Announcements

• WQ7 due tomorrow
• HW7 due on Wednesday
• HW8 (last one!) will be out on Wednesday

• Final on 12/12 (Monday), 2:30-4:20pm
  – Location TBD
  – You can bring 2 sheets of notes
HW8 Preview

• HW8 will require using Amazon EC2 compute cloud
  – If you do not already have an Amazon account, go to http://aws.amazon.com/ and sign up.
    • Amazon will ask you for your credit card information.
  – Apply for credits on https://aws.amazon.com/education/awseducate/apply/ (choose students)
    • Use your @uw.edu email address
Outline

• Review of transactions

• Parallel databases
Transactions Recap

• Why we use transactions?
  – ACID properties

• Serializability
  – Conflict-serializable / non-serializable

• Implementation using locks
  – 2PL and strict 2PL
  – Serialization levels
In-Class Exercise

Given these 3 transactions: (Co = commit)
- T1 : R₁(A), R₁(B), W₁(A), W₁(B), Co₁
- T2 : R₂(B), W₂(B), R₂(C), W₂(C), Co₂
- T3 : R₃(C), W₃(C), R₃(A), W₃(A), Co₃

And this schedule:
- R₁(A), R₁(B), W₁(A), R₃(C), W₃(C), R₃(A), W₃(A), Co₃, W₁(B), R₂(B), W₂(B), Co₁, R₂(C), W₂(C), Co₂

Determine:
- If the schedule conflict-serializable? If yes, indicate a serialization order. 1, 3, 2
- Is this schedule possible under the strict 2PL protocol?
Parallel DBMS
Why compute in parallel?

• Multi-cores:
  – Most processors have multiple cores
  – This trend will increase in the future

• Big data: too large to fit in main memory
  – Distributed query processing on 100x-1000x servers
  – Widely available now using cloud services
Big Data

• Companies, organizations, scientists have data that is too big, too fast, and too complex to be managed without changing tools and processes

• Complex data processing:
  – Decision support queries (SQL w/ aggregates)
  – Machine learning (adds linear algebra and iteration)
Two Kinds of Parallel Data Processing

• Parallel (relational) databases, developed starting with the 80s (this lecture)
  – OLTP (Online Transaction Processing)
  – OLAP (Online Analytic Processing, or Decision Support)
    – Will only cover parallel query execution in 344
    – Parallel transactions and recovery (444)
    – Schema design for parallel DBMS (544)

• General purpose distributed processing: MapReduce, Spark (next lectures)
  – Mostly for Decision Support Queries
Performance Metrics for Parallel DBMSs

$P = \text{the number of nodes (processors, computers)}$

- **Speedup:**
  - More nodes, same data $\Rightarrow$ higher speed

- **Scaleup:**
  - More nodes, more data $\Rightarrow$ same speed

- **OLTP:** “Speed” = transactions per second (TPS)
- **Decision Support:** “Speed” = query time
Linear v.s. Non-linear Scaleup

Batch Scaleup

# nodes (=P) AND data size

x1  x5  x10  x15

Ideal
Challenges to Linear Speedup and Scaleup

• **Startup cost**
  – Cost of starting an operation on many nodes

• **Interference**
  – Contention for resources between nodes

• **Skew**
  – Slowest node becomes the bottleneck
Architectures for Parallel Databases

- Shared memory
- Shared disk
- Shared nothing
Shared Memory

Interconnection Network

Global Shared Memory

D  D  D

process b
cisks

CSE 344 - Fall 2016
Shared Disk

Interconnection Network
Shared Nothing

Interconnection Network

P
M
D

P
M
D

P
M
D
A Professional Picture...

From: Greenplum (now EMC) Database Whitepaper

SAN = “Storage Area Network”
Shared Memory

• Nodes share both RAM and disk
• Dozens to hundreds of processors

Example: SQL Server runs on a single machine and can leverage many threads to get a query to run faster (check your HW3 query plans)

• Easy to use and program
• But very expensive to scale: last remaining cash cows in the hardware industry
Shared Disk

• All nodes access the same disks
• Found in the largest "single-box" (non-cluster) multiprocessors

Oracle dominates this class of systems.

Characteristics:
• Also hard to scale past a certain point: existing deployments typically have fewer than 10 machines
Shared Nothing

• Cluster of machines on high-speed network
• Called "clusters" or "blade servers"
• Each machine has its own memory and disk: lowest contention.

NOTE: Because all machines today have many cores and many disks, then shared-nothing systems typically run many "nodes" on a single physical machine.

Characteristics:
• Today, this is the most scalable architecture.
• Most difficult to administer and tune.

We discuss only Shared Nothing in class
We study only intra-operator parallelism: most scalable
Single Node Query Processing (Review)

Given relations $R(A,B)$ and $S(B, C)$, no indexes:

- **Selection**: $\sigma_{A=123}(R)$
  - Scan file $R$, select records with $A=123$

- **Group-by**: $\gamma_{A,sum(B)}(R)$
  - Scan file $R$, insert into a hash table using $A$ as key
  - When a new key is equal to an existing one, add $B$ to the value

- **Join**: $R \bowtie S$
  - Scan file $S$, insert into a hash table using $B$ as key
  - Scan file $R$, probe the hash table using $B$
Distributed Query Processing

• Data is horizontally partitioned on many servers

• Operators may require data reshuffling
Horizontal Data Partitioning

Data:

<table>
<thead>
<tr>
<th>K</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Servers:

1  2  ...  P
Horizontal Data Partitioning

Data:

<table>
<thead>
<tr>
<th>K</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Servers:

1

<table>
<thead>
<tr>
<th>K</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2

<table>
<thead>
<tr>
<th>K</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

... ...

P

<table>
<thead>
<tr>
<th>K</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Which tuples go to what server?