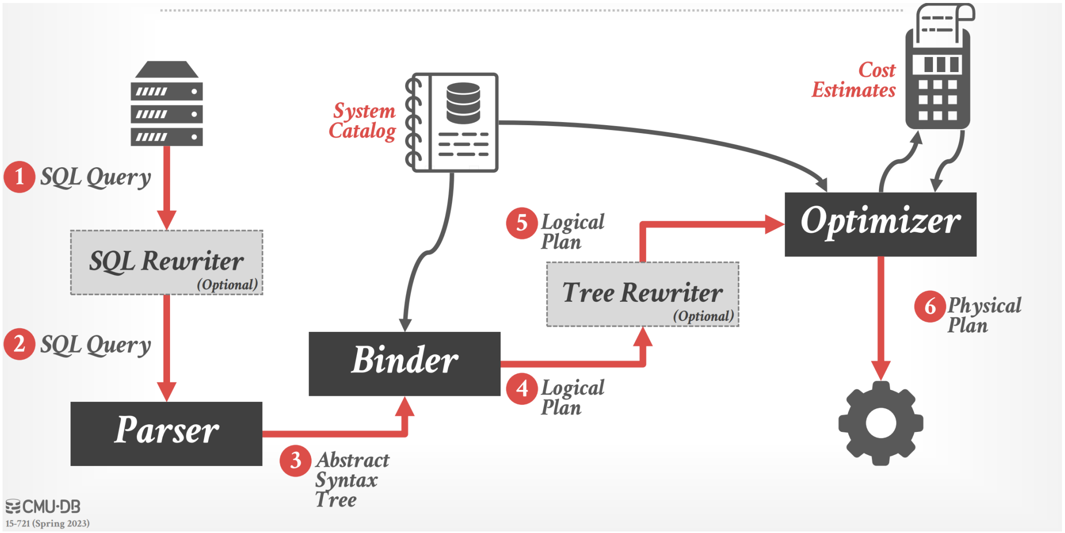
Query Optimisation

PRESENTED BY: RUT
DISCUSSION LEAD: WILSON

* Some slides borrowed from Rachel's Presentation

Architecture of a Query Optimiser



Source: https://15721.courses.cs.cmu.edu/spring2023/slides/16-optimizer1.pdf

What is Logical/Physical Plan?

Logical Plan

- Action to carry out
- E.g. JOIN

Physical Plan

- Algorithm/Implementation to use to perform action
- E.g. Merge-Join, Hash-Join

Starburst: Motivation

- DBMS were unable to support non-administrative (e.g. engineering or science-related) applications
    - Needed extensions to data types
- Needed extensibility to support arbitrary applications
- Note: Publication time when emphasis on OODBMS

Starburst: Features and Components

Features (supports extensions for):

- Languages (e.g. Data Types)
- Data Management (e.g. Access/Storage Methods)
- Internal Processing

Components:

- Corona: Query Language Processor
- Core: Data Manager (similar to RDS/RSS in System R)

Starburst: Query Language - Hydrogen

- Similar to SQL but more orthogonal
- Introduces Table Expressions
- Extensions of Functions:
  - Scalar Functions (e.g. Area of a rectangle, Mean/Stdev)
  - Set Predicate Functions (e.g. “Majority of”)
  - Table Functions (e.g. SAMPLE(table, int))

Discussion (4 people)

- Starburst envision having a small group of more knowledgeable people to work on it instead of general programmers.
- Do you think this is a better direction to take in terms of extensibility?
- Can you think of other examples of this in open-source projects?
- What are some good/bad things you see with this model. Why or why not do you think it will work here.

Starburst: Language Processing

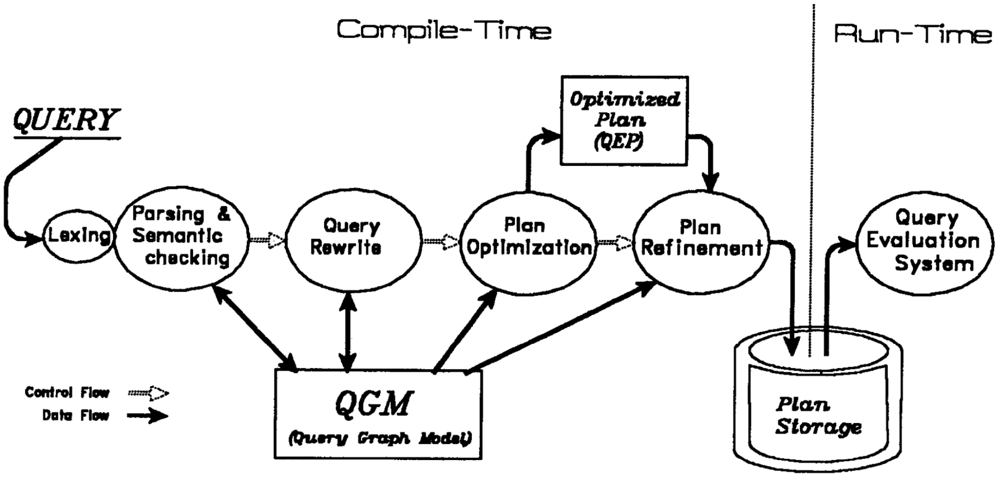
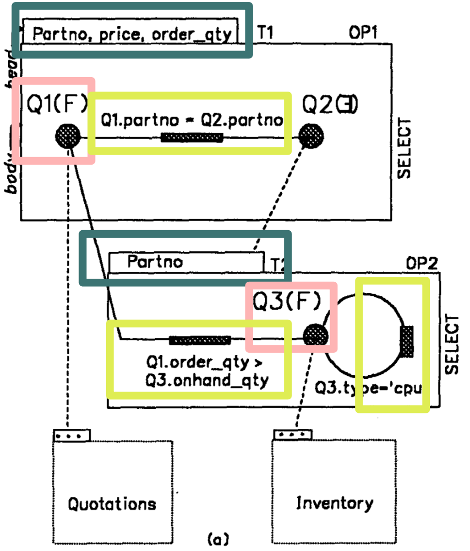


Figure 1: Phases of Query Processing

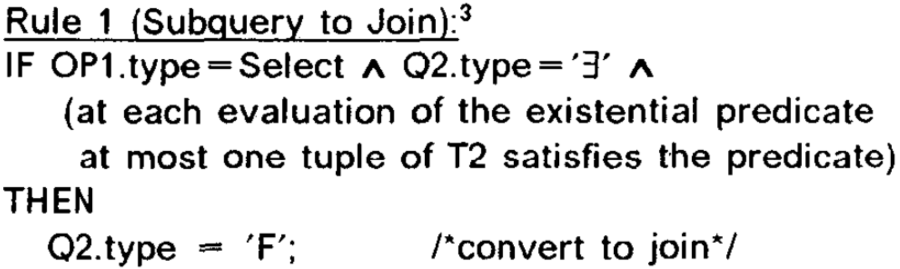
Starburst: Example of The Query Graph Model

```
SELECT partno, price, order_qty
  FROM quotations Q1
  WHERE Q1.partno IN
  (SELECT partno
    FROM inventory Q3
     WHERE Q3.onhand_qty < Q1.order_qty
           AND Q3.type = 'cpu')
```



Starburst: Rewriting the Query

- New Rule System
- Language: C
- Rule = Condition + Action
- Transformation: QGM -> QGM



<u>Rule 1 (Subquery to Join):</u>[3]

```
IF OP1.type = Select ∧ Q2.type = '∃' ∧
    (at each evaluation of the existential predicate
     at most one tuple of T2 satisfies the predicate)
THEN
   Q2.type = 'F';            /*convert to join*/
```

Starburst: Rewriting the Query

- Pre-defined query rewrite classes
  - **Predicate Migration**
  - **Projection Push-down**
  - **Operation Merging**

Discussion (2 people)

- We can see that the query rewrite rules are quite verbose and exhaustive.
- What do you think of this design choice? Do you have any recommendations instead?

Starburst: Optimised Query Graph Model

```
SELECT partno, price, order_qty
 FROM quotations Q1
 WHERE Q1.partno IN
   (SELECT partno
     FROM inventory Q3
     WHERE Q3.onhand_qty < Q1.order_qty
           AND Q3.type = 'cpu')
```

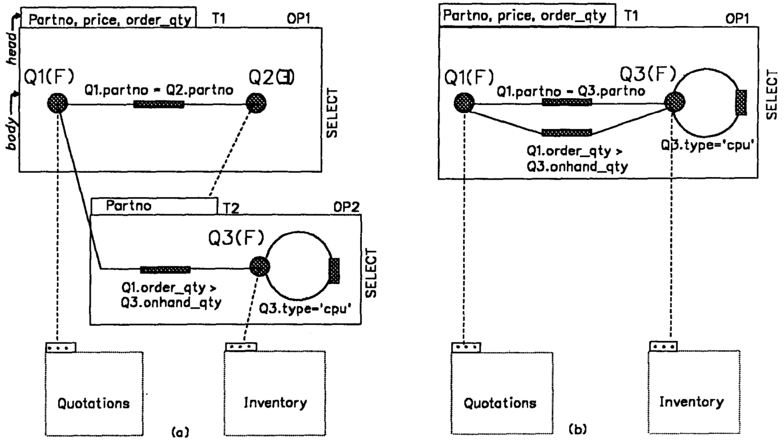


Figure 2: (a) QGM of a query (b) Same QGM after rewrite rules applied

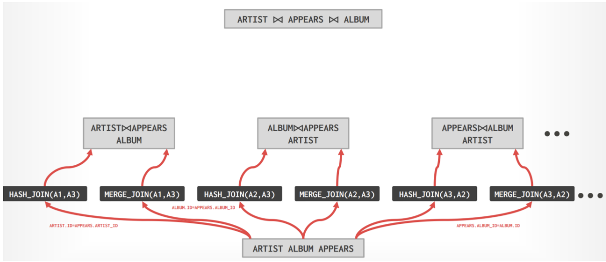
Starburst: Query Execution Plans

- **STARs**: Strategy Alternative Rules (Grammar for Execution Plans)
- **LOLEPOP**: Terminal operators/functions/implementations (e.g. SCAN, SORT, JOIN)
- **Query Execution Plan**: Nested invocations of LOLEPOPs

Starburst: Example QEP – Bottom-Up Optimisation

Retrieve names of artists that are in Andy’s mix ordered by artist ID

```
SELECT ARTIST.NAME
  FROM ARTIST, APPEARS, ALBUM
  WHERE ARTIST.ID=APPEARS.ARTIST_ID
    AND APPEARS.ALBUM_ID=ALBUM.ID
    AND ALBUM.NAME=“Andy's Drill Remix”
    ORDER BY ARTIST.ID
```

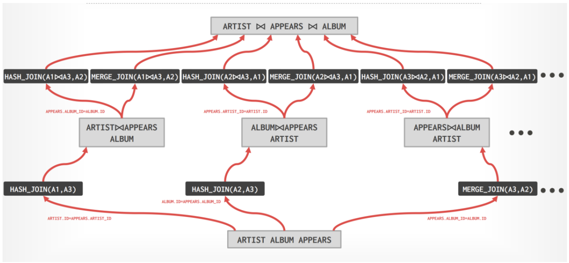


Source: https://15721.courses.cs.cmu.edu/spring2023/slides/16-optimizer1.pdf

Starburst: Example QEP – Bottom-Up Optimisation

Retrieve names of artists that are in Andy’s mix ordered by artist ID

```
SELECT ARTIST.NAME
  FROM ARTIST, APPEARS, ALBUM
  WHERE ARTIST.ID=APPEARS.ARTIST_ID
    AND APPEARS.ALBUM_ID=ALBUM.ID
    AND ALBUM.NAME=“Andy's Drill Remix”
    ORDER BY ARTIST.ID
```

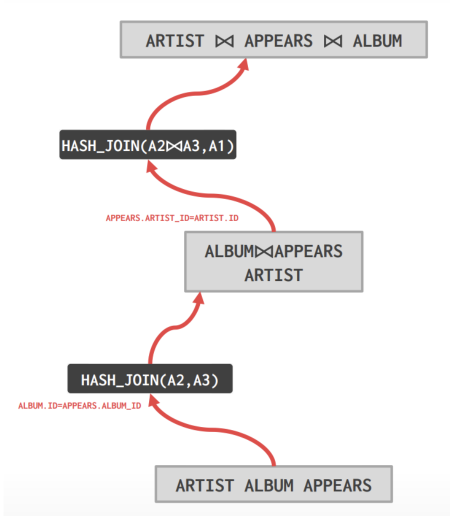


Source: https://15721.courses.cs.cmu.edu/spring2023/slides/16-optimizer1.pdf

Starburst: Example QEP – Bottom-Up Optimisation

Retrieve names of artists that are in Andy's mix ordered by artist ID

```
SELECT ARTIST.NAME
  FROM ARTIST, APPEARS, ALBUM
  WHERE ARTIST.ID=APPEARS.ARTIST_ID
    AND APPEARS.ALBUM_ID=ALBUM.ID
    AND ALBUM.NAME="Andy's Drill Remix"
    ORDER BY ARTIST.ID
```



Source: https://15721.courses.cs.cmu.edu/spring2023/slides/16-optimizer1.pdf

Starburst: Accounting for Costs

- No need to evaluate all Query Execution Plans (QEPs)
- Table has a summary, used by the cost model
- Add cost when moving up the QEP graph
- Exhaustive searching?

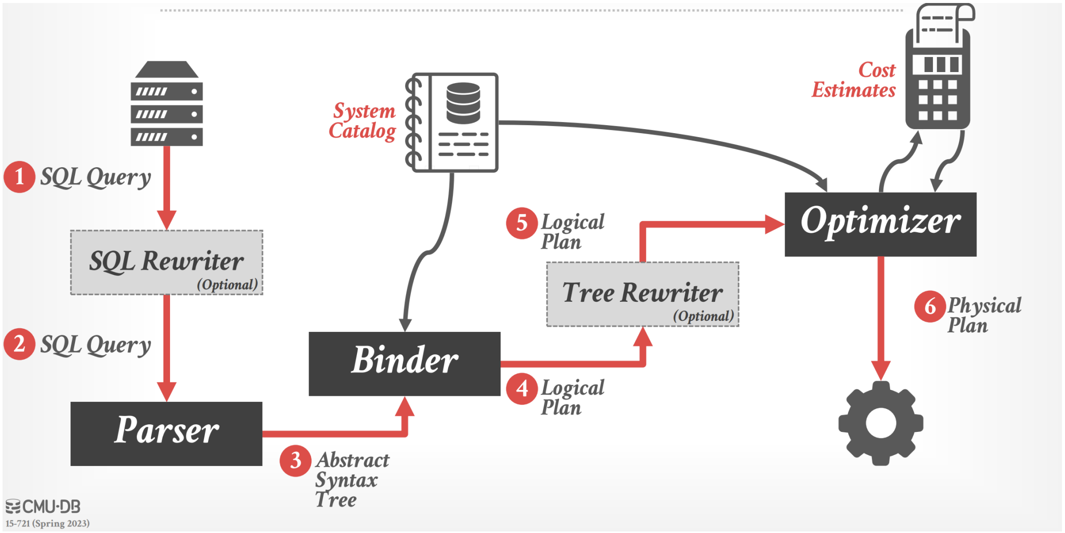
Discussion (2 people)

- Who do you think would be a DBC (what kind of education, skills)? What are some challenges with needing such specialized skill set?

Volcano: Motivation

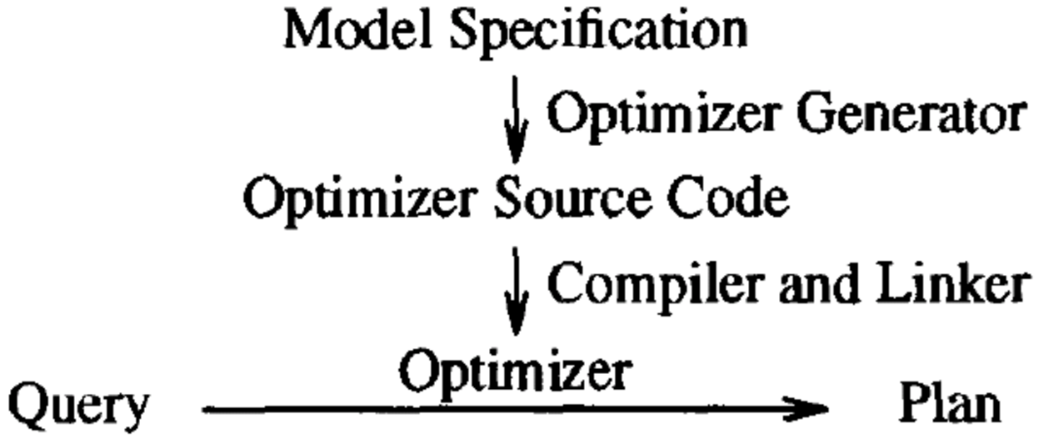
- General Purpose DBMS Optimiser Generator
- Application-specific optimisations:
  - Scientific and OO-DBMS the goal (but can generalise)

Architecture of a Query Optimiser



Source: https://15721.courses.cs.cmu.edu/spring2023/slides/16-optimizer1.pdf

Volcano: Overview of the Optimiser Generator



Volcano: Design Principles

1. Query processing based on algebra, even in OODBMS
2. Rules are expressive for extending/creating optimisers
3. Query Rewrite: Equivalent Algebraic expressions
4. Rules are compiled (not interpreted)
5. Directed Dynamic Programming to search optimal plan

Volcano: Inputs and Outputs

Input:

- Set of Logical Operators
- Algebraic Transformation Rules
- Algorithms/Enforcers
- Implementation Rules (Operators to Algorithms)
- Cost Functions
- Applicability Function for each algorithm/enforcer

Output: Generated Optimiser

Discussion (2 people, optional)

- Notice that the cost function can be customized to what we want. Usually estimated number of I/O is used, but can you think of other cost functions that might be helpful?

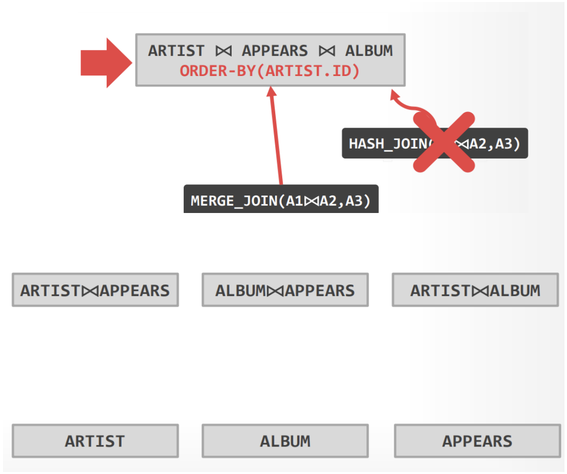
Volcano: Search Engine

- Same search engine for all optimisers
- Directed Dynamic Programming (backward chaining)
- Find cost of useful moves only

Volcano: Example QEP – Top-down Optimisation

Retrieve names of artists that are in Andy's mix ordered by artist ID

```
SELECT ARTIST.NAME
  FROM ARTIST, APPEARS, ALBUM
  WHERE ARTIST.ID=APPEARS.ARTIST_ID
    AND APPEARS.ALBUM_ID=ALBUM.ID
    AND ALBUM.NAME="Andy's Drill Remix"
    ORDER BY ARTIST.ID
```

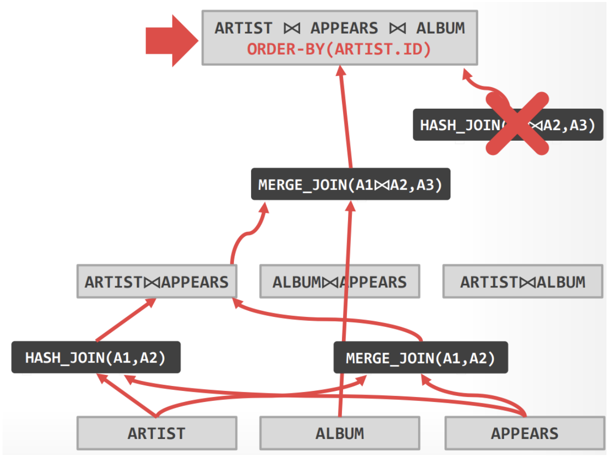


Source: https://15721.courses.cs.cmu.edu/spring2023/slides/16-optimizer1.pdf

Volcano: Example QEP – Top-down Optimisation

Retrieve names of artists that are in Andy's mix ordered by artist ID

```
SELECT ARTIST.NAME
  FROM ARTIST, APPEARS, ALBUM
  WHERE ARTIST.ID=APPEARS.ARTIST_ID
    AND APPEARS.ALBUM_ID=ALBUM.ID
    AND ALBUM.NAME="Andy's Drill Remix"
    ORDER BY ARTIST.ID
```

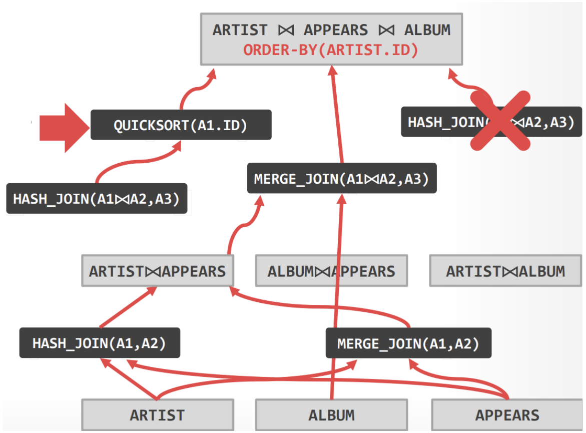


Source: https://15721.courses.cs.cmu.edu/spring2023/slides/16-optimizer1.pdf

Volcano: Example QEP – Top-down Optimisation

Retrieve names of artists that are in Andy's mix ordered by artist ID

```
SELECT ARTIST.NAME
  FROM ARTIST, APPEARS, ALBUM
  WHERE ARTIST.ID=APPEARS.ARTIST_ID
    AND APPEARS.ALBUM_ID=ALBUM.ID
    AND ALBUM.NAME="Andy's Drill Remix"
    ORDER BY ARTIST.ID
```

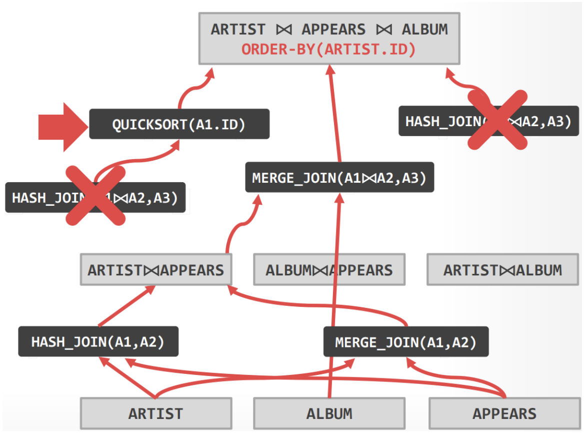


Source: https://15721.courses.cs.cmu.edu/spring2023/slides/16-optimizer1.pdf

Volcano: Example QEP – Top-down Optimisation

Retrieve names of artists that are in Andy's mix ordered by artist ID

```
SELECT ARTIST.NAME
  FROM ARTIST, APPEARS, ALBUM
  WHERE ARTIST.ID=APPEARS.ARTIST_ID
    AND APPEARS.ALBUM_ID=ALBUM.ID
    AND ALBUM.NAME="Andy's Drill Remix"
    ORDER BY ARTIST.ID
```



Source: https://15721.courses.cs.cmu.edu/spring2023/slides/16-optimizer1.pdf

Discussion (4 people)

- Do you think by today's standard this paper would be published? At what point is writing code research and what point is it considered implementing a product? What is the distinction and has the line blurred/became clearer over the years?
- (An interesting point someone brough up is the paper reveals author identity)

Volcano: Comparison to Startburst

Starburst	Volcano
• Bottom-up optimization	• Top-down Optimisation
• Hierarchy of LOLEPOPS (Difficult to understand)	• Algebraic equivalence (Easier to understand)
• Query rewrites based on heuristics only (no cost)	• Query rewrite considers cost of execution

Volcano: Evaluation of the system

- Comparison with Exodus
- Used similar data model descriptions for both
- Exodus took a lot of memory
- Exodus does not exploit physical properties for QEP

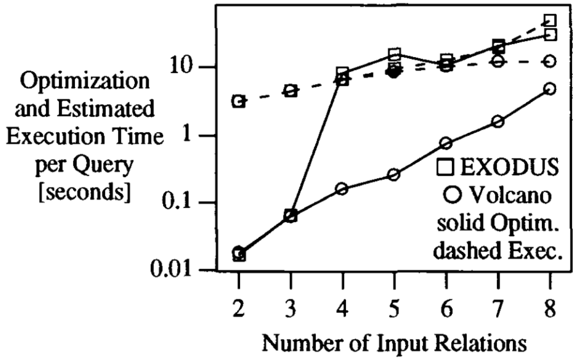


Figure 4. Exhaustive Optimization Performance.

Discussion (2 people)

- The benchmark the paper use is quite small. Do you think it's representative of the performance? How would you test it instead to show the effectiveness?

Summary

- Tools for extending DBMS
- Separation of logical and physical properties
- Heuristics vs Cost-based optimisations
- Consider physical properties for optimisation, too