatively high, and with mission-critical systems that must be highly available. It's not practical, for example, to make Tokyo stop doing business just because the network link from the New York main office is down, or for that matter, to cripple New York operations because backups are being taken on Tokyo's server. Oracle, Sybase, and Informix have chosen to go beyond 2PC and offer data distribution solutions based on another "abstraction," the replication paradigm. (Interbase plans to incorporate replication into a future release.) Replication is a process wherein several servers keep identical copies of a database at all times. It's useful for several purposes, such as ensuring data availability, dedicating servers to specific tasks, and minimizing network traffic by keeping a complete database copy at a remote site (see diagram Replication strategies). Until recently, many users confused replication with 2PC, and some vendors even touted 2PC as a replication mechanism. You can use 2PC to do replication when the data really must be exactly the same on all machines at all times; for example, if you are using one server as a "hot spare" for another, or for transaction processing where debits in one table must match credits in another at all times. But replication is fundamentally different from 2PC. It is a looser coupling of the servers, with less-stringent constraints about how identical the data must be on all of them. Replication guarantees only that the distributed databases be identical at certain times or under certain, occasional conditions. One technique for replication is Oracle7's utility called Table Snapshots, which copies only changed (differential) parts or all of selected data tables at discrete times and then propagates the snapshot to other machines in the network. Snapshots can be useful, but they incur the risk of logical inconsistencies between databases. For example, unless a transaction is carefully defined for a multiple-server system in your application code (which it may not be if it was written for a single server), there's a chance that incomplete transactions will appear to be logically consistent when in fact they aren't. Informix's replication mechanism is similar to Oracle's "snapshot" scheme. Like various other features of Informix's On-Line database, its replication scheme takes advantage of as much pre-existing functionality as possible: Informix uses information on database changes in backup logs to determine what data to replicate. At the appointed intervals – preset by the database administrator – the replicating database does a "roll-forward" over the Informix-Star network to the remote machine(s). This mechanism has the virtues of simplicity and proven technology, but it has the same drawbacks related to logical data consistency as Oracle's snapshots. Oracle recommends an alternative approach to replication, known as "store-and-forward," which works rather like electronic mail. The mechanism lets programmers set up replication so it takes place at transaction boundaries, rather than at fixed time intervals like a snapshot scheme, thereby greatly improving the chances that all databases are in logically consistent states. One server is designated as the master. When changes are made on the master database, they are propagated to the other servers as messages in a queue. If a remote machine goes down, the messages accumulate until it comes back up and can apply them to its database. It's possible to implement a store-and-forward-style replication mechanism with triggers and some extra code, such as a special Unix cron job. Oracle announced they soon will be offering an add-on package (currently unnamed) that automates the process by eliminating much of the need to write extra code. Sybase System 10 has an add-on component, called Replication Server, that goes to great lengths to implement a store-and-forward scheme in a reliable way. A special server is given the entire task of managing replication, and an alternate master is designated for immediate "hot-spare" failover if the master goes down. The objectives, of course, are fault tolerance and high availability. Replication Server, in combination with System 10's database administration add-ons, will help Sybase's customers go a long way toward taming their data-distribution beasts. However, Replication Server is so inherently complex that its own ramifications and limitations have yet to be discovered. And Sybase will need to expend a lot of effort to educate and train database administrators on Replication Server's care and feeding. Oracle's snapshot and Informix's roll-forward schemes, in contrast, are clean and simple, and they may satisfy many organizations' needs. Replication Server and the other products represent only the beginning of an entirely new genre of products that embody higher-level abstractions of data distribution, many of which have yet to be discovered. Replication is only one of several data-distribution paradigms needed for the next generation of distributed-database systems. Another important distribution issue arises, for example, when users need access to different data on different machines.

1.4 Interoperability

Interoperability is probably the most definitive issue of computing in the '90s. Despite plans by vendors like IBM and Digital for world conquest through monolithic enterprise networking, most businesses are not going to abandon investments in their current, variegated collection of computers and LANs. Things get more interesting when you access data on a machine running a different server or a different data source altogether, such as a hierarchical database, an indexed data manager on a mainframe, and so on. Database vendors have stopped dreaming about homogeneous, proprietary systems. Instead, they are scrambling to determine which connectivity standards to bet on. From the perspective of database server vendors, interoperability also is confusing because it has two principal aspects that overlap: interoperability with other database servers (relational to hierarchical, for example) and interoperability among client applications (spreadsheets and flat-file PC databases, for example). Also, the distinctions between client and server are becoming more and more artificial – and less relevant – because of the increasing variety of software that ought to be able to interoperate with database systems and the decreasing importance of machine classifications by CPU size. The various client/server database vendors have adopted different strategies for dealing with interoperability. Borland has the widest range of products, so it's not surprising the company has developed (in consort with IBM, Novell, and WordPerfect Corp.) its own connectivity standard, the Integrated Database Application Programming Interface (IDAPI). As a programming interface, IDAPI lets you access relational tables a row at a time, dBASE-style – or as Borland calls it, using the "Xbase navigational" style. On the other side, IDAPI lets you use SQL commands on Xbase records and get relations as answers – instead of just the first row that meets the condition, as Xbase databases would normally do. IDAPI purports to be a superset of its most important competitor: Microsoft's ODBC (Open Database Connectivity). ODBC, in turn, supports SAG-CLI, the SQL Access Group's Call Level Interface. Another important competing standard is IBM's DRDA (Distributed Relational Database Architecture), which IDAPI also supports. Because IDAPI promotes interoperability among a wide range of products, including two PC databases, a spreadsheet, and a number of programming- language systems in addition to Interbase, Borland is really blurring the distinction between client and server connectivity. This is a sensible strategy. The same is true of ODBC – though to a lesser extent, since it doesn't support navigational access. However, Borland's delays in releasing IDAPI have given ODBC a distinct advantage: Many third-party products and Microsoft's own database products already support ODBC. And Microsoft's trump card is that its high-end database, SQL Server, is a port of Sybase for PCs, giving ODBC a gateway to Sybase's world, as well. Interestingly, Sybase seems to be ignoring the Microsoft Connection. The company's interoperability products perpetuate the client-side versus server-side connectivity dichotomy by offering two distinct paradigms based, respectively, on their Open Client and Open Server architectures, which have been Sybase features for some time. Open Client and Open Server are essentially programming interfaces that serve to "expose" Sybase's proprietary client/server networking scheme so developers can create components on either end. For example, a program written to Open Server can act as a middleware layer between client applications and an SQL Server, or it can replace SQL Server. Most of Sybase's System 10 add-ons, including the Replication and Navigation Servers, are written as Open Servers – programs that issue Open Server API calls. The Open Client API is easier to explain. It is Sybase's own call-level relational interface, la SAG-CLI or ODBC (see above). Sybase has attempted to promote the Open Client and Open Server APIs as proprietary connectivity standards, offering developers flexibility as long as they take the effort to make their products Sybase-interoperable. This strategy hasn't paid off very well, however. So with System 10, Sybase is reaching out and offering better interoperability through products that work on top of Open Client and Open Server. On the server side, Sybase's new OmniSQL Gateway is an interface that makes other RDBS, including Informix, Oracle, and various mainframe-based nonrelational products, look to the client almost exactly like a Sybase SQL Server. This means client applications written in Sybase's Transact-SQL dialect also can use data on non-Sybase data servers. Sybase is offering a number of client layers above Open Client that make it look like IDAPI, ODBC, or SAG-CLI. Informix also announced ODBC support and offers connectivity to mainframe-indexed data managers through Informix DataExtract, an add-on tool. Oracle7's interoperability offering on the server side is SQL*Gateway, which is similar to Sybase's OmniSQL Gateway and accesses a similar variety of data sources. The company also announced a joint effort with Novell called OracleWare, which addresses desktop operating environments and messaging as well as the database server. The curious thing about OracleWare is Oracle's entry into the messaging/e-mail sweepstakes. The partnership does offer interoperability between Oracle Office and Novell NetWare Global MHS messaging. But many developers already are impatient with the messaging API war (Microsoft's MAPI versus the Lotus-led consortium's VIM), which currently is the most important technical roadblock to groupware development. Another messaging interface merely adds to the confusion. Oracle's client-side interoperability product is Oracle Glue, a programming utility that works with object management protocols on Windows and Macintosh clients. With Oracle Glue, programmers can make data on an Oracle server available in Windows applications, without extra programming. Although Dynamic Data Exchange (DDE), Object Linking and Embedding (OLE), and Apple's analogous Publish and Subscribe were originally developed to promote application-level interoperability on a single machine, distributed filesystems have made them into something else. You could think of these mechanisms as higher layers on a network protocol stack, somewhat like the elusive sixth and seventh layers of the OSI networking model. In this context, Oracle Glue is an important innovation. It makes Oracle the only database vendor, other than Borland, that is addressing application-level interoperability with today's desktop environments. It's becoming obvious that advanced systems users need more than just ad hoc middleware solutions that address limited nooks and crannies of the connectivity world. Unfortunately, the free market demands that standards wars must be fought before any progress is made. The good news is database vendors are dealing with these issues instead of clinging to now-outmoded notions of proprietary interfaces.

Besides database performance, data distribution, and interoperability, server vendors also must wrestle with several areas outside their expertise, including CPU and network scalability, data complexity, and application-development tools. We'll examine this and more in Part 2.

2 Part Two: Unix RDBMS - Developing a future

Enterprisewide systems, the second generation of client/server architectures, demand expertise in many technologies – a variety that may humble current relational database management system (RDBMS) vendors who dominate the Unix information systems world. What are they doing, and what do they need to do to survive in that next generation? In the first of this two-part series, we introduced the leading Unix RDBMS vendors (Informix, Sybase, and Oracle) as well as a dark horse, Borland. In this month's conclusion, we look at how these vendors are addressing issues of scalability, data complexity and integrity, and application development, while we glance into the future of Unix database servers.

2.1 Scalability

In today's expanding Unix database market, the emphasis is on more – not necessarily bigger. So a major issue now confronting vendors who are designing the next generation of RDBMS is how to accommodate more and more of the variety of machines added to the network with as little effort as possible. Today, all of our selected RDBMS vendors provide for scalability – as long as you run their database on all of your servers. Informix, Interbase, and Oracle let users set up tables on remote machines and access them by means of synonyms, which are local aliases for remote tables. Sybase will offer similar functionality through an add-on component called Navigation Server in System 10. A synonym-like feature in all these database products also lets programmers perform distributed joins, possibly without even knowing they are distributed, and move tables around from one server to another without having to rewrite existing SQL code (see diagram Data-location independence). One of the tasks for next-generation scalability planning is determining in advance which data to put on which machines. More often than not, this is done for purely pragmatic reasons like geographic necessity. But it's also an opportunity to partition local data onto multiple servers to speed up queries. This could involve putting separate data tables on separate machines, replicating some tables on multiple machines, or a combination of the two. Either way, the task is a coarser-grained variation on the parallelization theme discussed earlier (see "Unix RDBMS: The next generation," February 1994). Programmers also might parallelize multiple-server queries manually by experimenting with table locations until a given task takes the least amount of time – a very tedious job. Sybase's Navigation Server, when it is released, will help automate the process when used in conjunction with another System 10 component called the Configurator. The Configurator generates statistics on server use and performance, and can be used by the Navigation Server to determine where to put the data for maximum performance. The Navigation Server also can be more effective than a manual approach because it's able to partition the database more finely by actually slicing up tables into parts that are put on separate machines (PDQ technology was also discussed in last month's article). It does all of this transparently to application programs, which aren't even aware on what server(s) the data resides.

2.2 More datatypes

Objects, objects everywhere! Nearly everywhere, that is, except in Unix RDBMS. Standard SQL, the programming language of choice for all of our vendors' systems, handles only simple datatypes like integers, floating-point numbers, and fixed-length character strings. Yet, users are demanding database services for more and more complex datatypes – publishing documents, graphics, images, and other mixed forms of information. There's nothing in the underlying relational model that places limitations on what kinds of data an RDBMS can support. So the question is how to extend the data model in a way that supports complex types, yet preserves the relational model and does not hinder the database's industrial-strength qualities. Enter the BLOB: Binary Large Objects let programmers reserve arbitrarily large amounts of database space for lumps of binary data that have no special meaning. (In an object model, by contrast, semantics – how the data is arranged and how it can be manipulated – are in interface definitions.) You can store anything you like in a BLOB; in most cases, you can treat a BLOB like any other data field in relational queries. BLOBs are a relatively new feature invented by Interbase in 1987. Now all four vendors support them. Interbase goes one step further than the rest, however, with its concept of the BLOB filter. Normally, BLOBs must be managed by client applications. Filters, on the other hand, run on the server and give BLOBs some semantics while still remaining true to the relational model. For example, a BLOB filter on the server might convert text to and from a special desktop publishing format, or compress data to save storage space. The market for development tools grew by 78% from '91 to '92. (Also see News, March 1994, for two other approaches to extend datatypes, by object database vendor Objectivity and by RDBMS start-up Montage.) Another much-needed datatype that standard SQL doesn't support is the array, which is crucial for many applications, especially in science, engineering, and finance. In programs created with other common languages that contain the array datatype, arrays occupy only the space required by their data elements, and access to any element in an array takes just one processing step. Without the array datatype, however, an RDBMS must handle even the simplest array list as a two-column table, where one column is the actual data element and the other is an index to that data element – a colossal waste of space. And it leads to unnecessarily complicated and abstruse programming. Interbase is the only one of the four RDBMS we profile that handles arrays. Its proprietary programming language, GDML, which can be intermixed with SQL within database programs, supports multidimensional arrays by treating them internally as BLOBs. Closely related to the need for more datatypes is the issue of data integrity – the assurance that data adheres to a set of restrictions and conditions. The SQL89 Level 2 standard includes several features related to data integrity, and all four vendors conform to these standards. Sybase also supports part of the new SQL92 standard, which has extended data-integrity features. The SQL89 and SQL92 standards let you specify what to do when, for example, a department row is deleted from the database: The employees whose department number is being deleted can have their department numbers reset to some default value or to NULL. Or, as is more relevant to the business practices of the early '90s, the employee records might be deleted as well. This referential data integrity is called a cascade in the SQL92 standard; you could implement it with triggers if your database doesn't support SQL92. The check domain integrity feature of SQL92 is much more interesting: It lets you define a condition that the data must satisfy on input. For example, an employee's sex must be "M" or "F," and their salary must be less than that of his or her boss. Interbase's GDML implements similar domain-integrity features and adds a few more. Most important is an edit mask, similar to those used in report-generating tools that constrain the values of character strings. For example, a telephone number might be "(xxx) xxx-xxxx," where all of the xs must be digits.

2.3 User and developer tools

The revolution in application development and graphical user-oriented database querying tools has become an extremely important issue for vendors in the RDBMS market. Now that many organizations have gotten over the initial rush to downsize from mainframes and minis, they are getting down to the business of re-implementing legacy programs and developing applications for their new client/server architectures. International Data Corp. research shows that the market for development tools grew by a stunning 78 percent from 1991 to 1992, and was expected to grow 45 percent in 1993. Today, there is a bewildering array of client application-development tools. But developers aren't the only ones with choices to make. Database vendors must choose how much they want to get involved in the highly volatile and competitive market, let alone what to offer. Although database-server vendors obviously can profit by selling their own client tools, that market is dominated by independent vendors. There are two main reasons why this is so. First, until recently, most database vendors – if they invested in tools at all – put their resources in computer-aided software engineering (CASE), a market that has done poorly in the transition from mainframes to the client/server realm. Second, the independent start-ups who have delivered more up-to-date tools chose not to seek backing from database vendors because it would limit their tools to a particular database. Instead, the independents often aligned themselves with client operating system vendors, as Easel did with IBM, or they took advantage of Wall Street's infatuation with client/server technologies and sought money there, as Powersoft, KnowledgeWare, Gupta, and others did. As a result, database vendors are struggling not only against fleet-footed, independent tool start-ups, but also against their own lack of experience with next-generation client tools. In general, the latest application-development tools include a range of products from simple "point-and-shoot" query-building tools meant for users, such as Gupta's Quest, Trinzic's Forest & Trees, and Powersoft's PowerViewer, to highly sophisticated, full-featured development environments for professional programmers, like Gupta's SQLWindows, Powersoft's PowerBuilder, and KnowledgeWare's ObjectView. The popularity of these third-party tools has skyrocketed – especially those in the professional category, including GUI builders with complete programming-language systems (usually resembling a 4GL or BASIC), interoperability with other GUI desktop applications like spreadsheets and word processors, and "back-end" software that translates queries into SQL for any of the major networking platforms. Most of them also include some form of a programmable spreadsheet-style window (called a DataWindow in PowerBuilder parlance) for viewing query results. Since a spreadsheet is a familiar interface and applies very well to table-oriented relational data, the DataWindow is a very handy tool. PowerBuilder-style tools also can slash development time and produce powerful, appealing, and user-friendly – albeit often inefficient – applications (see "GUI-based development tools," April 1993). RDBMS vendors now must scramble to produce tools that match or exceed these third-party benchmarks if they are to compete at the high end of the tools market. So far, the RDBMS vendors have been able to identify ways to add significant functionality to the PowerBuilder paradigm, but they have not been able to deliver timely products based on those new ideas.

2.4 Grow or buy?

Sybase entered the tools market by acquisition rather than by growing its own expertise. In 1992, Sybase bought Gain Technologies, makers of the GainMomentum visual programming environment. The tool featured an object-oriented development paradigm and support for multimedia objects. But Gain used its own proprietary object-oriented database, and was only recently modified for direct integration with Sybase's SQL Server. Another problem with GainMomentum is that it runs on Unix clients rather than on DOS- or Windows-based PCs or Macintoshes. Sybase has wisely decided to move beyond GainMomentum. The company moved its tool-building activity in house, and is working on a tool, nameless so far, that has various new features, including support for multiprogrammer project management. But Powersoft has already introduced similar capability in its latest release of PowerBuilder, so apparently Sybase must still play catch up. Oracle has been doing a better job of building in-house tools expertise, presumably by leveraging its considerable experience in mainframe CASE tools. So far, the company has produced a number of different development tools and cobbled them together into a loose framework called the Cooperative Development Environment (CDE), which includes interface portability between Windows, Macintosh, and Motif. However, none of Oracle's tools has all the desirable features in one package. It updated Oracle Forms from a character-based form builder to a GUI builder. But users must purchase other Oracle tools to handle such things as report generation (Oracle Reports) and multimedia (Oracle Graphics). By comparison, Informix appears to have a tools strategy with the best chance of competing with the independents. Like Oracle, Informix has a good amount of in-house tools-development expertise. But Informix's experience lies in the 4GL market – where the company enjoys a leading reputation – rather than in CASE tools. They have decided to build directly on that experience by offering a tool called 4GL++ that not only includes all of the right stuff, including object orientation, team-programming support, and a full-featured debugger, but also is backward-compatible with the company's highly successful 4GL language. (4GL++ is scheduled to ship in mid-1994.) Informix's 4GL++ and Oracle Forms share another significant feature: database independence. Both vendors have pledged support in their respective tools for RDB interoperability standards, such as Microsoft's Open Database Connectivity. That way, they potentially might compete head-to-head with the independents. Informix and Oracle also are acknowledging the increasing reality of heterogeneous installations, wherein proprietary tools need not apply. Nevertheless, all of the database vendors will have a difficult time commandeering a share of the high-end programming tools market away from the independents. They are, however, likely to do better at the low end – the market for user tools. While success in the high-end tools market seems predicated on adding more and more features, the low-end market is more elusive: Users do not want an endless parade of bells and whistles. Instead, they want ease of use and views of the data that make sense. Satisfying these desires requires experience with corporate users who aren't technological wizards. The independent tool makers don't necessarily have more user experience than the database vendors; so the RDBMS vendors stand a better chance of competing in the low-end than in the high-end tools market. Both Oracle and Informix offer highly competitive, user-oriented querying tools. Oracle's Oracle Card gives the user a hypertext-style interface to data – including multimedia – in the manner of Apple's well-known HyperCard tool for the Macintosh. Informix's HyperScript Tools are similar, except they add more programming functionality. Both Oracle Card and HyperScript Tools run on a variety of GUI platforms, including Windows, Macintosh, and Motif. Oracle and Informix also offer spreadsheet-style data-viewing tools called the Oracle Browser and the Wingz spreadsheet, respectively. Informix's newest tool, ViewPoint, occupies a very interesting middle ground in the continuum between professional programmer and user tools. Viewpoint's SuperViews feature extends user data views beyond those obtainable through standard SQL by supporting more descriptive data-naming conventions, and by allowing more types of table joins, for example (see screen shot above). Thus, Informix takes the position that a modicum of programming effort put toward hiding the grungy details of an RDB goes a long way toward making that database a much more useful decision-support tool. ViewPoint's Superviews is an intriguing, new approach that still must stand the test of time in the user-market. (ViewPoint currently runs on Windows only, but Informix has announced its intention to port it to Macintosh, Motif, and Windows NT.) Sybase has no serious product in the user-tools market. Interbase, on the other hand, seems to have the best chance of success in that arena, since Borland has more experience with user software in general than any other database vendor. And the company has an enormous amount of experience in the software-tools market, being a leading maker of programming-language systems for the PC. So Borland has positioned Interbase in the center of a tools strategy that, unlike the other vendors' efforts, starts with the needs of the user and works its way down to the database server, rather than the other way around. Borland's BOCA (Borland Object Component Architecture) promises integration of the company's popular dBASE IV and Paradox databases andwill integrate its Quattro Pro spreadsheet with Interbase. However, it has been slow to deliver on that promise. So far, Borland only has released SQL Link, a client/server interface for Paradox 4.0. SQL Link gives users transparent access to relational data on a server. The Interbase version of SQL Link also adds support for BLOBs and other features unique to Interbase. Borland also has been late in delivering their Integrated Database Application Programming Interface (IDAPI), the company's response to Microsoft's ODBC database interoperability standard that combines the functionality of RDBMS with navigational, flat-file systems like dBASE. With IDAPI, Borland recognizes the increasing importance of flat-file PC databases in companies' enterprisewide integration strategies. But its inability to deliver tools based on that keen observation has forced many of the users and developers to turn – once again – to independent tool vendors for their solutions.

2.5 The future of RDBMS

Many of the issues we have covered in this two-part article just weren't considered important in first-generation client/ server systems. Today, however, users and developers are demanding radical changes toward enterprisewide, heterogeneous client/server architectures and demand that RDBMS fit in or perish. As discussed last month, database performance is in the forefront: Without dramatic improvements brought by technological advances for multi- and parallel-processing of queries, the RDBMS just won't be able to keep up with the frenzy of transactions envisioned for future highly interactive, distributed database systems. Data distribution and interoperability, which we also discussed last month, are both directly related to the movement from homogeneous departmental LANs to heterogeneous client/server systems throughout the entire enterprise. Only part of the technology the vendors are bringing to bear on these problems relates directly to the database server itself; many other solutions lie outside database vendors' traditional domains. Indeed, application-development tools take the RDBMS vendors even further away from their core competence. And problems related to data complexity and scalability, like the replication issues for distributed databases we detailed last month, are only beginning to be understood, let alone addressed. Initial attempts at solutions are impressive, but the database vendors have quite a long road ahead in these areas. Even so, all of our selected RDBMS vendors – Borland, Informix, Oracle, and Sybase – have reached a plateau: Each offers basic features which are needed to support next-generation, distributed client/server information architectures. For those of you who are ready to make the leap from departmental enterprisewide client/server RDBMS, it would be hard to go wrong with any of them. The most significant differences between these vendors' systems have more to do with interoperability than with performance, robustness, or other traditional parameters. Right now, Informix is in third place, Sybase second, and Oracle leads in the Unix RDBMS market. Oracle has a more comprehensive set of products for the next generation of enterprisewide, client/server database systems. The company is also well established in the mainframe and midrange markets, and is working to dominate in the increasingly important desktop arena. Oracle's PL/SQL language is the most advanced of the vendors' extended SQLs, and the company has developed an impressive array of in-house tools expertise that has led to a wide product line. Oracle just has to make sure its many products and services are sufficiently integrated; otherwise, it will suffer from IBM's problem of too many divisions trying to do too many things with too little in common. The CDE framework shows that Oracle is moving in the right direction. Borland's acquisition of Interbase, the dark horse in the RDBMS race, shows the company's desire to approach the next generation of database systems from a unique direction – from user applications and programmer tools out to the enterprise. Interbase has some truly unique features, especially in the database-programming area, that can give them significant market momentum; but only if Borland markets those products properly. Interbase's peer-to-peer architecture puts it in an excellent position to solve the data-distribution puzzles of the future, which will undoubtedly require a more flexible networking approach that even surpasses the client/ server paradigm. Like Oracle, Borland has a wide spectrum of excellent products that would be formidable if integrated properly. Unlike Oracle, however, Borland has the advantage in BOCA and IDAPI, which promise to deliver a high level of systems and software integration. Now if Borland would just deliver: Although IDAPI is the most widely applicable database interoperability standard, its lateness and fluidity have led some analysts to dismiss it as "marketecture." Borland has a difficult job ahead selling its interoperability products against Microsoft's established ODBC. Meanwhile, Sybase and Informix are starting out somewhere in the middle of the hike toward next-generation systems. Both hope to gain from the very recent movement by open systems vendors towards "superservers" like Sun's SPARCcenter 2000, which they hope will replace mainframes and minis. Systems like that will allow Sybase's and Informix's respective midrange-server products to graduate to the big money and big-CPU world with little effort on the vendors' part. Sybase and Informix also offer products for the low end of the hardware spectrum. Informix will capture the MS-DOS market by fiat with Informix-SE, while Sybase, whose low-end product only runs on OS/2, will benefit from its close relationship with Microsoft. With System 10, Sybase is expanding on its original strength as a client/server vendor. The company is taking several, very ambitious steps in the hope of leading the industry in advanced data distribution, including replication, location transparency, and PDQ. However, System 10's ancillary products are very complex, so Sybase will need to expend a lot of marketing effort educating potential customers of their benefits. But the effort will pay off if the high-level abstractions defined in System 10 turn out to be the right ones for meeting real-world data-distribution needs. Informix's distribution features, on the other hand, are simple, robust, and cautious. They seem to be taking the conservative position that the hard problems in real-world data distribution are largely unknown at this time and, therefore, currently proposed solutions are premature. So, the company has decided not to expend effort on large distribution solutions but to concentrate instead on development tools. With 4GL++ and ViewPoint, the company's solid reputation in the tools area looks like it will continue well into the future.

3 Glossary

Atomicity: The guarantee that a transaction runs either to completion or not at all.

Cluster: A group of computers connected by a high-speed network that work together as if they were one machine with multiple CPUs.

Locking: The traditional way to guarantee read consistency gives one process exclusive access to data and makes others queue up and wait for one-after-another access to the database.

Multithreading: Threads are small ("lightweight") processes that can be quickly and easily switched by the operating system. This software method of parallel processing also maximizes use of a single CPU to process queries more efficiently.

Parallel Query Decomposition (PDQ): A software method by which a database query is broken into components that can be executed in parallel by multiple CPUs.

Read consistency: The concept that data in use by one process is not irreconcilably modified by another process.

Remote procedure call (RPC): A way of executing a program on a remote computer, such as in client/server architectures; clients often execute RPCs on servers.

Replication: Method by which copies of a database are distributed to remote servers.

Schema: A set of definitions of data tables that, taken together, serve as the template for all data in the database.

Stored procedure: A program stored along with the database so that it can be executed by a client usually via an RPC.

Threads: Lightweight processes, distinguished by the fact that they can be switched more quickly than whole processes.

Transaction: A complete set of database operations that make all necessary modifications to the database, takes the database from one logically consistent state to another.

Trigger: A specific type of stored procedure that automatically executes when a certain event occurs – when a record is inserted into a given table, for example.

Two-phase commit (2PC): A mechanism for enforcing transaction atomicity. In the distributed-database world, the two-phase commit also is used as a way of ensuring that all servers have identical copies of the database at all times.

Versioning: This new way of guaranteeing read consistency without locking gives each process its own version of the data, and reconciles modifications later.

back end: The database that resides on the server.

cascade: A transitive closure operation wherein an action taken on records in one table of data summarily effects related data records in other tables. For example, when a data record in Table 1 is deleted, corresponding, related records in Tables 2 and 3 also are deleted.

data view: A "virtual" database table that looks like the real thing to the user, but actually is constructed from several real data tables.

distributed join: A link between database records that are stored in different tables on separate machines.

join: A basic database operation that links records in one table with records in another table; for example, employee names in an employee database table with department number in its respective table.

synonym: An alias or nickname for a table, usually used to make it appear as though it resides on the local machine, whereas it actually resides in another remote location.

4 RDBMS players

Informix OnLine 6.0, the company's high-end offering released at the end of 1993, runs on most flavors of Unix and Novell NetWare version 3.11. Informix-SE, the low-end database server, runs on MS-DOS and NetWare 2.2, as well as a variety of Unix for PCs. Informix Software Inc., 4100 Bohannon Dr., Menlo Park, CA 94025, 415-926-6316.

Interbase 3.3 is available for most Unix platforms, VAX/VMS, and HP/Apollo Domain. By the time this article is published, ports to Novell NetWare 4.0 and NextStep for Intel 386/486 should be available. Borland International Inc., 1800 Green Hills Rd., Scotts Valley, CA 95067-0001, 408-438-5300.

Oracle7, released in late 1992, runs on, as a colleague of mine put it, "just about everything, including a box of corn flakes." Supported platforms include most major flavors of Unix, including SVR4 for Intel processors and AU/X for Macintosh, OS/2 2.1, Novell NetWare, Digital VAX OpenVMS (the primary platform), IBM MVS and VM, and a few other mainframe platforms. A Microsoft Windows NT port is planned for release soon. Oracle Corp., 500 Oracle Pkwy., Redwood Shores, CA 94065, 415-506-7000.

Sybase System 10 runs on most flavors of Unix and VAX/VMS. It consists of SQL Server 10, the latest version of the database engine, and several ancillary products that are mostly implemented on Sybase's proven Open Server architecture. Sybase Inc., 6475 Christie Ave., Emeryville, CA 94608, 800-879-2273.