WEBARCH 253: Storage Systems

Peter Bailis
pbailis@cs.berkeley.edu
26 October 2012

Who am I?

Peter Bailis
Getting Ph.D. in Computer Science @ Cal
Study distributed systems and databases

Who am I?

Peter Bailis
Getting Ph.D. in Computer Science @ Cal
Study distributed systems and databases
[complicated games of Telephone]

Who am I?

Peter Bailis
Getting Ph.D. in Computer Science @ Cal
Study distributed systems and databases
[complicated games of Telephone]

Appreciate interruptions and questions!

Who are you?

Who are you?

Engineering?

Who are you?

Engineering?

Product?

Who are you?

Engineering?

Product?

Management?

Who are you?

Engineering?

Product?

Management?

Legal?

Who are you?

Engineering?

Product?

Management?

Legal?

Entrepreneur?

Who are you?

Engineering?

Product?

Management?

Legal?

Entrepreneur?

Thought leader?

Who are you?

Engineering?

Product?

Management?

Legal?

Entrepreneur?

Thought leader?

Other?

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

Why data storage?

Why data storage?

- When we make a web request, where do we get the data from?
- When we create data, where do we put it?
- Where do “resources” live?

bit.ly clone

bit.ly clone

- Lots of data to store
  - shortcut to url mapping
  - statistics about links
  - information about users

bit.ly clone

long url	http://news.google.com
short url	http://bit.ly/awekl
hit count	482240

long url	http://facebook.com/user/profile/
short url	http://bit.ly/czasw
hit count	11023

long url	http://msnbc.com/news/article/
short url	http://bit.ly/olkjpl
hit count	1232

Data Storage Design

Data Storage Design

- What is the storage format?
- How do we lay out data?
- How do we access data?

Why use a file?

Why use a file?

```
http://news.google.com, http://bit.ly/awekl, 482240

http://facebook.com/user/profile/id/..., http://bit.ly/czasw, 11023

http://msnbc.com/news/article/12/20/..., http://bit.ly/olkjpl, 1232

...
```

Why use a file?

```
http://news.google.com, http://bit.ly/awekl, 482240

http://facebook.com/user/profile/id/..., http://bit.ly/czasw, 11023

http://msnbc.com/news/article/12/20/..., http://bit.ly/olkjpl, 1232

...
```

Pros?

Why use a file?

```
http://news.google.com, http://bit.ly/awekl, 482240

http://facebook.com/user/profile/id/..., http://bit.ly/czasw, 11023

http://msnbc.com/news/article/12/20/..., http://bit.ly/olkjpl, 1232

...
```

Pros?

Cons?

Problems with Files

- What if we want to add another field?
- What if we want to query different parts of data? How efficient is this?
- What if we have concurrent accesses?
- What data structures should we use?

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

Data Independence

- Databases: apps shouldn't have to worry about these problems!
- Underlying storage format independent of application-level logic

Relational Data Stores

- RDBMS

Relational Data Stores

- **RDBMS:** Relational Database Management System
- Invented in the 1970s
- e.g., Oracle, MySQL, Postgres, IBM DB2, Microsoft SQL Server

Relational Model

- Reason about sets of facts, or "tables"
  - Each fact is a "row"
- Attributes are "columns" of row

long_url	short_url	hit_count
http://news.google.com	awekl	482240
http://facebook.com/user/	czasw	11023
http://msnbc.com/news/article/	olkjpl	1232

long_url	short_url	hit_count
http://news.google.com	awekl	482240
http://facebook.com/user/	czasw	11023
http://msnbc.com/news/article/	olkjpl	1232

relation

long_url	short_url	hit_count
http://news.google.com	awekl	482240
http://facebook.com/user/	czasw	11023
http://msnbc.com/news/article/	olkjpl	1232

row

relation

column

long_url	short_url	hit_count
http://news.google.com	awekl	482240
http://facebook.com/user/	czasw	11023
http://msnbc.com/news/article/	olkjpl	1232

row

relation

SQL Query Language

- High-level query language over tables
- Declarative: say "what" you want computed, not "how"

SQL Query Language

- High-level query language over tables
- Declarative: say "what" you want computed, not "how"

Why is this good?

SELECT example

SELECT example

```
mysql> SELECT * from links;
+--------------------------+-----------------------+-----------+------------+
| long_url                 | short_url             | hit_count | created    |
+--------------------------+-----------------------+-----------+------------+
| http://www.google.com    | qwelmw                | 2         | 2012-10-19 |
| http://www.facebook.com  | adfer                 | 45        | 2012-10-22 |
+--------------------------+-----------------------+-----------+------------+
```

SELECT example

SELECT example

```
mysql> SELECT * from links WHERE hit_count < 20;
+----------------------+----------------------+-----------+------------+
| long_url             | short_url            | hit_count | created    |
+----------------------+----------------------+-----------+------------+
| http://www.google.com | qwelmw               | 2         | 2012-10-19 |
+----------------------+----------------------+-----------+------------+
1 row in set (0.00 sec)
```

INSERT example

INSERT example

```
mysql> insert into links VALUES ("http://www.twitter.com", "eovle", 0,
CURDATE());
Query OK, 1 row affected (0.00 sec)
```

UPDATE example

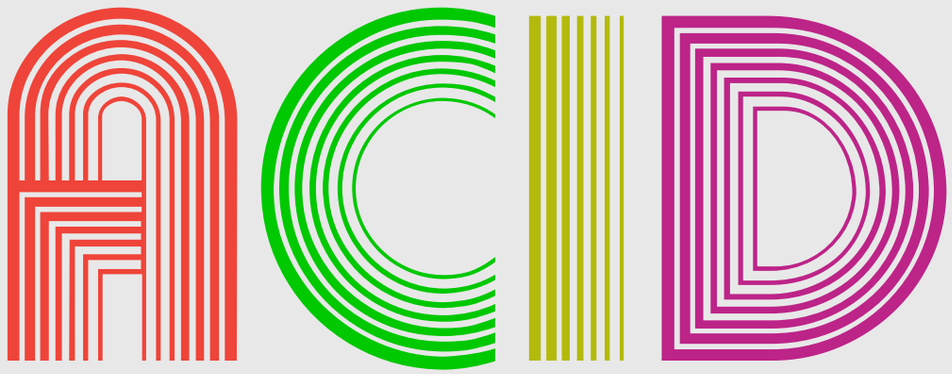
UPDATE example

```
mysql> UPDATE links SET hit_count = '0' WHERE created > '2012-10-22';Query OK,
23 rows affected (0.00 sec)

Rows matched: 23  Changed: 23  Warnings: 0
```

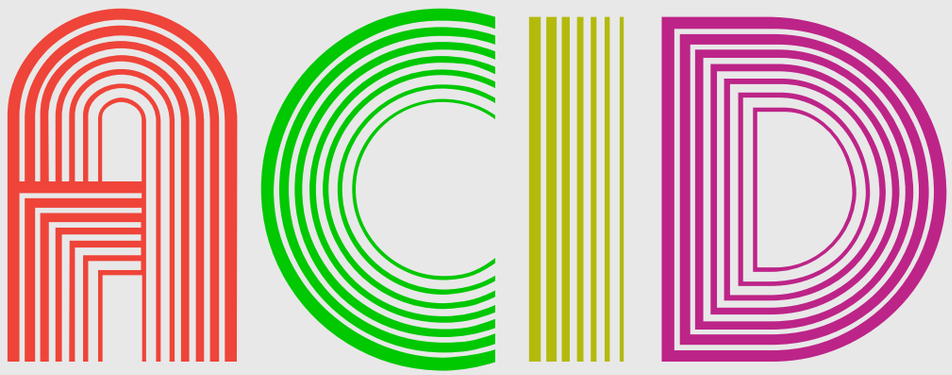
Useful Properties

Useful Properties



Useful Properties

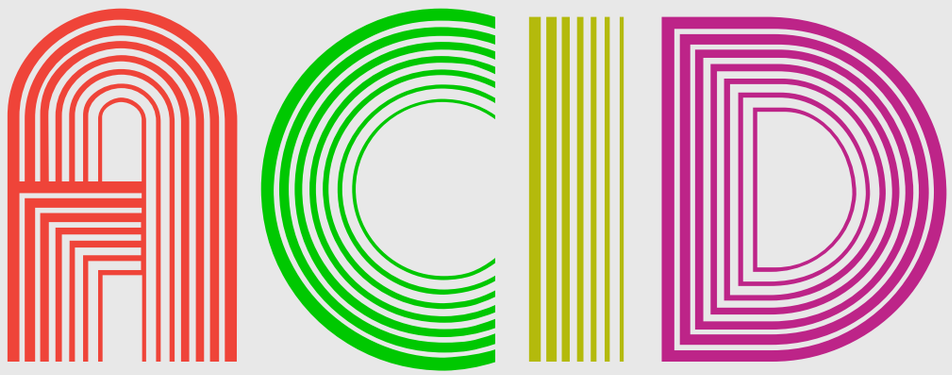
Atomicity: all updates happen or none do



Useful Properties

Atomicity: all updates happen or none do

Consistency: easy to reason about database

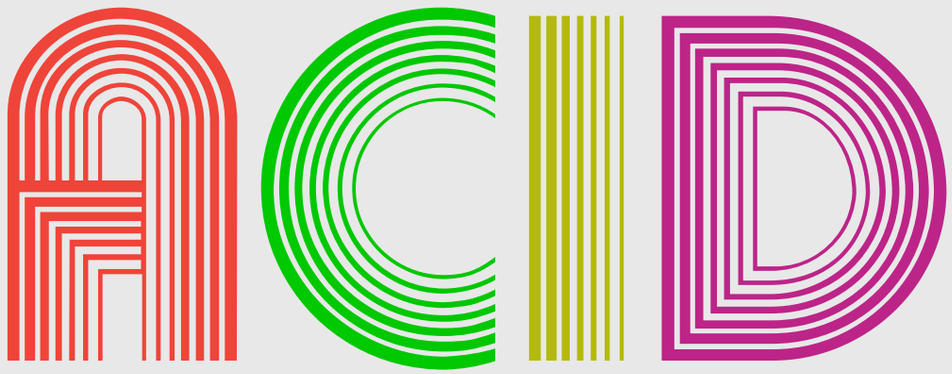


Useful Properties

Atomicity: all updates happen or none do

Consistency: easy to reason about database

Isolation: operations are separated from each other



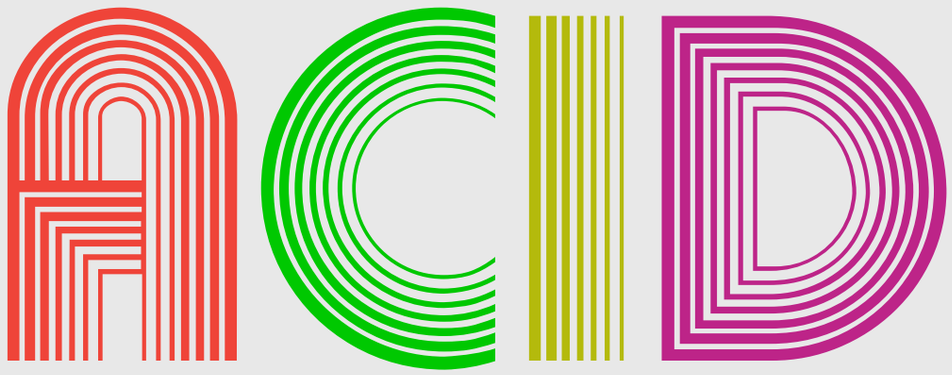
Useful Properties

Atomicity: all updates happen or none do

Consistency: easy to reason about database

Isolation: operations are separated from each other

Durability: updates won't disappear



RDBMS Pros and Cons

RDBMS Pros and Cons

PROs

RDBMS Pros and Cons

PROs

- High-level query language
- Data independence
- Isolation of users

RDBMS Pros and Cons

PROs

- High-level query language
- Data independence
- Isolation of users

CONs

RDBMS Pros and Cons

PROs

- High-level query language
- Data independence
- Isolation of users

CONs

- Have to define schema at start
- Few open source multi-server implementations
- Often require complex tuning

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

IRL Data Management

IRL Data Management

Organizational roles often dedicated
to “looking after the data” (e.g., DBA)

IRL Data Management

Organizational roles often dedicated
to "looking after the data" (e.g., DBA)

PROs

IRL Data Management

Organizational roles often dedicated to "looking after the data" (e.g., DBA)

PROs

data is "safe"

protect institutional interests

sharing of data is controlled

IRL Data Management

Organizational roles often dedicated to "looking after the data" (e.g., DBA)

PROs

data is "safe"

protect institutional interests

sharing of data is controlled

CONs

IRL Data Management

Organizational roles often dedicated to "looking after the data" (e.g., DBA)

PROs

data is "safe"

protect institutional interests

sharing of data is controlled

CONs

slow changes (e.g., schemas)

more overhead

"feral databases"

Let's Pretend...

Let's Pretend...

you run a supermarket:

Let's Pretend...

you run a supermarket:

1. you store your data in a database
   what do you put in it?

Let's Pretend...

you run a supermarket:

1. you store your data in a database
   what do you put in it?

2. you lose all of your data
   what breaks?

Let's Pretend...

you run a supermarket:

1. you store your data in a database
   what do you put in it?

2. you lose all of your data
   ~~what breaks?~~ what **doesn't** break?

Let's Pretend...

you run a supermarket:

1. you store your data in a database
   what do you put in it?

2. you lose all of your data
   ~~what breaks?~~ what **doesn't** break?

3. how costly is it to restore your data?

Let's Pretend...

you run a supermarket:

1. you store your data in a database
   what do you put in it?

2. you lose all of your data
   ~~what breaks?~~ what **doesn't** break?

3. how costly is it to restore your data?
   what can('t) be restored?

Reliability

Companies pay for databases
not just for the database

Reliability

Companies pay for databases
not just for the database

Pay because data has
immense business value

Reliability

Companies pay for databases
not just for the database

Pay because data has
immense business value

$$$ buys

insurance + someone to yell at on the phone

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

Typical Web Architecture

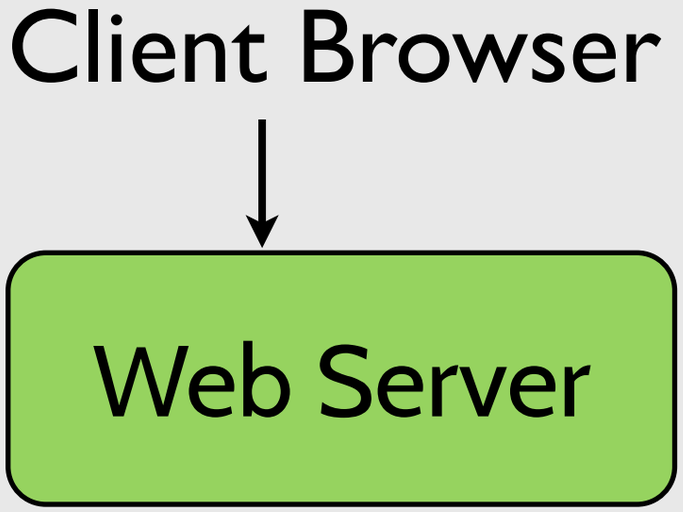
Typical Web Architecture

Client Browser

Typical Web Architecture

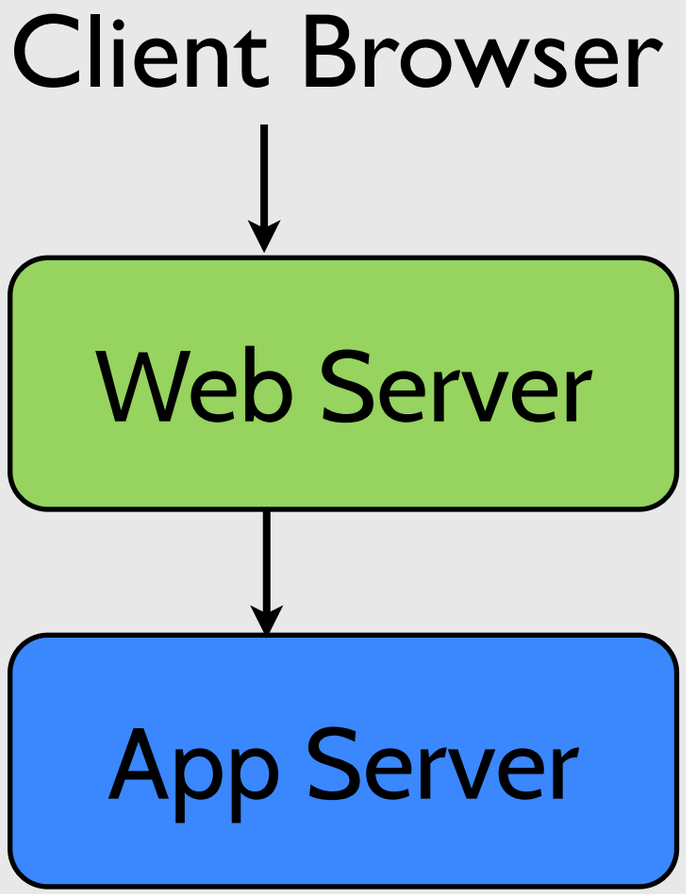
Client Browser

Typical Web Architecture



Accepts request from client

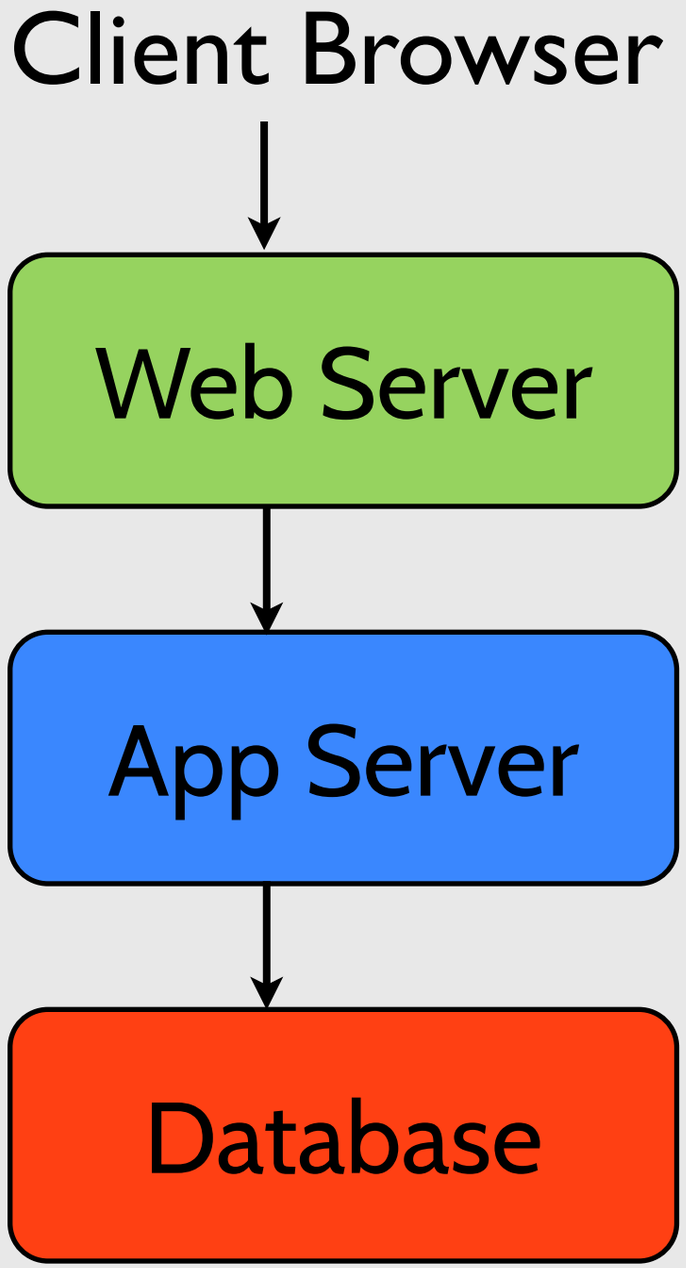
Typical Web Architecture



Accepts request from client

Decides what to fetch and how to present it

Typical Web Architecture

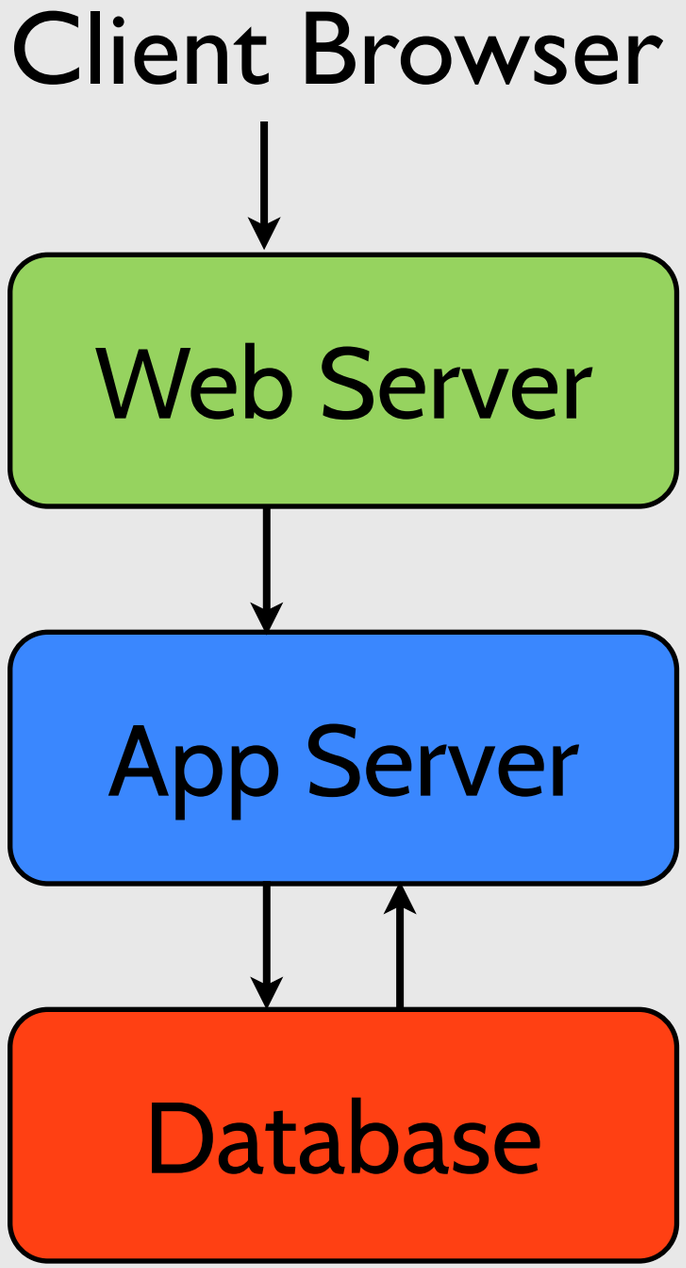


Accepts request from client

Decides what to fetch
and how to present it

Fetches data from storage

Typical Web Architecture

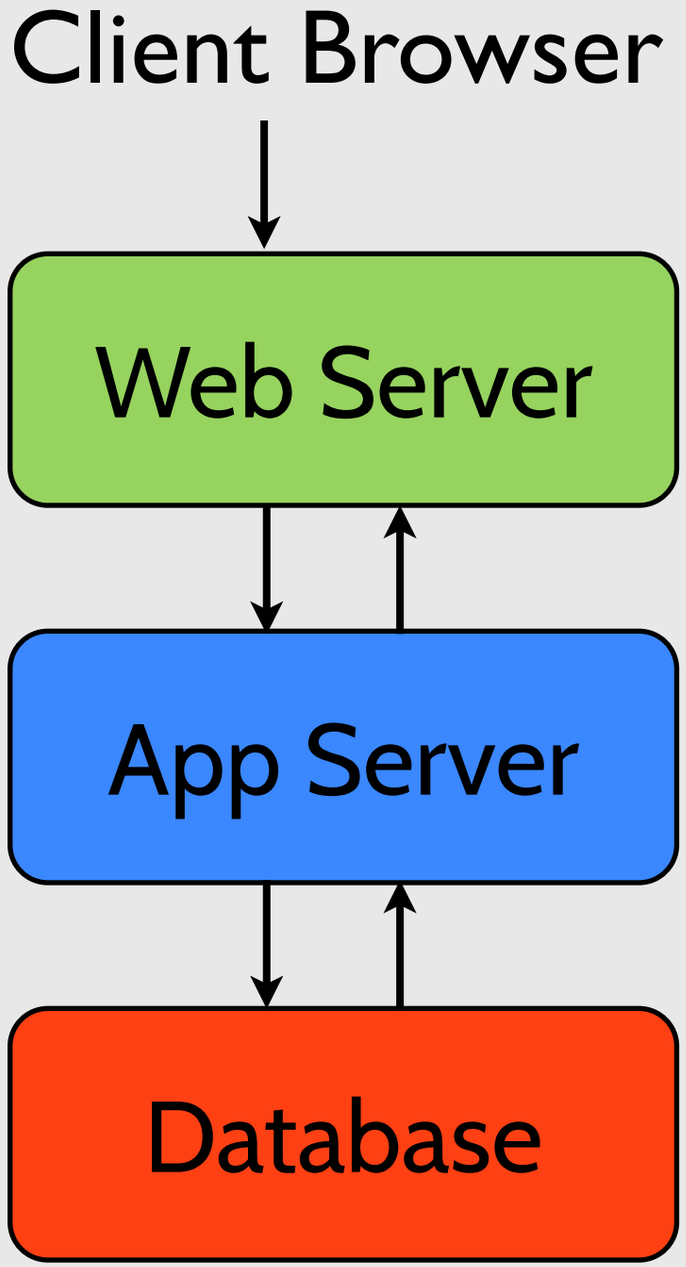


Accepts request from client

Decides what to fetch
and how to present it

Fetches data from storage

Typical Web Architecture

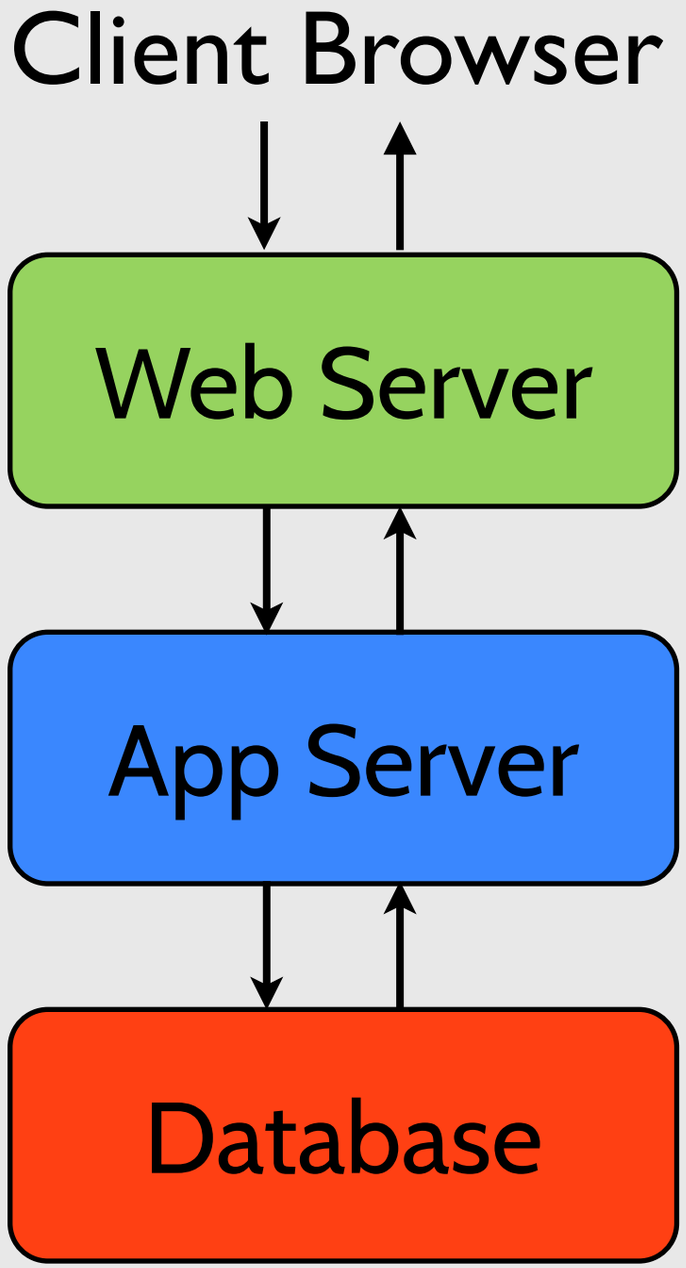


Accepts request from client

Decides what to fetch
and how to present it

Fetches data from storage

Typical Web Architecture



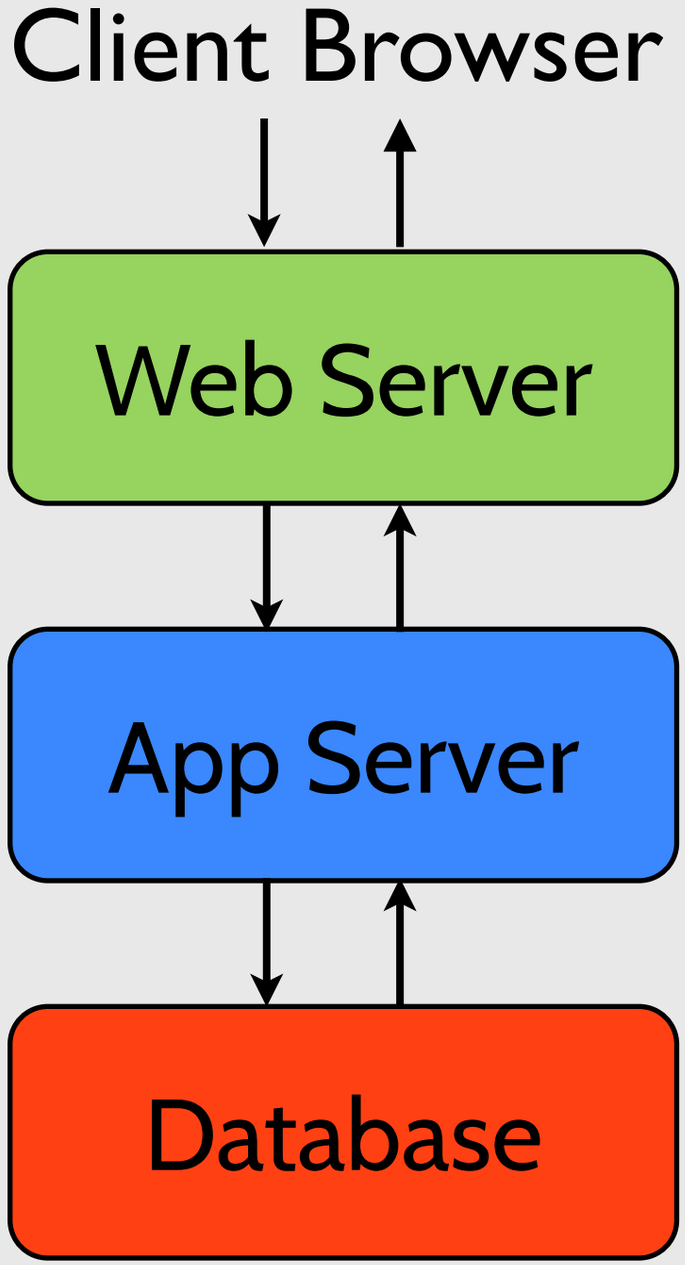
Accepts request from client

Decides what to fetch
and how to present it

Fetches data from storage

Typical Web Architecture

HTTP

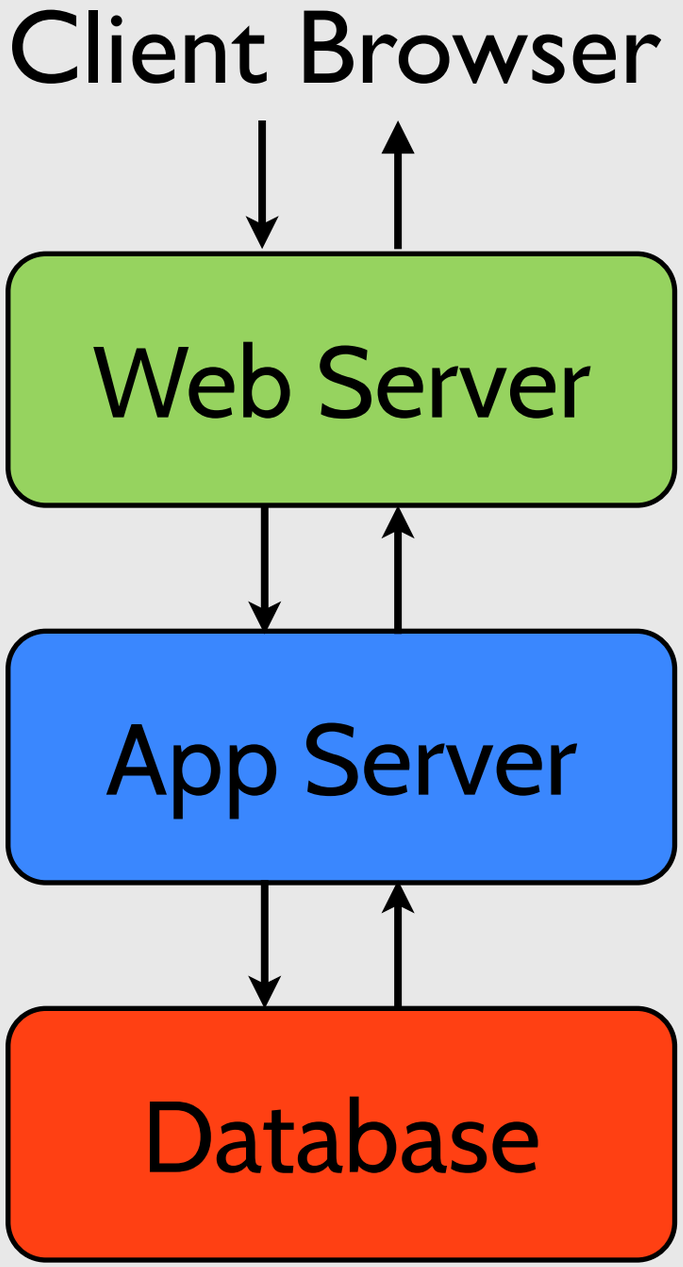


Accepts request from client

Decides what to fetch
and how to present it

Fetches data from storage

Typical Web Architecture

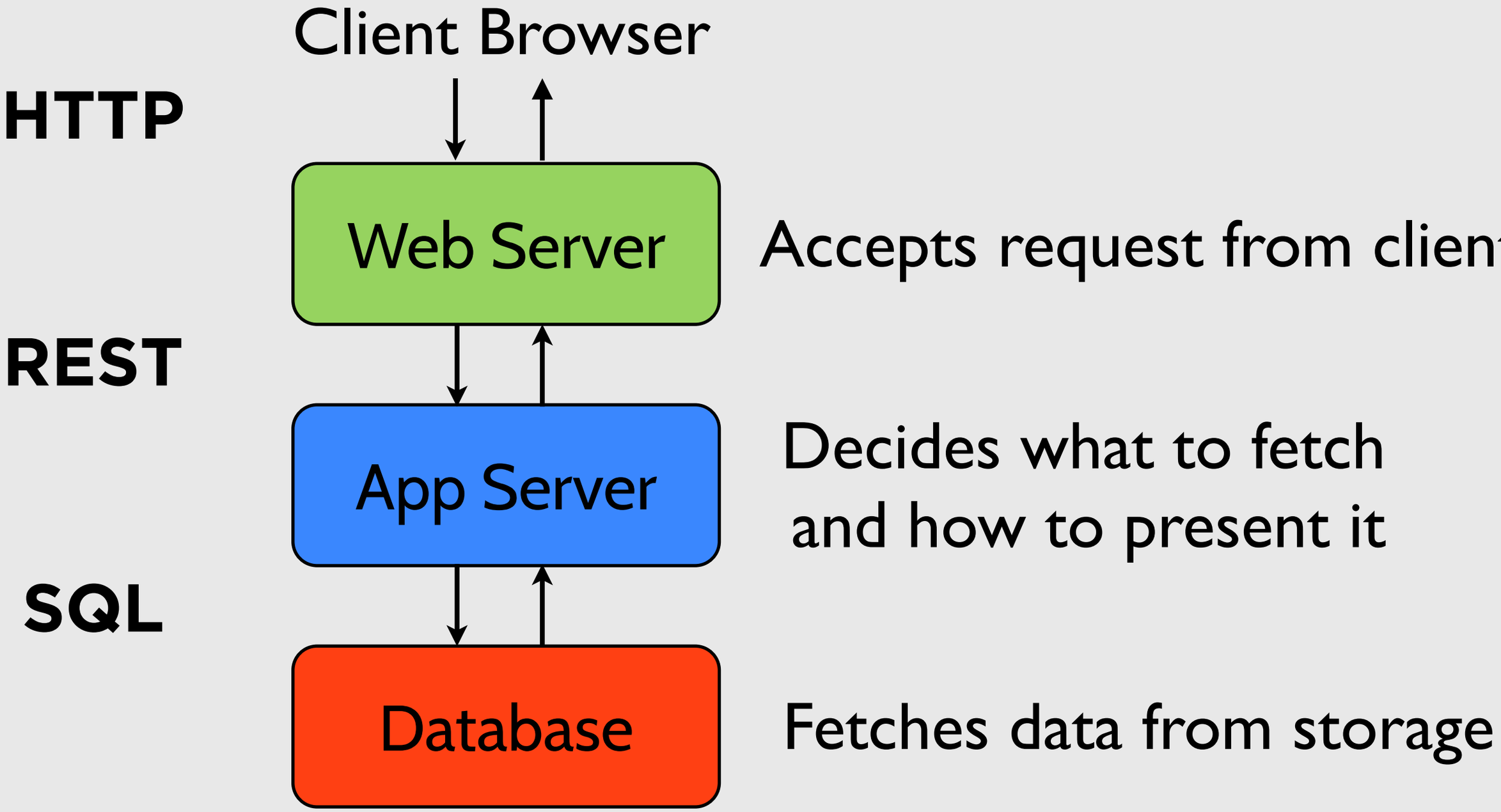


Accepts request from client

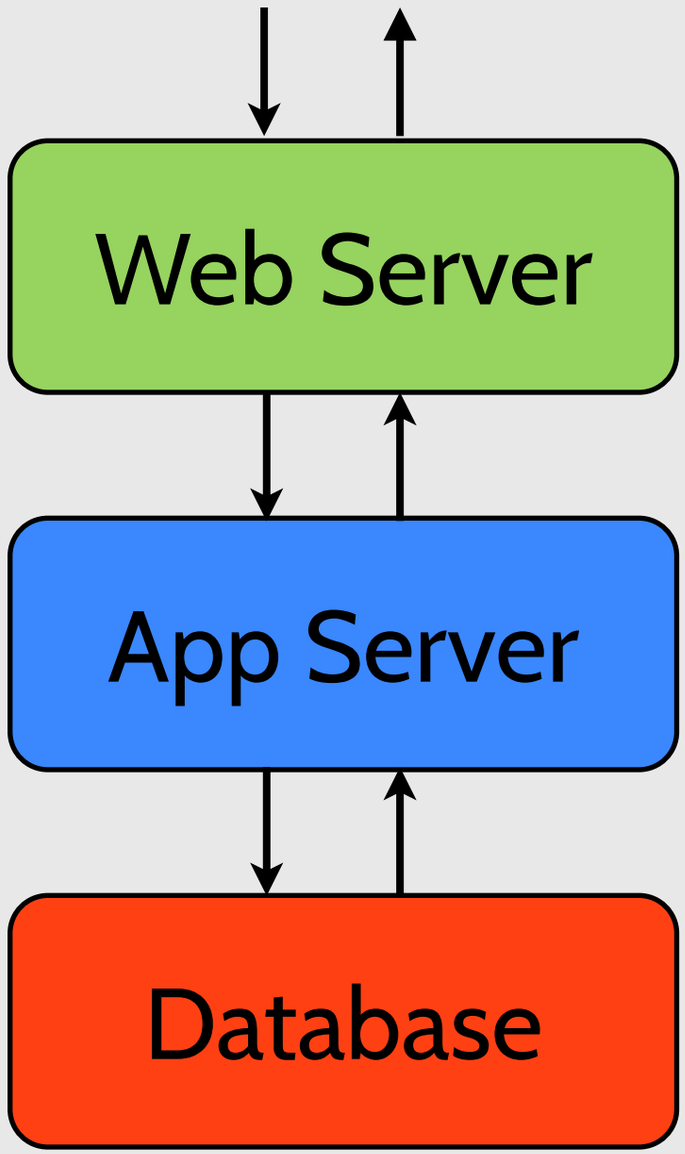
Decides what to fetch
and how to present it

Fetches data from storage

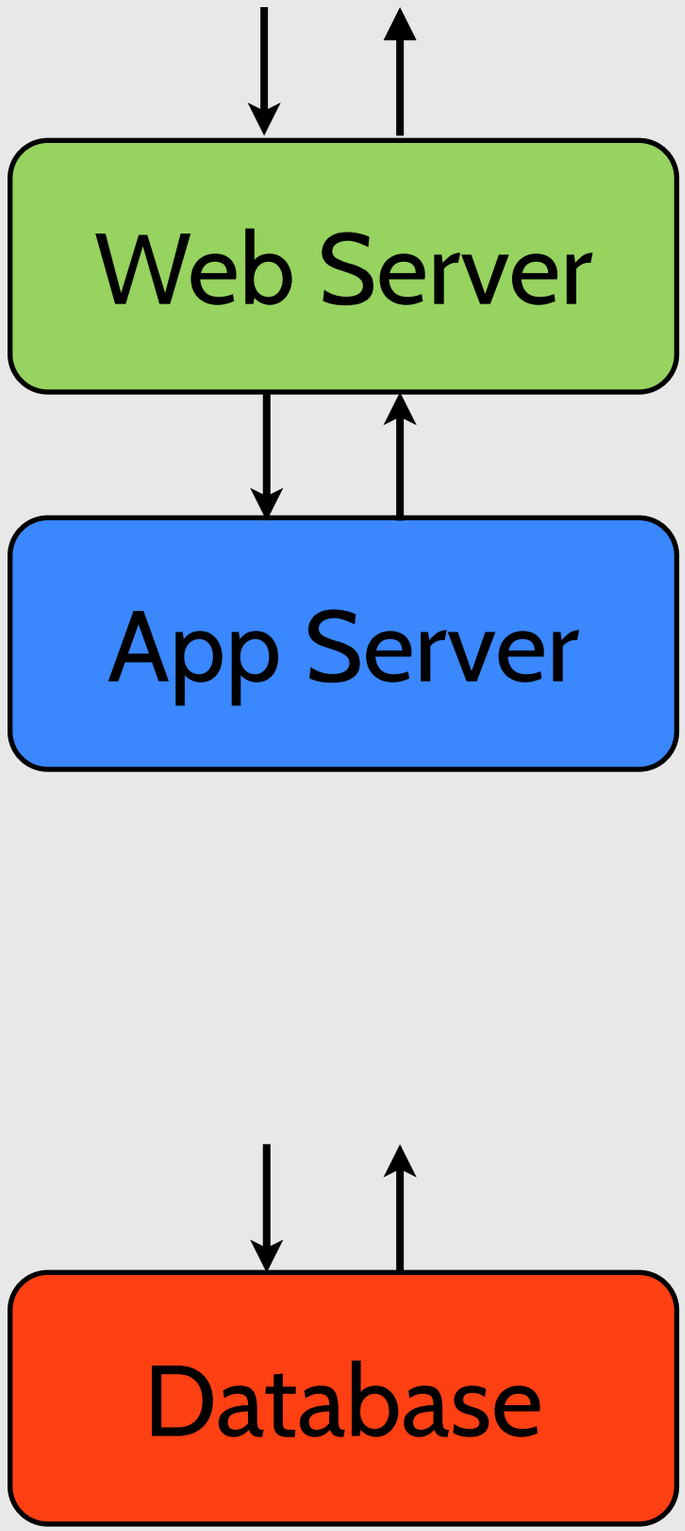
Typical Web Architecture



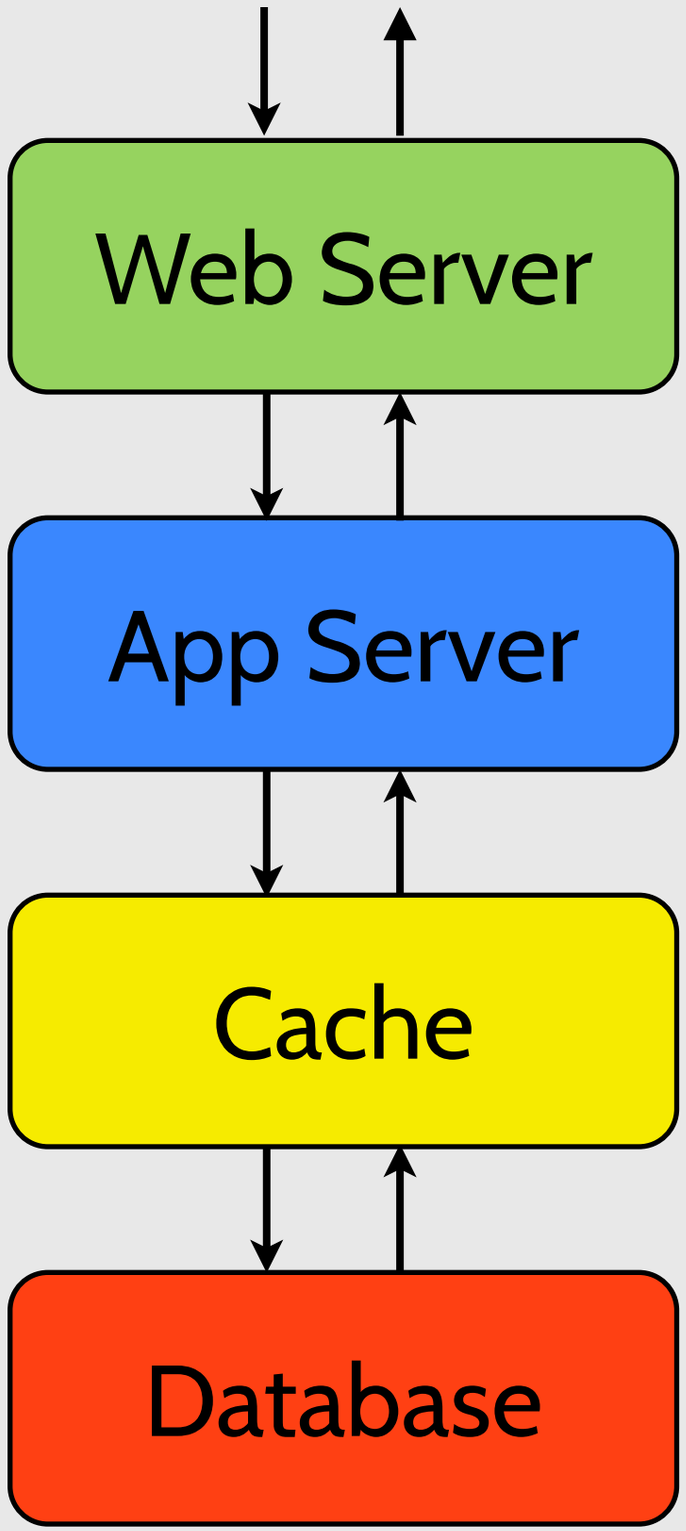
Caching



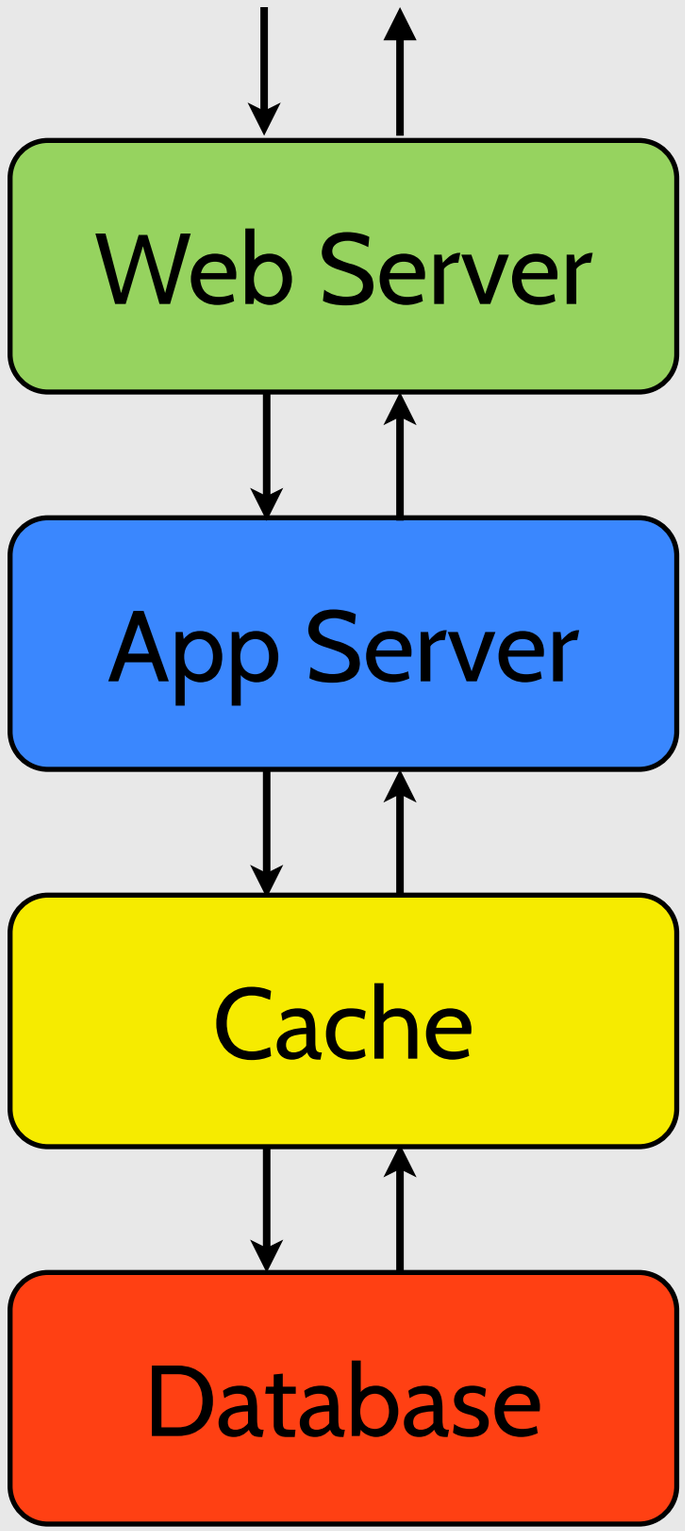
Caching



Caching

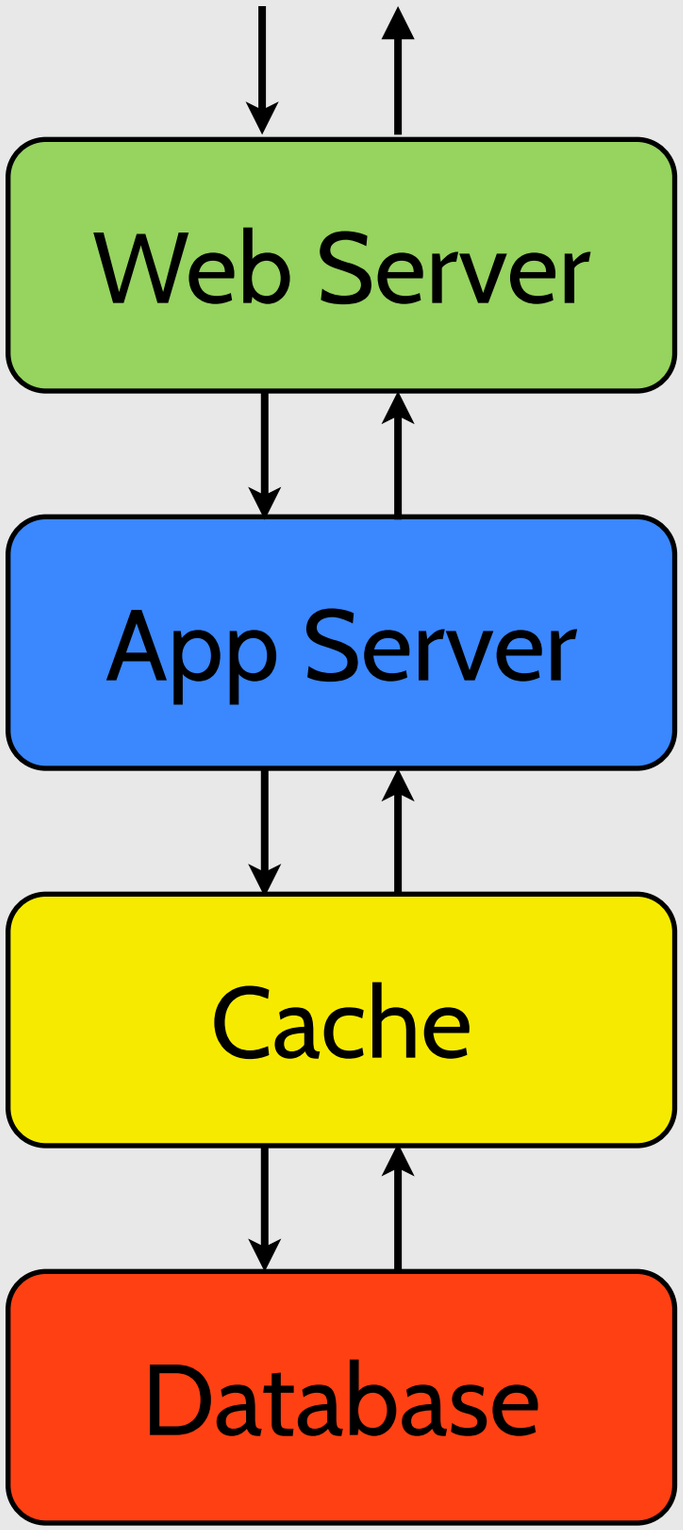


Caching



REST

Caching

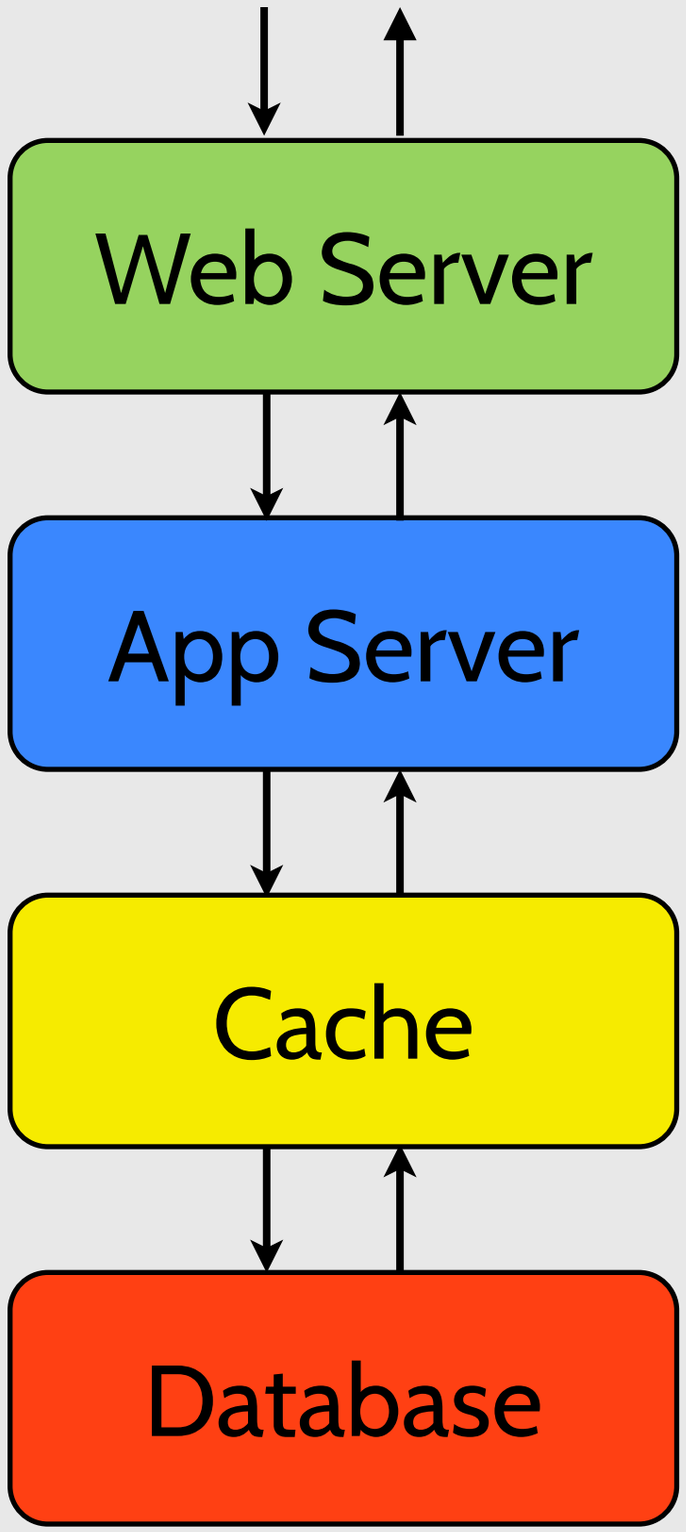


REST

SQL

Caching

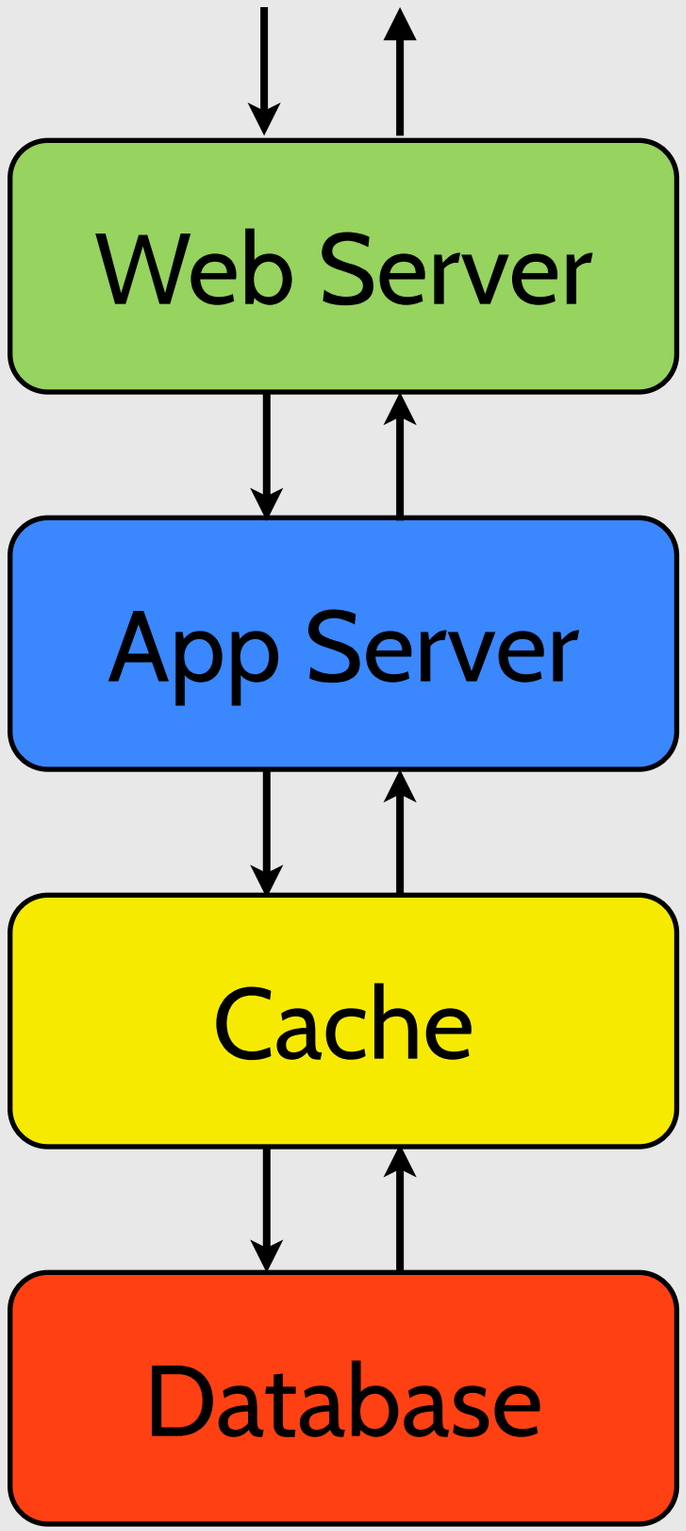
reduce load on database
by placing cheap copies
in front of DB



REST

SQL

Caching



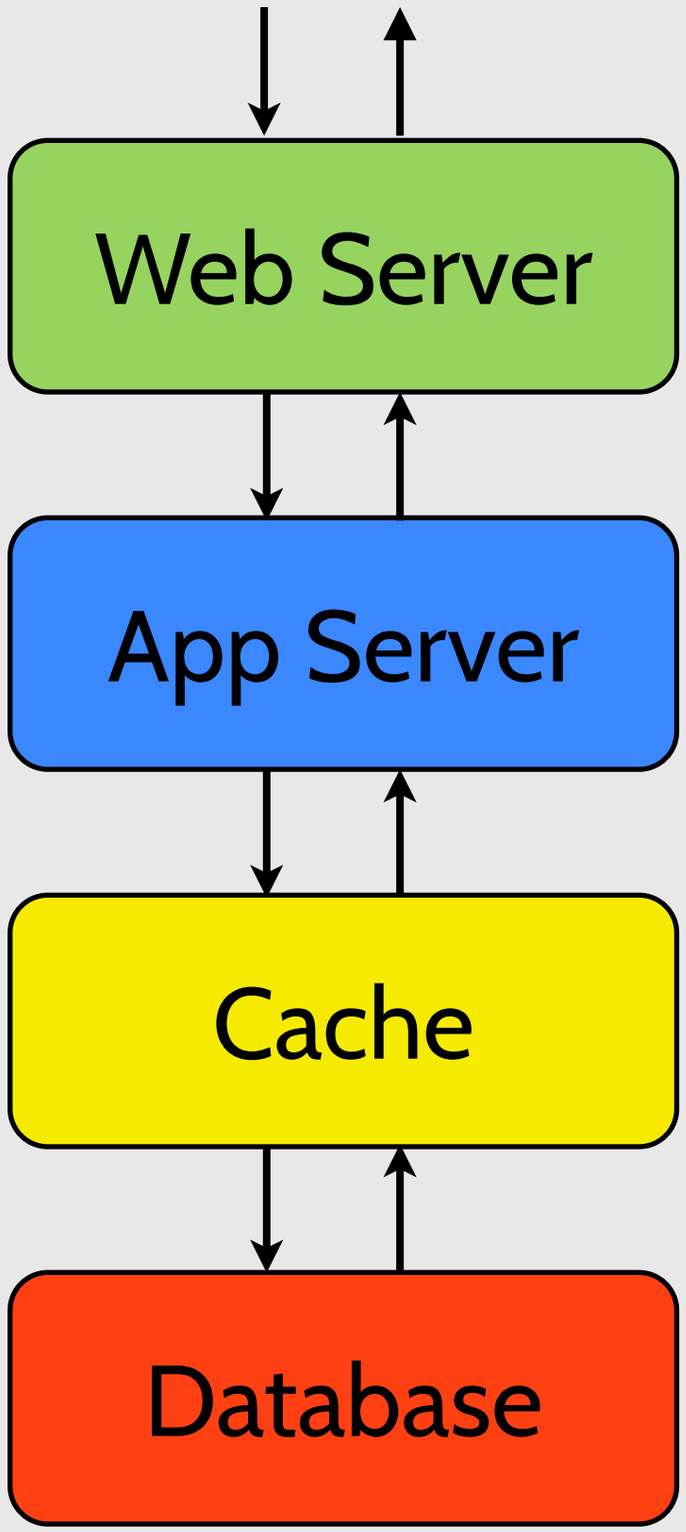
REST

SQL

reduce load on database
by placing cheap copies
in front of DB

problem?

Caching



REST

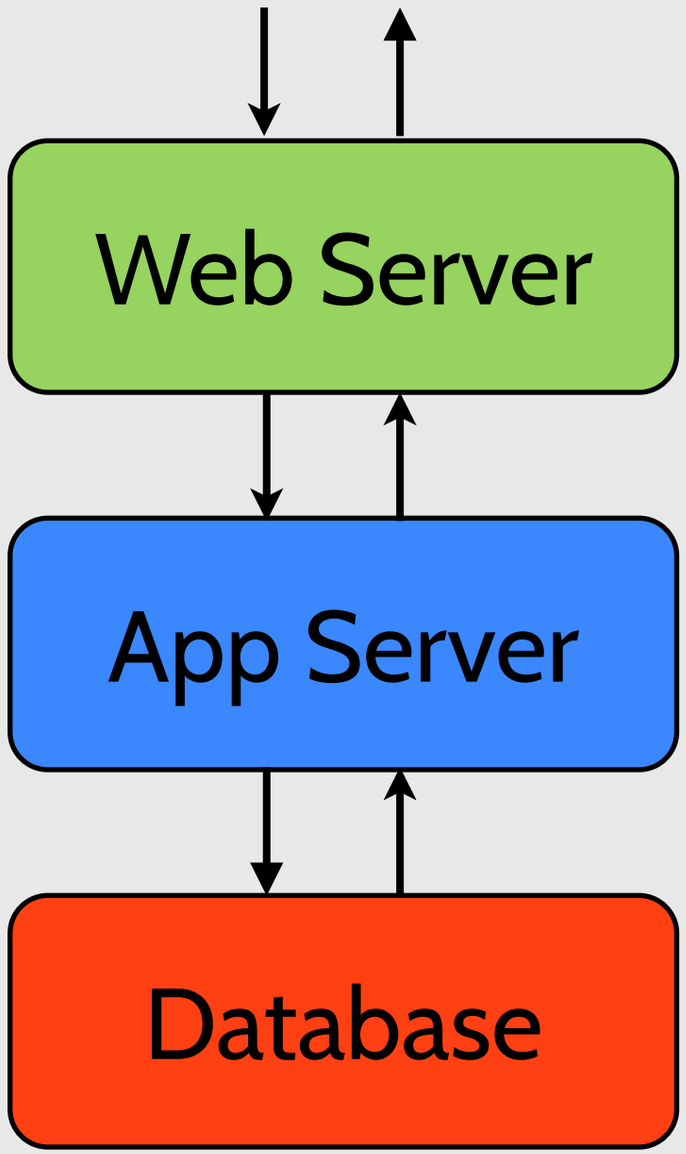
SQL

reduce load on database by placing cheap copies in front of DB

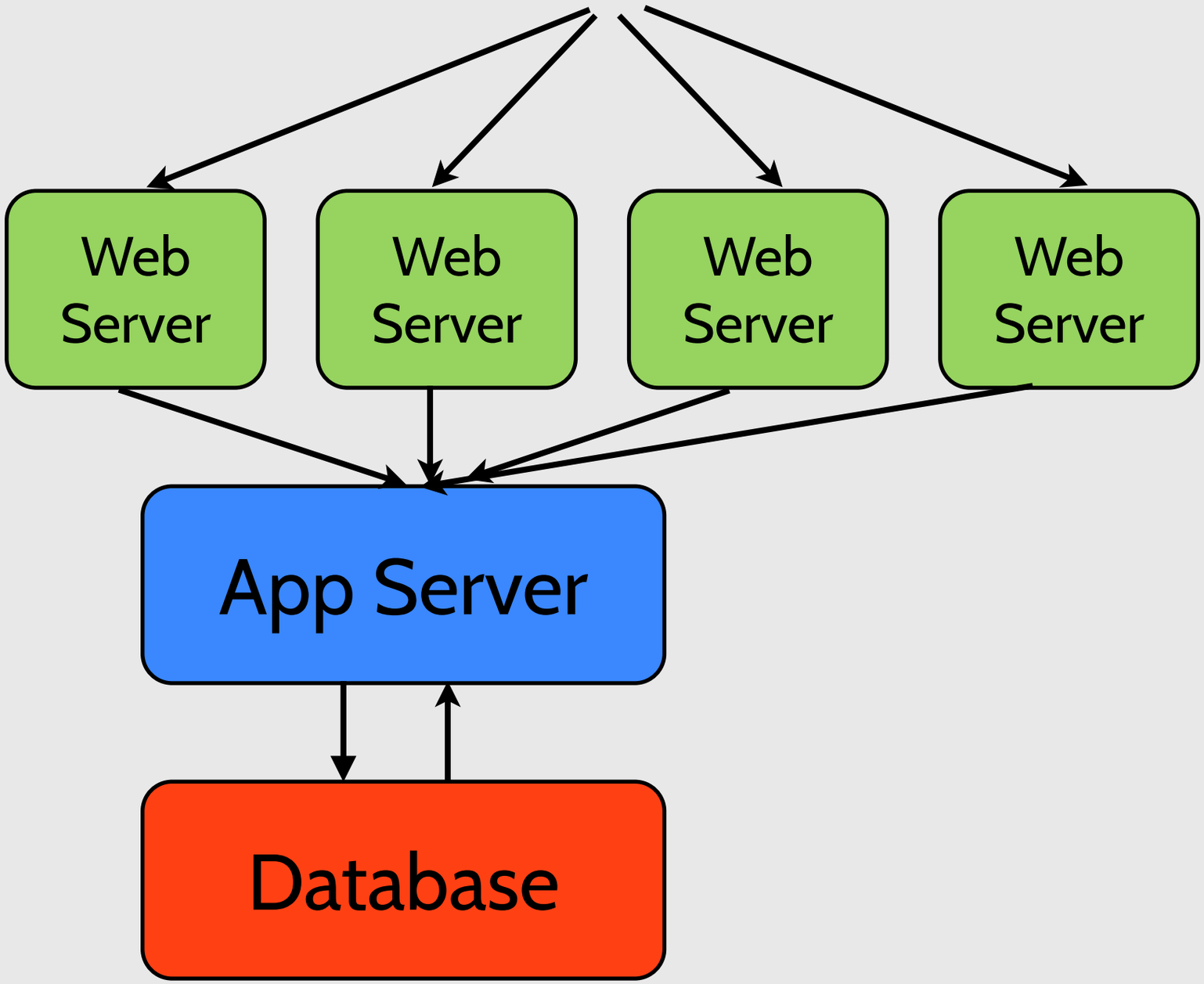
problem?

have to keep cache(s) up to date...

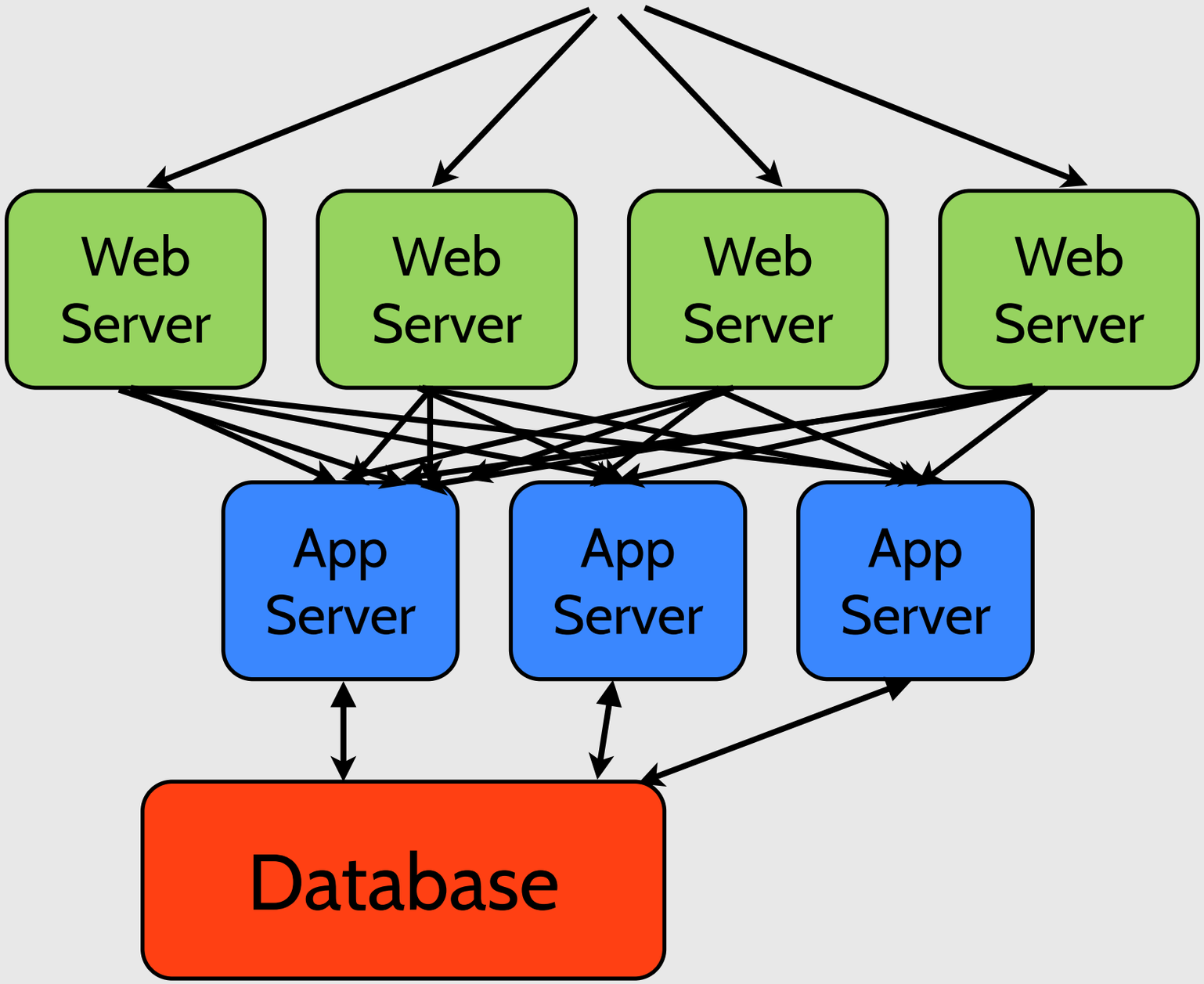
Horizontal Scale-out



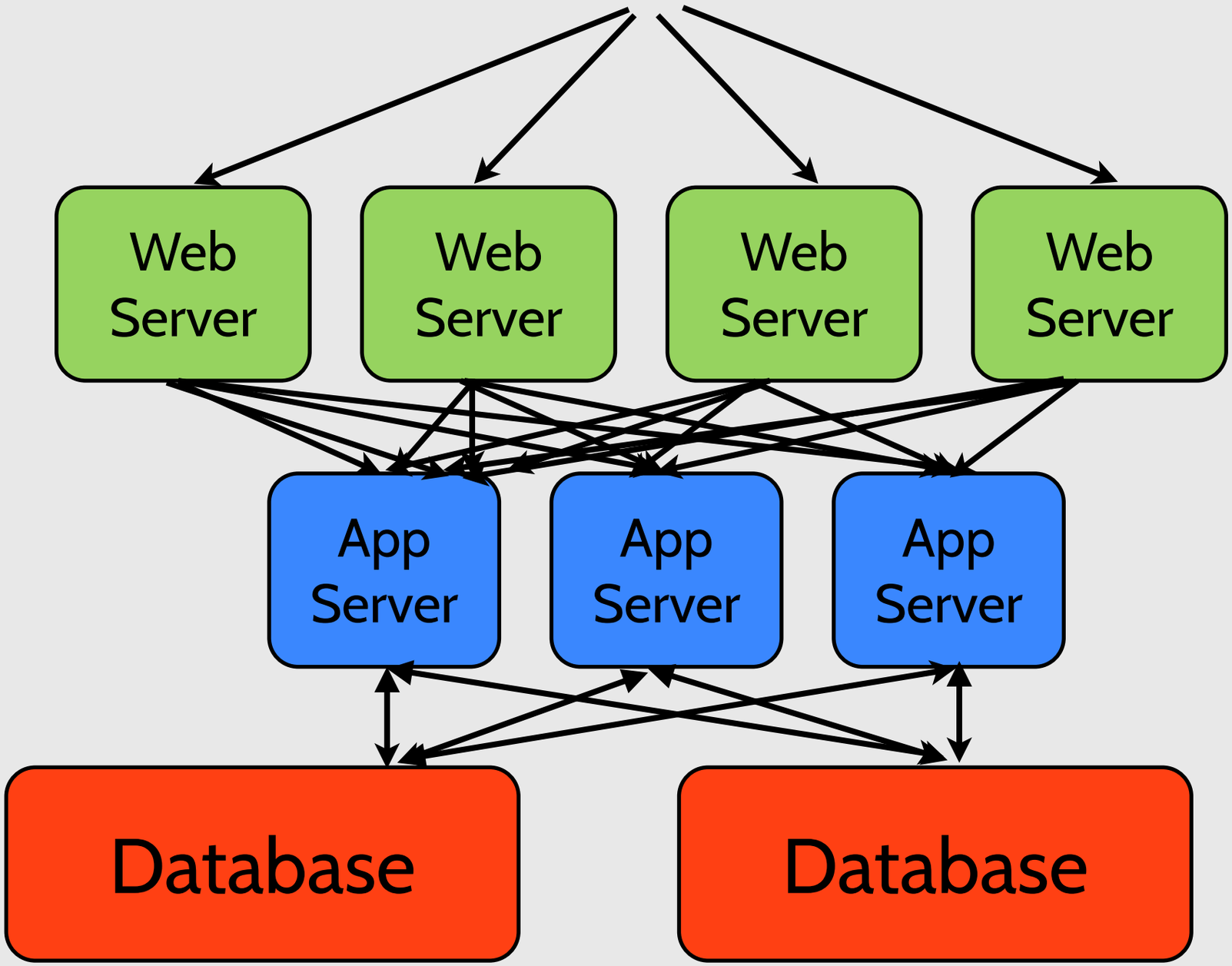
Horizontal Scale-out



Horizontal Scale-out



Horizontal Scale-out



Statelessness

Front-end, mid-tier are often stateless--why?

Statelessness

Front-end, mid-tier are often stateless--why?

Simplifies programming, reasoning about services

DBs can manage complexity of state management; promote reuse of complex code

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

Scaling

- What if data can't fit on a single server?
- What if a server goes down?
- What if a machine fails completely?

Fun: "Why is it hard to scale a database in layman's terms?" http://qr.ae/8jOUH

Scaling

- What if data can't fit on a single server?
- What if a server goes down?
- What if a machine fails completely?

A good problem to have, but not easy!

Fun: "Why is it hard to scale a database in layman's terms?" http://qr.ae/8jOUH

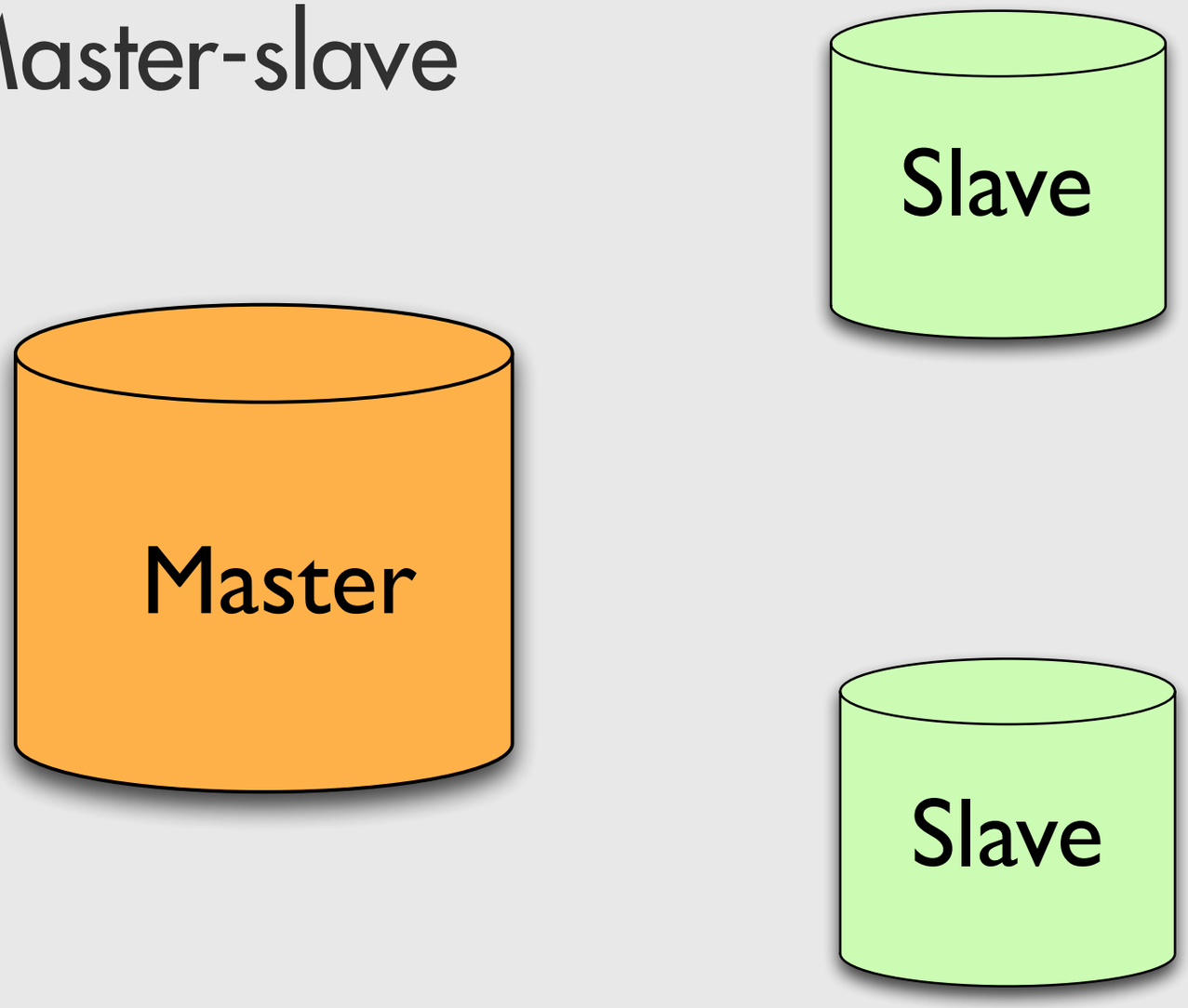
Replication

Replication

- Provides durability: don't lose data
- Provides capacity: multiple servers
- Leads to many interesting challenges

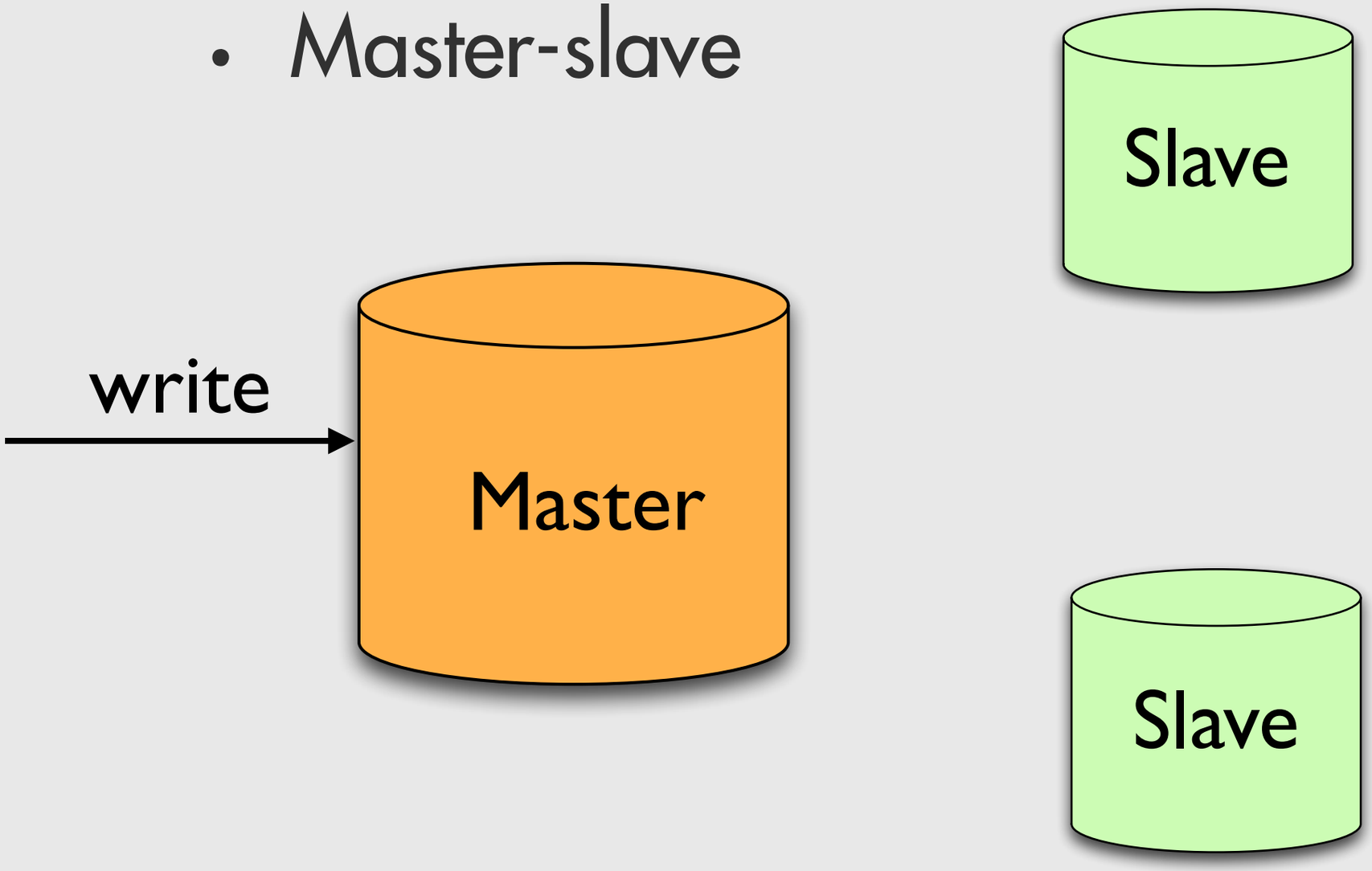
Typical Replication

- 3-way
- Master-slave



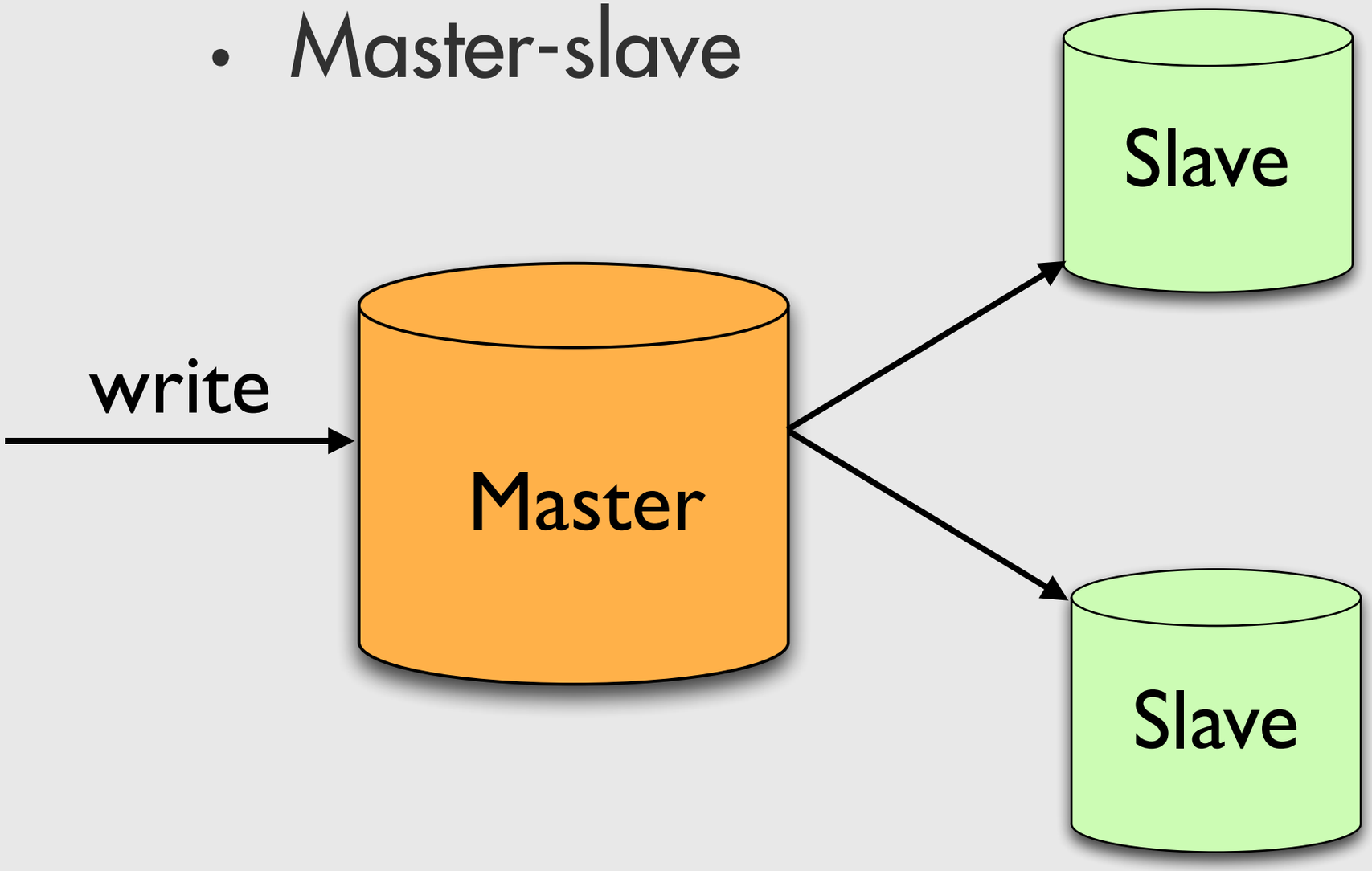
Typical Replication

- 3-way
- Master-slave



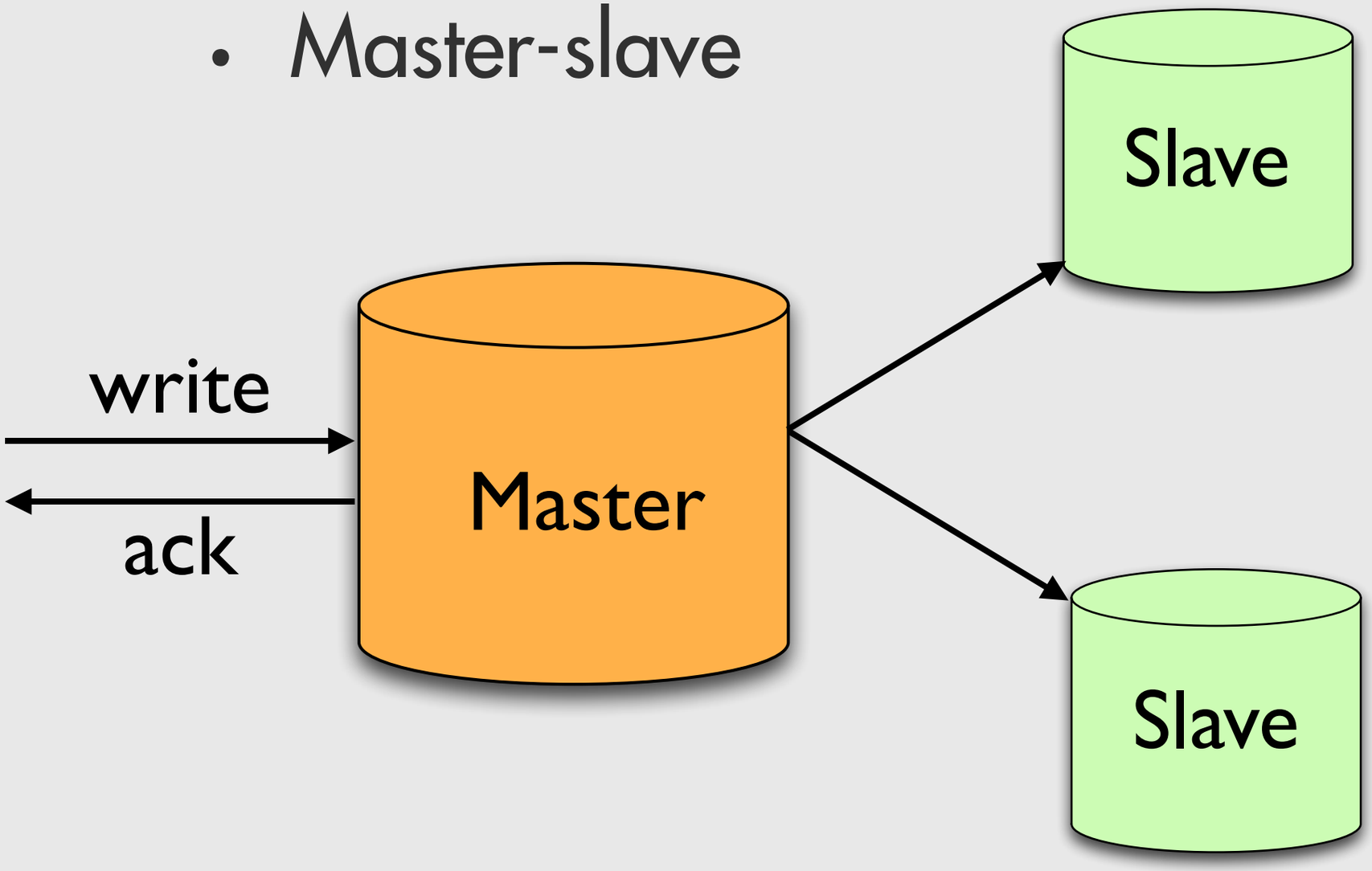
Typical Replication

- 3-way
- Master-slave



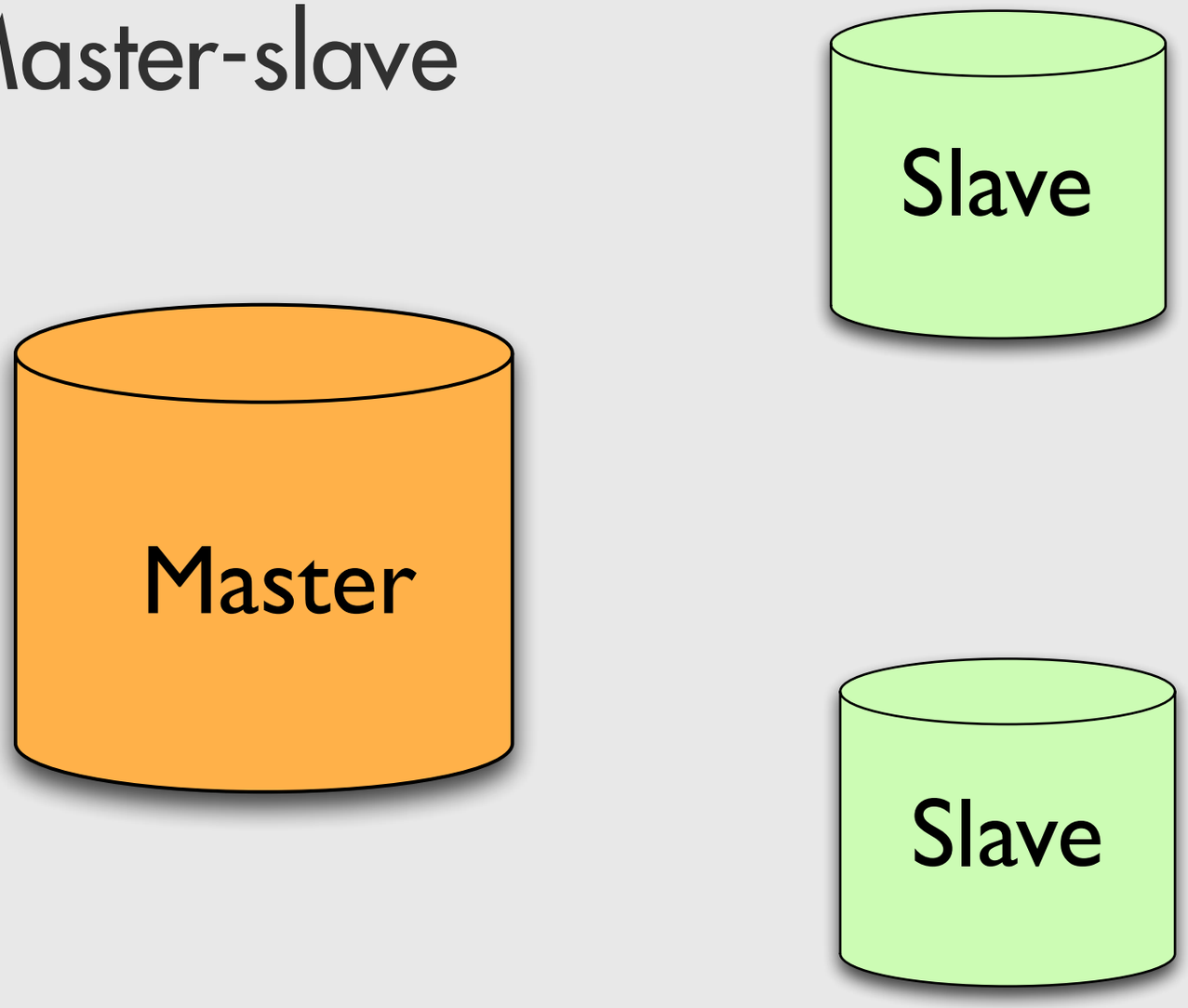
Typical Replication

- 3-way
- Master-slave



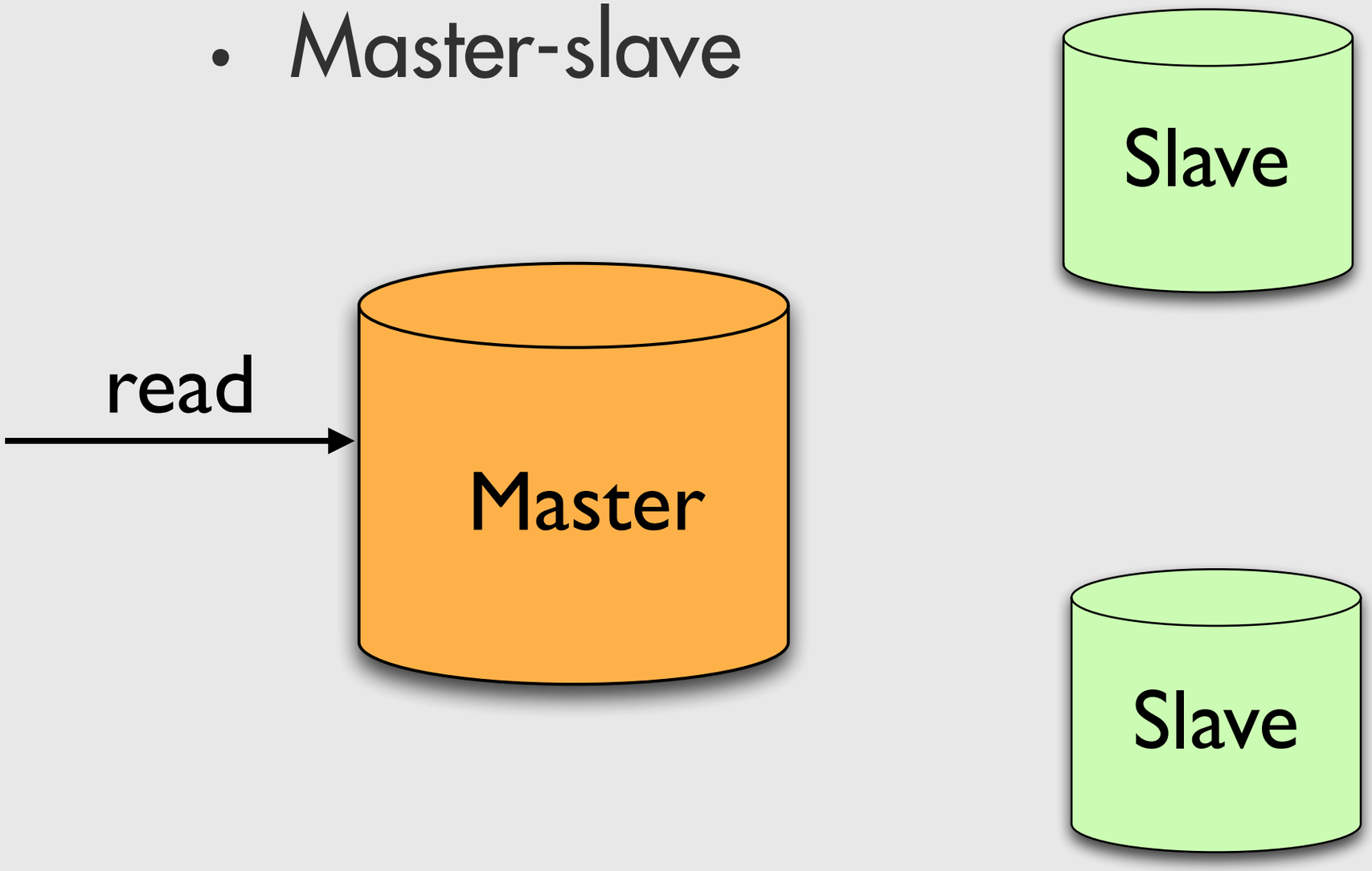
Typical Replication

- 3-way
- Master-slave



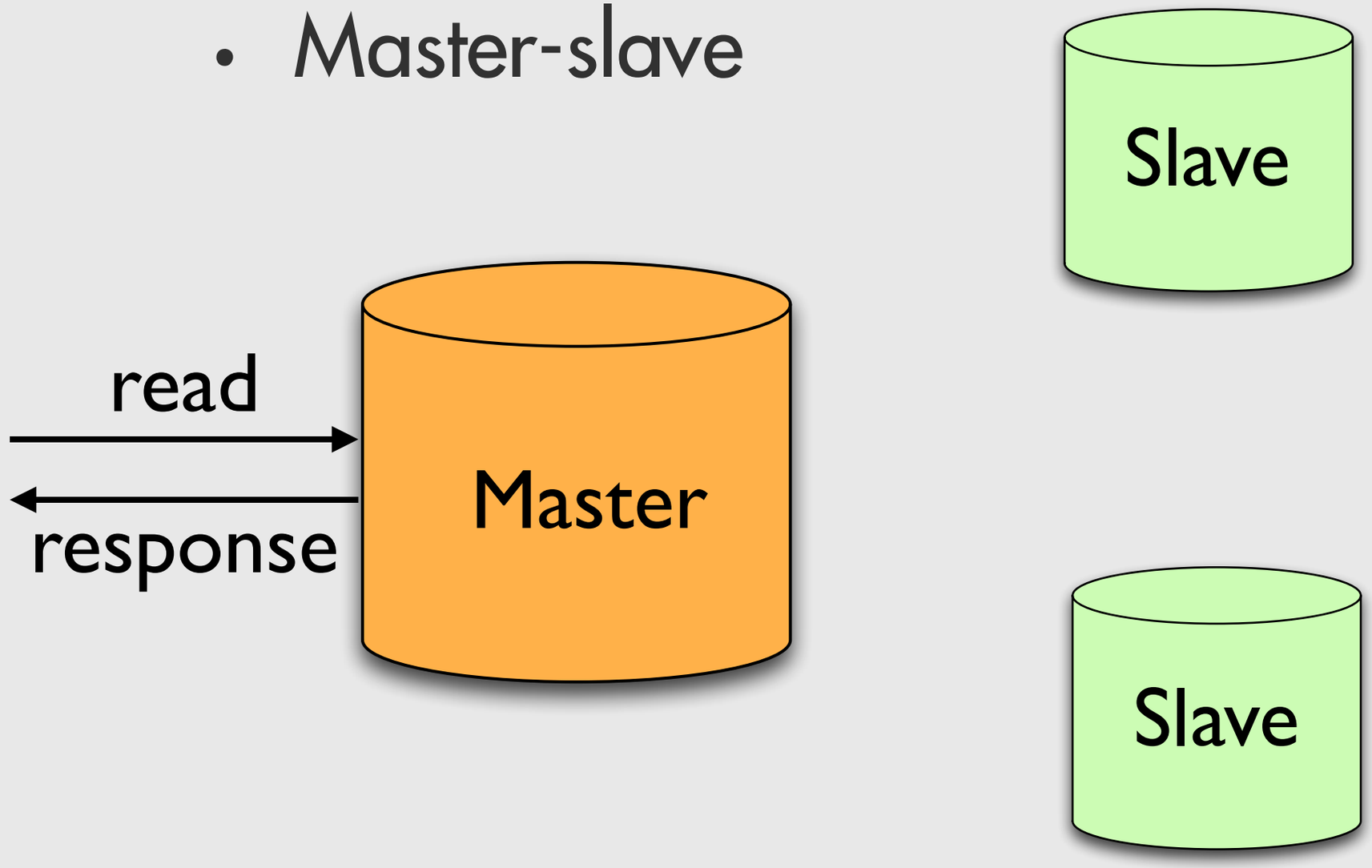
Typical Replication

- 3-way
- Master-slave



Typical Replication

- 3-way
- Master-slave



Data Placement

- Which servers get what data?
- How many copies of the data?

Data Placement

- Which servers get what data?

Data Placement

- Which servers get what data?

assign students to servers based on age:

Data Placement

- Which servers get what data?

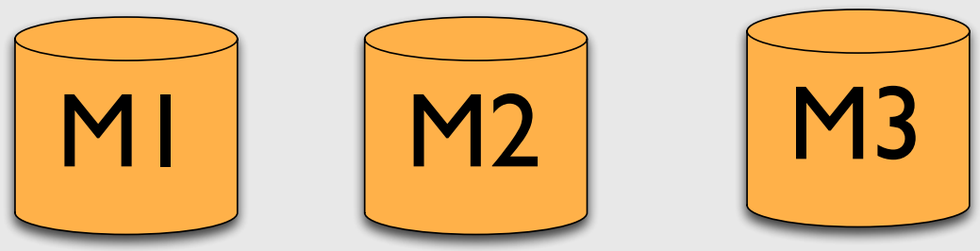
assign students to servers based on age:

don't stress, this is just to show a taste of what's required...

Data Placement

- Which servers get what data?

assign students to servers based on age:



Data Placement

- Which servers get what data?

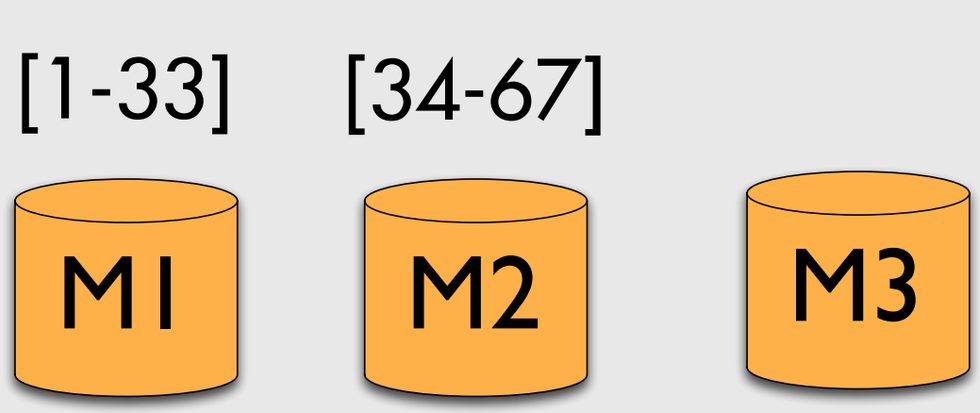
assign students to servers based on age:

[1-33]

Data Placement

- Which servers get what data?

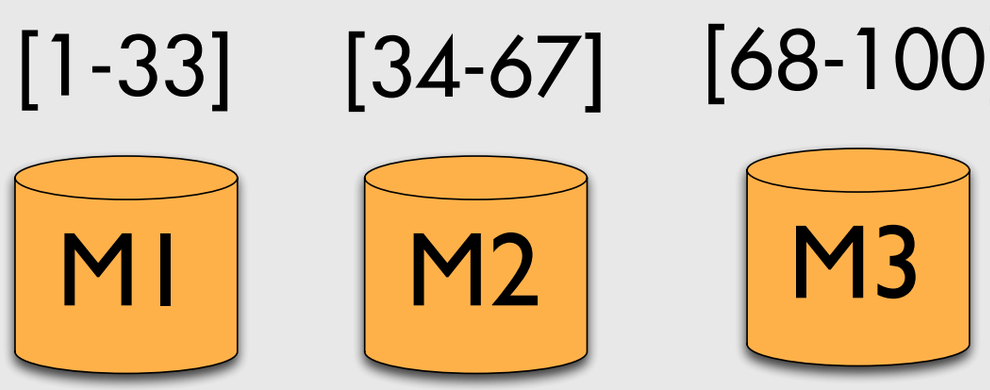
assign students to servers based on age:



Data Placement

- Which servers get what data?

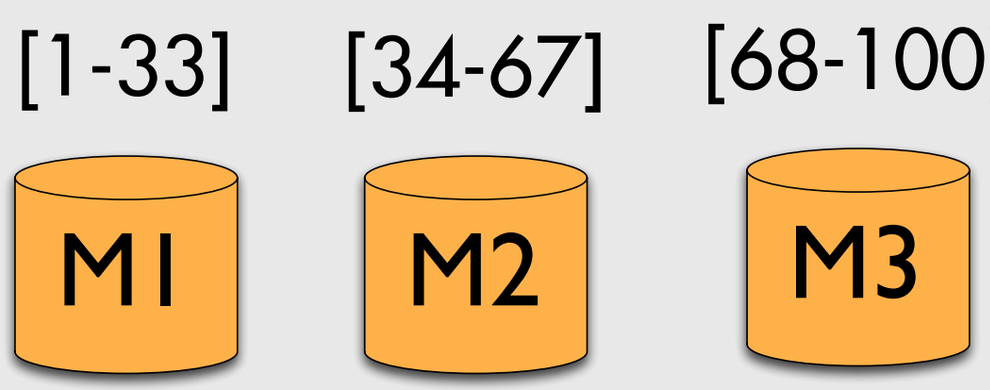
assign students to servers based on age:



Data Placement

- Which servers get what data?

assign students to servers based on age:



RANGE partitioning

Data Placement

- Which servers get what data?

assign students to servers based on age:

RANGE partitioning

server = hash(age) % 3

Data Placement

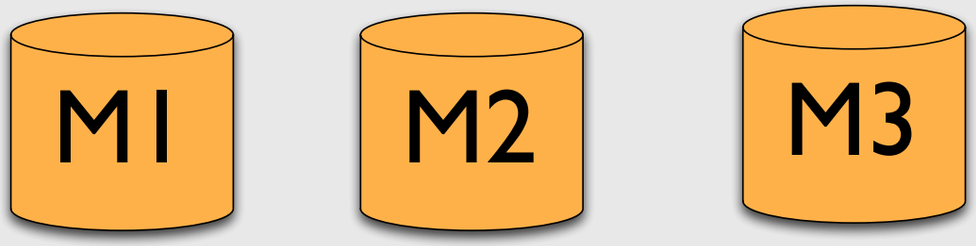
- Which servers get what data?

assign students to servers based on age:

[1-33] [34-67] [68-100]

RANGE partitioning

server = hash(age) % 3



Data Placement

- Which servers get what data?

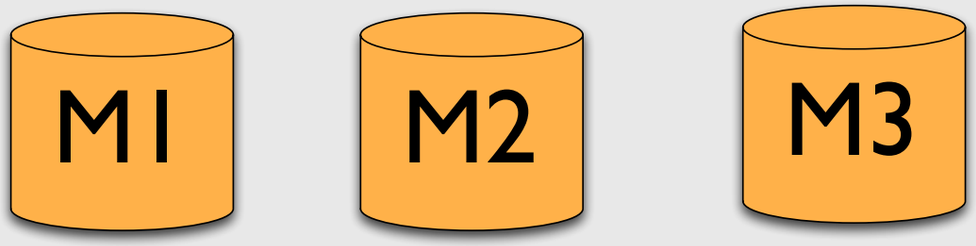
assign students to servers based on age:

[1-33] [34-67] [68-100]

RANGE partitioning

server = hash(age) % 3

[1,8,...]



Data Placement

- Which servers get what data?

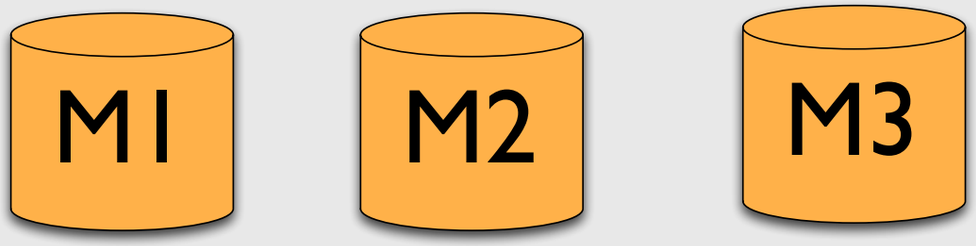
assign students to servers based on age:

[1-33] [34-67] [68-100]

RANGE partitioning

server = hash(age) % 3

[1,8,...] [2,4,...]



Data Placement

- Which servers get what data?

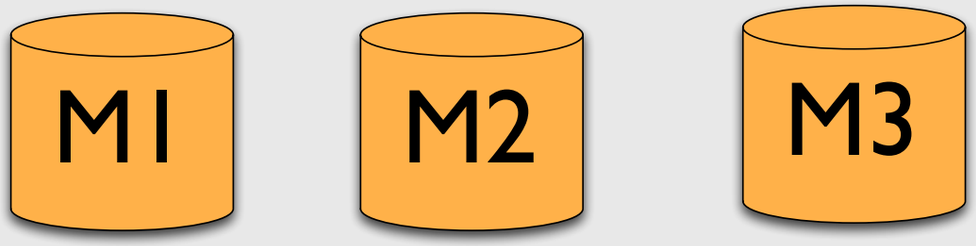
assign students to servers based on age:

[1-33] [34-67] [68-100]

RANGE partitioning

server = hash(age) % 3

[1,8,...] [2,4,...] [3,5,...]



Data Placement

- Which servers get what data?

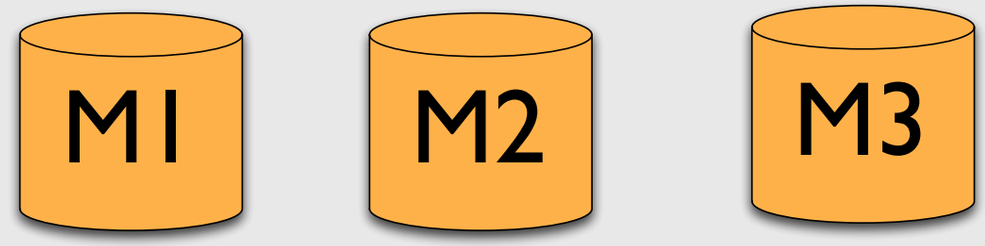
assign students to servers based on age:

[1-33] [34-67] [68-100]

RANGE partitioning

server = hash(age) % 3

[1,8,...] [2,4,...] [3,5,...]



HASH partitioning

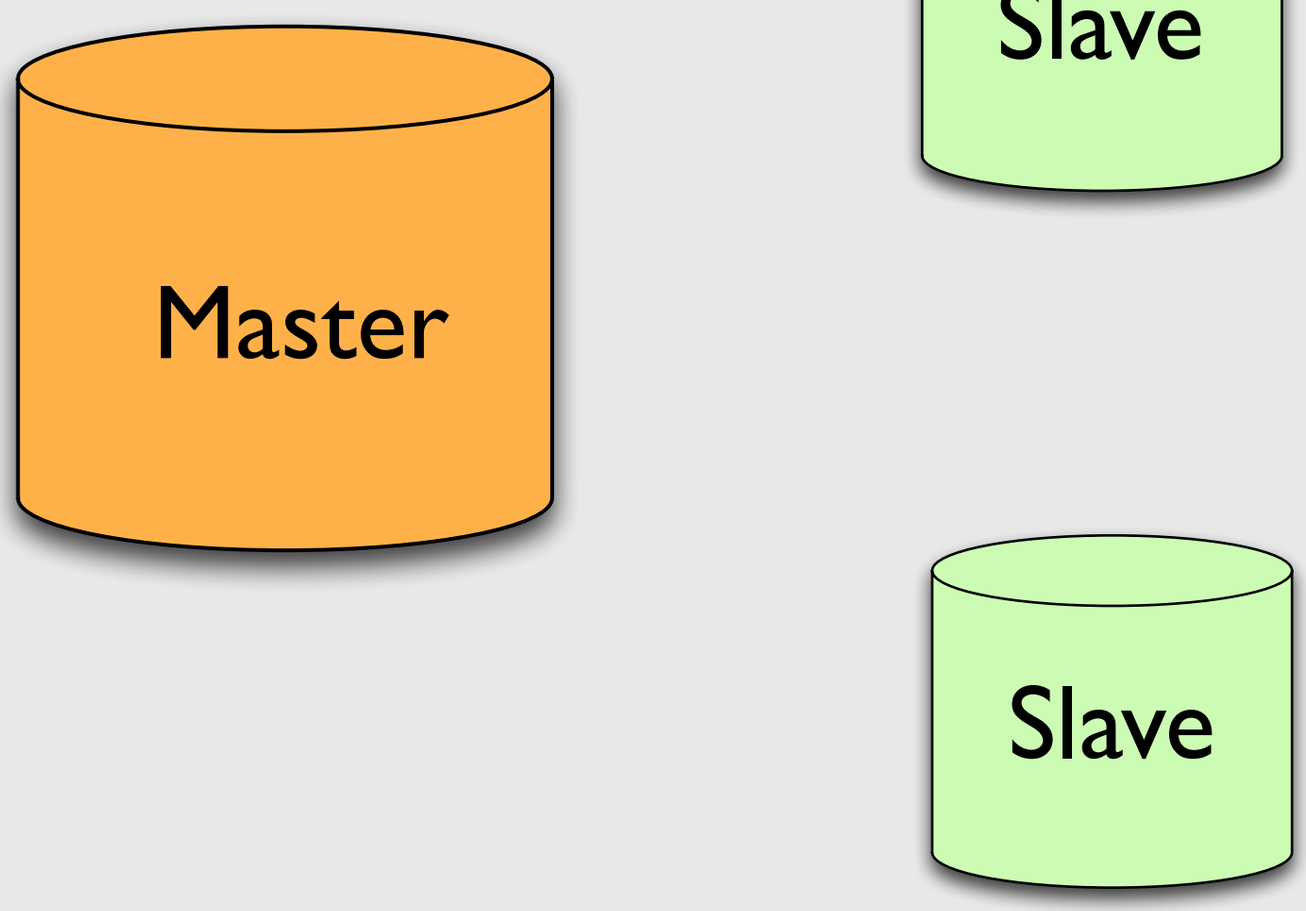
Data Placement

- Which servers get what data?
  - range vs. hash vs. ?

- How many copies of the data?
  - Durability: how many failures?
  - Capacity: how many requests?

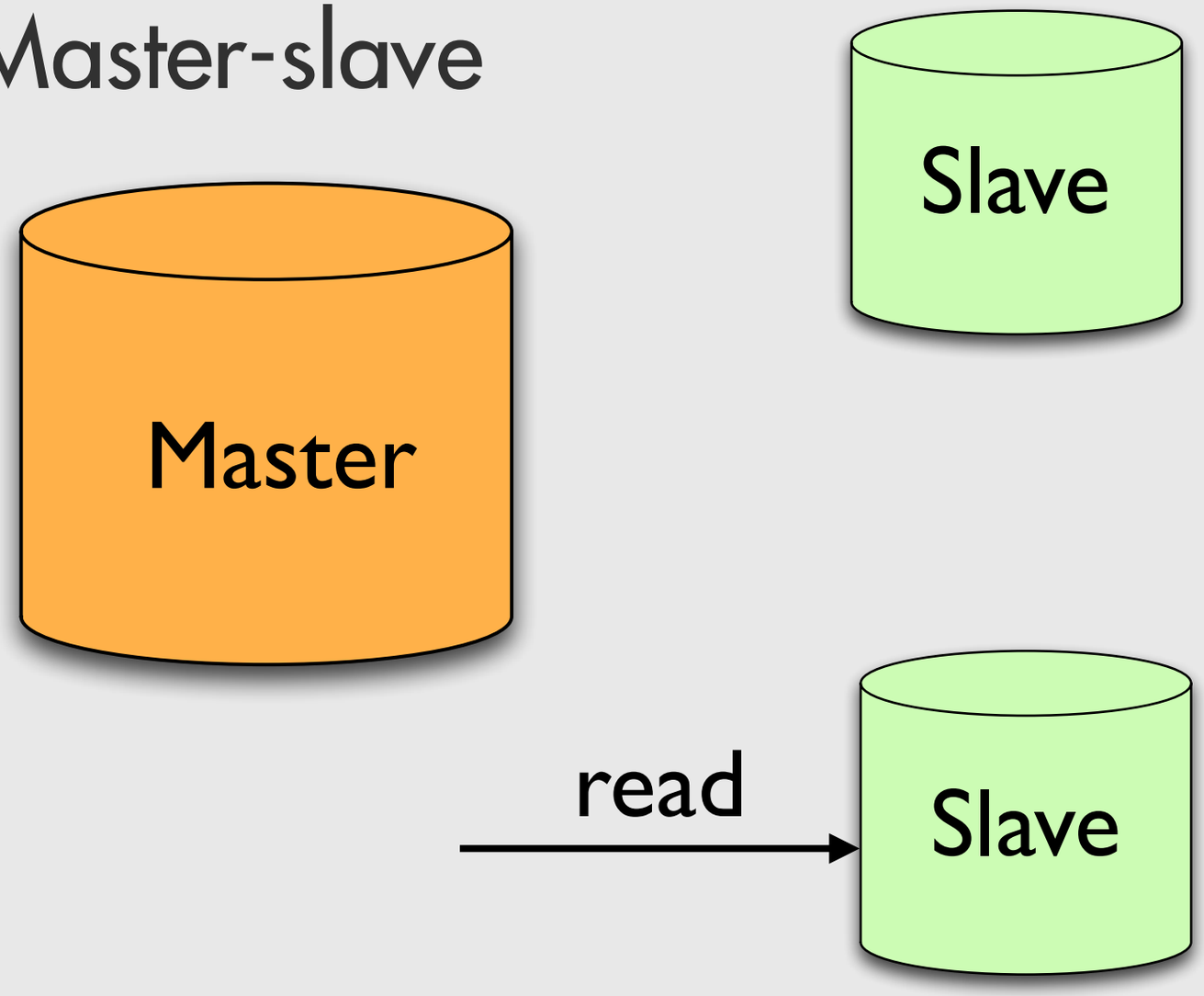
Typical Replication

- 3-way
- Master-slave



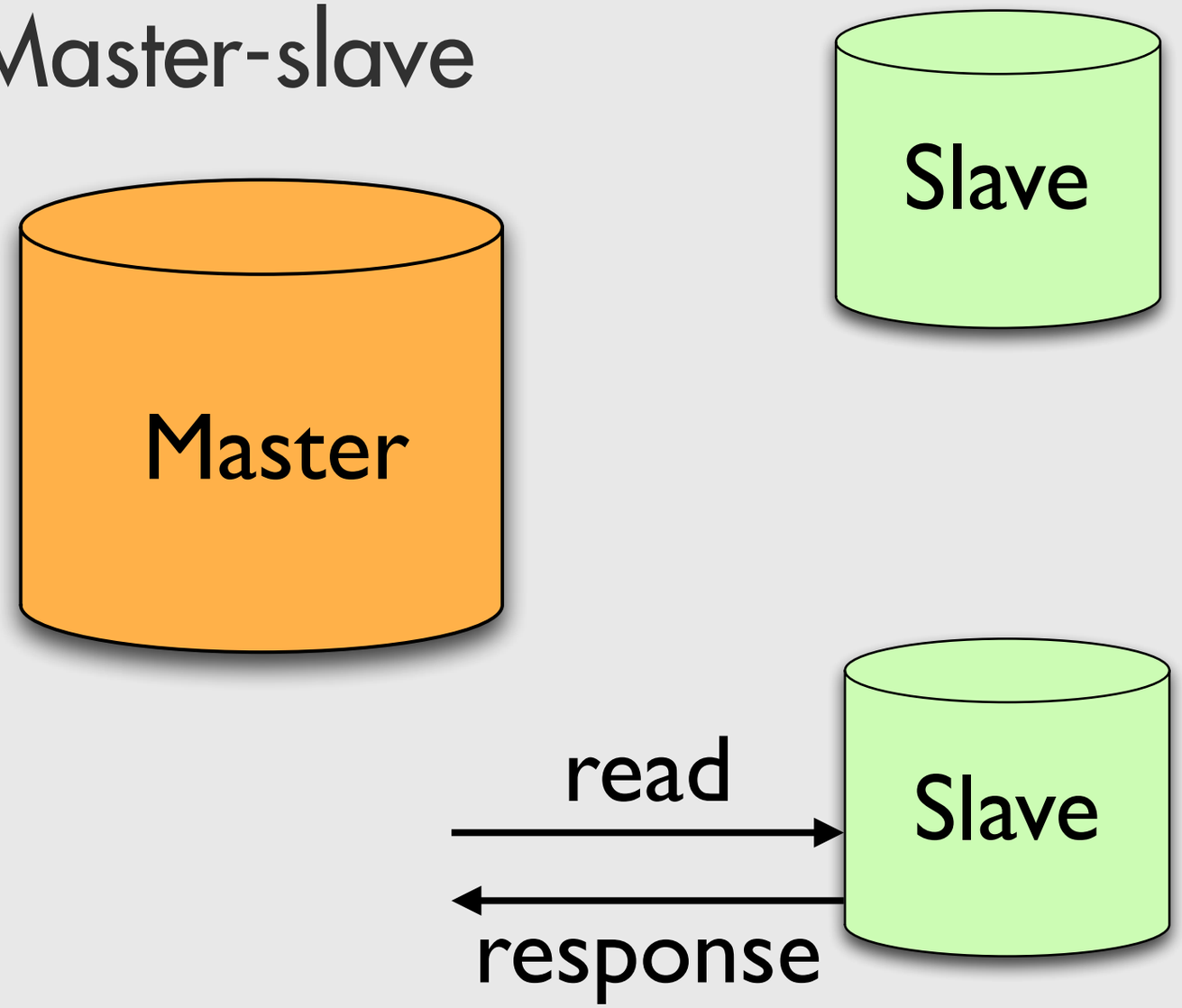
Typical Replication

- 3-way
- Master-slave



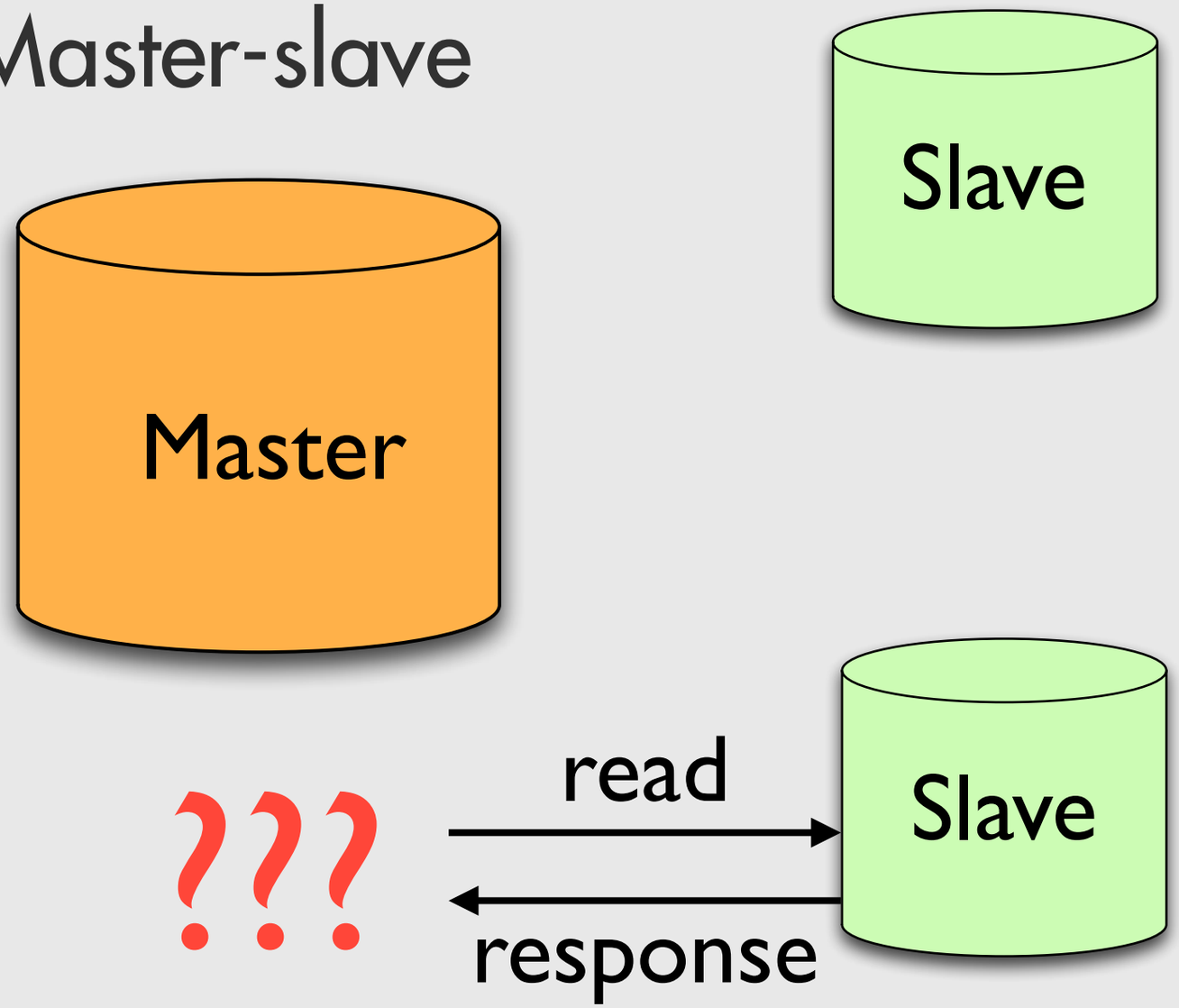
Typical Replication

- 3-way
- Master-slave



Typical Replication

- 3-way
- Master-slave



Consistency

- Need to keep replicas up to date

Consistency

- Need to keep replicas up to date
  - May be slow or impossible!
  - Very expensive if servers are located around the world!

Consistency

- Need to keep replicas up to date
  - May be slow or impossible!
  - Very expensive if servers are located around the world!

Compromise:
latency versus staleness
latency vs consistency
(CAP lecture)

Why data storage?
Classic Data Mgmt
IRL Data Mgmt
Break
Web Arch
Scaling
NoSQL

Database Usage

- Different storage systems are optimized for different tasks
  - Random access
  - Sequential access
- Different kinds of data
- Different load & query volume

What do we put in a DB?

- URL mappings?
- View counts?
- Email addresses?
- Pictures?
- Server logs?

What is NoSQL?

- "New" approach to data storage
- Simple but predictable data models
- Often have to build own features
- Designed for massive scale-out

When it comes to NoSQL, there's lots of hype and some merit
A reasonable, short answer to "What is NoSQL": http://stackoverflow.com/a/1164798
A longer answer: http://static.usenix.org/publications/login/2011-10/openpdfs/Burd.pdf

Key-Value Store

Key-Value Store

```
put(key,value)
```

Key-Value Store

`put(key,value)`

`get(key)`

Key-Value Store

`put(key,value)` `get(key)`

PROs

Key-Value Store

`put(key,value)` `get(key)`

PROs

- Simple API
- Easy to understand perf
- Easy to scale, implement

Key-Value Store

`put(key,value)` `get(key)`

PROs

- Simple API
- Easy to understand perf
- Easy to scale, implement

CONs

Key-Value Store

```
put(key,value)        get(key)
```

PROs

- Simple API
- Easy to understand perf
- Easy to scale, implement

CONs

- Simple API
- Handle own schema management, complex features (e.g., WHERE)

Key-Value Store

```
put(key,value)        get(key)
```

PROs

- Simple API
- Easy to understand perf
- Easy to scale, implement

CONs

- Simple API
- Handle own schema management, complex features (e.g., WHERE)

e.g., Riak, Voldemort, Memcached

Document Store

```
{“long_url” : “http://
   news.google.com”,
 “short_url” : “http://
    bit.ly/awekl”,
  “hit_count” : 482240}
```

no pre-defined schema,
but store handles layout of arbitrary fields

Document Store

```
{“long_url” : “http://
   news.google.com”,
 “short_url” : “http://
    bit.ly/awekl”,
  “hit_count” : 482240}
```

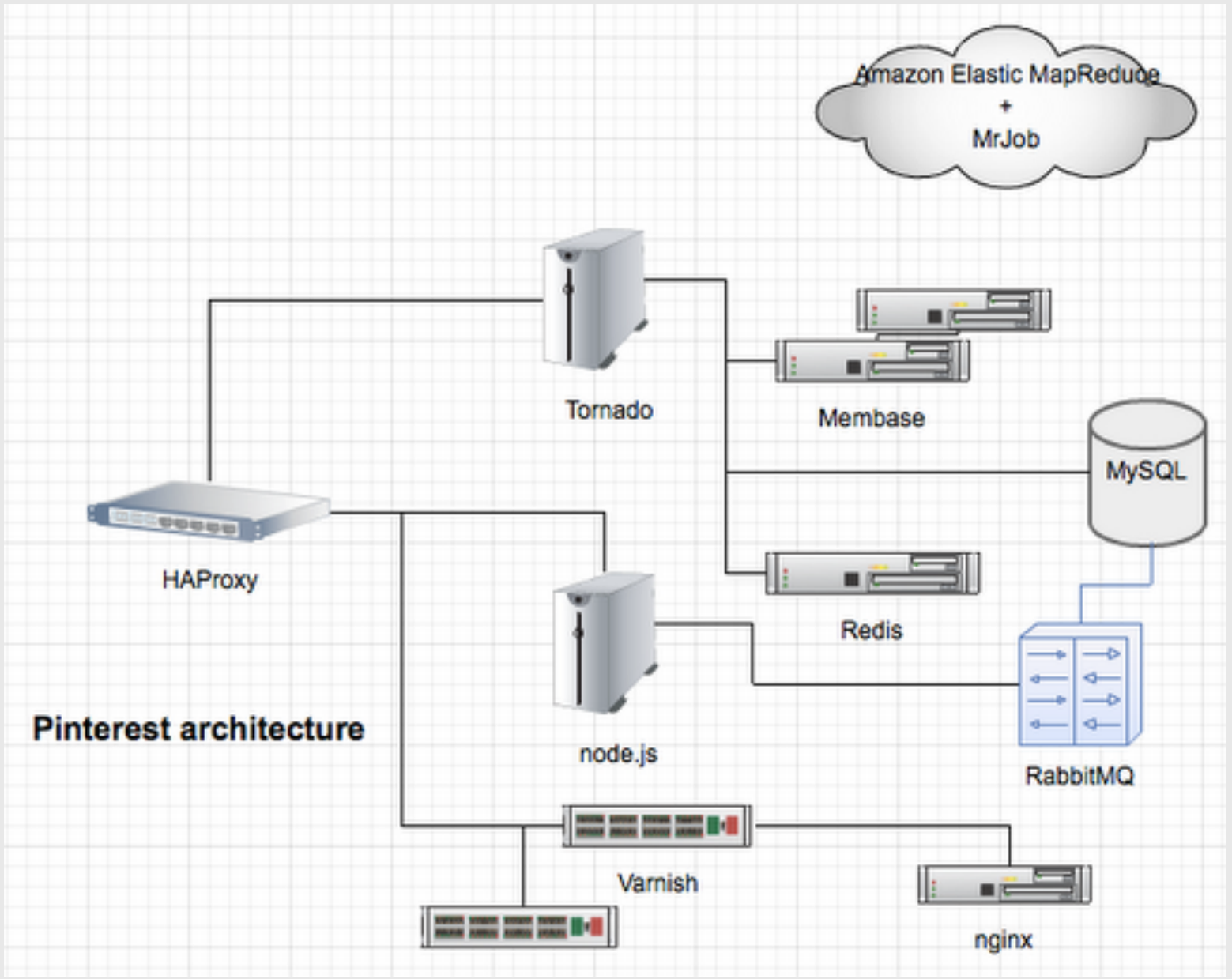
no pre-defined schema,
but store handles layout of arbitrary fields

e.g., MongoDB, CouchDB, Cassandra, Redis,
Amazon DynamoDB

Blob Store

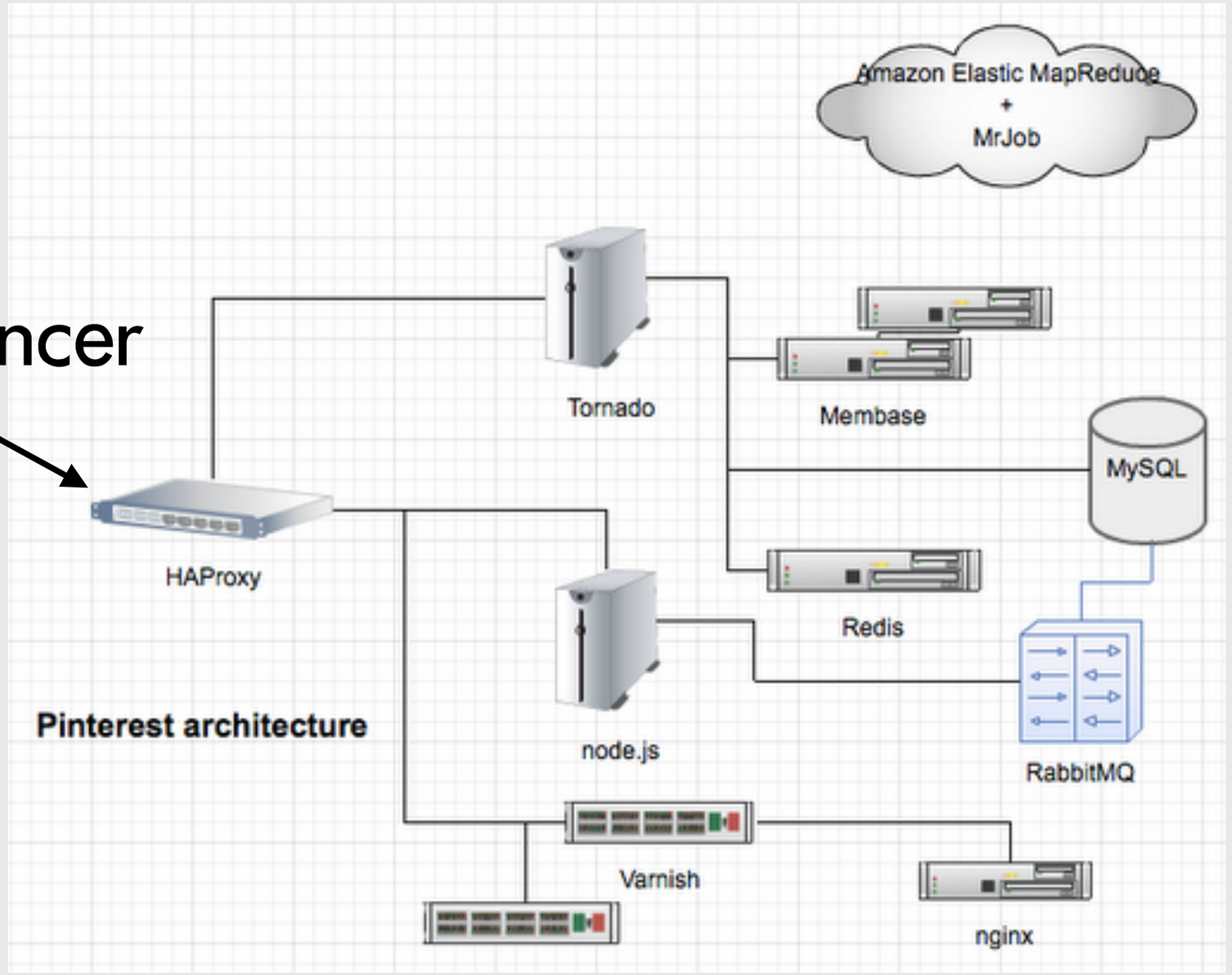
optimized for larger files

e.g., Hadoop File System,
Amazon S3 (Dropbox, Netflix),
Riak CS,
MS Azure Blob Storage



http://nosql.mypopescu.com/post/1765841584 7/polyglot-persistence-at-pinterest-redis-membase

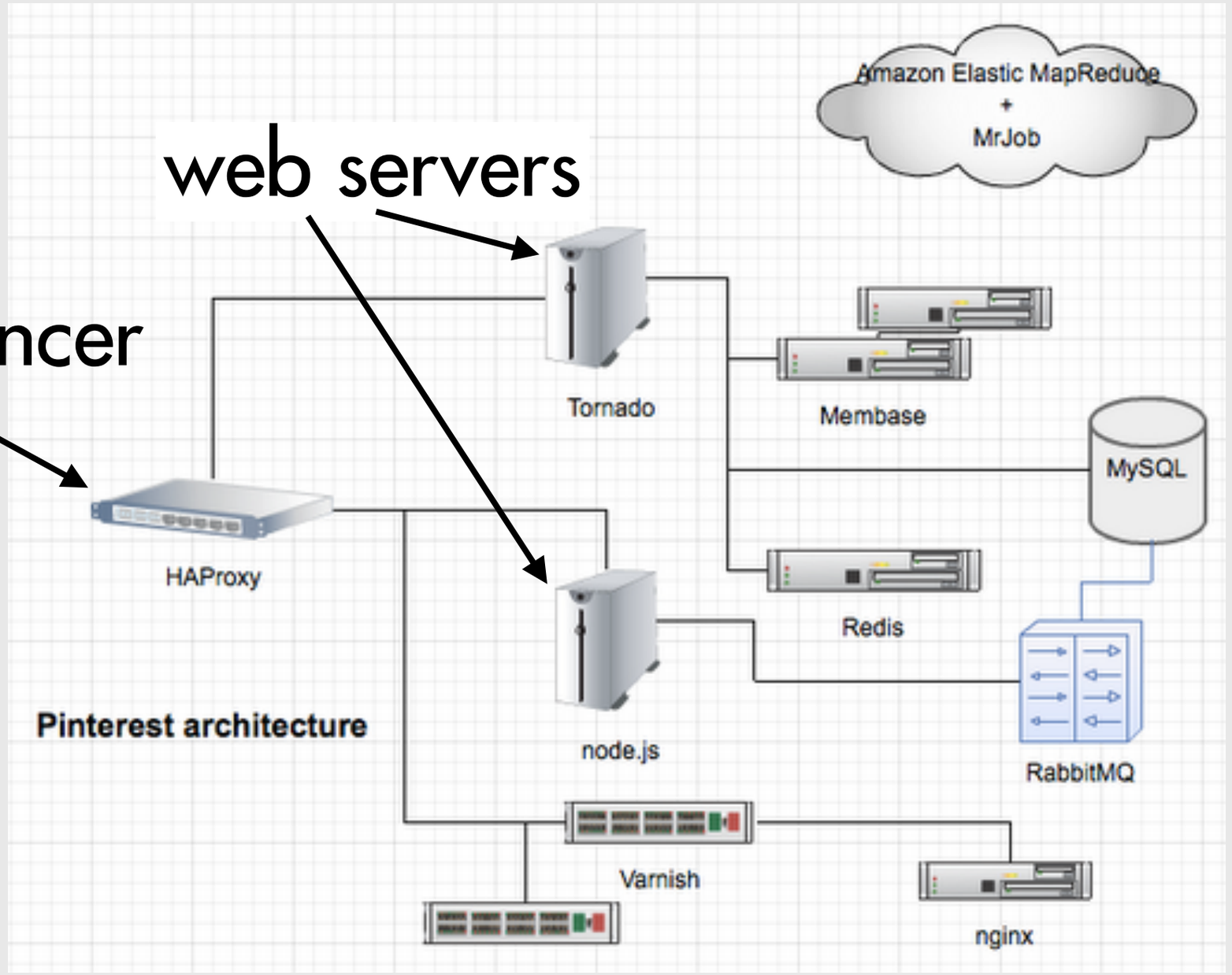
load balancer



http://nosql.mypopescu.com/post/1765841584 7/polyglot-persistence-at-pinterest-redis-membase

load balancer

web servers

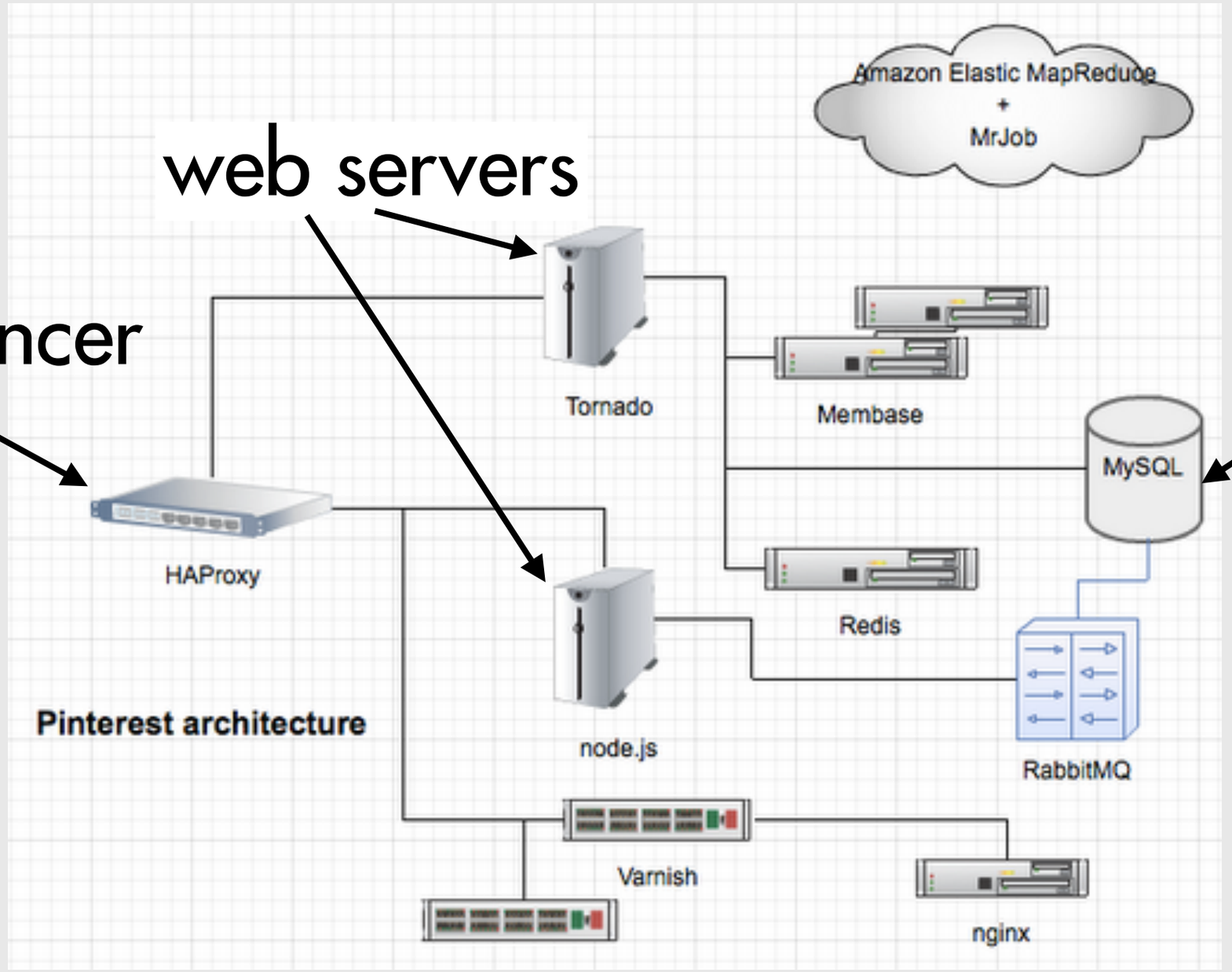


http://nosql.mypopescu.com/post/1765841584 7/polyglot-persistence-at-pinterest-redis-membase

load balancer

web servers

database



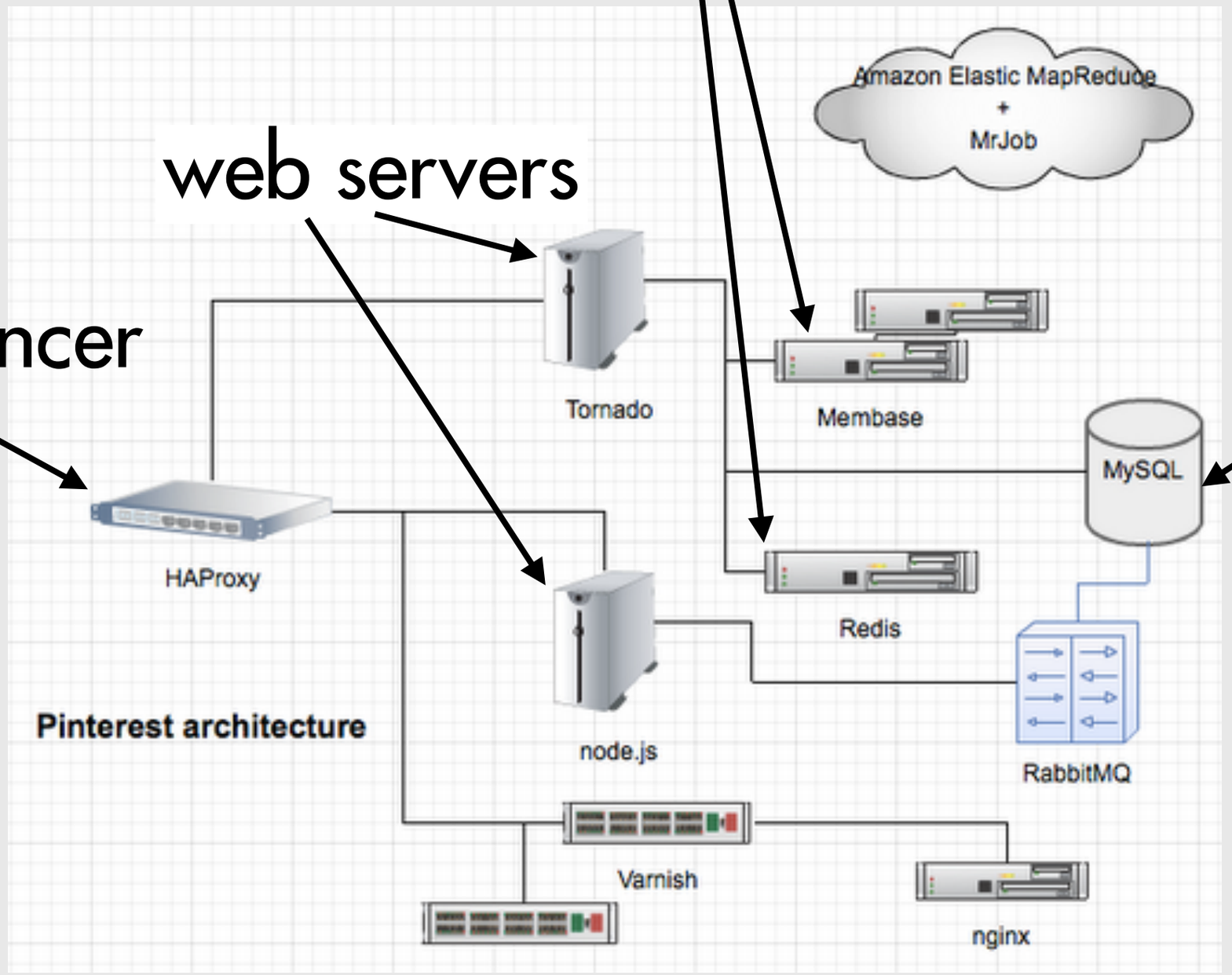
http://nosql.mypopescu.com/post/1765841584 7/polyglot-persistence-at-pinterest-redis-membase

DB caches

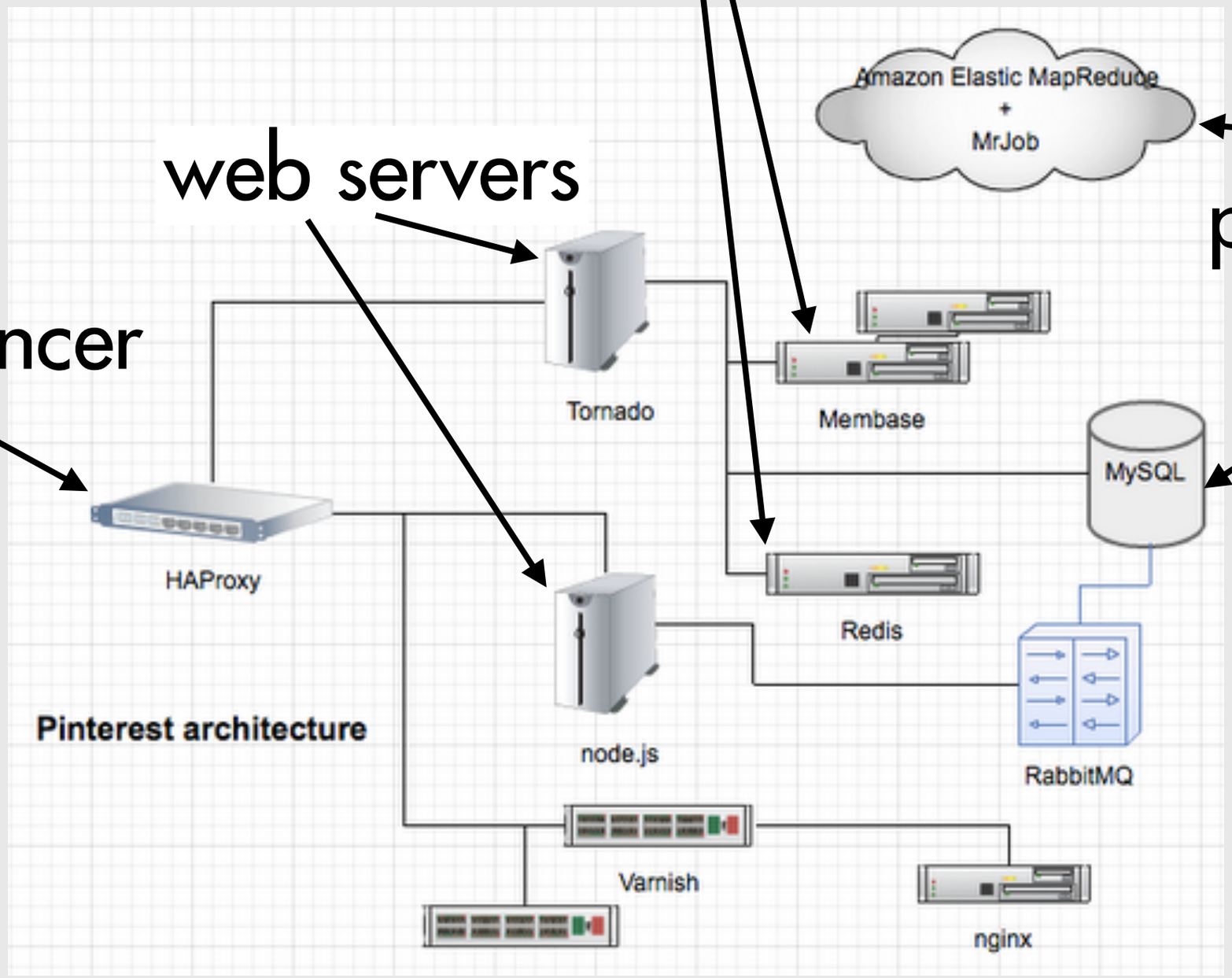
web servers

load balancer

database

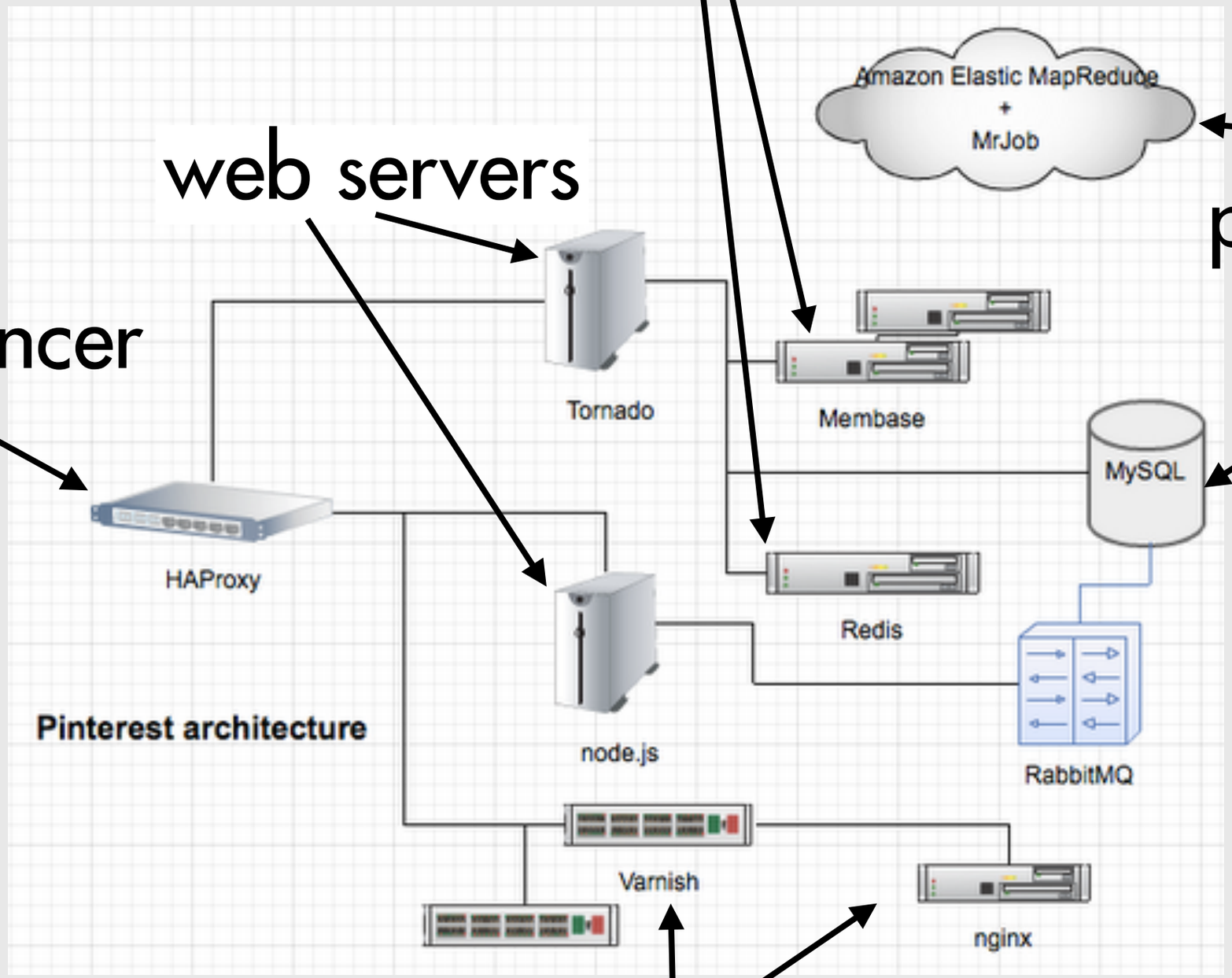


http://nosql.mypopescu.com/post/1765841584 7/polyglot-persistence-at-pinterest-redis-membase



http://nosql.mypopescu.com/post/1765841584 7/polyglot-persistence-at-pinterest-redis-membase

DB caches



data processing

load balancer

database

HTTP caching

http://nosql.mypopescu.com/post/1765841584 7/polyglot-persistence-at-pinterest-redis-membase

Facebook

- MySQL
- Memcached
- HBase
- Custom blob store (Haystack)

well-documented, e.g.,
http://www.quora.com/Facebook-Engineering/What-is-Facebooks-architecture

What's a database, anyway?

Summary

- Databases designed to solve many common data storage problems
- Storage comes in many flavors; right choice is often specific to use case
- When in doubt, start simple!

Peter Bailis
pbailis@cs.berkeley.edu