Fragmentation, replication and transparency

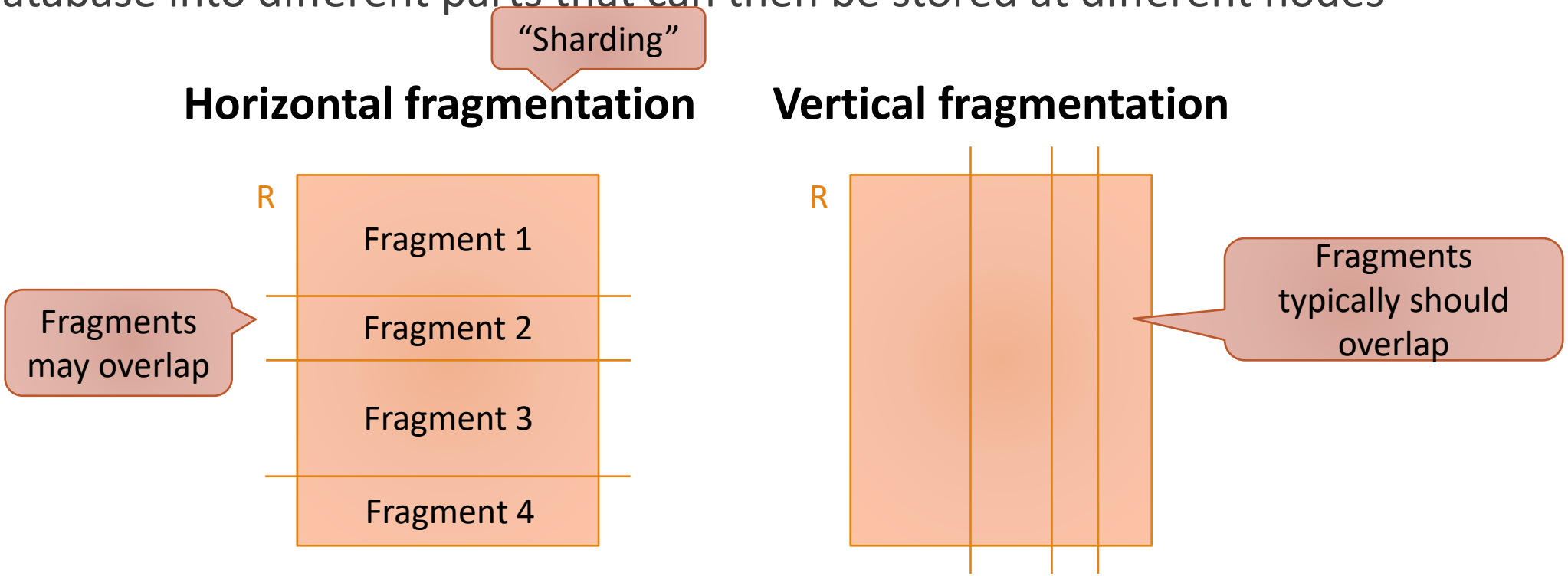
Overview over this video

This video covers transparency, i.e. things that the DDBMS (distributed DMBS – distributed database management system) keeps hidden from people accessing the database

- Fragmentation and replication from the title are important examples of such

Fragmentation

Split database into different parts that can then be stored at different nodes



Users don't see fragments, just the full relations

"Fragmentation transparency"

Horizontal Fragmentation Example

The chain of stores from our running example might jointly store the Transaction relation

Does not exist physically!

e_id	c_id	date	t_id	
124	3247	2020-07-10	643682	← @Liverpool
513	6343	2020-11-15	855321	← @Manchester
1235	7542	2020-11-22	934342	← @Manchester
1643	856	2020-04-03	324456	← @London
865	2347	2018-05-07	86545	← @Liverpool
…	…	…	…	

Each site stores a relation that contains a subset of the tuples

Entire relation = union of relations at the different sites

Horizontal Fragmentation Example

Typically, tuples stored at different sites can be distinguished

- by the value of one or a few attributes; or
- by other conditions that are easy to test

store	e_id	c_id	date	t_id
Liverpool	124	3247	2020-07-10	643682
Manchester	513	6343	2020-11-15	855321
Manchester	1235	7542	2020-11-22	934342
London	1643	856	2020-04-03	324456
Liverpool	865	2347	2018-05-07	86545
…	…	…	…	…

Vertical Fragmentation Example

The stores might also jointly store a relation

name	t_id	c_id	price
Cheese	1324	1245	250
Bread	32412	5678	100
…	…	…	…

Does not exist physically!

@Central Office

name	price
Cheese	250
Bread	100
…	…

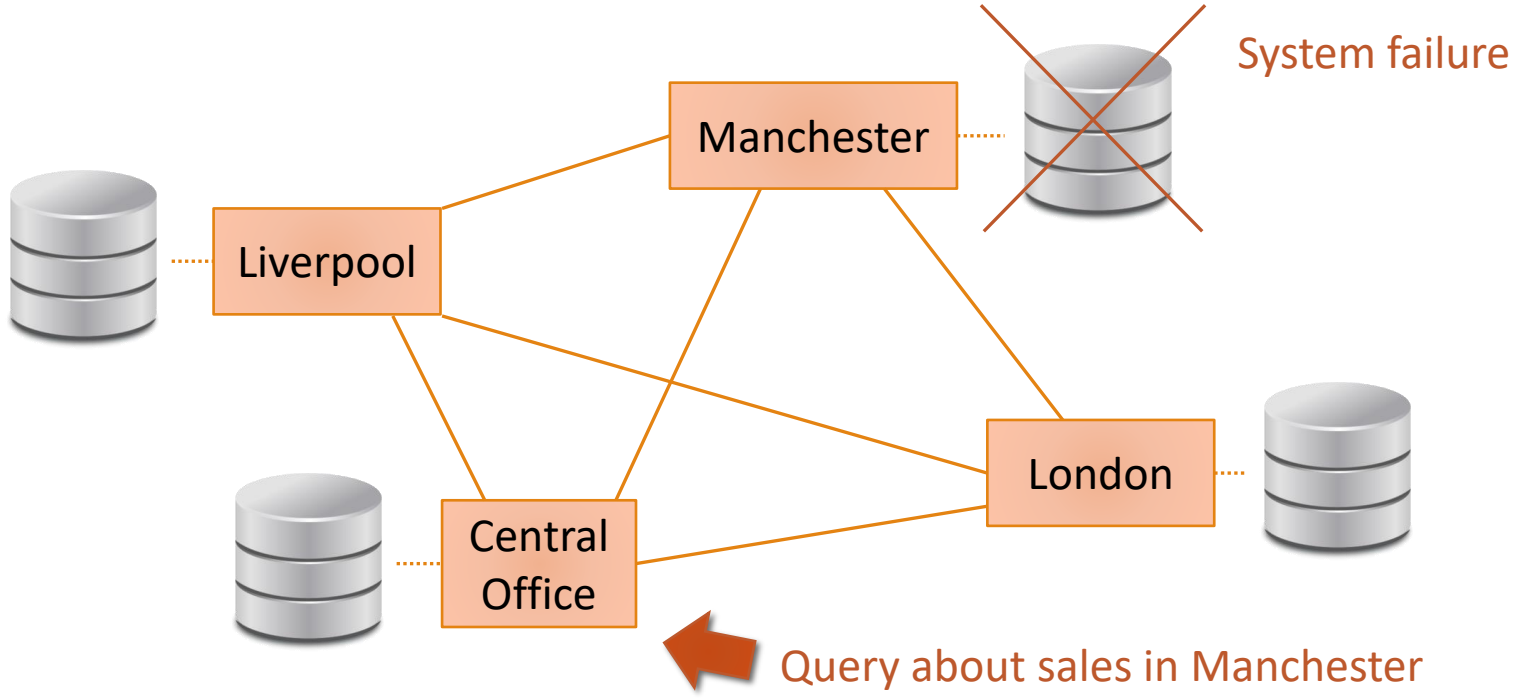
Stored at other sites

name	t_id	c_id
Cheese	1324	1245
Bread	32412	5678
…	…	…

This could then be horizontally fragemented

Original relation ≈ join of the fragments

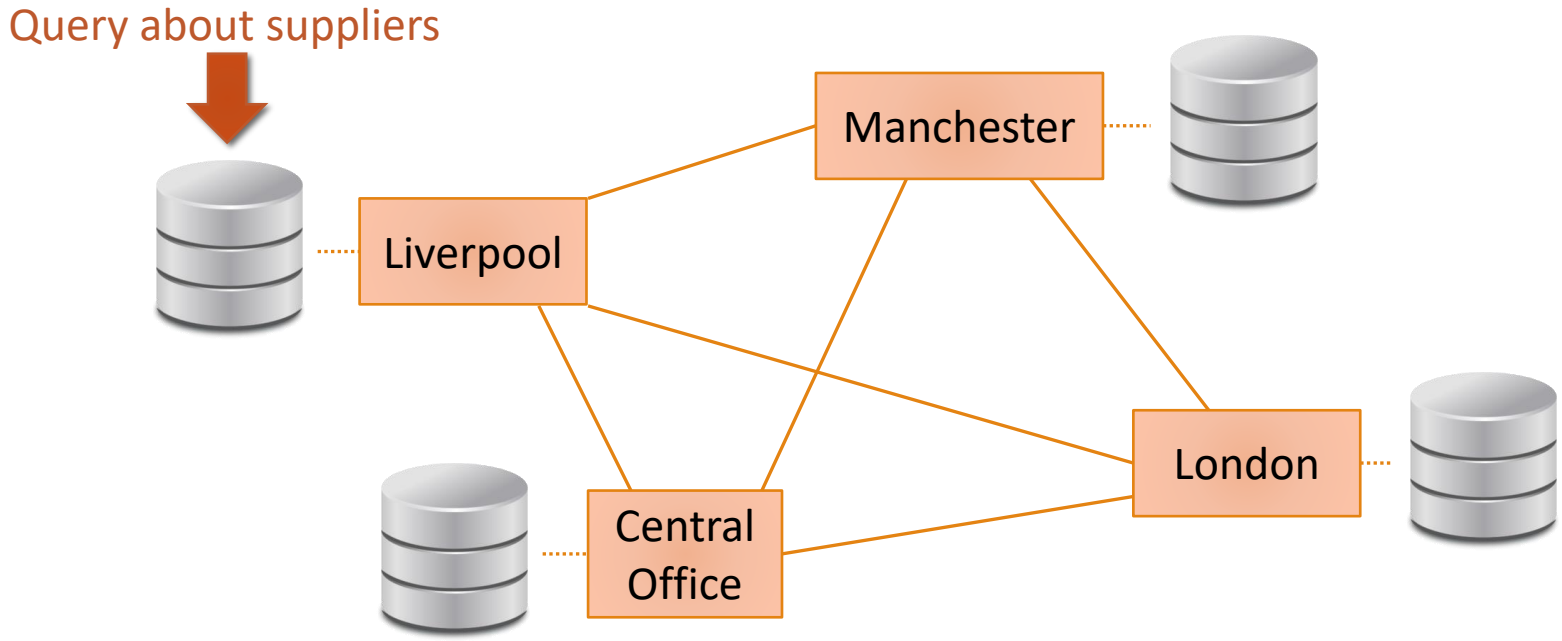
Redundancy Improves Resilience



Other sites keep copies of fragment stored at Manchester

Allows us to answer queries involving data from Manchester

Redundancy Increases Efficiency



Other sites keep copies of data about suppliers

Allows stores to answer queries involving suppliers without establishing a connection to the central office

Replication

Controls how many sites keep a copy of a fragment

Full replication

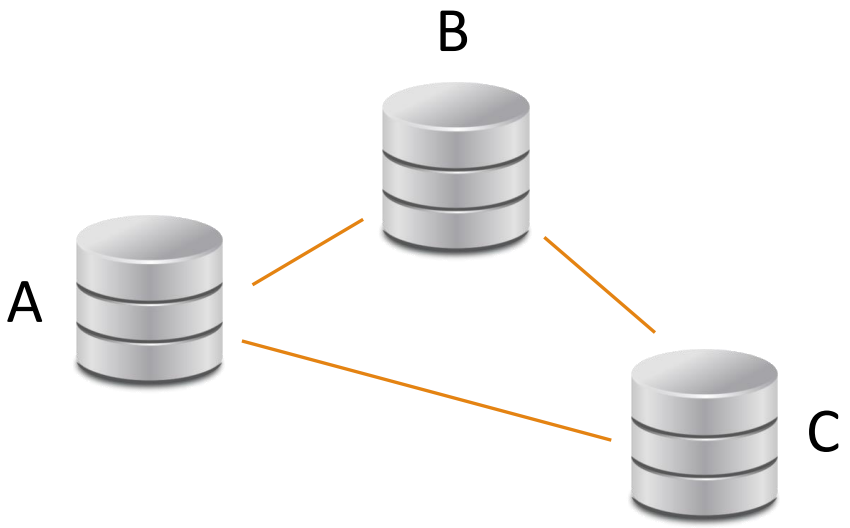
- Each fragment stored at every site (→ there are no fragments)
- Faster query answering
- Very slow updates: consider *every* copy

No replication

- Each fragment stored at a unique site
- Crashes are a big problem…

Wide spectrum of **partial replication**

- Limit number of copies of each fragment
- Replicate only some fragments, …



Transparency

DDBMSs ensures that users do not need to know certain facts then creating queries

Transparency at different levels

- **Fragmentation transparency**
  - Fragmentation is transparent to users
  - Users pose queries against the entire database
  - The distributed DBMS translates this into a query plan that fetches the required information from appropriate nodes
- **Replication transparency**
  - Ability to store copies of data items / fragments at different sites
  - Replication is transparent to users
- **Location transparency**
  - The location where data is store is transparent to the user
- **Naming transparency**
  - A given name (e.g. of a relation) has the same meaning everywhere in the system
- **…. and others**

Summary

There are four main kinds of transparency:

- **Fragmentation**
- **Replication**
- **Location**
- **Naming**