



Extensible Query Processing in Starburst

Original Slides

Presentation: Kati

Discussion: Andrew

Modified by: Rachel Pottinger



Outline

- Motivation
- Solution: Extensible DBMS
- Language Processing
- Query Graph Model
- Query rewrite
- Summary



Motivation

- DBMSs inability to support other applications than administrative ones
- No sufficient support for the functions and data types needed by the engineering's applications
 - additional functions and data types needed



Starburst Project

- Extensibility
 - Language extensions
 - Internal processing extensions
 - Data management extensions
- Worth noting the time when it came up
 - Object-oriented phase



Two major components

- Corona: the query language processor
- Core: data manager



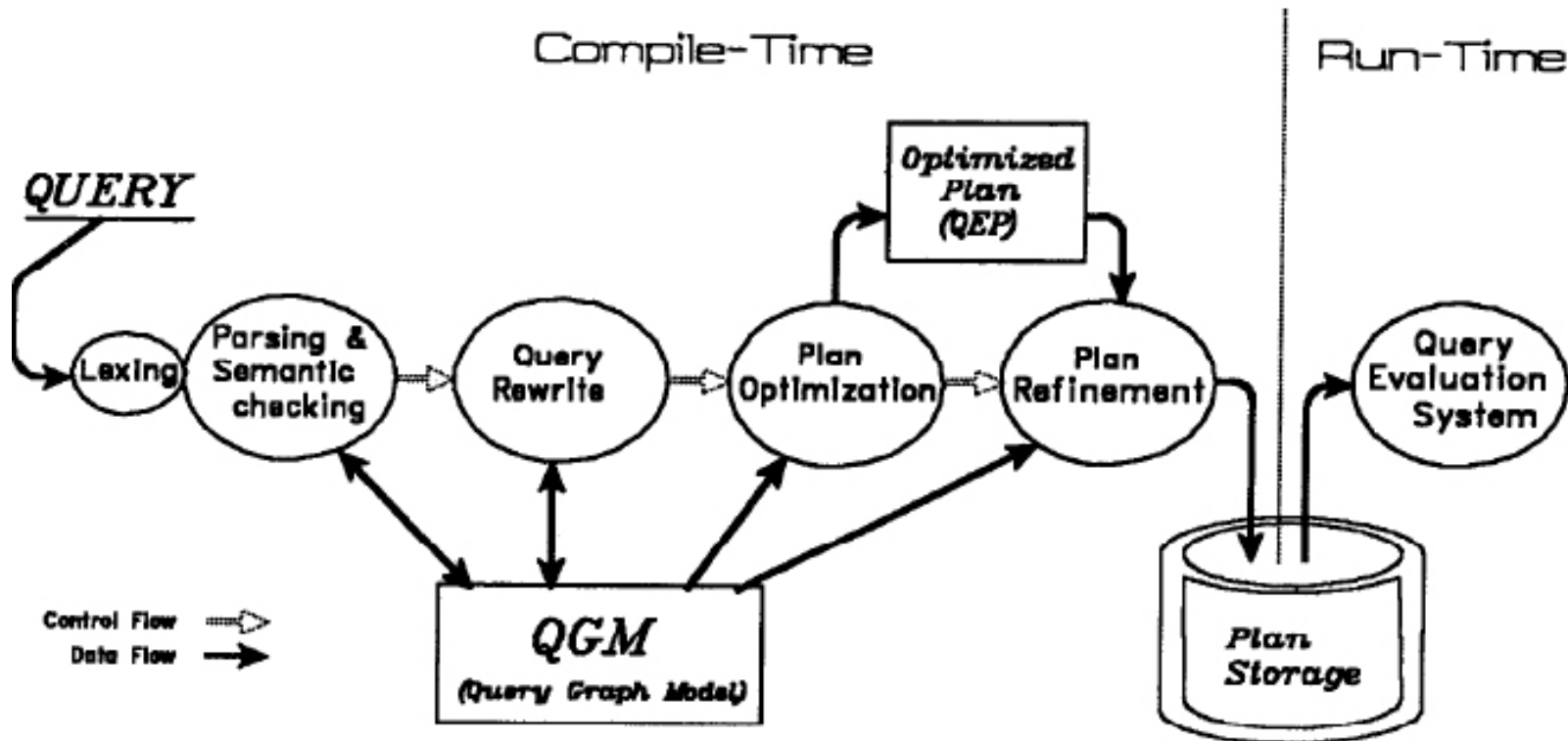
Starburst's Language: Hydrogen

- Based on SQL
- Orthogonal
- Extensible
- Table expression
- Table function

→ very complex queries possible

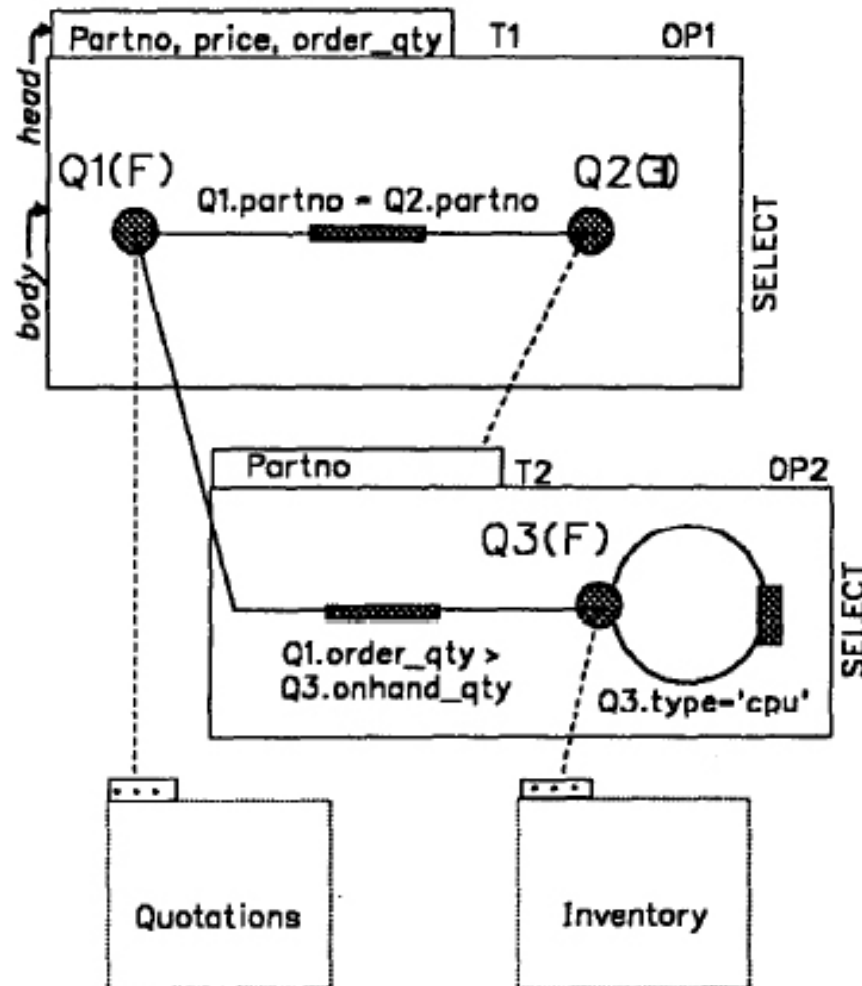
Language processing

- Two stages: compilation and execution



Query Graph Model

- Vertices
- Edges
- boxes





Discussion (pair & share)

- Jason: The ability of DBCs to "influence" internal processing (section 4) seems counter to the declarative approach of query optimizers while the QGM still makes cost estimates to choose cheaper plans (section 6). What would be the advantages/disadvantages of allowing a query optimizer to be influenced from the user level on how it makes decisions?

- From a user point of view, still very declarative
- Could allow the user to specify the techniques
- People who know nothing vs. know everything - looking at the know everythings.
- Better to have things that are related to the machine
- See that this is the way that people are creating queries, optimize for that, abstraction
- the ability to add extensions affects stability
- if it is optional, they don't *have* do it
- having domain knowledge might be a benefit.



Query Rewrite

- A form of optimization and a big challenge
- New transformations required
- Rule-based approach
 - Creation of new rule system
 - Greater scope of optimization and improved execution plan



Rule-based Approach

- Rule language is C
- Two parts: condition and action, each written a C function
- Consistency
- Rule classes → Modularization



Rules- three classes

- Predicate migration
- Projection push-down
- Operation merging



No(!?) cost-based query-rewrite

- All alternatives are generated
- At the plan level cost-based
- BUT interaction desired
- SINCE number of alternatives grows tremendously



Cost-based optimization

- Plan generation
- Plan costing
- Search strategy
- Designed to be orthogonal



Plan generator

- Strategy alternative rules (STARs)
- A general-purpose STAR evaluator
- A search strategy that chooses the next STAR to evaluate
- An array of STARs



Summary

- Starburst: extensible DBMS
- Extensions to the language, the language processing and the data manager
- Table expressions allow orthogonality
- Orthogonality & Extensibility → complex queries possible



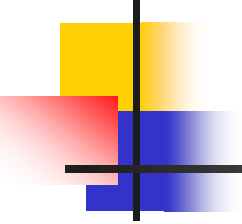
Summary

- Query internally a QGM
- QGM simplifies the DBC's task, give him a great deal of flexibility and power
- Rule-based query rewrite
- Grammar-like rules to generate plans



Discussion (group)

- Sepher: Under what business models does it make sense to make one particular DBMS extensible like that? What skills would a Database Customizer (DBC) have to have? Who would they be?

- 
-
- System assumes that you have such a person, but then what if that person leaves? (maintainability)
 - Salesforce has a special position to customize the ERM. Whether they'll have regular work is problematic
 - People have it in their resumes. But not very general.
 - Very applied skill set. Need domain knowledge. But consultants could help, especially for an industry
 - Education is important. Especially for people in other domains.

