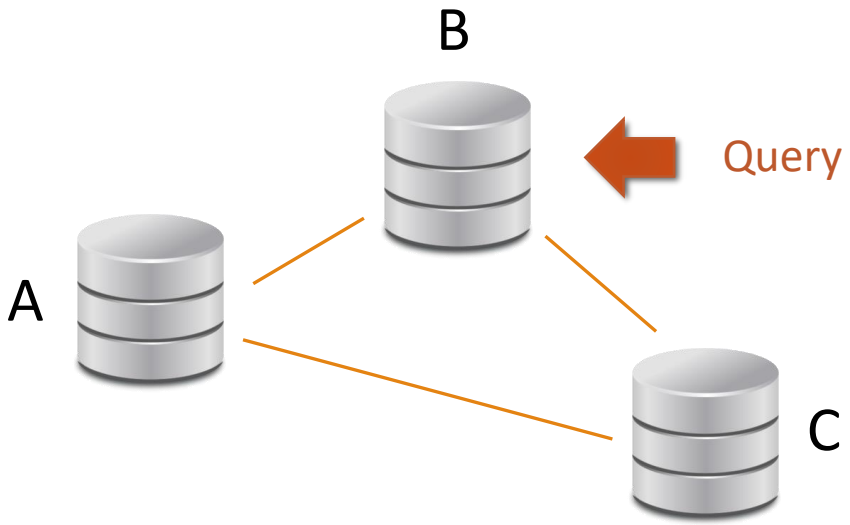
Query processing for DDBMS

Overview over this video

We discuss some problems that comes up in DDBMS for query processing and an example of how to do better than the naïve approach in some cases

Query Processing in Distributed DBMS



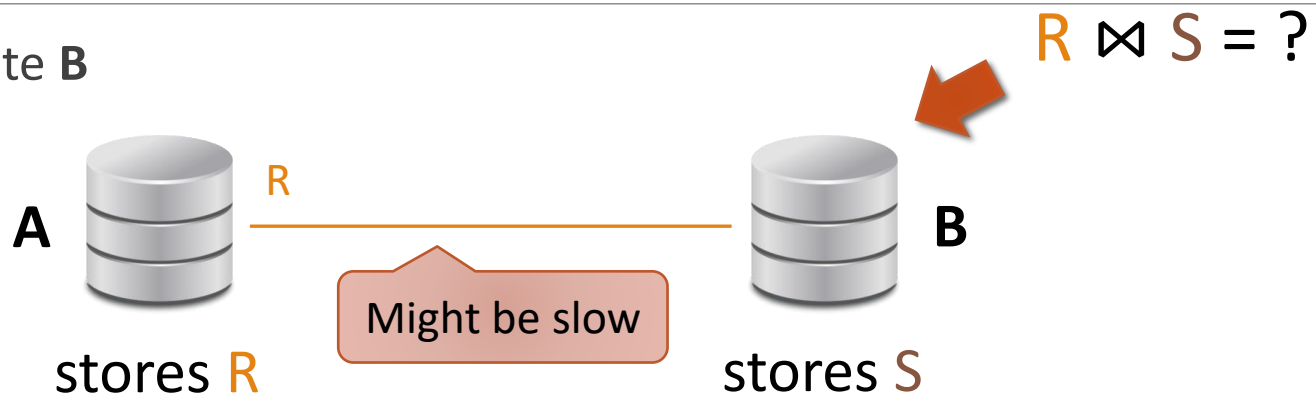
Try to answer query at site where query is raised

If not possible: request information from other sites

- Slow → design database to reduce this as much as possible
- Most expensive: joins

Joins

Goal: compute join at site **B**



Obvious approach:

- Site **B** asks site **A** to send R
- Site **B** computes R ⋈ S

R might be very large – do we have to send all tuples?

- Better: only send data that is actually required

Semijoins (⋉)

$R \ltimes S := R \bowtie \pi_{\text{common attributes of R and S}}(S)$

Modules

module	year
COMP105	1
COMP201	2
COMP207	2

Lecturers

name	module
J. Fearnley	COMP105
S. Coope	COMP201

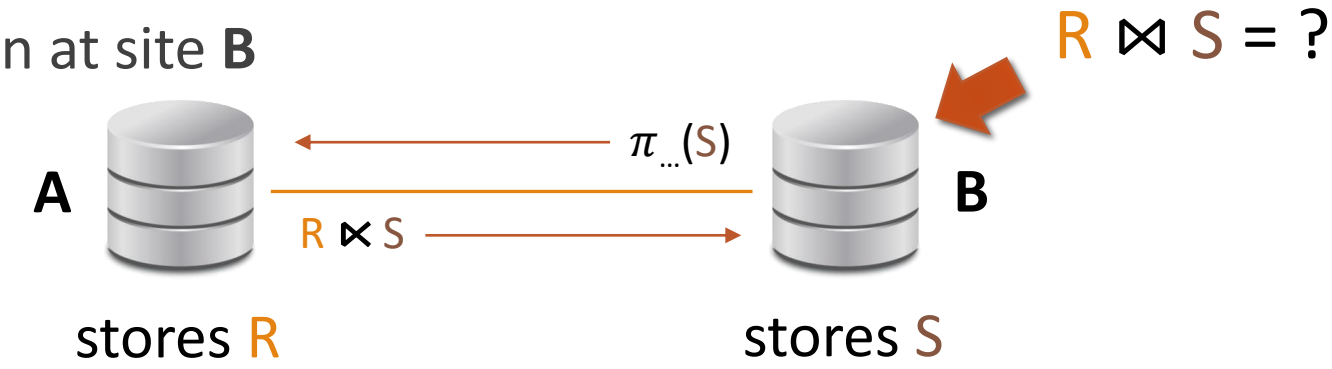
Modules ⋉ Lecturers =

module	year
COMP105	1
COMP201	2

Intuition: R ⋉ S = set of all tuples in R that NATURAL JOIN at least one tuple in S

Semijoin Reduction

Goal: compute join at site **B**



With semijoins:

- Site **B** sends S' := $\pi_{\text{common attributes of R and S}}(S)$ to site **A**
- Site **A** sends R' := R ⋉ S (= R ⋈ S') to site **B**
- Site **B** outputs R' ⋈ S

Communication costs ≈

$$|S'| \times (\text{size of tuple in } S') + |R'| \times (\text{size tuple in } R')$$

Efficiency

Is this more efficient than computing the join in the obvious approach way (exchanging relations)?

Depends:

- Is the projection much smaller than the full relation?
  - Many duplicates to be eliminated? I.e., do many tuples of S share values of the common attributes? Not the case if one of the join attributes is a key…
- Is the size of the semijoin much smaller?
- In general: $|\pi_{\text{common attributes}}(S)| + |R \ltimes S|$ should be smaller than $|R|$
- Note that: You can calculate if sending $\pi_{\text{common attributes}}(S)$ is less than sending R, if so, you can try. In the worst case you end sending twice as much as needed (e.g. in case $\pi_{\text{common attributes}}(S)$ is close to the size of R and $R \ltimes S$ is R)

Example

Films $\bowtie \sigma_{\text{city='Liverpool' AND date=2020-12-3}}$(InTheatres)

Films(film_title, genre, …)

InTheatres(film_title, theatre, city, date, …)

Procedure:

- At **B**, send S' := $\pi_{\text{film\_title}}(\sigma_{\text{city='Liverpool' AND date= date=2020-12-3}}(\text{InTheatres}))$ to **A**
- At **A**, send R' := Films $\ltimes$ S' to **B**
- At **B**, output R' $\bowtie \sigma_{\text{city='Liverpool' AND date= date=2020-12-3}}(\text{InTheatres})$

Communication costs:

- Assume |Films| = 10,000, |S'| = |R'| = 20, and 1000 bytes per tuple
- Communication cost = (20+20) × 1000 = 40,000 bytes

10,000,000 bytes for obvious approach

Summary

Query processing have new challenges in DDBMS, because of the distributed nature of the database

- We saw how to use Semijoins to in some cases speed up joins

Also, if you are watching these in order, this was it for the required part on distributed databases

- There are some more videos on MapReduce – special programming framework that fits somewhere between distributed databases and the next topic of NoSQL databases (i.e. not only SQL databases)
  - Those are not required for the exam though