A History and Evaluation of System R

Presented by Ankita Sakhuja
Let’s time travel to 1970

• What was the state of Database Systems then?

The leading commercial database of its time was a hierarchical one: IBM’s IMS

| Data was linked in a tree-like structure. It’s still a success today | ✓ |
| Laborious to maintain relationships between data | ✗ |
| Locations of data had to be known | ✗ |
| Physical Links had to be maintained and making changes to the database were daunting | ✗ |
| Queries were crude | ✗ |
• Codd's approach was rather different, favoring a declarative model.
• This meant the programmer "declared" relationships and the computer would be expected to implement them in bits and bytes.
In 1973 Big Blue decided to do something about this, and consolidated its database research in San Jose, California.

IBM gave its research staff beautiful buildings in serene locations - San Jose.

Codd had joined IBM’s research labs in 1970, and the move brought him into contact to some clever engineers.
About this Research Paper

What

- Prototype System R designed by IBM (Started from 1974 – 1979)
- Experimental DB system constructed for transition of Relational Model from theory to reality

Where & When

- IBM's Research Lab at San Jose, California
- Communications of the ACM (Association for Computing Machinery), October 1981
- Lots of PhD researchers out of which 16 researchers published this paper

Why

- Data Independence
- User Productivity
Key points

1. Three Principal phases of the system R project
2. Lessons learned from system R about design of database systems in general
3. Our Views regarding contributions of System R to the present era Relational DB system
SYSTEM R CASE STUDY IN A NUTSHELL

Phase 0: 1974-1975
- Initial Single User Prototype, Try out ideas and find issues

Phase 1: 1976-1977
- Throw away version 1.0 once prototype is ready
- Full-function Multi-user prototype

Phase II: 1978-1979
- Evaluation and Feedback, Lots of good lessons learned

Evolved into SQL/Data System, a RDMS product offered by IBM in the DOS/VSE operating system
System Modules Identified

• Log/Recovery System
• Lock Manager
• Data Storage
• View Management
• Access Methods
• Query Optimizer
Basic Design Decisions
- System Catalog
- XRM as Access method

Challenges
- Optimizer Algorithm

Result
- Usability of SQL

Lessons Learned
- Optimizer Cost calculation
- Need of "Join" in SQL
- How can the complexity of Optimizer be reduced
End of Phase 0 (1974 – 1975)

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Phase 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Method</td>
<td>XRM</td>
</tr>
<tr>
<td>Concurrency</td>
<td>Single User</td>
</tr>
<tr>
<td>Lock Method</td>
<td>N/A</td>
</tr>
<tr>
<td>Recovery</td>
<td>N/A</td>
</tr>
<tr>
<td>Query Complexity</td>
<td>Supports Subquery but not “Join”</td>
</tr>
<tr>
<td>Interactive level</td>
<td>Standalone Query Interface</td>
</tr>
<tr>
<td>Security</td>
<td>N/A</td>
</tr>
<tr>
<td>Compilation Approach</td>
<td>N/A</td>
</tr>
</tbody>
</table>
- **Architecture**: RSS, RDS
- Designed to support multiple concurrent users
- Locking sub-system
- Authorization sub-system
- Recovery Sub-system
- Compilation Approach
Compilation Approach

- Inspired by R. Lorie’s observation in early 1976
- SQL statements are compiled into efficient machine language routines and further packed into access module
- When program goes into execution, access module is invoked to the RSS
RSS Access Paths

- RSS provides indexes.

- Indexes are maintained automatically in the event of updates to the database.

- RSS access path includes three different kind of scans:
  - Index Scans
  - Relation Scans
  - Link Scans

- "Search Arguments"
Important Sub-systems which were introduced:

- **Views and Authorization**
  - Power and Flexibility
  - Grant and Revoke

- **Recovery Sub-system**
  - Disk Failure
  - System Failure
  - Transaction Failure

- **Locking Sub-system**
  - Intention Lock
  - Exclusive Lock
## End of Phase 1 ( 1976 – 1977 )

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Phase 0</th>
<th>Phase 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Method</td>
<td>XRM</td>
<td>RSS, RDS</td>
</tr>
<tr>
<td>Concurrency</td>
<td>Single User</td>
<td>Multiple User</td>
</tr>
<tr>
<td>Lock Method</td>
<td>N/A</td>
<td>Intention Locks, Exclusive Locks</td>
</tr>
<tr>
<td>Recovery</td>
<td>N/A</td>
<td>w.r.t. Disk, System, Transaction failures</td>
</tr>
<tr>
<td>Query Complexity</td>
<td>Supports Subquery but not “Join”</td>
<td>Joins were introduced</td>
</tr>
<tr>
<td>Host- Capability</td>
<td>Standalone Query Interface</td>
<td>Supported PL/I, Cobol, Standalone QI (Called UFI)</td>
</tr>
<tr>
<td>Security</td>
<td>N/A</td>
<td>Views were introduced</td>
</tr>
<tr>
<td>Compilation Approach</td>
<td>N/A</td>
<td>Yes</td>
</tr>
</tbody>
</table>

• Evaluation phase consisted of two parts:
  1. Experiments performed on the system in the Lab
  2. Use of R system on IBM’s several internal sites and at 3 selected customer sites.

• General User Comments:
  1. Easy installation
  2. High level user language
  3. Ability to reconfigure the DB
  4. SQL was identified to be simple, user-friendly and data independent.
  5. SQL code was platform independent (w.r.t. adhoc query, application programs)
  6. Compilation approach was applauded for it’s approach to make the compilation easy
Suggestions

- Need of functions like “Exists”, “Like” operators. Which we currently hold now in RDBMS systems
- Need for “Outer Joins”. Exist with us now
- Authorization could be augmented to the group instead of users. Access is now provided to groups
- Creating new table out of existing table. We have functionality of %InsertSelect now
- Shadow pages concept introduced in Recovery sub-systems could have an alternative of simply keeping the logs of all the updates.

*These suggestions with SQL language were later presented as a paper published in 1980 as a user-experience with SQL data sublanguage

Conclusions

- Cost was defined as weighted sum of number of page fetches and RSS calls
- Each user could be provided a three-level lock system as part of Locking sub-system.
## End of Phase 2 (1978 – 1980)

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Phase 0</th>
<th>Phase 1</th>
<th>Phase 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Method</td>
<td>XRM</td>
<td>RSS, RDS</td>
<td>RSS, RDS</td>
</tr>
<tr>
<td>Concurrency</td>
<td>Single User</td>
<td>Multiple User</td>
<td>Multiple User</td>
</tr>
<tr>
<td>Lock Method</td>
<td>N/A</td>
<td>Intention Locks, Exclusive Locks</td>
<td>Three Level Locking</td>
</tr>
<tr>
<td>Recovery</td>
<td>N/A</td>
<td>w.r.t. Disk, System, Transaction failures</td>
<td>Shadow Page and Log Mechanism</td>
</tr>
<tr>
<td>Query Complexity</td>
<td>Supports Subquery but not “Join”</td>
<td>Joins were introduced</td>
<td>Subqueries, Joins, Outer Join need was proposed</td>
</tr>
<tr>
<td>Host- Capability</td>
<td>Standalone Query Interface</td>
<td>Supported PL/I, Cobol, Standalone QI (Called UFI)</td>
<td>Supported PL/I, Cobol, Standalone QI (Called UFI)</td>
</tr>
<tr>
<td>Security</td>
<td>N/A</td>
<td>Views were introduced</td>
<td>Need of Groups was proposed</td>
</tr>
<tr>
<td>Compilation Approach</td>
<td>N/A</td>
<td>Introduced</td>
<td>Was a big hit in System R</td>
</tr>
</tbody>
</table>
Contributions of System R- Our Views

- Bringing Theory to practice
- High Level Query Language was introduced
- System Research in “Action”
  - Macro: Design a complete system architecture
  - Micro: Identify key problems and provide solutions
Thankyou !