
MONTANA
STATE UNIVERSITY

Database Clustering

•••

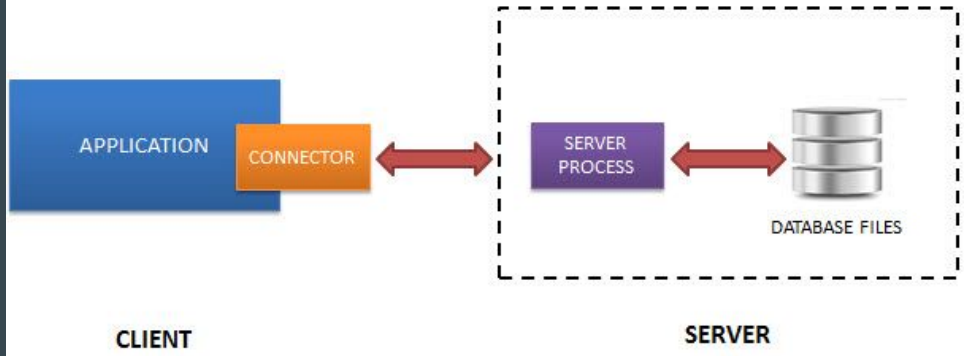
Multiple Databases

SQLite

- In this course we have been working with SQLite
  - A great, simple database
- However SQLite is unique amongst major databases in that it does not provide a network interface

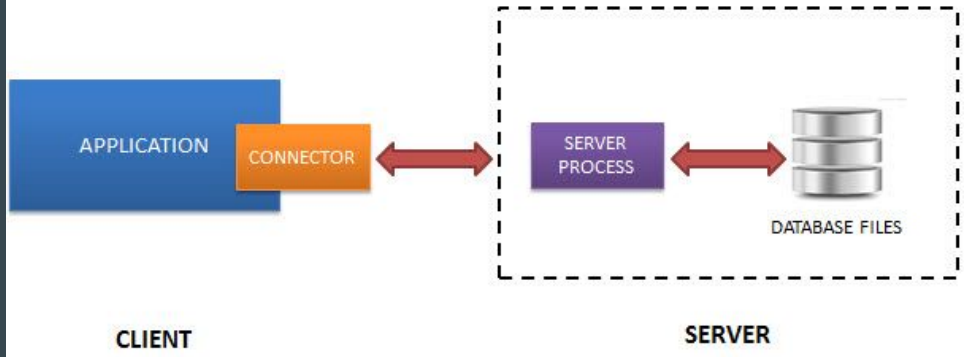
Networked DBMS

- Most other RDBMS systems (as well as NoSQL systems) offer a network API
  - Client/Server architecture
  - JDBC abstracts this away so java code would look very similar with another RDBMS



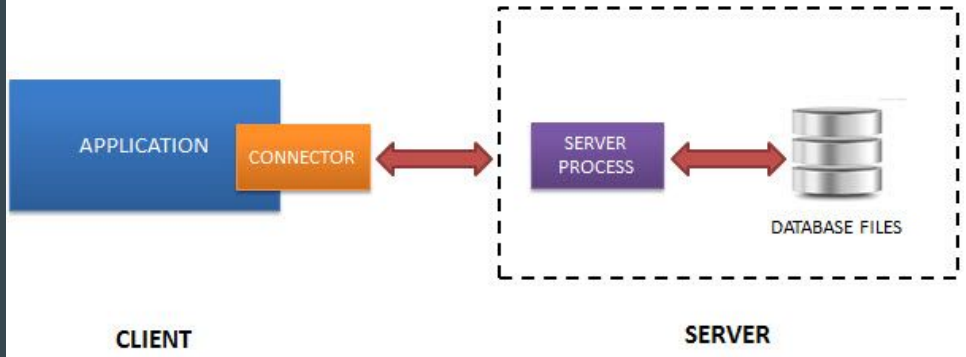
Networked DBMS

- Since most databases are on the network, they aren't constrained to only talk with clients
- They can also talk with other databases!



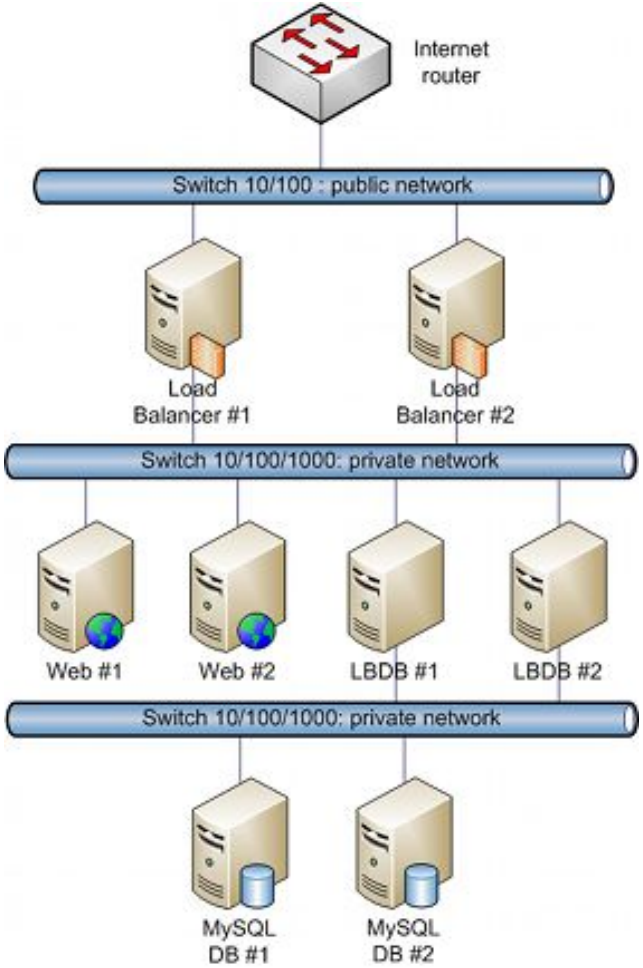
Networked DBMS

- Why would a database want to talk to another database?
  - Replicate data between one another?
  - Offload queries?
  - Offload other stuff?



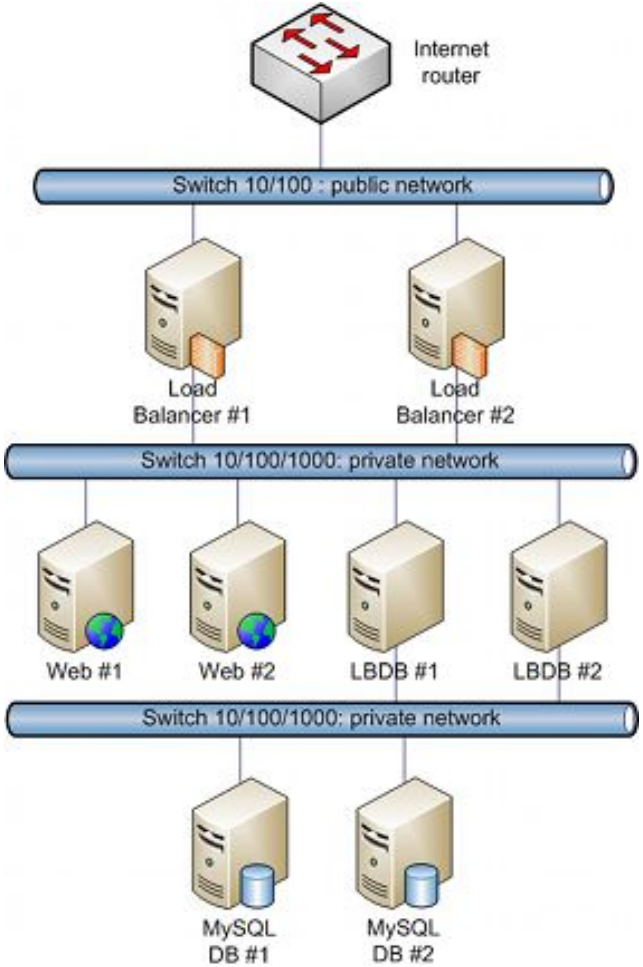
Clustering

- Here we see a standard cloud layout
  - Load Balancer
  - Web App
  - Database
- These two databases can communicate with one another over the ethernet network as a “Cluster”



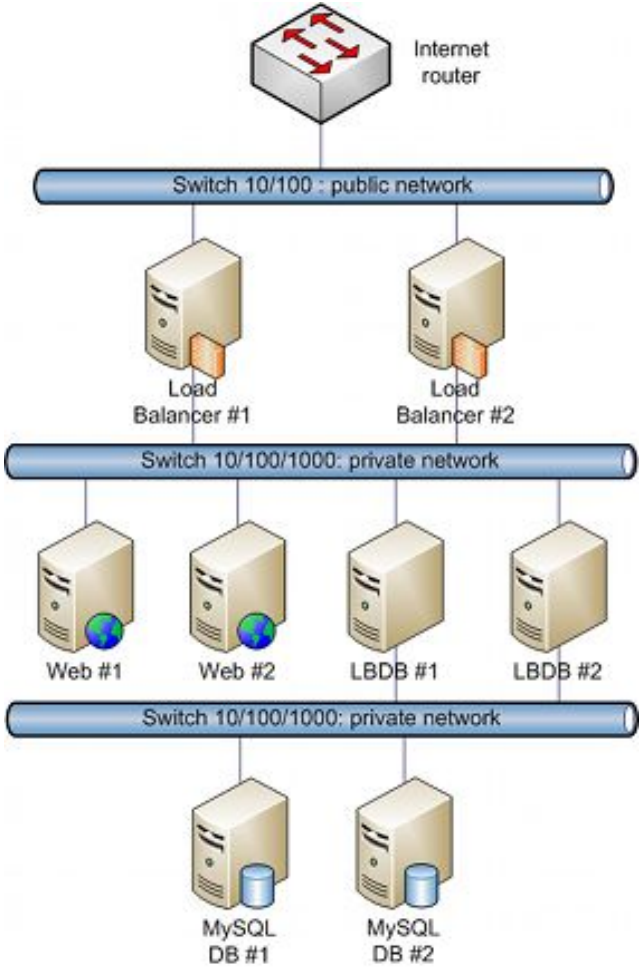
Clustering

- A cluster refers to a group of server (in this case databases) that are connected through a network to behave as a single resource



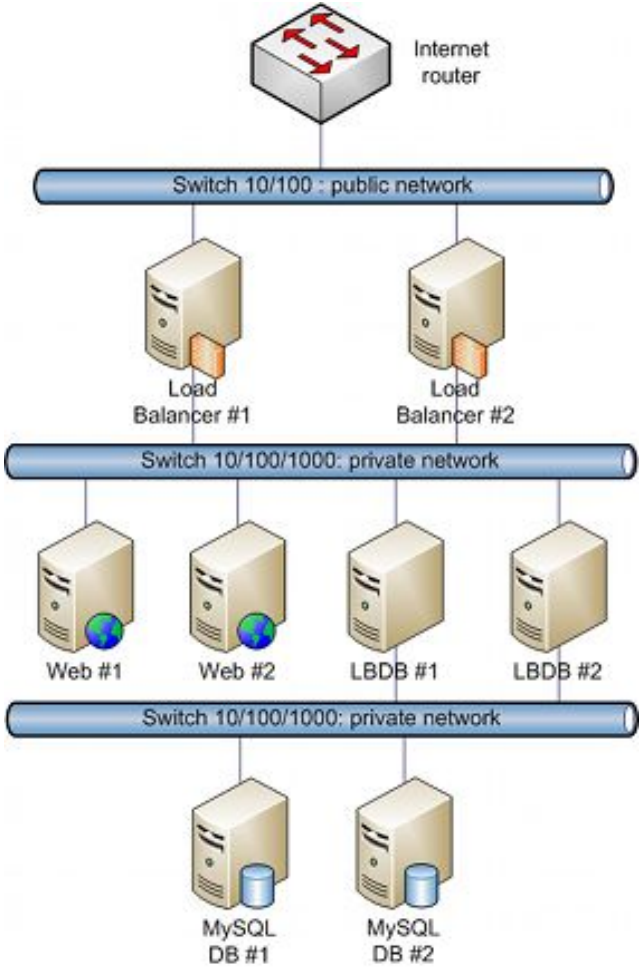
Clustering

- Advantages of clustering
  - High Availability
    - If one db fails, another db can take its place
  - Load Balancing
    - Queries and writes can be distributed across machines allowing for better performance



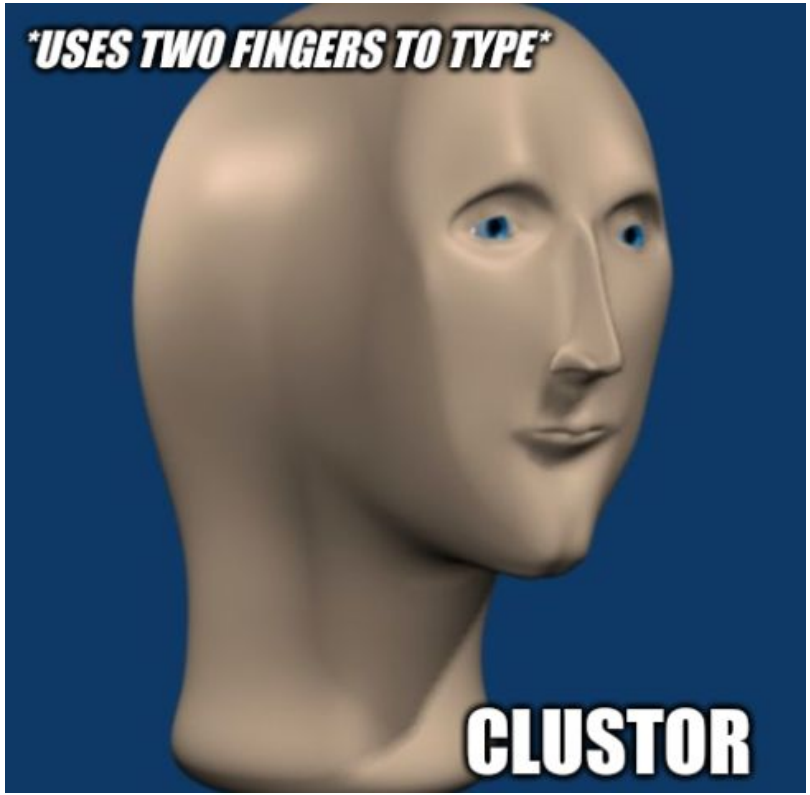
Clustering

- Advantages of clustering
  - Parallel Processing
    - Jobs can be split across machines to take better advantage of parallelism
  - Scalability
    - Once a cluster has been established, adding and removing a new server is relatively easy



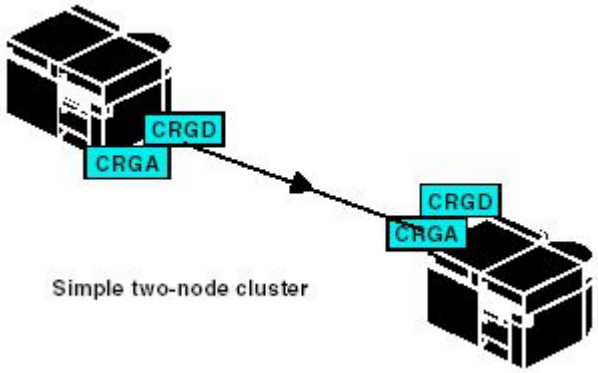
Clustering

- Advantages of clustering
  - Parallel Processing
    - Jobs can be split across machines to take better advantage of parallelism
  - Scalability
    - Once a cluster has been established, adding and removing a new server is relatively easy



Clustering Configurations

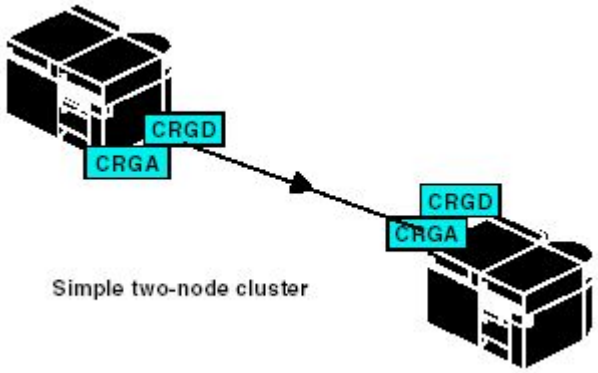
- The simplest cluster setup is simply a primary and backup server
  - No traffic is routed to the backup server in normal operation
  - If the primary fails, the system moves to the backup server
- The primary traffic between clustered servers is *replication*



Simple two-node cluster

Clustering Configurations

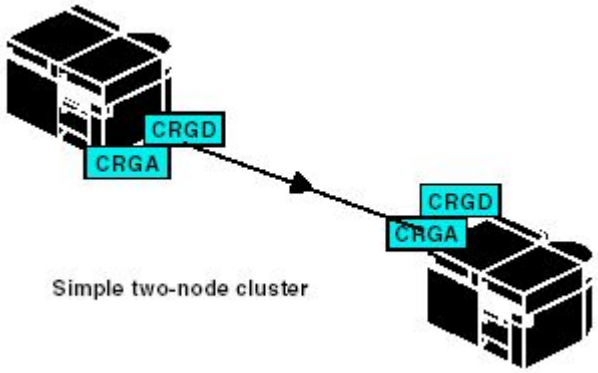
- Note that here the traffic is a one-way feed from the primary to the backup server
  - Simple and effective



Simple two-node cluster

Clustering Configurations

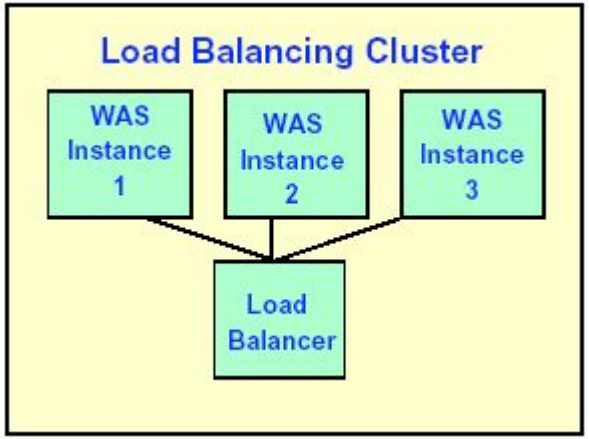
- No load balancing is done in this configuration however



Simple two-node cluster

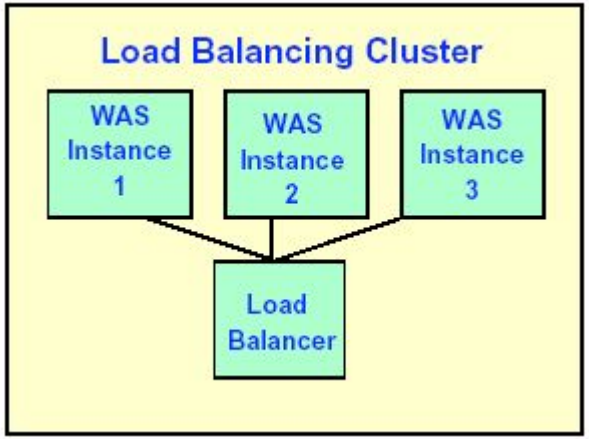
Clustering Configurations

- To load balance a cluster requires, well, a load balancing layer as well
- Note that all clustered servers are *peers*
  - This implies data replication between all nodes
  - Much more complex!
    - Learn more in networking



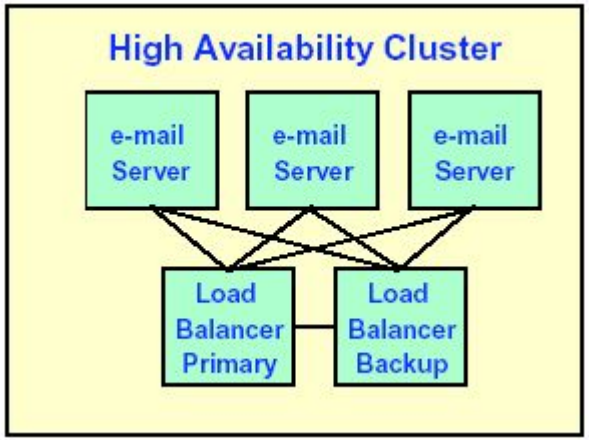
Clustering Configurations

- We still have a single point of failure in this cluster setup: the load balancer
- To achieve a high availability we need to remove this single point of failure



Clustering Configurations

- Now we have the previous primary/backup architecture on the load balancer level with peer clustering among servers
- This is a *High Availability* cluster



MySQL Clustering

- MySQL is a popular open source database
  - Purchased and managed by Oracle
- MySQL Cluster
  - Clustering solution for MySQL
  - Originally developed by Ericsson for the telecom world

MySQL Clustering

- “Shared Nothing” Architecture

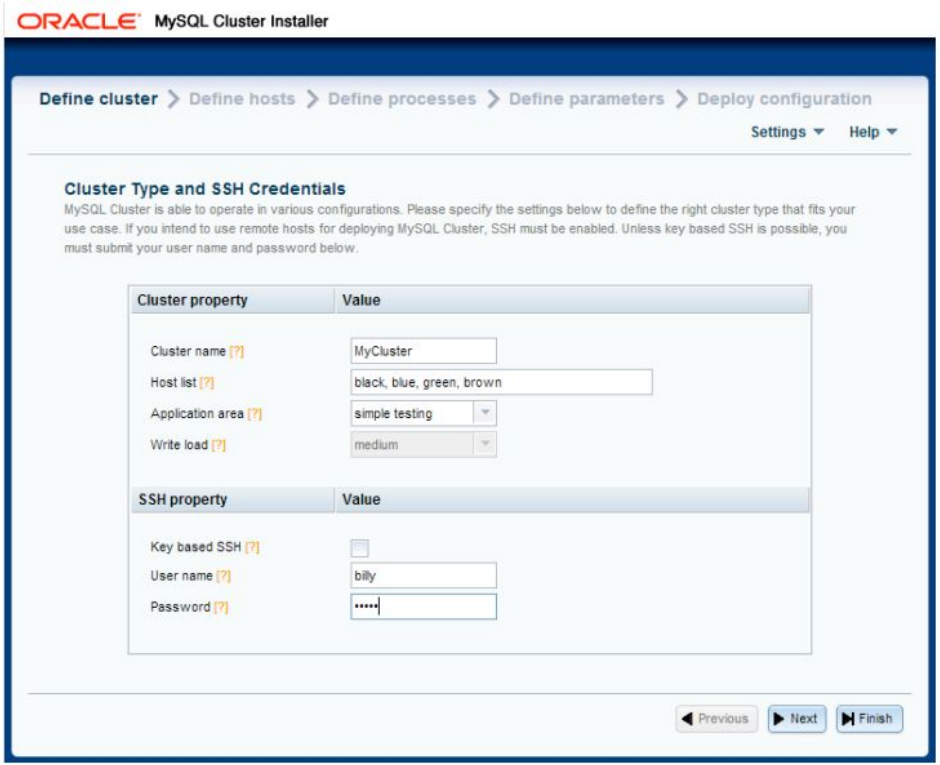
MySQL Cluster is designed to have no single point of failure. Provided that the cluster is set up correctly, any single node, system, or piece of hardware can fail without the entire cluster failing. Shared disk (SAN) is not required. The interconnects between nodes can be standard Ethernet, Gigabit Ethernet, InfiniBand, or SCI interconnects.

MySQL Clustering

- Multi-Master
  - Any node in the cluster can accept writes
  - Data is partitioned redundantly across entire cluster transparently
  - Uses optimistic concurrency control for replication

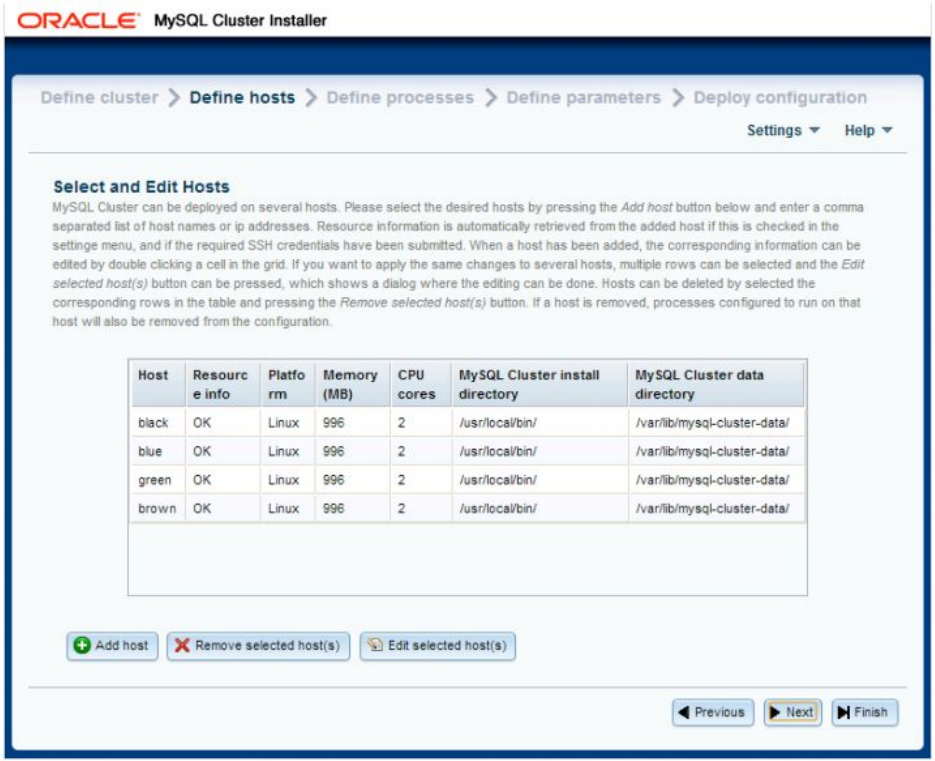
MySQL Clustering

- Installation
  - Step 1
    - Set up hosts
    - Set up cluster type
    - Configure SSH key access between servers



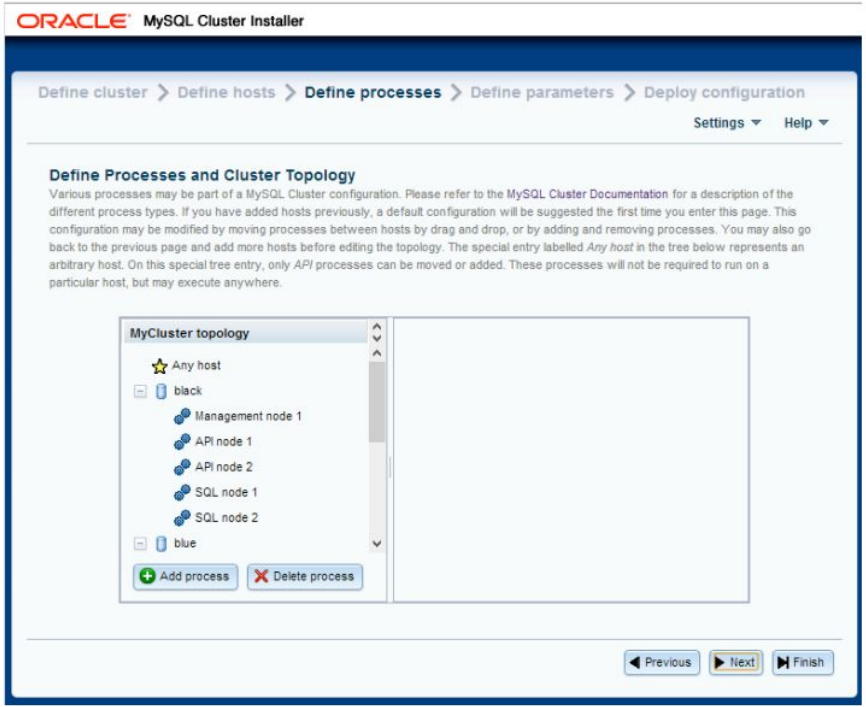
MySQL Clustering

- Installation
  - Step 2
    - Configure individual host machines
    - Recall, multi-master so each node can act as its own master



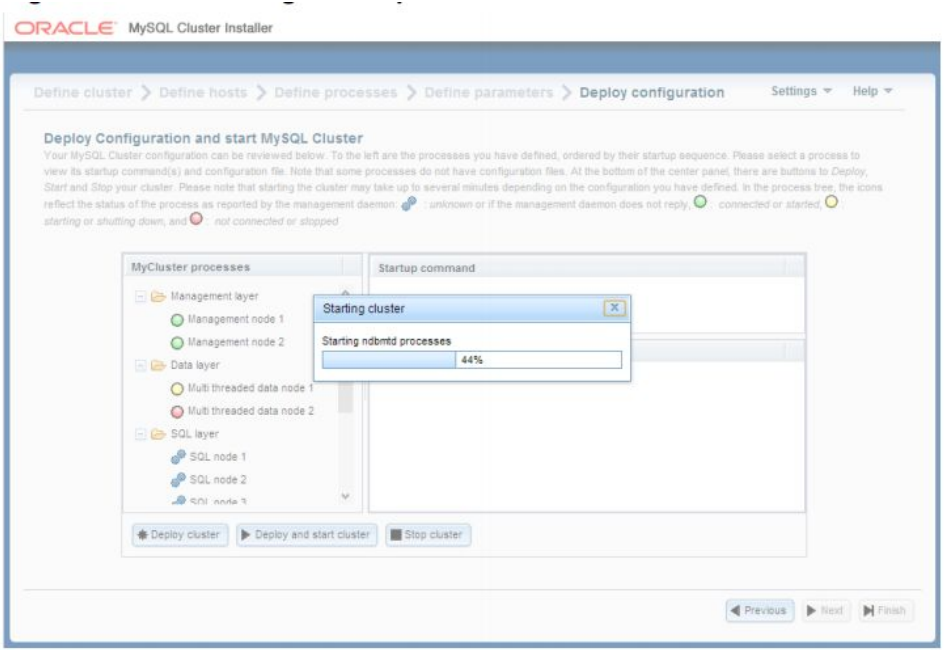
MySQL Clustering

- Installation
  - Step 3
    - Configure “Topology” of the cluster
    - Determine which nodes will fill which roles within the cluster, etc



MySQL Clustering

- NB: Cluster creation & management is typically done by a DBA or DevOps, not by developers
- As a developer, the cluster is typically hidden from you behind an abstraction such as JDBC



Redis Clustering

- Redis also offers clustering
- This redis.conf file enables clustering in an instance
  - cluster-enable enables the clustering technology
- Command line starts the cluster
  - 6 total servers
  - 3 masters, 3 replicas

```
port 7000
cluster-enabled yes
cluster-config-file nodes.conf
cluster-node-timeout 5000
appendonly yes
```

```
redis-cli --cluster create 127.0.0.1:7000 127.0.0.1:7001 \
127.0.0.1:7002 127.0.0.1:7003 127.0.0.1:7004 127.0.0.1:7005 \
--cluster-replicas 1
```

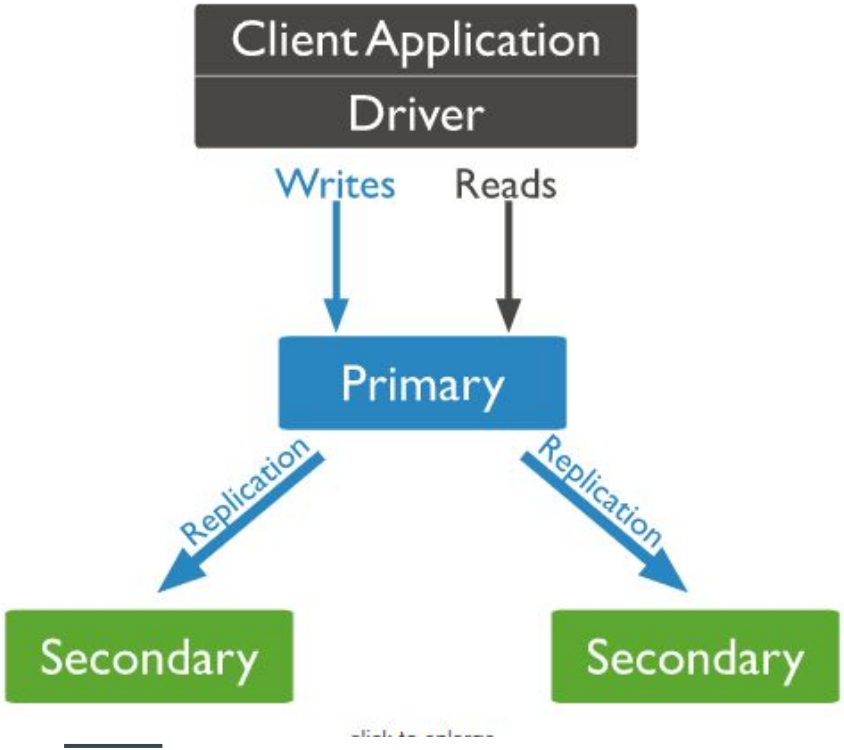
Redis Clustering

- Connecting to the clustered redis instance will now distribute data across multiple servers
- This data is *sharded*
  - We will discuss sharding in more detail next time

```
$ redis-cli -c -p 7000
redis 127.0.0.1:7000> set foo bar
-> Redirected to slot [12182] located at 127.0.0.1:7002
OK
redis 127.0.0.1:7002> set hello world
-> Redirected to slot [866] located at 127.0.0.1:7000
OK
redis 127.0.0.1:7000> get foo
-> Redirected to slot [12182] located at 127.0.0.1:7002
"bar"
redis 127.0.0.1:7000> get hello
-> Redirected to slot [866] located at 127.0.0.1:7000
"world"
```

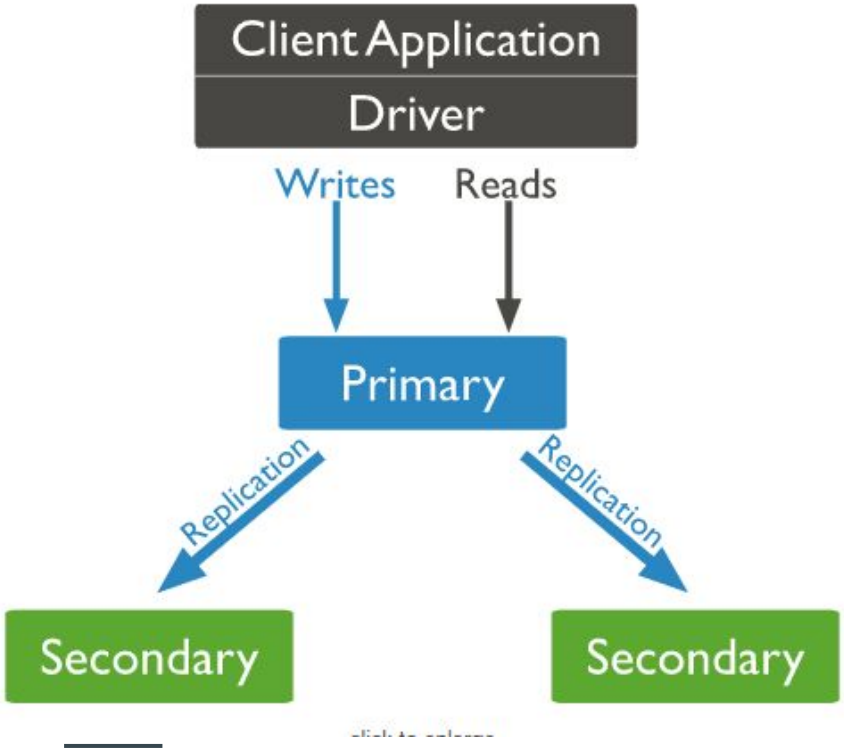
MongoDB Clustering

- MongoDB has a long history of offering clustering support
- One of the killer features early on in Mongo adoption



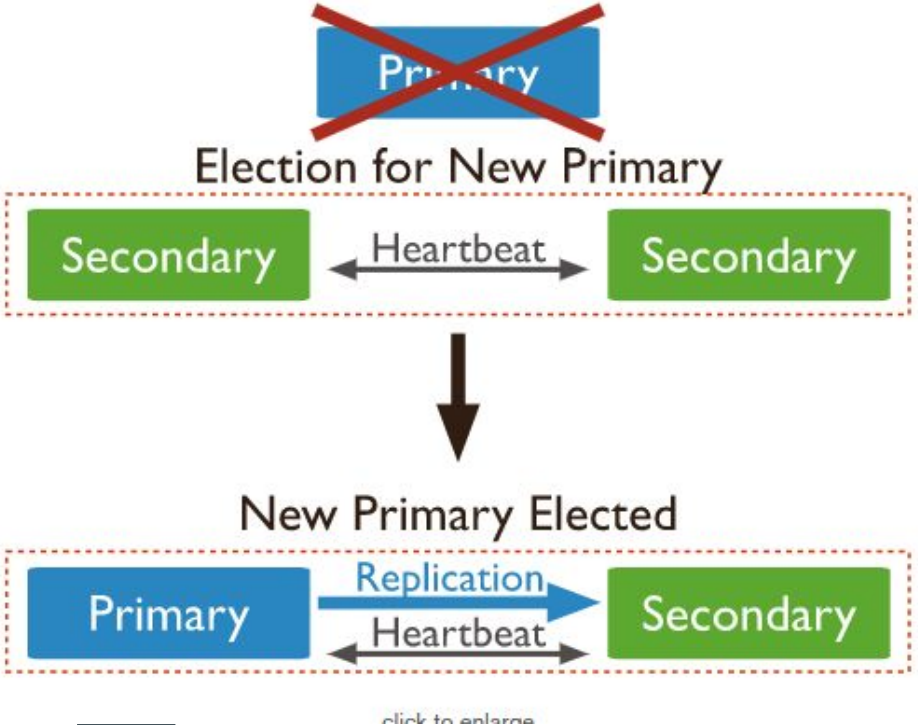
MongoDB Clustering

- MongoDB calls servers participating in a cluster as data stores as Data Bearing Nodes
- One Data Bearing Node is declared Primary Bides
- Other instances are Secondary Nodes



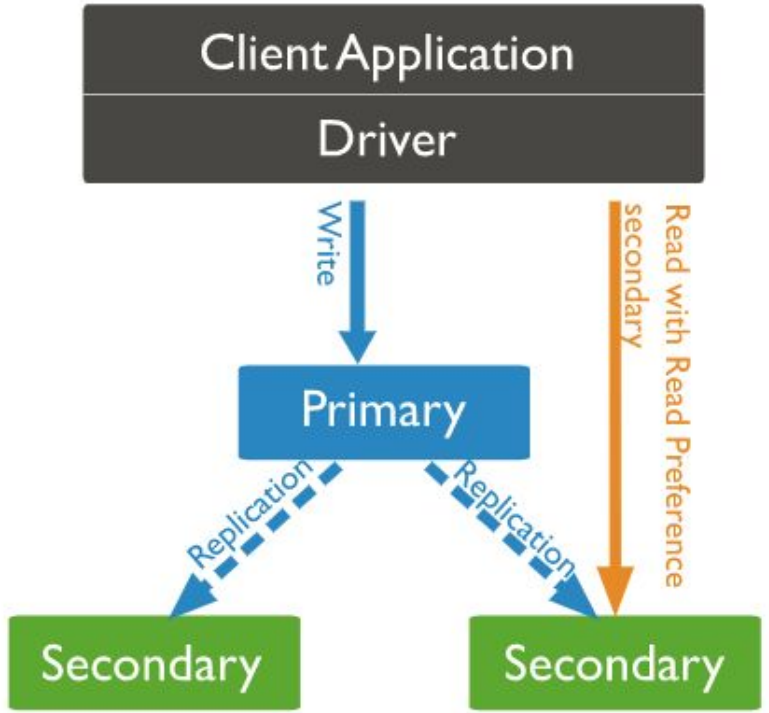
MongoDB Clustering

- In case of a failure of a primary node, an “election” is held to promote one secondary node to the new primary node



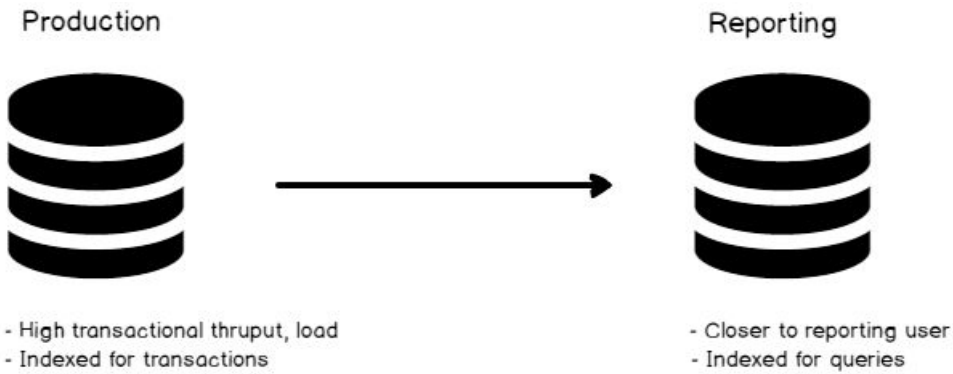
MongoDB Clustering

- Mongo clients can specify a read preference to send reads to the secondary database
  - May be faster than a read against the primary database and “good enough”



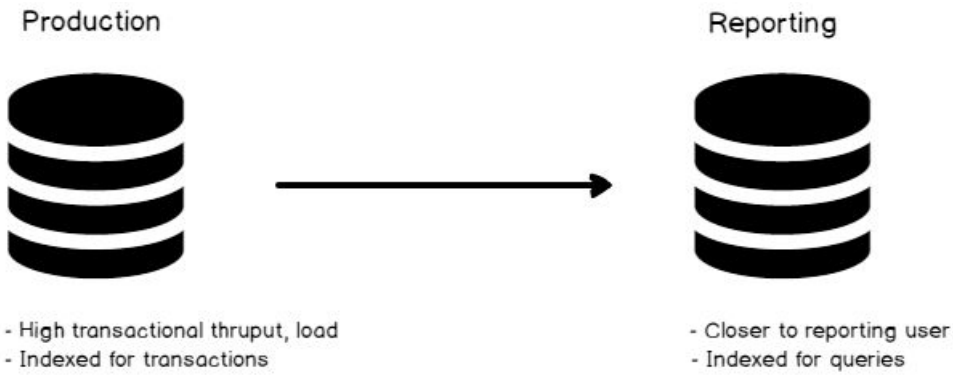
Reporting Database

- Not typically considered a cluster, but related
- Data is replicated from a primary database to a secondary reporting database
  - One way replication



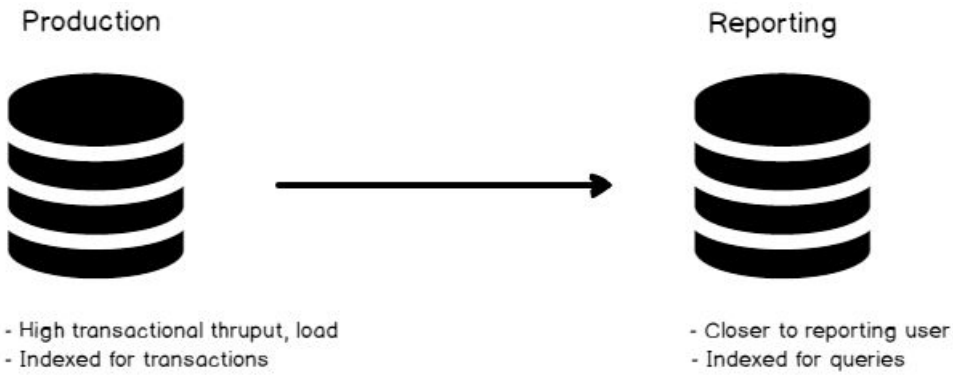
Reporting Database

- Production Database is tuned for high throughput
  - Indexed for application requirements
  - Expensive indexes may be omitted to maximize insert/update speeds



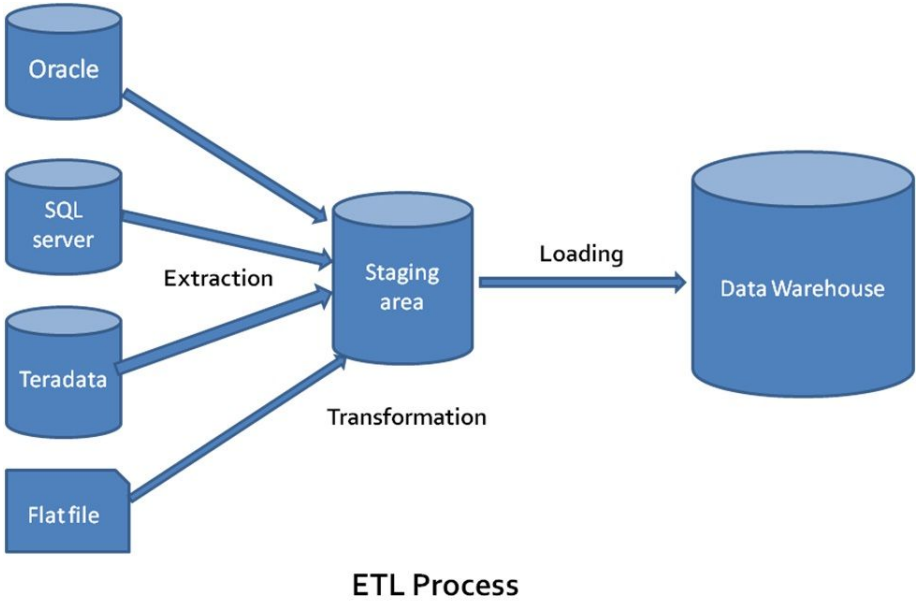
Reporting Database

- Reporting Database is tuned for reporting needs
  - May be physically closer to reporting users
  - Indexed for query speed rather than insert/update
  - Long running queries that take up lots of DB resources are OK



Data Warehouses

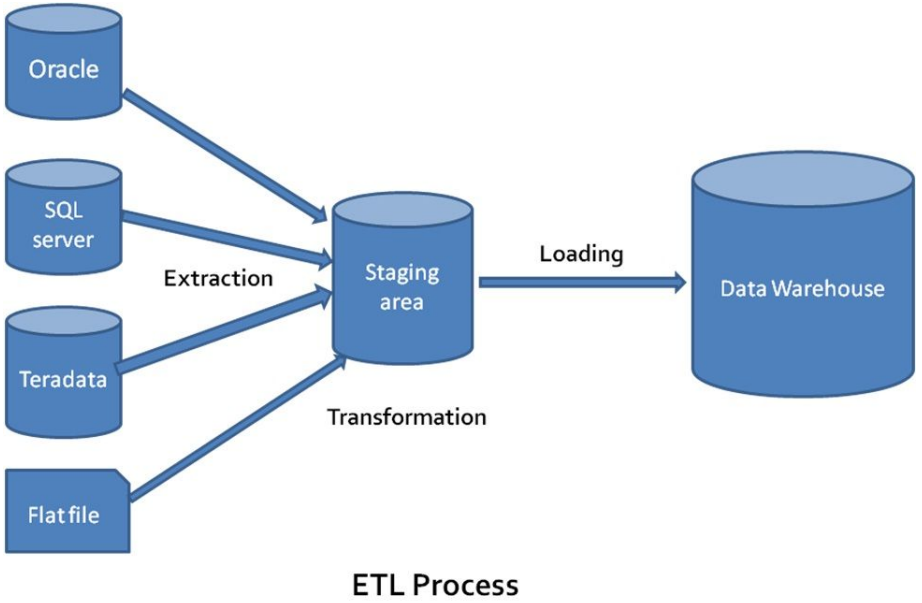
- Related to the concept of reporting databases is the notion of Data Warehouses
- A data warehouse is an aggregation of multiple data stores across an organization
- Typically used in Read Only mode for reporting



ETL Process

Data Warehouses

- ETL
  - Extract (pull data from servers)
  - Transform (standardize data)
  - Load (load into the data warehouse)
- ETL processes and tools are a major part of the operations of large companies



ETL Process

Clustering

- Today we discussed clustering
  - Using multiple machines to provide
    - Redundancy
    - Parallelism
    - Scaleability
- We looked at clustering in MySQL, Redis & MongoDB
- We discussed Reporting DBs
  - Optimized for reporting needs, allow production database to optimize for throughput
- Finally, we discussed Data Warehouses


MONTANA
STATE UNIVERSITY