

Lab 3 - Partitioning

DB/AI Bootcamp

2018 Summer

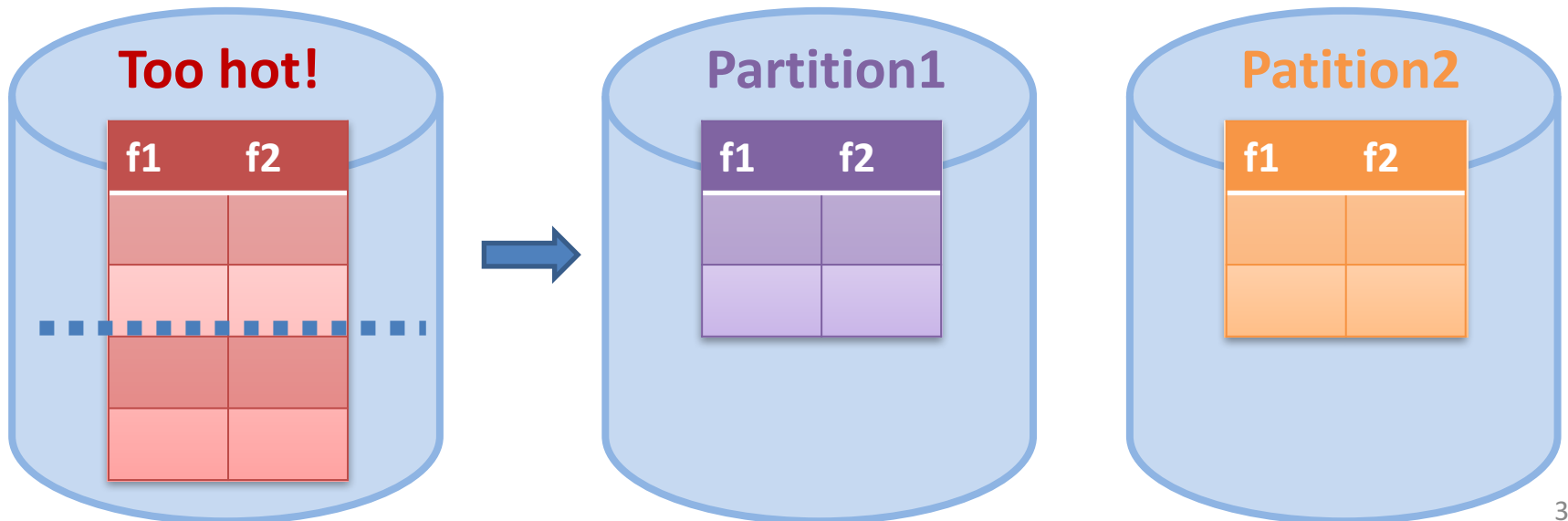
DataLab, CS, NTHU

SAE

- Remember we mentioned that a cloud database should ensure **SAE**:
- ***high Scalability***
 - High max. throughput
- ***high Availability***
 - Stay on all the time, despite of machines/network/datacenter failure
- ***Elasticity***
 - Add/shutdown machines and re-distribute data on-the-fly based on the current workload

Scalability and Partitioning

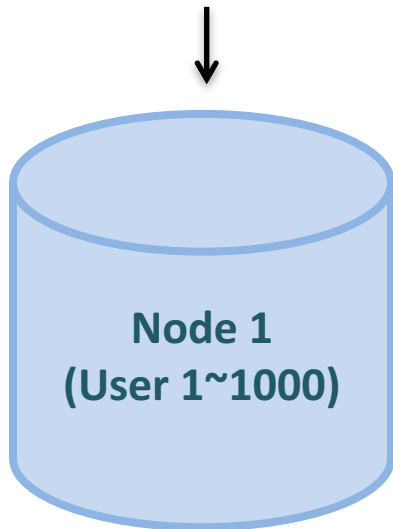
- We increase scalability by *partitioning* a database
 - Usually horizontally
 - Distribute read/write load to different servers



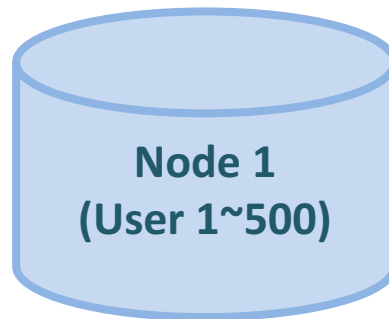
Distributed Transactions

- It may cause that a transaction needs to access multiple partitions.

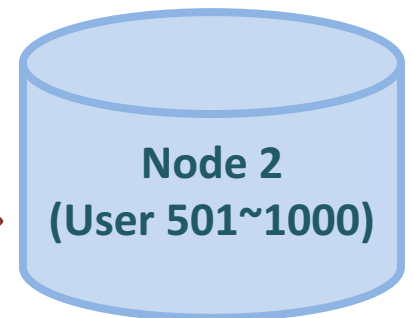
T_3 : write(user 596, user 4)



T_3 : write(user 596, user 4)



2PC



Good Partitioning

- We want to partition data such that
 - the system increases scalability.
 - each machine gets equal load.
 - #dist. txs are minimized.
- How? Hire an DBA?
 - No. Too expensive.

Automation: Graph Partitioner

- Model the recent workload as a graph
 - Nodes: data objects, with weight denoting their access frequency
 - Edges: common access by txs, also weighted
- Find the minimal balanced partition of this graph
 - Machines are evenly loaded
 - Dist. txs are minimized

Schism [SIMGOD'10]

GRAPH REPRESENTATION

transaction edges

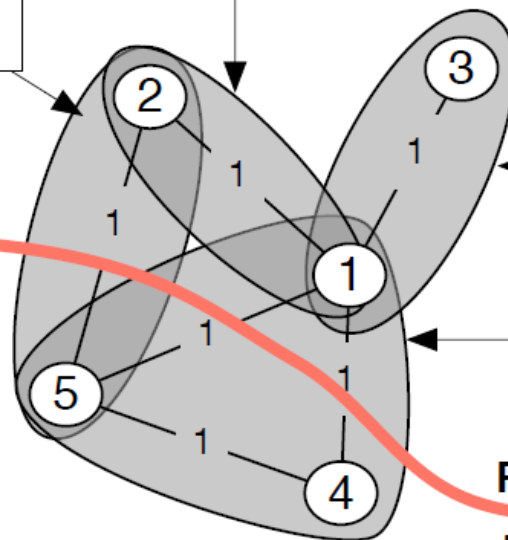
```
BEGIN
UPDATE account SET bal=60k
  WHERE id=2;
SELECT * FROM account
  WHERE id=5;
COMMIT
```

account		
id	name	bal
1	carlo	80k
2	evan	60k
3	sam	129k
4	eugene	29k
5	yang	12k
...

```
BEGIN
UPDATE account SET bal=bal-1k WHERE name="carlo";
UPDATE account SET bal=bal+1k WHERE name="evan";
COMMIT
```

```
BEGIN
SELECT * FROM account
  WHERE id IN {1,3}
ABORT
```

```
BEGIN
UPDATE SET bal=bal+1k
  WHERE bal < 100k;
COMMIT
```



PARTITION 0

PARTITION 1

Problems of Schism

- Schism generates a partition table.
 - Input: a key (e.g. primary key)
 - Output: a partition id
- The size of this table \propto the size of the DB
 - This table would become extremely large if the DB was large.
 - It may not fit in memory.
- Schism cannot handle unseen data.

Lab

- Task 1:
 - Given a workload, we firstly use Schism to generate a partition table.
 - Train a NN to mirror the table.
 - Much smaller
 - Able to handle unseen data
- Task 2:
 - Find the best partitioning without using Schism.

<https://github.com/yschang1206/dbai-labs>