RDBMS Workshop: Technology in the 1980s and 1990s

Moderator:
Burton Grad

Recorded: June 13, 2007
Mountain View, California

CHM Reference number: X4069.2007

Table of Contents

RDBMS Workshop: Technology in the 1980s and 1990s

Conducted by Software Industry SIG – Oral History Project

Abstract: A group of relational database technology pioneers from Oracle, Informix, Sybase and Ingres discuss the technology advances made by their companies during the late 1970s through the 1990s. They discuss the major new technological problems which they had to deal with including the need to improve performance particularly for OLTP applications and for complex queries. There is a lengthy discussion of the role played by the various standards committees, both domestic and international. There is also a discussion on security issues and on other database models which were developed.

Participants:

Name	Affiliation
Burt Grad	Moderator
Michael Blasgen	IBM
Marilyn Bohl	IBM, Ingres
Don Chamberlin	IBM
Sharon Codd	IBM
Chris Date	IBM
Roy Harrington	Informix
Jerry Held	Ingres, Tandem, Oracle
Ken Jacobs	Oracle
Bruce Lindsay	IBM
Stu Schuster	Sybase
Jim Strickland	IBM
Moshe Zloof	IBM
Peter Capek	Historian, IBM research
Michael Mahoney	Historian, Princeton University
Jan Phillips	SI SIG, DEC

Burt Grad: Yesterday we talked about the early RDBMS developments primarily at IBM. Now we will focus on RDBMS technology outside of IBM. Basically, what happened from

1983 on? What were the major technical developments, not the business issues, except as they affected why we did things technically? What were the major technical developments over the ten to 15 years after 1983? What happened? What changed in the relational products? How did they get different, better, in what ways? So I'm ready to start whenever anybody wants to.

Sybase Technology

Stu Schuster: Well I can start. Sybase came into the market later, so it had to identify an area where there was a hole in the marketplace, an uncontested area. We concluded that the open area was a portable general purpose relational database that could perform transaction processing and networking functions. Other proprietary companies, Tandem for instance, had pioneered some of these ideas in their database file system, but without SQL. So Sybase hung its themes on SQL, networking and online transaction processing. We began development in 1984 and shipped in 1986. The main points that we were pushing were performance, integrity, availability, distributed architecture and a tool set which wasn't the strongest part of it but it was one of the first graphical tool sets. On the performance side, the breakthrough was designing, from the Britton-Lee base, a database operating system. The server ran as a single process in a VAX and UNIX environment and had its own internal scheduling, queuing, multitasking and locking concurrency control, all within a single OS process. Because of this design, spawning another process within our system was very efficient. We could do 20, 30, 40 users on small VAX's very easily.

Grad: You picked the VAX platform for what reason?

Schuster: We did all our development on Sun and UNIX. Sun offered a networked architecture, one of the first true networked architecture. Our Britton-Lee base was a back end machine and so separating the application development user interface from the server was a natural extension of the Britton-Lee architecture. Sun had the best networking architecture for UNIX computing at the time. We targeted VAX as a necessary element because Sun wasn't in the commercial marketplace and DEC was. I do believe Sybase helped Sun dramatically in their efforts to get into the commercial marketplace. But VAX was where the commercial work was being done, so a port was instigated very early in the development cycle and the product came out simultaneously on Sun and VAX VMS.

Grad: It was VAX VMS and Sun UNIX?

Schuster: Yes. You could get Sybase in any flavor you wanted as long as it was one of those two. So were up against people that had 30 or 40 platforms already and plans to go to IBM's platforms as well. So we had to make a virtue out of this and so we said to people, "If you want to do transaction processing we're the only game in town."

Grad: Stop a second. We haven't covered the Britton-Lee thing in this technical session. Can you give us a little background?

Schuster: I can't really.

Grad: All right. Just a note though we should pick up some of that information about Britton-Lee because that was mentioned to me as one we should pick up here but, again, we set fences around how many companies we could pick up but we should get that information. Okay, go ahead.

Schuster: From a performance point of view it was extremely fast relative to other competitors on updates, and as you added users, performance didn't degrade. Sybase was the first for online client/server applications, pretty much the whole client/server concept got associated with the Sybase architecture. We also had data integrity because all of the triggers and stored procedures, things that were used to maintain the integrity of the database separate from an application, could be now programmed into the database itself. Our approach to high availability was borrowed from the other high performance systems and we had the first set of mirrored disc technology in a portable relational database. As far as distributed databases, other people had done this before, but we were the first to introduce the two-phase commit to the portable relational database for distributed transactions across servers in a network. This story together gave Sybase the ability to come successfully into the marketplace even though we were late. We didn't know how big this market could be. That was kind of unknown because in the mainframe market it was the decision support systems that came first and then the transaction processing systems later. These two markets shared the mainframe database market about equally. So the argument to the investors was, "Well, we probably will see the same thing in relational databases because the current applications were light update, heavy query, because that was the strength of relational in the beginning and we felt that the same pattern would hold true in the smaller relational market. We thought the market would start with lighter-weight, query intensive applications. All the benchmarks, in fact, for relational databases prior to Sybase were primarily the DeWitt benchmarks from Wisconsin, which was a series of increasingly complex queries when Sybase entered the market. We went after applications that other people couldn't. Soon the focus began to shift to the work that Jim Gray did on the TP1 [also ET1] benchmarks, the transaction-based benchmarks. Over time, systems matched each other's performance but I think that was our big contribution, the de-separation of the application and front end which gave credence to the whole concept of client/server architecture, putting the application on a PC or a workstation and then the server and then distributed servers and high performance.

Grad: Where did the technical people come from? What was their background and so forth?

Schuster: Most of the people founding Sybase came from Britton-Lee where they were doing high performance back-end database work as an off loading server. But the Britton-Lee design was very different; it was a direct connect to a minicomputer as opposed to a general purpose network and it had very specific hardware with software on it as opposed to a general purpose computer. A lot of the people came out of UC-Berkeley, students and people that were in the software industry. It was clearly a Berkeley thing, you know, with Bob Epstein's and Britton-Lee's culture and environment. We also picked up a few people from Ingres.

Grad: Did you get any of the people from IBM?

Schuster: No, but we did some for our sales organization. We picked up a lot of people from the mainframe independent software companies. That's where a lot of the people came from.

Sharon Codd: Actually you did get one person from IBM and that's Bill Shaw who was the rep to the ANSI Standards Committee. I know that he went to Sybase. I don't know the year.

Grad: The ANSI thing is mid-1980s right? He was a very active member of that committee, apparently.

Ken Jacobs: Very much so. And of course Bill Shaw ended up at Oracle.

Grad: Well, you got all the good people obviously.

Schuster: Back to the Sybase approach, almost the entire marketing pitch was done on two slides, for both fundraising and selling. One slide showed non-relational decision support versus OLTP products and the other slide showed decision support versus OLTP products in relational. There was a big hole and we could argue, and people believed it, that we had the performance and the features to fill that hole. On the marketing side, what's the difference between decision support and online applications? We had a chart that showed one was update intensive and one was query intensive. One was simple queries versus complex queries; one had a larger number of users, updating versus small and that was basically almost the entire pitch.

Grad: You had a two-year implementation period. Did you have serious technical issues that you had trouble overcoming during that period?

Schuster: Yes. We talked about this in the company session. When we went out to beta test, Mark Hoffman got a phone call when he was on the road and I had Tom Hagen, one of the technical founders, come into my office and he said, "Well there's good news and bad news.

The good news is the beta sites are all in development. The bad news is multi-threading concurrency doesn't work. We had just done our announcement of what we were going after because we had 12 companies developing products doing transaction processing prep. I was thinking that this is going to be great. We're going to have the fastest online transaction processing system for one user. And it was about a three to five month period where I wasn't sure if we had a fundamental design flaw and we would be basically dead in the water in delayed revenue, delayed sales. The only thing that saved us was that the customers really didn't know because when they were in development they weren't really taxing that part of the system. And all the early first customers for six months were essentially doing development. Nobody had deployed yet.

Grad: That's interesting.

Schuster: Ultimately it was a set of bugs that just needed to be fixed and then it worked beautifully.

Grad: It was not a design flaw then.

Schuster: It wasn't a design flaw, no, but it was pretty scary.

Grad: When did it first ship, Stu? When was the first production shipment? You said 1986. Did you mean it?

Schuster: It was either late 1986 or early 1987. I'm not sure because the first full year of sales was 1987 and that was a $5.7 million year. Obviously, things worked pretty well because the next year we did much better, almost $25 million in revenue.

Grad: I'm going to break off from Sybase. What was Informix doing during this period in terms of the technical stuff? How were they doing?

Informix Technology

Roy Harrington: Informix started off as a UNIX-based database, so that was our niche. We were going to be UNIX and we ran on a whole pile of different machines. The first parts were built straight onto the UNIX operating system, the UNIX file system using ordinary files, using UNIX to do the concurrency control which was very rudimentary in the beginning. And in about 1984, 1985 it was clear we needed to get further down into the system in order to get performance. Benchmarks were becoming more important and we needed to be able to control all those things, so we went accessing raw discs directly, doing the whole file system on the disc, that particular disc, doing big shared memory that was accessed by multiple processes.

We didn't have client/server exactly. We had multiple server processes all working on one shared memory so it was a multi-threaded backend but the only thing that was shared was the memory, as opposed to the whole process. At that same time we converted to SQL as a database sublanguage.

Grad: That was a transition you were making then.

Harrington: Both of those things kind of happened together.

Grad: You started with SQL at Sybase?

Schuster: Yes, but it turned out that the SQL standard hadn't completely jelled and we had some deviations from it. And we had to deal with things that hadn't been invented yet, like stored procedures which were critical to us. Also, we had page level locking so our transaction model was slightly different. The basic syntax was the same but we had to make changes in I think our fifth version to get to ANSI compliance.

Grad: The fifth version would have been when, late 1980s, 1990s?

Schuster: Yeah, late 1980s, yeah.

Grad: With Informix, were you looking at the online transaction processing at this point in time or strictly the query type thing?

Harrington: I think we started doing the transaction benchmarks somewhere in the mid-1980s. I think it was before going public, so it must have been 1985.

Schuster: Yes, we started to force that issue. We basically were the first to publish a TP1 benchmark and then it started to be something that everyone needed to do. I think Sybase pioneered pushing that whole thing as part of a marketing strategy.

Grad: I want to try and bring in all the companies here. IBM is continuing with its development?

Codd: Well, IBM announced DB2 in June of 1983 and things had gone so well in the test that they moved the announcement from September to June to catch everyone before summer vacation. I think Marilyn [Bohl] became the product manager, was it May or June of 1983?

Marilyn Bohl: I don't remember.

Codd: Oh, well maybe later, 1984. The reason I know this is that I was in the International Executive Briefing Center at the time and I did not get along with the current DB2 product manager and so basically I was waiting. I knew a change was going to occur. I knew that from the lab director and so they held me in the briefing center and as soon as Marilyn was announced I went over.

Grad: So you could work with her?

Codd: In fact, no, I wasn't held in the briefing center. I worked in the IMS organization for about three to six months.

Grad: They allowed you to do that? That sounds practically traitorous.

Codd: Absolutely, absolutely.

Chris Date: Were you a mole?

Codd: No, I was not a mole.

Grad: Marilyn, let me ask you a question about this. What were the technical things going on there in DB2 during that period?

Bohl: You need to get Don Haderle down here. He can run circles around both of us. You really need to get Don Haderle.

Grad: Okay, well we'll follow that up later then.

Bohl: I had two folks, Don Haderle for the database related things and Bob Jackson for the system related things.

Grad: Do we have someone here from Ingres at that time.

Schuster: Well I was there in product marketing. You're talking about the early 1980s?

Grad: I'm talking about the early to mid-1980s. When did you go to Sybase?

Schuster: I went to Sybase in 1986.

Grad: Oh, really because you were not a founder then?

Schuster: No, no. I was one of the first outside executives to come.

Grad: Okay, well tell us what was going on in Ingres then because you were there?

Ingres Technology

Schuster: Yes, I was. I was at Ingres from early 1983 through mid-1986 when I joined Sybase.

Grad: Okay, that's a good time. What was happening technically there?

Schuster: A big emphasis really was on the tools. They were involved in the QUEL language at that point and, in fact, the poster for Ingres at the company was the BNF of the QUEL syntax. It was up on a wall. Actually it was quite beautiful I mean to look at because it was a pretty diagram the way it was done. So the company had a very technical base. From an engine point of view it was mostly performance improvements and adding features and functions, but I think the real emphasis was on their tool set. That's what distinguished Ingres in the early days -- they had a great productivity tool set, very strong productivity tool set and that really was their strength. Somebody made the point yesterday in business meetings that in Europe, the tool set was the major evaluation tool for a lot of the competitive wins because they were more interested in productivity than performance. They weren't running benchmarks there in those days. They were really evaluating tool sets, whereas in the U.S. we always had benchmark wars.

Codd: I wanted to ask a question and that is at what point did Ingres switch from QUEL to SQL? Was that while it was still a prototype?

Date: I can talk about that. When I left IBM in mid-1983 one of my first jobs was consulting with Ingres and I did that for 'much of 1983 and 1984. We were doing presentations for them and writing about them and so on. But towards the end of that time they were getting concerned about SQL and QUEL. They saw their competitors across the Bay going very strong with SQL and so they decided and I remember the phrase, "At least for cosmetic reasons" they had to have an SQL interface and the way they would do it initially was to map SQL to QUEL. My job was to define those mappings. Now there's irony for you but that's what I did.

Schuster: It was definitely emulation in the beginning.

Date: It must have been around 1987 or so.

Bohl: I got to Ingres in 1989. Release 6 was a major rewrite that they had treated like

it was a maintenance update. It was out in many, many sites and very, very buggy and no performance at all. That was the first native SQL implementation. That was like release 6. It had been in beta sites for over a year I think when I got there, so it was the release before that that was the emulation.

Oracle Technology

Jacobs: I wanted to pick up on some things Stu was talking about. Stu, you certainly did identify an opportunity where we were not playing but in each of the areas that you spoke about, you left us an opportunity to come back. For example, you did the data integrity with stored procedures and triggers but did not do declarative referential integrity. We were able to show with one declarative ANSI compliant declaration how database integrity could be defined versus your solution of a whole set of code and triggers. The same was true about two phase commit. In Oracle it happened transparently as we did a distributed transaction. In terms of concurrency when we did version 6, we implemented row level locking. You guys had page level locking. And your process architecture, which was beautiful for the uniprocessor, didn't work on an SMP. So those design decisions actually turned out to leave an opportunity for us to come back with Oracle 7. We had even laid a lot of the groundwork for that in 6, which we started writing in 1986. In 6 we didn't have stored procedures, so we ended up spending a lot of time talking in the 1980s about Oracle 7 which delivered the stored procedures, the declarative integrity and other features, the transparent two phase commit. We took the opportunities you left us and plugged those.

Date: You reminded me of something. Wasn't there a period of time with Oracle where you had declarations on integrity but no semantics at all?

Jacobs: Yes, that was true in Oracle 6 actually. We put a stake in the ground that we were going to do declarative integrity and we made a weak argument that by declaring it, it was maintained in the schema and tools could see it but it was sort of a no implementation.

Codd: Is that what we call vaporware?

What were the Advantages of RDBMS

Bruce Lindsay: Referring to the paragraph discussing this workshop there's a question, "What new features and functions enabled the market for RDBMS to grow and replace other DBMS solutions." We should really emphasize what really did make the RDBMS products displace all these other Nomads and file-based systems. I believe recovery and multi-step transactions were perhaps the key that made this market grow.

If a company was using a system, a DBMS, that didn't have any recovery integrity, all they were getting was fast processing of the stuff they typed in. They couldn't throw away their paper and so it's almost like why type it in? Maybe you could do queries but it hardly paid for itself at that point because you're still maintaining the paper. The second thing that made these systems supplant the predecessors, especially the ones that had recovery, like IMS and IDMS, was online data definition or schema change and ad hoc query. The predecessor systems didn't have any of those features. The first case, the recovery, that was attack your paper pile and the second one was attacking the application backlog which in the 1980s every major IT user had--I mean there was a six, eight, ten month backlog to get a simple change to an application.

Codd: Even more.

Lindsay: The only way that I know people were successful in getting their application changes was to bring gifts to the DBA.

Grad: That worked wonderfully.

Lindsay: Bottles of alcohol were also good.

Date: Oh, my. Not at IBM I wouldn't do that.

Lindsay: Well I'm talking about we didn't have applications.

Date: You didn't have DBAs right?

Lindsay: Another example of how did the marketplace and users influence the shift that took place, and this is more towards the late 1980s, was automatic selection, independence from physical access paths, the planned selection for queries and data updates that you didn't have to know what indexes there were. You didn't have to manage them yourself. Any query that was legal would be executed however efficiently or inefficiently the system figured out to do it, but it would get executed even if you didn't have good indexes or nice layouts. That was important for usability again, attacking the application backlog. It became easier to develop applications.

Another feature, driven by the market and users' influence, something that I certainly know that we spent an enormous effort on, was online utilities.

Bohl: Right, online backup and online recovery versus the IMS batch window that we fought in IBM for years of how do we get that batch window smaller, smaller, smaller.

Jacobs: A lot of these considerations that were important to high end mainframe customers started to migrate down to departmental and other users. Oracle version 6 had online backup and so we weren't yet running big mainframe scale applications but we had to address some of the high availability requirements and security requirements of our customers. As the Internet came around that was even universal that everybody had to deal with those kinds of issues. But I think some of the very same considerations that you were facing in the mainframe space were starting to become important in the departmental systems.

Lindsay: This applies to all users. They don't want to have DBA administrative activities and they don't want to shut down their system.

Jacobs: For a lot of department users it didn't matter. They only worked 9:00 to 5:00.

Grad: That was one of the things I'd like to explore. You were going into DB2 for the large corporate customers, the typical IBM thing. At Oracle you were going to some of those same customers but you were going to their departments and not to glass houses, right?

Lindsay: I think what Ken is saying is that they seemed to have the same requirements as the mainframe guys. Or were there real differences?

Jacobs: I think there were real differences. I mean people weren't 24 by 7, but 9 by 5. The databases weren't as big. The systems that we were delivering probably had more knobs than they needed but they weren't going to be tuned for the kinds of workloads that you guys dealt with.

Schuster: Yes, often you were doing work that ended up getting migrated back and forth. I mean Gateways became a big marketing issue, because once these departmental systems started doing real work then there was this need to talk back to the mainframes through IBM protocols and interface either in a batch or real-time mode, especially in the broker trading systems that Sybase was involved in. Often a lot of things had to be resynchronized back with corporate mainframes.

Harrington: The amount of availability time made it so the small systems could afford to shut down for backup or remaking indexes and things like that.

Security

Grad: Let me go into the area of security for example. Some of your customers obviously needed high security. In the typical IBM large account security would have been a concern, an issue, I assume. Was that so in the departmental areas with these smaller users

that you were running into? Is that something they worried about?

Schuster: What do you mean by security?

Grad: In terms of people hacking in, people getting into the systems, that kind of security.

Schuster: Well there were a number of efforts to do a highly secure version and, in fact, we had a research grant to do a highly secure version and that became a little bit of an issue because there were almost no customers for this highly secure database. People were using the basic same stuff that they did for authentication and access control that they had been using in others. On the hacking issue, it's actually a real vulnerability because somebody could easily have put in a stored procedure or a trigger and leaked data all over the place and we did have these military contracts which could do something to delete that. But that version of the product never really made it commercially and nobody really wanted the overhead associated with that. But I remember there was very famous case when I think there were a series of major bugs published about how Microsoft could be penetrated.

Jacobs: The slammer. That was later though. That was much later I think, in the1990s sometime.

Schuster: Oracle started with CIA. Didn't they worry about hacking?

Jacobs: It started with the CIA. Remember this was before the Internet. There weren't even firewalls. You weren't connected. It was all inside the company so some of the public awareness, the mainstream awareness of security was more limited. But as I said in our session yesterday, when I had started working with NSA as well as CIA, I sat down with the Orange Book which gave criteria for evaluating operating systems and tried to apply it to database; so I invented an audit capability that the guys out here built. I collaborated with them. We subsequently did "roles" which are collections of privileges. The guys working on that inside Oracle you might remember, maybe not Jerry, was Ken, Bob, and Gordon, so it was the KGB project. I took that to the ANSI SQL committee, and I want to spend a little time at some point talking about ANSI and what happened there at the SQL committee, but let me finish the security topic. So, yes, we had a lot of early interest in security, partly because of our clientele, the military intelligence community, partly because I took an interest in it and kept banging on people's doors. We did build this multilevel secure version of Oracle that allowed intelligence agencies to have unclassified, sensitive, and top secret data all on one database and we implemented the Bell-LaPadula model, a model of write up, read down. It was a very interesting, but probably not very practical product in the long run. It had a modest success but things like trusted Solaris and stuff like that also had modest success. They were just not very workable. The only secure thing is a brick. If you don't turn it on, it's secure as long as it's

nailed down.

Grad: I'm trying to work with our timing. We haven't talked about the ANSI SQL.

Jacobs: There's that and I also wanted to talk about the benchmarking wars.

Grad: Okay.

ANSI Standards Committee

Jacobs: As I often said the most difficult activity I did in my career was TPC, but I think the ANSI process was rather interesting because when I joined the committee in 1985 we had a standard that was this big and it covered the basics. I mean insert, update, and delete and in 1989 we added declarative referential integrity and I think that was a very, very challenging thing to do, update cascade.

Grad: Who was on the committee?

Jacobs: Well, we had IBM. We had Ingres, Informix, and Sybase. We had government agencies.

Grad: I thought Stonebraker said he wouldn't put anybody on the committee.

Jacobs: No, Ingres put Carol Joyce on the committee.

Grad: There's a direct quote from him in the book I just read, interesting.

Jacobs: So we had this very simple standard but when we were doing referential integrity in particular, it used to give me headaches. I mean every time we'd get into discussing cascaded updates and how you could end up visiting the same row multiple times, it was very challenging.

Grad: The "how you can be your own grandfather" kind of issue?

Jacobs: Yes, sort of it's a cyclic RI chain. You can go back and see the same row multiple times and the system had to detect that and properly enforce the semantics and that's where the difference between Codd's theory and the ANSI specification and the implementation in products all were different takes on the same problem. Some very interesting work was done there.

Schuster: It wasn't the first time it came up. It's called the Halloween problem, and was first documented at IBM Research in the 1970s. .

Jacobs: Similar kind of thing, exactly. But there was a lot of inventiveness going on and we ended up building things over time in SQL that nobody ever implemented and over time the standard grew to the point where it became so large that it was clear no one product was going to comply with all of it. There was a point where we had to subset it down to entry level, SQL 92. That's the common framework that everybody pretty much complies with. At the boundaries there were some cases where things were a little different.

Grad: What timeframe is that, Ken?

Jacobs: Well, I joined the committee in 1986, 1985, picking up from Jerry Baker who had been at Oracle after he was at IBM. I stayed on until 1993 when I hired Phil Shaw and Jim Melton from Sybase to come to Oracle. Jim is still on the committee and Phil is retired.

Grad: So when is the first public release of an SQL standard?

Date: Sequel 86 is the first.

Jacobs: I think it actually goes back much further than that because Don George is the present chairman and has always been the chairman of the committee on database. He had worked on a network model to define the CODASYL standard. At some point, he managed to shift to a relational framework. And then, even after that, they didn't first do SQL, as I understood it. They went off and did something else and then came back and did SQL. You know the details.

Date: I was going to comment on that. Ken is right, of course. This committee had been around a long time and there were folks in it pushing to do some relational standard. They called it RDL at the time, relational database language. But other folks were saying, "Well, if you're going to do that, you got to do CODASYL stuff as well," so eventually some kind of compromise was worked out and they came out with what they call NDL, network database language. It was not exactly CODASYL and it's never been implemented as far as I know in any product. But we worked on the IDL thing for a while and round about, oh, it was before I left IBM I guess, it must have been 1982 or so, Phil Shaw got me to go to talk about IBM SQL. The committee said, "Okay, here we have something comparative and concrete. Instead of wasting your time with RDL let's focus on SQL." That's what happened and they came out with the first standard in 1986, SQL 86. It was characterized at the time as the intersection of existing implementations. It was very low function and wherever possible they tried to be compatible with IBM. That was a good ground rule.

Jacobs: So it's a fair statement, Chris, despite your position, this is the second irony I guess in your career that you actually are a good reason why SQL is the standard.

Date: Yes, I'm well aware of that.

Grad: Let me clarify here. Was QUEL submitted as a candidate to the committee at that time for consideration?

Date: I'm not aware of that.

Grad: So maybe that was the Stonebraker statement I read is that they wouldn't submit QUEL because it would limit creativity if it was made a standard.

Date: I don't know. That would be consistent I think with Mike's outlook.

Codd: I wanted to eliminate some confusion and that is that CODASYL was not part of ANSI. In other words, there are the American National Standards committees for all kinds of things. CODASYL was a separate organization that was started for the COBOL community and it stood for Common Data System Language. The fact that the ANSI committee chose to pick up the CODASYL work was not a choice made for them. This is something they did of their own volition.

Grad: Help me here because I worked with the CODASYL committee when they were doing COBOL. I was at IBM then. How did they get into the Bachman stuff on IDS, does anybody know?

Codd: The database task group was formed. Actually what happened in CODASYL was that they created subcommittees to examine different technologies and the database task group, by the way, consisted of Charlie Bachman.

Date: I'd like to write a footnote. The database task group when it was first formed was actually called the list processing task group. Their job was to define list processing extensions to COBOL and then they said, ah, but if we put these lists on the disc we have a database system.

Codd: And George Dodd from General Motors, who later became the manager of their whole research lab, was on that database task group committee. He came in with all the lists.

Grad: I appreciate the clarification because I was confused. I remember that CODASYL worked on a lot more than COBOL through its subcommittees and that was separate from

ANSI.

Codd: Right.

Grad: Okay, we're at ANSI. We got the SQL standard. It becomes basically very much the IBM standard or does it match with the others or what is it?

Date: SQL 86 was, as I said before, the intersection of existing implementations at the time so that all the vendors could claim overnight that they supported it. Wherever possible they wanted to be compatible with IBM. That was in 1986. SQL 89 was an extension to SQL 86 where they added primary and foreign keys and a limited form of other kind of integrity checking. It was called the integrity enhancement feature, IEF. The thing about IEF was that support was optional.

Grad: That's a wonderful statement.

Date: If you're writing a code in your product and you would say, "Now I support SQL 89." You have to state whether or not you support it.

Grad: How does that happen? Were these basically technical people or political people?

Date: Technical people.

Schuster: Politically and technically savvy people.

Codd: Well every vendor is protecting his implementation.

Testing for Compliance

Jacobs: The interesting thing was at that time NIST was conducting compatibility tests and they actually had a test suite that would determine whether you complied or not. Thus there was a hurdle you had to get across to comply. Well at some point, the government in its wisdom decided they weren't going to fund that anymore and the game was wide open to making claims without really complying.

Grad: Why was that decision made?

Jacobs: The government decided they weren't going to be in the business of funding. Let

NIST do the tests.

Grad: And the organizations themselves weren't going to fund that because it might not be to their interest.

Jacobs: Well, I think there were people who were trying to find a way to set up an independent organization because each of us felt we were compliant and we wanted to prove the other guys weren't, so it would have been helpful for a period of time to have an independent body that continued these tests. But it just could never happen.

Grad: Did you all get much concern about the compliance with SQL?

Schuster: Yes. At Sybase we weren't 100 percent compliant with the standard at one point and complying became a major part of one of the releases. It was a major release where we had to adapt our syntax and be compatible with old syntax as well as the new.

International Standards

Don Chamberlin: Could you talk a little bit about the relationship between ANSI and ISO? They each had a standard where it was the same I think but how did that happen?

Jacobs: Well the way the American National Standards works is we as an American organization have one vote in the international body. Chris you probably understand this better than I do.

Date: Yes, it's what Ken said. The International Standard Organization has representation from various national standards bodies, so ANSI is just one of those in the system. The U.K. and Germany and Japan, France, and so on, have equal votes in ISO. But, of course, they try all the time to be compatible, so technically it's the ISO ANSI standard and it's the same thing.

Chamberlin: The technical work was done in ANSI?

Jacobs: Both ISO and ANSI. We would pass papers, change proposals in ANSI and then submit them to the next ISO meeting and papers would sometimes flow in the other direction as well, but there would be progress toward a common standard, so indeed the first few standards that I recall were ANSI ISO standards. Over time what became a frustration for the Americans was that the Europeans and other international organizations used paid consultants, not vendors; they contributed pretty elaborate ideas that were maybe not very practical to implement. Also, as an Oracle partisan it was difficult to compete with the power

that IBM had in the international arena.

Date: There were some sneaky things going on because in principle the representation in ISO is on a national basis, but if the American rep works for IBM and the French rep works for IBM and the British rep works for IBM, you know, difficulties would arise for the non-IBM vendors.

Jacobs: We just didn't have that power. In terms of ANSI and ISO it still is somewhat of a contentious issue of where is the locus and do we work collaboratively or do we finish something and then say, "Take it or leave it?"

Jim Strickland: Let me go back one step here. Maybe you've discussed this but back in the early 1980s IBM was still very proprietary and the idea of a standard wasn't something that we necessarily wanted to do. I'm a little confused on the timing but it seems to me it was about 1981 that I think it was Bob Engle came forward and said, "We really ought to standardize SQL and either he or Phil Shaw or somehow both were involved and they didn't work for me but they got me involved. The idea was that I would get IBM to agree that we ought to sponsor standardization of SQL for a whole good variety of standardization reasons. That was like 1981 or 1982. Not that it happened then, but that's when it started.

Codd: IBM had its own problems internally because you had SQL in DOS and VM. You had SQL in DB2 and you had other versions of SQL on some of their other platforms and I believe at one point there was a meeting to try to standardize SQL within IBM itself that didn't turn out too well.

Bohl: I mentioned it yesterday. There was an ongoing effort between the AS/400 SQL and the DB2 SQL and the Austin PC SQL and then the SQL/DS in Toronto and Endicott. My perception is that the big thing, the toughest thing there wasn't getting agreement on what should be standardized, but it was getting agreement on the priorities of implementation because the marketplaces were different. The users were different on the AS/400 than they were on DB2 and we all had our own other priorities other than the SQL language that we were going to put in a particular release.

Chamberlin: We also had the error code problem.

Jacobs: So it wasn't just Oracle that had that?

User Groups

Michael Mahoney: The way we've been talking it's the industry driving standardization. Did

any user groups begin to form and was there pressure from them for standardization?

Jacobs: There were plenty of users on the committee at various times, government mostly, a couple of big industrial companies, but never a lot of participation. I think that's changed most recently. There are more users involved which is curious because some might say the standard is irrelevant at this point, but mostly it was vendors. I'm not aware of any independent efforts to standardize or to get users together to put pressure on the committees.

Mahoney: There were no users groups formed?

Jacobs: I'm not aware of any input that ever came to the committee from those organizations.

Codd: The IBM users group was formed in 1989.

Mahoney: Okay that's kind of late.

Jacobs: I was actually involved in the Oracle user group as well.

Bohl: But Share and Guide were important to us from a DB2 standpoint.

Mahoney: It's interesting that the user groups formed around the products rather than ideas. There was an Oracle group or Informix or IBM, formed around the vendors as opposed to forming around the tool or around the language. We don't have an SQL users group the way you might have a UNIX users group.

Grad: I don't know of any real exceptions to that. It's almost always vendor specific to my knowledge.

Applications Built on RDBM Systems

Schuster: One of the things that drove some of the standards effort though was that by the early 1990s the applications started to become more and more important, especially the standard business applications and those vendors started putting pressure on people to be more conformant because they wanted to minimize their reporting to different databases. It was a huge issue for Sybase. The biggest strategic mistake Sybase made was not adopting the row level locking and a couple of other issues and we were very late with PeopleSoft, and we never got SAP because of the different transaction model. It was a bigger issue than any other issue.

Jacobs: But in addition to the apps it was the tools vendors. And the analytic tools and

things like that were really putting pressure, not on the committee, but on the vendors.

Schuster: On the vendors to be compliant.

Grad: What's happening though? We have this transition point appearing to take place at the end of the 1980s, beginning of the 1990s where the whole size and range of the applications being built are starting to get to be mainstream stuff, right?

Schuster: Well, people were starting to become the majority adopters. I forget the term. The early adopters rolled their own applications and, in fact, the application vendors in the early days did not focus on database. The exception was Oracle which built their own apps because they didn't have the vendor base. For Oracle that was both a revenue expansion and an issue, I believe, because the mainframe app vendors or AS/400 app vendors weren't interested yet in Oracle. But as the relational database became the corporate standard, the customers said, "Okay, now we got to have applications to run on this because we want to move to this UNIX platform. We want to move to this VAX platform. We want to move to relational database as our standard building blocks." This became a big opportunity for application vendors and they started to put huge pressure on the database vendors to start to look more and more alike.

Jacobs: And at the same time they coded the applications to a least common denominator. They didn't exploit unique features of the product which was endlessly frustrating.

Lindsay: That was really frustrating. You put in something really cool--

Jacobs: And they never used it.

Lindsay: And they'd say, "Well I can't use that. Oracle doesn't have it."

Technology Advances in the 1990s

Grad: Let me now move again looking at the timeframe. I'd like to at least do some coverage of what happened during the 1990s. What were the major technical achievements? I had some questions because this again shows my lack of knowledge. Are data types an issue? I'd like to at least cover that first half of the 1990s while we're here together if we could. Were there major changes in performance other than through the hardware improvements? Were there changes in the kinds of queries?

Jacobs: Parallel query was a big innovation that Informix led and, of course, Teradata did that in the early 1990s.

Grad: What is that one?

Jacobs: Parallel query, being able to take a query statement and break it up so it executed in multiple threads on different subsets of the data. And first it was just query and then it was DML updates as well. Certainly there was a lot of pressure put on Oracle because Teradata had been making a lot of hay about being able to execute SQL across multiple fronts.

Schuster: It was kind of ironic because SQL started in query intensive applications. Then the OLTP stuff came along and networking and gateways. And then it went back to the data warehousing craze and so people were adding a lot of features for that segment of the marketplace.

Jerry Held: Sort of the mid-1990s was when all of a sudden data warehousing became the focus.

Schuster: There were products that were implemented to specifically meet certain needs. Sybase had a parallel product line for heavy decision support applications.

Jacobs: It was a separate code base.

Held: Ingres had three or four products. They had a very high end one.

Jacobs: They had XPS. And Informix had multiple products.

Harrington: Well we had our old product, the original product, and people kept buying it and we didn't update it for years and years. I think it's still available.

Held: One of the big projects was the one that Gary Kelly led up in Portland.

Schuster: The scalability project.

Held: It was probably one of the big attempts to do high performance.

Harrington: And they did a very horrible job on that.

Bohl: Pyramid was one of two hardware companies that had their heyday in the late 1980s, early 1990s and they were really courting the relational database management systems to show that their processor would run a database management system faster.

Jacobs: Multi-processors.

Harrington: Yes, we did a lot of work with them.

Bohl: And scalability was a big thing.

Jacobs: That was OLTP and data warehousing and it was around that time that the benchmarking wars really got into gear.

Schuster: Because the numbers were getting astronomical then.

Jacobs: I remember sitting in the back of a taxi after leaving the Concorde. We launched Oracle version 6 on the Concorde. We took analysts and press for a flight to nowhere for dinner and it was fast. We sat in a taxi with Omri Serlin and he was describing the notion of forming this transaction processing performance council which was to standardize the TP1 benchmark.

Bohl: That's right.

Jacobs: And so they came out with the TPC, transaction processing council benchmark. That had an A and a B that had been kind of very minimal and C became the mainstream transaction processing benchmark and it still today is used to measure transactions. It's interesting if you look at the charts of historical performance. 100 TPM, transactions per minute, was a big number in the early days and now we're at well over a million transactions per minute.

Schuster: We're up to three million I think.

Jacobs: And price/performance is down to like 78 cents per transaction per minute instead of hundreds of dollars in transactions per minute.

Schuster: You have to include the hardware price along with it so a lot of it was not just the software.

Jacobs: A lot of it was disc in particular.

Held: Really the beginning of that whole thing was at Tandem in the 1980s. I don't know if you were around or not, Stu.

Schuster: I was there.

Benchmarking

Held: We had what we felt was the highest performance transaction processing but we couldn't get any comparison point, so Jim Gray along with a few other people went off and did this thing which was anonymously published in Datamation of all places. It was the original banking debit/credit transaction. TP1.

Held: The TP1 benchmark evolved over many, many years.

Schuster: TP1 was in place fairly early. I was at Tandem from 1977 to 1982 and I believe it could have easily been 1979, 1980, somewhere in that range.

Held: There was no organization to enforce compliance with benchmarking. Everybody took that article and interpreted it a little differently and so there were myths of benchmarking. What was wrong with your version of the TP1 and why you couldn't compete?

Schuster: Well there was a whole industry that got created of people that certified a benchmark. It was one of the spin-off industries.

Schuster: The last thing I did at Ingres was to launch Ingres Star in the marketplace, which was the first attempt to sort of come up with an implementation of a distributed database, virtual distributed database effort. Now, they never got the update parts of that done but we marketed it anyway. But the retrieval part of taking a query, breaking it apart from a meta dictionary, a distributed dictionary, and sending it to different computers was actually demonstrated and that was actually launched in 1986. And it included gateways to mainframes so that you could send a query off to a mainframe, a DBMS, and multiple VAX's and UNIX. It was well received.

Held: That's another one of these examples. The query side was probably somewhat well received but there are many examples of sort of dead end paths down this whole history. Tandem, which is off in its proprietary world, had done a lot of these things much earlier, in the very early 1980s; we had not only distributed queries but distributed updates across whole networks of Tandem systems. We made a big deal of it. We went to the customers and they said, "We don't want this. We're not going to let you update a database that spans multiple sites." We went down these paths, we invented this great technology and then some of it just sort of died, died for ten years.

Schuster: Another area was replication technology. A handful of customers got excited about replication.

Bohl: It seemed to me that distributed data was a solution looking for users that cared

about the product. Initially, at Ingres, I had some of my best people working on it but you couldn't justify it from the market requirements.

Schuster: When I left, I was working for Pete Tierney, who ended up at Oracle, and the announcement was so successful, from a press point of view, it was the most amount of press I've ever been involved with in my life. I mean, the press book was inches thick of clippings on this announcement. And they said, "Well, we're going to position the whole company on this," and I said, "That would be a big mistake because I believe it makes great copy but it doesn't sell software."

Bohl: Right. Right on.

Schuster: And, you know, stick to the things that you're really good at. It was the tool set that was the powerful part of Ingres for a long time.

Distributed Databases

Lindsay: Just two topics. One to discuss what happened at IBM Research with respect to distributed database. We had what we called the R Star project. And we did the zero case. That was easy. We did one. That was system R. We found that two was a lot harder. But R Star eventually worked. We did updates. We did query deep composition. It wasn't federated in the sense that what went from one node to another wasn't pure SQL because we needed to do the nested join in with block nested loop. We got a federated product much, much later but we did this and I had two phase commit and the feedback we got from our customer is, "Two phase commit? That's disease spread. Why would I, if I have, you know, three 9s of reliability on this machine, and three 9s of reliability on this machine, you put them together with two phase commit, you've got half of three 9s availability. So, if I get sick, you get to have sympathy sickness." That didn't fly with the customers. Federated stuff was done a little bit better but, even there, they don't like the two phase commit. They don't like that, if your machine goes sick, it holds resources on my machine that can't be released until your machine comes back up. People didn't buy that.

Schuster: In a way, the whole extraction-transformation-load industry (ETL) provides tools for migrating big blocks of data; cleaning and loading it can be thought of as a distributed database. People solved this distributed problem by batch updates to another machine. That was straightforward.

Lindsay: Or replication technology.

Held: That was the one thing that we did, again, at Tandem. We did a remote disaster

recovery feature. This was a distributed thing but it allowed you to have another site that could come up if you had an elephant step on your machine or whatever and that, actually, has caught on over the years.

Lindsay: That's another technology we might want to get into is disaster recovery technologies that were developed in the 1990s. I wanted to come back to Burt's point about data types, just briefly, because that's certainly been a problem in the standards area. From the beginning, it was realized that, gee, Oracle and DB2 don't have the same data types, the numeric types in Oracle are completely different than DB2 and the other products, Ingres, Sybase, Informix. The same is true with date and time. IBM has the dumbest date time scheme you could imagine. It was really bad. <laughter>

Jacobs: I like that. My version of SQL allowed things like February 39th.

Lindsay: The standards people managed to wash that away somehow. I can't imagine how they could have ignored this.

Held: But then there was another whole component of data types which was sort of there early to mid-1990s where, after a big success in relational database over the previous decade, there was this movement to say oh, well, we had hierarchical, network, relational--there must be, obviously, another one.

Lindsay: Let's go back to network.

Object Databases and Data Types

Held: There was this whole bunch of companies that started up to do object databases and probably five, six, seven companies got started in the object space.

Schuster: Every major company had an object. I'm sure you did. We had a big object.

Lindsay: Well, certainly the relational vendors reacted to that by adding, let me call them, abstract data types as a column type -- I call it regress and de-evolution. <laughter> Because, remember, we started off with the hierarchical model, IMS, and then that evolved to the network model, CODASYL, and then to the flat model, relational and we thought, oh, good, things are getting better. But then came object database. That's network, right?

Everyone: Yes.

Lindsay: That was the early 1990s and the beginning of de-evolution. But it's even gotten

worse. Now, we have XML, which is hierarchical. <laughter> And so all the vendors are now, of course, doing something about XML, usually not actually selling a hierarchical model, although some people have gone a bit too far, I think, so that's my story about de-evolution.

Schuster: Well, some things just fit into networks and some things fit into hierarchies.

Held: But, from a company point of view, what happened was that there were these little companies trying to take a new model and there was the mid-1990s, where the universal database wars started and we started with Illustra being acquired by Informix and there was a race to see who could be the first universal database. It was Informix and Oracle and then it was the UDB from IBM.

Grad: What is Illustra technically?

Held: That was another Stonebraker company which was an object relational database. It was extensible data types with the blades that you could plug in.

Schuster: The quintessential example was the geographic data type, so you could now talk about things being within a certain distance of each other. There were whole new sets of operators.

Grad: That's a new data type. Halo is a new data type?

Schuster: Yes, but you could define it...

Held: It was an extensible data type.

Schuster: It was extensible. You could add to it.

Lindsay: They went quite deep in Illustra. They allowed the extension to actually do this new access method.

Schuster: Right.

Lindsay: Put in an R tree and say now I've got a spatial data type. That was actually pretty nice. It never really caught on because it was not a simple matter to define and install a blade.

Held: In the evolution of the industry, what happened right at that point was the small companies, instead of starting up new industry sectors, got subsumed into the bigger players.

So Informix, Oracle, and IBM all said, well, you don't need different databases. We'll put all those features into one and it kind of eliminated all of the object database companies. They all went under.

Schuster: Well, the number of applications was also very limited.

Lindsay: It got a little weird. Gem was a network database where objects actually had pointers. They weren't foreign keys, they were pointers, right?

Jacobs: But the other thing that happened to those object companies is that they didn't have any standards. There was no common language, as there was for relational, and that, I think, hurt them. They tried to standardize ODMG but it never really amounted to anything.

Lindsay: And I think that Don will underline the difficulty in developing query and update languages for non-relational data models, hierarchical in the case of XML.

Relational Model Becomes the Standard

Grad: These new data types, these new structures were coming back in the 1990s and, in effect, the relational model was now the foundation, do I have a correct view?

Bohl: Yes.

Grad: While IMS and some of the other hierarchical products were still selling for other purposes but the market had shifted and now relational was the primary database management system in use.

Lindsay: The relational response to object database was that it was no problem. You can have a tuple in a tuple called an abstract data type. And whether or not that had an inheritance or not, they took no position, but the way that objects referenced other objects was by foreign keys in the object relational, not by direct reference.

Jacobs: The ANSI committee spent a lot of time working on extending SQL with a lot of this object stuff and, ultimately, you know, there is a specification for all of this technology, inheritance, etc.

Grad: But how about XML? Someone mentioned that.

Jacobs: XML came much later. That's like late 1990s.

Lindsay: Someone will be interested in that in 15 years. But it's not my thing.

Grad: It's not old yet. Anyways, the work done by Harry Markowitz on entity-attribute-sets at various times, that was very similar, to my mind, to some of the definitions of object oriented work. Has anybody seen anything on that? Did that tie in with any of the work done here? That's a complete mystery to all you? Chris, in your work, in structuring and so forth and classifying in relational database and other database systems, is there a framework in which all these pieces come together that we've been talking about?

Date: Well, see, I'm prejudiced.

Grad: I know you are. I'm asking you to ignore your last three books and go back before that. <laughter>

Date: I think there is the relational model and there's everything else. And, I tried to say this briefly yesterday when I gave my introductory remarks, if you think abstractly about what a database is, it is a place where you want to put facts. What is a fact? It's the same thing as what the logicians call a proposition that happens to be true and, if you think about databases, those holding propositions, you are inexorably led to the position that it has to be a relational database. That's what relations are. Relations are sets of propositions. And all these other things, like the hierarchies, networks, objects, and, yes, XML and so on, are an anathema to me. They are attempts to surface implementation details and show them through to the user, which is the wrong thing to do. You can do all the stuff that you want to do with hierarchies, XML, whatever in the relational framework. So that's what I'd say of other models. Now, the stuff about entity-relationship and others, these were, well, a mixed bag. They mostly started out to be ways of looking at the world by people who hadn't really taken on board the relational model. The relational model does all this stuff perfectly well.

Schuster: Yes, they reinvented the world.

Entity-Relationship Models

Date: They're reinventing stuff that's already been very well done by other people. But that was particularly true about the so-called entity-relationship model. The first paper on that stuff by Peter Chen describes the entity-relationship model, but not terribly well, in my view. There are a lot of holes in the description but then he finishes off and says let's see how this compares with a relational model. Oh, look, it's the same thing. <laughter> And what I called value sets are really domains and so on. You know, absolutely outrageous, really, when you think about it. <laughter>

Held: So how did this paper get approved?

Date: I don't know.

Lindsay: Well, I think a lot of that stuff actually became useful in the modeling domain; when you try to model a complicated scheme, it begins to look like a network when you put it on a piece of paper.

Codd: One man's entity is another man's attribute. Then you wind up with some really peculiar relationships in data modeling if you take that approach. But they did build products using that approach and I think the fact that the products were built really promoted the approach.

Lindsay: Were they database products or were they modeling products?

Codd: They were modeling products.

Lindsay: Modeling products. They weren't really database...

Codd: They were modeling products that came out with these enormous sheets that showed the relationship between an entity and an attribute and the relationships and all of this and I went to a number of customers, particularly in the 1990s, and what I found was that they were using these modeling tools and they were talking about data administration and database administration and they'd have these charts in a room...

Lindsay: Covering the whole wall.

Codd: Right, all over the walls. When I thought back to the simplicity of Ted's approach, because, apart from the mathematical elegance of it and the fact that it really did solve a problem, what was so wonderful about it was it did it in a very simple type of way. Sure, you needed to understand mathematics to really appreciate what was there but there was an inherent simplicity about it. I was on the CODASYL database task group, and this reminded me of just that because, when you were on the CODASYL database task group, you understood what that whole report was. It was extremely complex and really impossible and so were these charts that were lining the walls.

Lindsay: Well, I'd take a little issue with that because if you've got an application that has 500 tables, each with an average of 50 columns, the only way you can represent that is on the whole wall. I mean, it's just the size -- the magnitude. It's not that it's non-relational. But I feel perfectly comfortable putting an arrow on a line or a line-- not an arrow -- on a piece of paper

between two tables. I'd feel comfortable doing that.

Date: I want to say that I do partly agree with what Bruce said. I have no problem with using informal notions like entity-relationship and so on if it helps you come up with your database structure, your database design. No problem with that. And the complexity Sharon's referring to, it may be inherent in the business you're dealing with. You may have to have, as Bruce said, a whole wall covering your diagram. The problem comes in if you try to make formal distinctions between entities and relationships and attributes. If you build in differences formally, you then have a very rigid structure, which is hard to change later on. In Ted's early papers he even talked about things called entity relationships and relationship relations. But the beauty, the genius, of Ted's work was, they were all relations and they were operated on in the same way by the same operators. So you didn't formalize the distinction. And that was the crucial thing. I agree, you can use these ideas to help you design. That's a separate realm.

Schuster: Just for completeness, there was another segment that evolved in the late 1980s and early 1990s that was successful. It was these relational enterprise document management systems like Documentum and there were a couple of other companies. And they're still around and it's a billion dollar industry and they almost-- I think they all use a relational database to store their metadata and some of the other information but to deal with these large blobs of documents, both for normal documents like we're producing here as well as web pages. This business became very big in web page configuration management. As a standard relational database, none of them became very successful, even with their extensions. They had to be embedded in and then enhanced to deal with the document management functions of version control and componentization.

Grad: The relational model became so fundamental that it basically became encoded within or used within all kinds of other applications, other kinds of capabilities.

Schuster: But for some reason it failed in this area, to be all encompassing itself.

Date: The criticism is a valid criticism but it's not a criticism of the model.

Schuster: I'm not criticizing the model. I'm talking about technology innovations.

Grad: We thank you all. We're cutting off the tape. You may not say anything important afterwards. Thank you very, very much. <laughter>

