Database system architectures

Lecture topics:

- monolithic systems
- client–server systems
- parallel database servers

References:

- text 3rd edition, chapter 24: sections 1, 2, 6, 8
- text 4th edition, chapter 25: sections 1–3, 6, 8

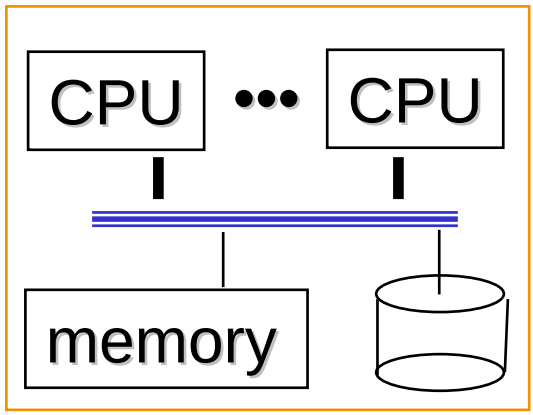
Multi-CPU and distributed systems

- CPU “computing power” does not scale linearly: a *2n*-MIPS (or MHz or GHz) CPU costs much more than twice an *n*-MIP (MHz/GHz) CPU
- To increase system throughput, increase number of CPUs, not speed of (single) CPU
- Two techniques:
  - parallel
  - distributed

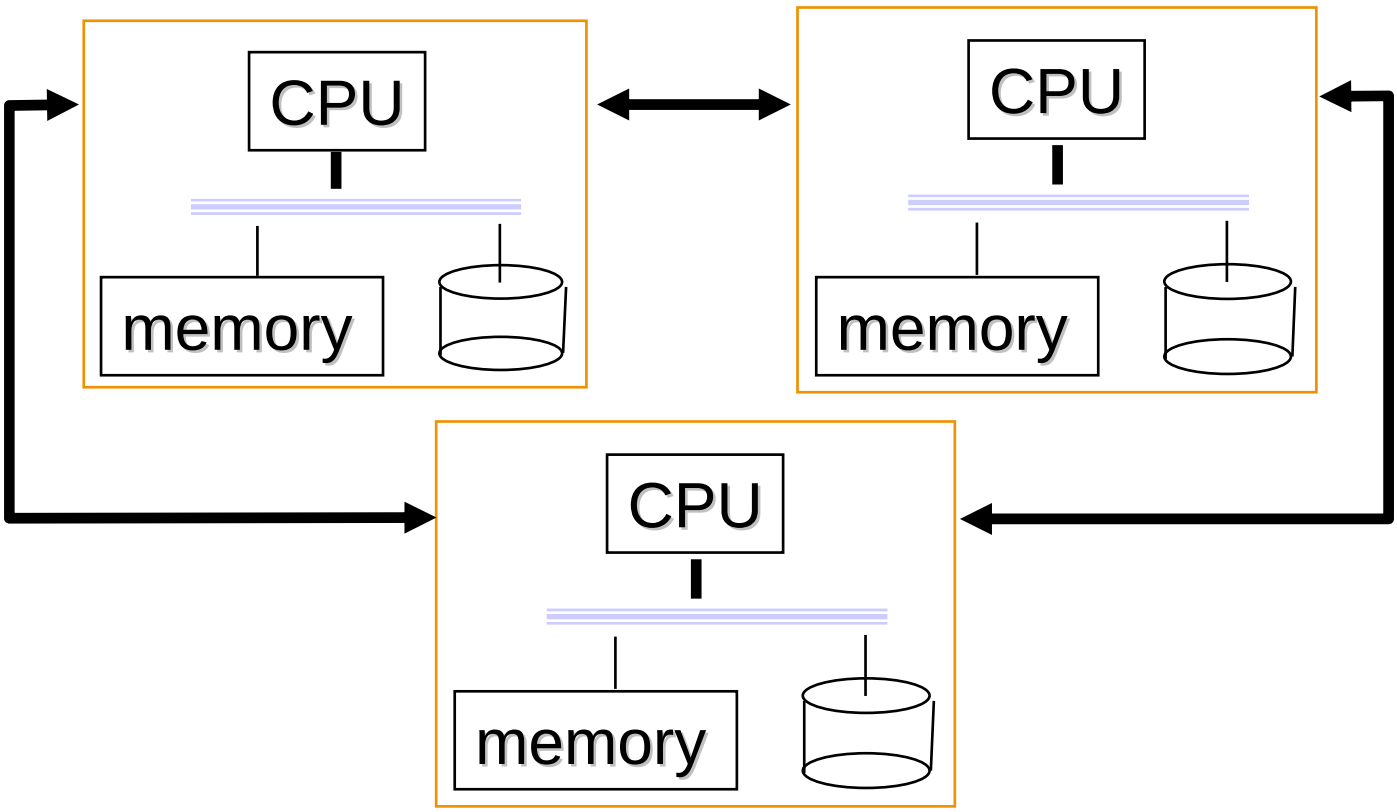
...continued

Parallel:

- single “chassis” with sharing



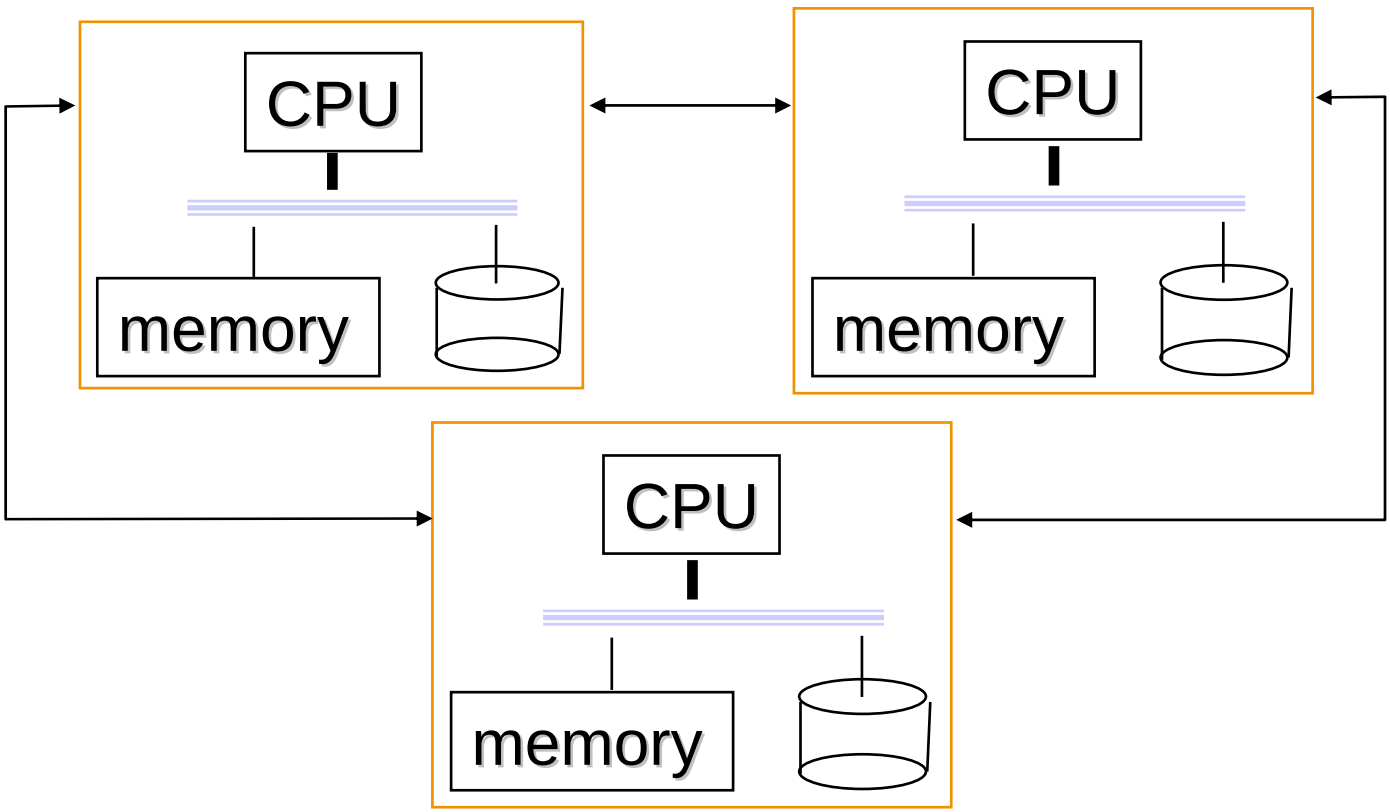
- many systems with high-speed LAN



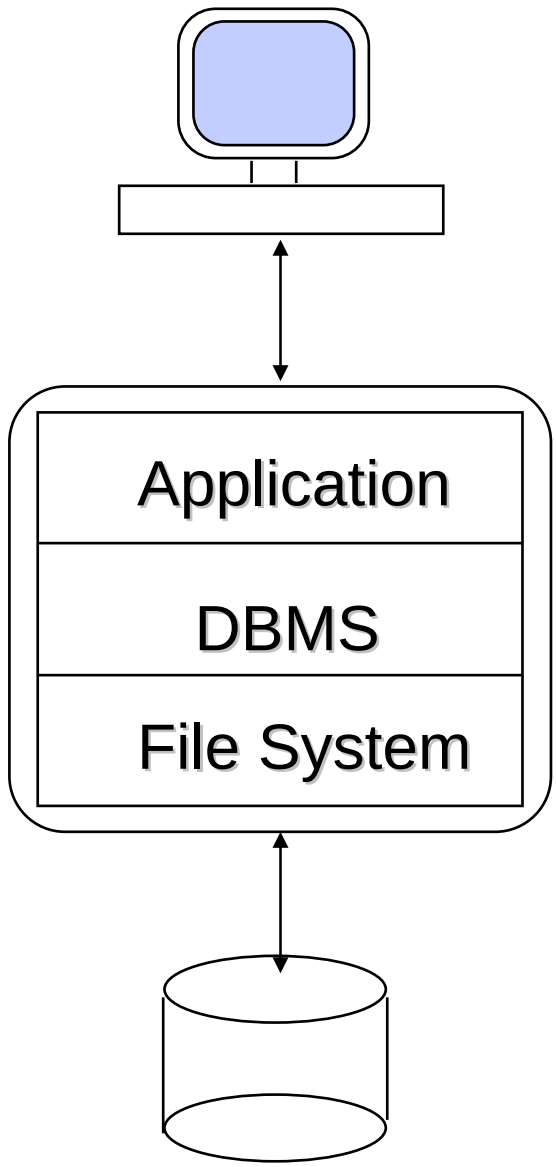
...continued

Distributed:

- many systems loosely-coupled with WAN



Monolithic system

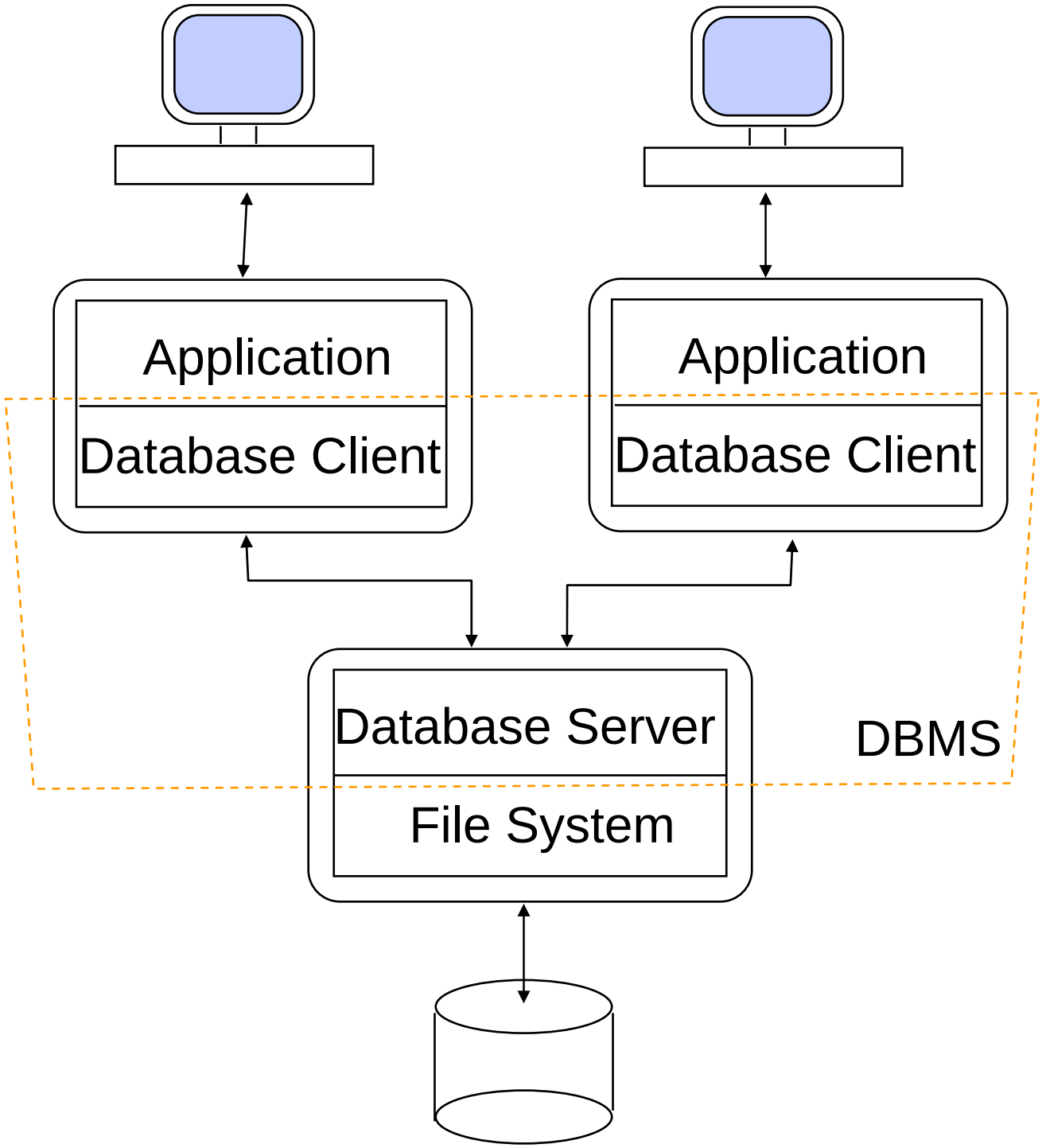


- Each component presents a well-defined interface to the component above

Component functions

- applications
  - user interaction: input of queries and data, display of results
  - application-specific tasks
- DBMS
  - query optimization: selection of one of many possible procedures for executing a query
  - query processing: execution of selected query
  - buffer management: allocation and control of memory
  - transaction management: concurrency control, rollback, and failure recovery
  - security and integrity management: access control and consistency checking
- file system
  - storage and retrieval of unstructured data on disks

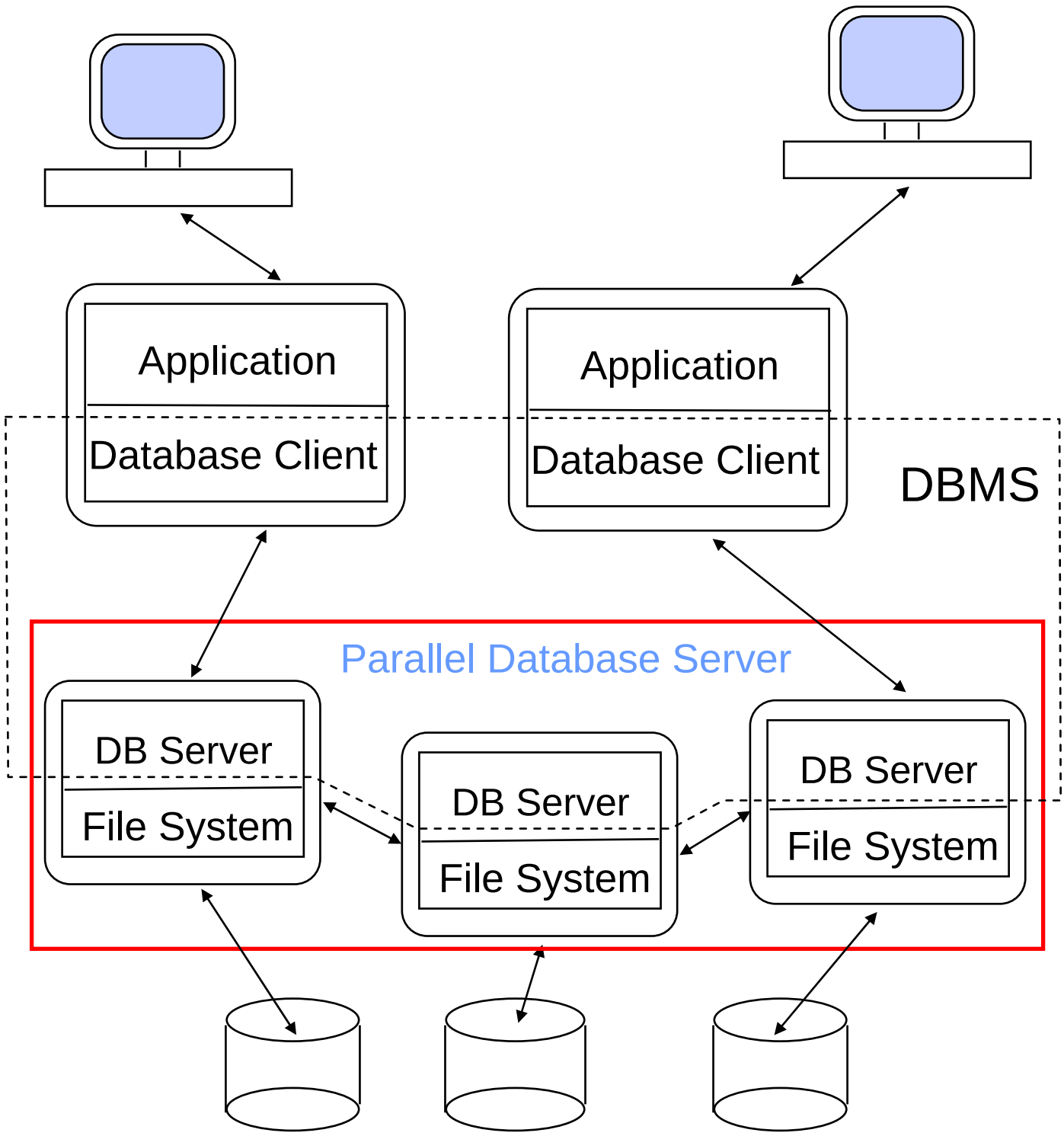
Client–server system



...continued

- DBMS client: packs application requests into messages, sends messages to server, waits for and unpacks the response; manages user interface
- DBMS server: all database system functions, including query processing and optimization, transaction management, security and integrity management, buffer management
- Client–server separation allows user interaction and database management to be performed by different processors

Parallel, distributed database server



...continued

- data is distributed across the **sites**
- relations may be fragmented
- relations (or fragments of relations) may be replicated at several sites
- clients perceive a single database with a single, common schema

Transparency:

- distribution of data is transparent
- distribution of computation is transparent
- replication is transparent
- fragmentation is transparent

Parallel vs distributed servers

- What multiprocessing architecture to use?

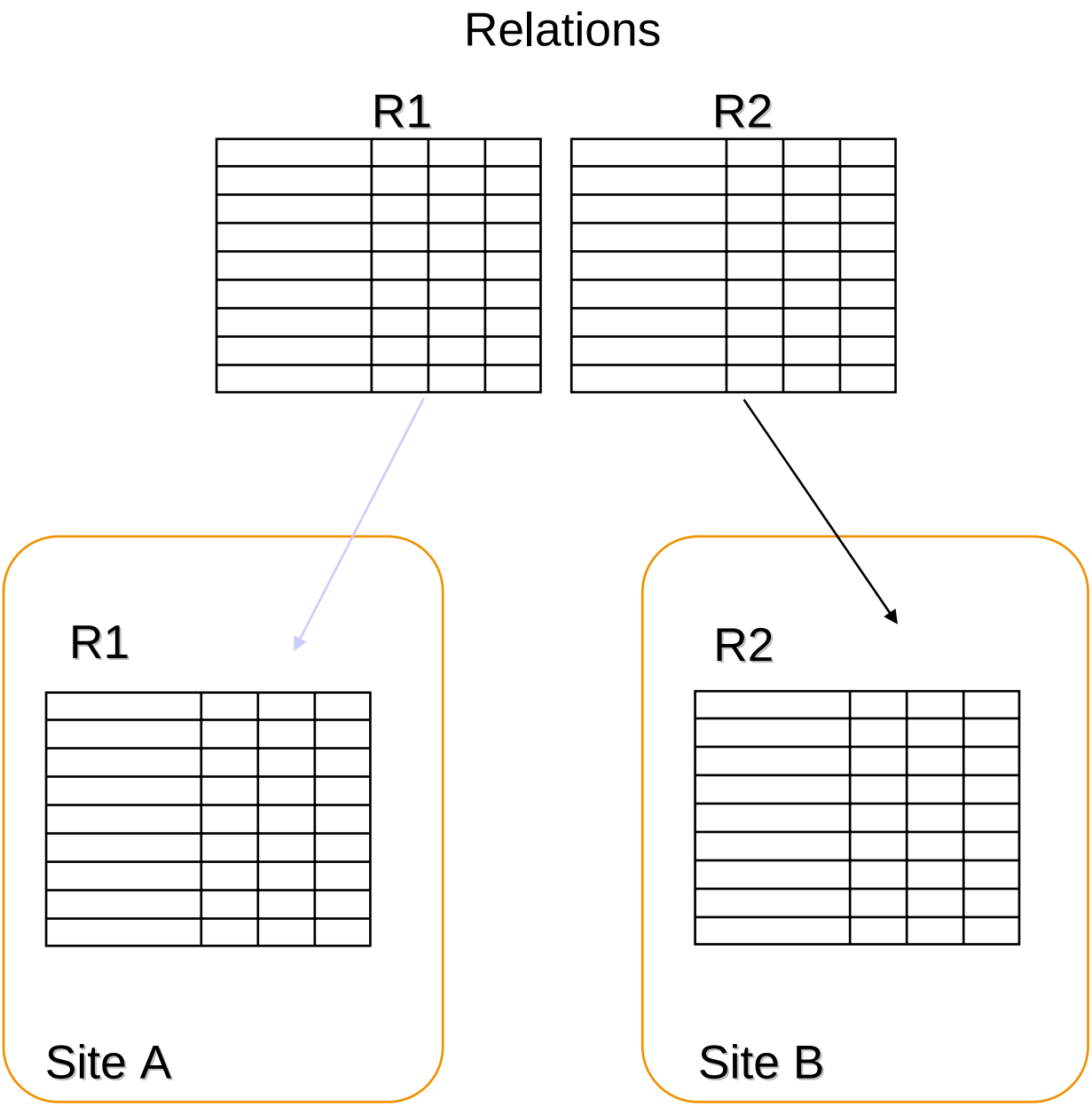
- parallel database server:
  - servers in physical proximity to each other
  - fast, high-bandwidth communication between servers, usually via a LAN
  - most queries processed cooperatively by all servers

- distributed database server:
  - servers may be widely separated
  - server-to-server communication may be slower, possibly via a WAN
  - queries often processed by a single server

Parallel, distributed: why?

- reliability and availability: if one server fails, another can take its place
- faster query processing: several servers can cooperate to process a query
- data sharing with distributed control: individual sites can share data while retaining some autonomy
- incremental database growth
- But:
  - processing overhead
  - difficulties enforcing integrity constraints
  - software complexity (cost and reliability)

Data distribution



Horizontal fragmentation

- **Complete relation:**

Vno	Vname	City	Vbal
1	Sears	Toronto	200.00
2	Kmart	Ottawa	671.05
3	Eatons	Toronto	301.00
4	The Bay	Ottawa	162.99

- **Horizontally fragmented relation (two sites):**

Site 1 (Ottawa site)

Vno	Vname	City	Vbal
2	Kmart	Ottawa	671.05
4	The Bay	Ottawa	162.99

Site 2 (Toronto site)

Vno	Vname	City	Vbal
1	Sears	Toronto	200.00
3	Eatons	Toronto	301.00

continued...

- Horizontal fragmentation stores subsets of a relation at different sites
- If R is divided into n subsets, labelled R_i (i=1..n), then:

$$R = R_1 \cup R_2 \cup \ldots \cup R_n$$

- Typical application in organizations with distributed management (e.g. local branch autonomy and control of data)

Vertical Fragmentation

- Complete relation:

Vno	Vname	City	Vbal
1	Sears	Toronto	200.00
2	Kmart	Ottawa	671.05
3	Eatons	Toronto	301.00
4	The Bay	Ottawa	162.99

- Vertically fragmented relation (two sites):

Site 1

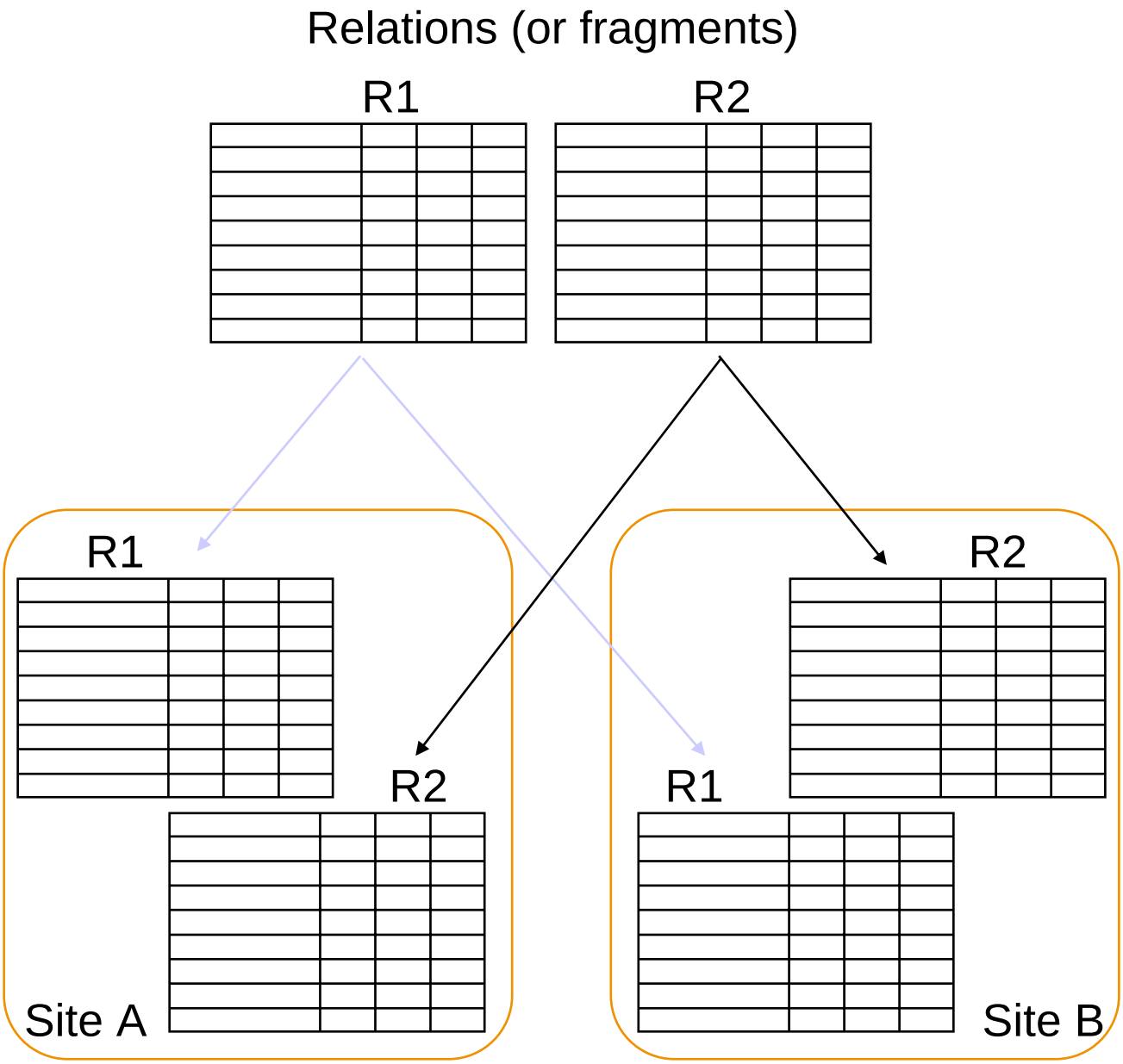
Vno	Vbal
1	200.00
2	671.05
3	301.00
4	162.99

Site 2

Vno	Vname	City
1	Sears	Toronto
2	Kmart	Ottawa
3	Eatons	Toronto
4	The Bay	Ottawa

- Vertical fragmentation is a lossless decomposition
  - for decomposition $R = \{R_1, R_2, \ldots, R_n\}$
    $R = R_1$ join R_2 join … join R_n
- Typical use in organizations where org. units only use partial information

Data replication



continued...

- Relations are copied to many sites
- Pros:
  - much better availability
  - faster (local) access
  - redundancy gives increased reliability
- Cons:
  - update much more complex and time-consuming
  - redundancy gives potential integrity problems
- Applicable especially in read-only transactions and infrequent changes to database