Principles of Distributed and Parallel Database Systems: Part 2

Distributed Query Processing

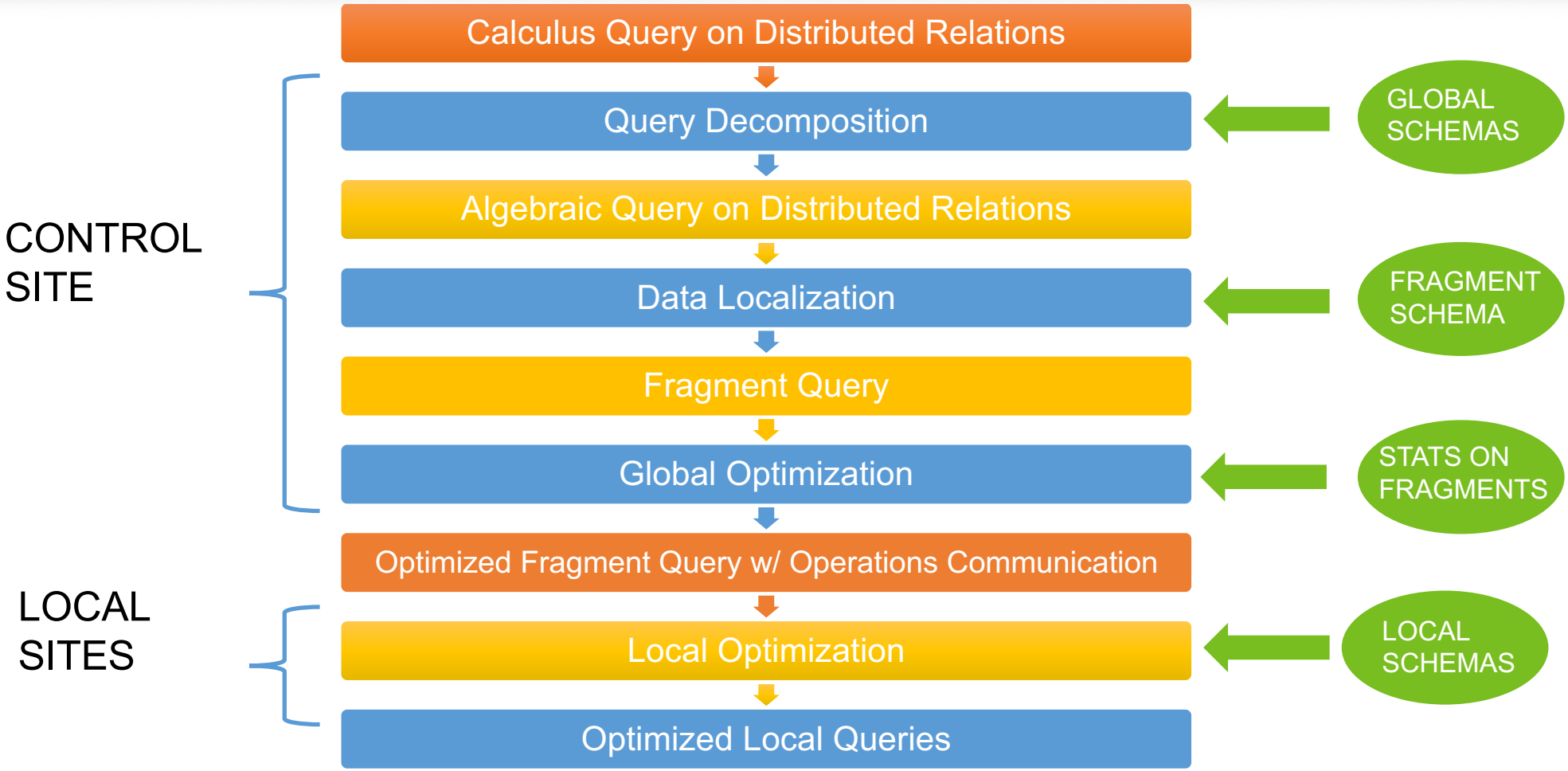
Ira A. Fulton Schools of Engineering
Arizona State University

Objectives

Objective

Illustrate how concurrency control is achieved in distributed databases

Distributed Query Processing



Query Processing

- Decomposition
- Localization
- Optimization

Decomposition

Same as in centralized system

- **Normalization**
- **Eliminating redundancy**
- **Algebraic rewriting**

Normalization

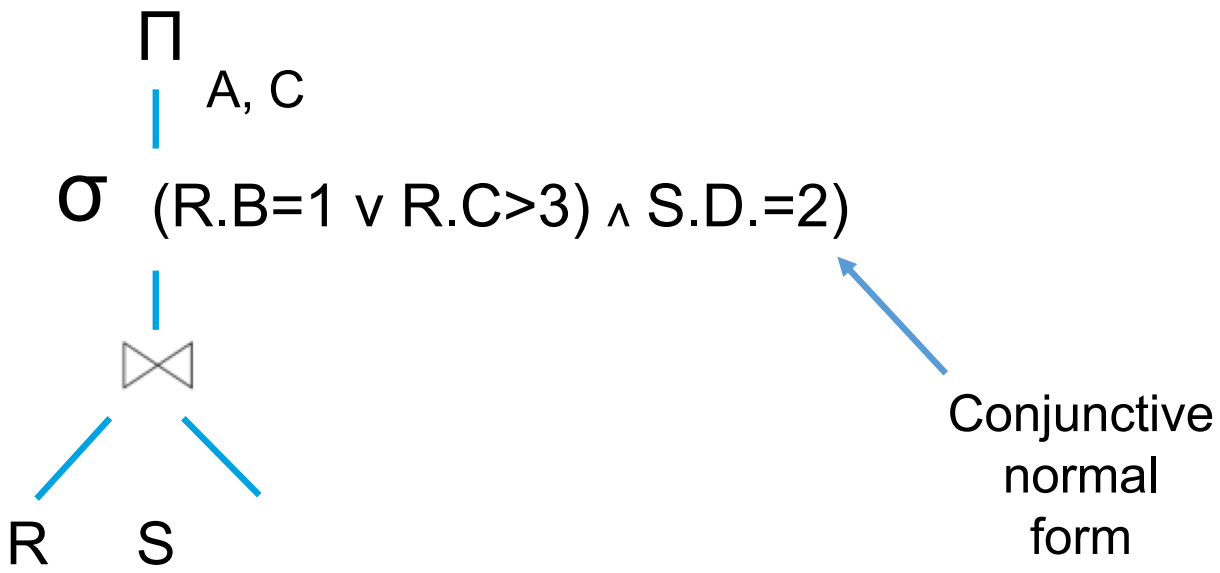
- Convert from general language to a “standard” form (e.g., Relational Algebra)

Example

Select A,C

From R,S

Where (R.B=1 and S.D=2) or (R.C>3 and S.D.=2)



Also: Detect invalid expressions

E.g.: Select * from R where R.A =3

- R does not have “A” attribute

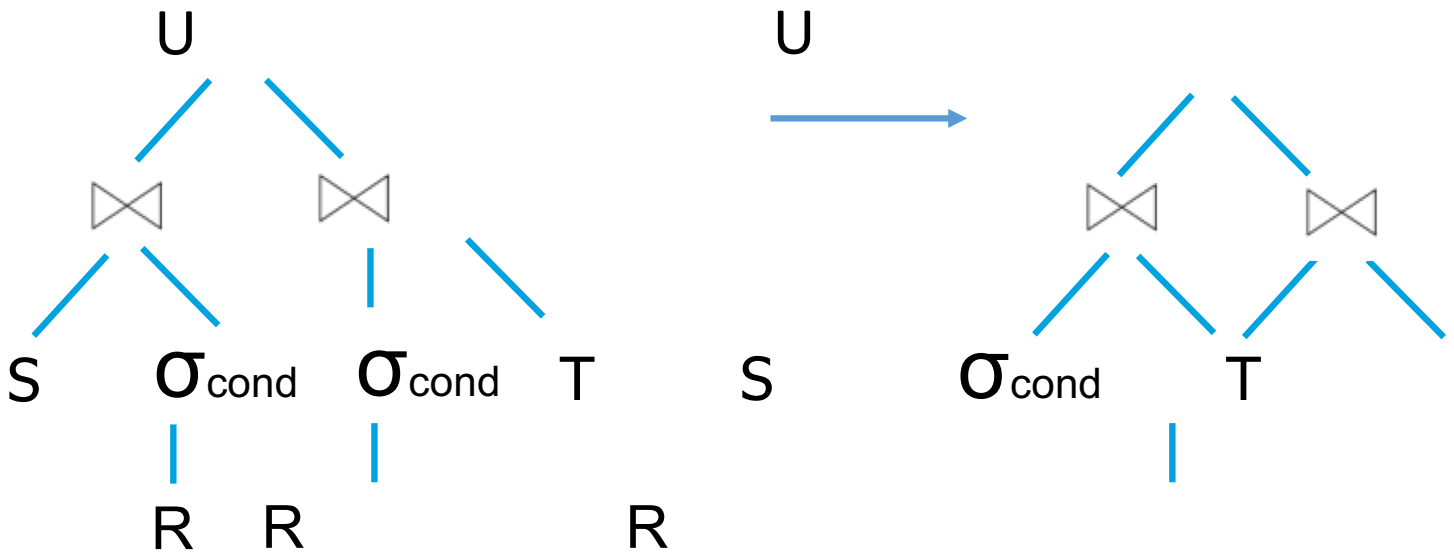
Eliminate redundancy

E.g. in conditions:

$(S.A=1) \wedge (S.A>5) \Rightarrow$ False

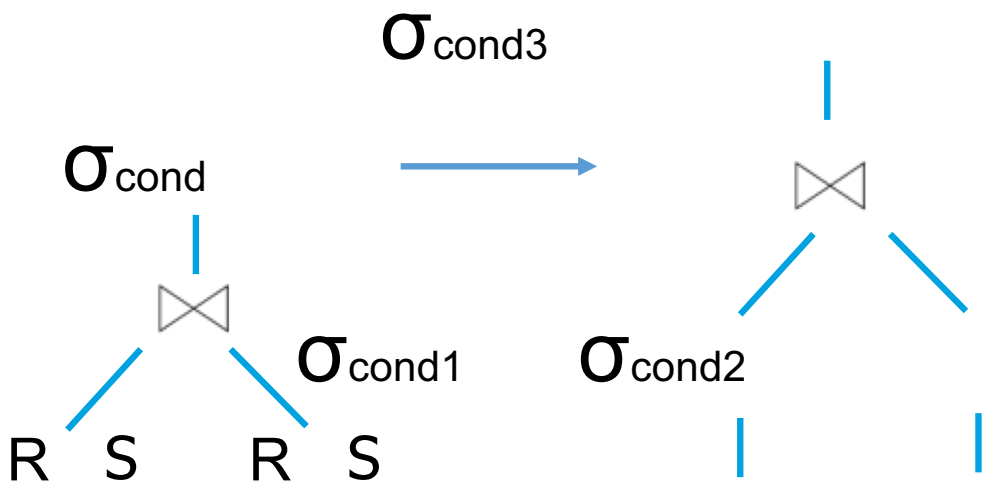
$(S.A<10) \wedge (S.A<5) \Rightarrow S.A<5$

E.g.: Common sub-expressions



Algebraic rewriting

E.g.: Push conditions down



Algebraic rewriting

After decomposition:

- One or more algebraic query trees on relations

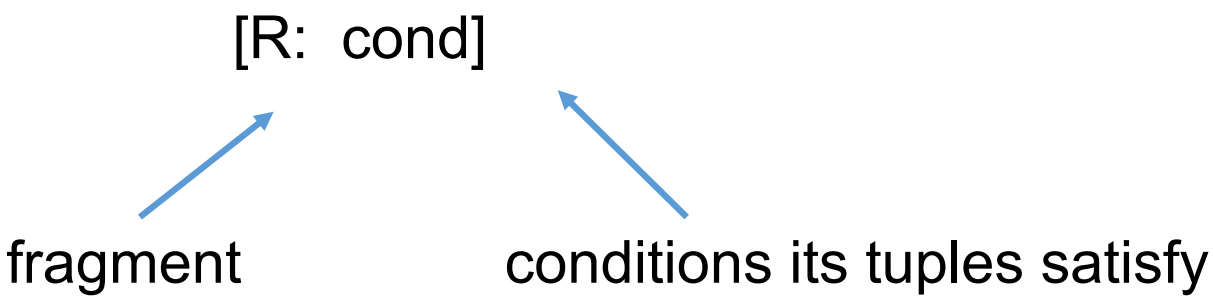
Localization:

- Replace relations by corresponding fragments

Localization steps

- Start with query
- Replace relations by fragments
- Push ∪: up ()
  - π,σ : down
- Simplify – eliminate unnecessary operations

Notation for fragment



Example A

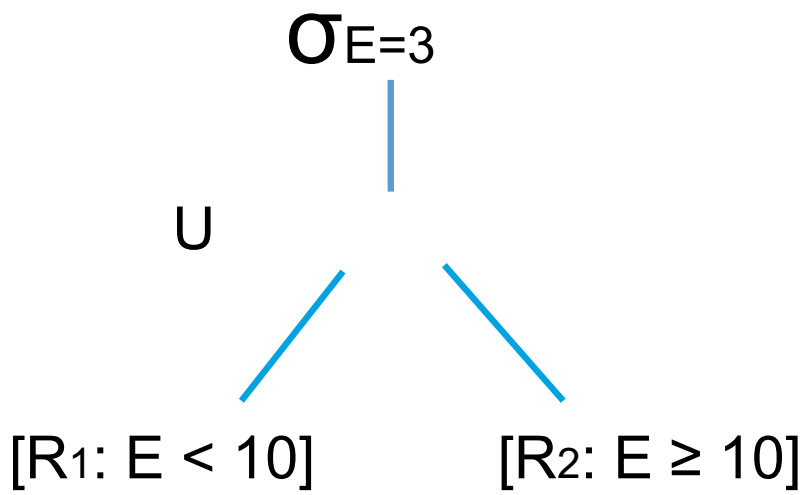
$\sigma_{E=3}$

|

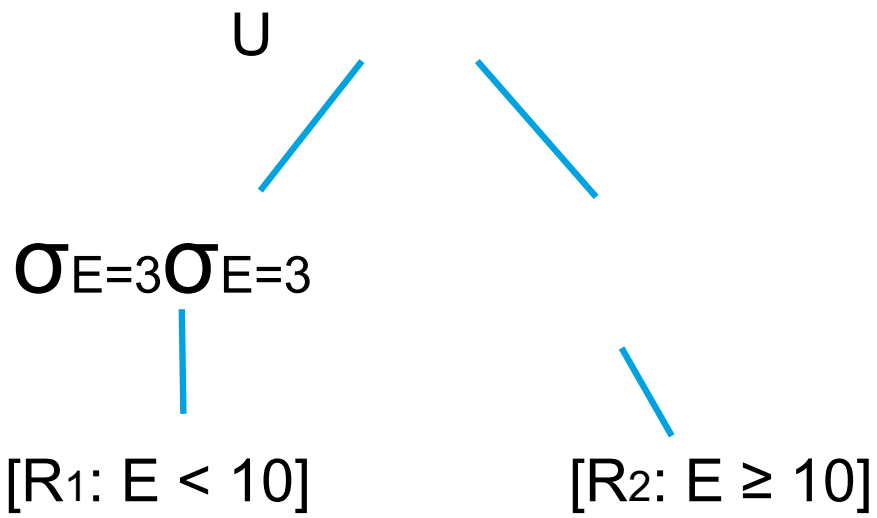
R

What if we have two Fragments
[R_1: E < 10] [R_2: E ≥ 10]

Example A

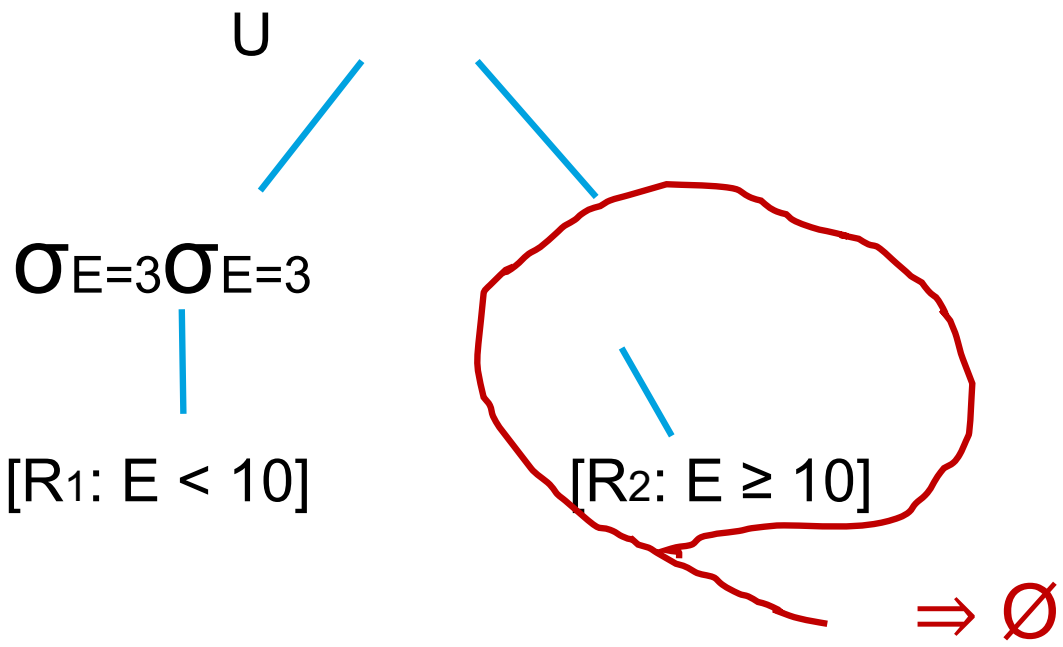


Example A



Anything Wrong here?

Example A



Example A

$\sigma_{E=3}$

$[R_1: E < 10]$