Slick Data Sharding

How to Develop Scalable Data Applications With Drupal

Tobby Hagler, Phase2 Technology

Don't Forget...

Official DrupalCon London Party

Batman Live World Arena Tour

Buses leave main entrance
Fairfield Halls at 4pm

Overview

- Purpose – Reasons for sharding
- Problems/Examples of a need for sharding
- Types of scaling and sharding
- Sharding options in Drupal

Scale:
Horizontal vs Vertical

Horizontal Scale
Add more machines of the same type

Vertical Scale
Bigger and badder machines

Sharding

- What is sharding?
- Types of sharding – Partitioning and Federation
- How sharding helps
- Vs. typical monolithic Drupal database

What Is sharding?

Simply put, sharding is physically breaking large data into smaller pieces (shards) of data.

The trick is putting them back together again…

Reasons for Sharding

- Sharding for scaling your application
- Sharding for shared application data
- Leveraging specialized technologies
  - Caching is a form of federated sharding

How Sharding Helps

- Scale your applications by reducing data sets in any single database
- Secure sensitive data by isolating it elsewhere
- Segregates data

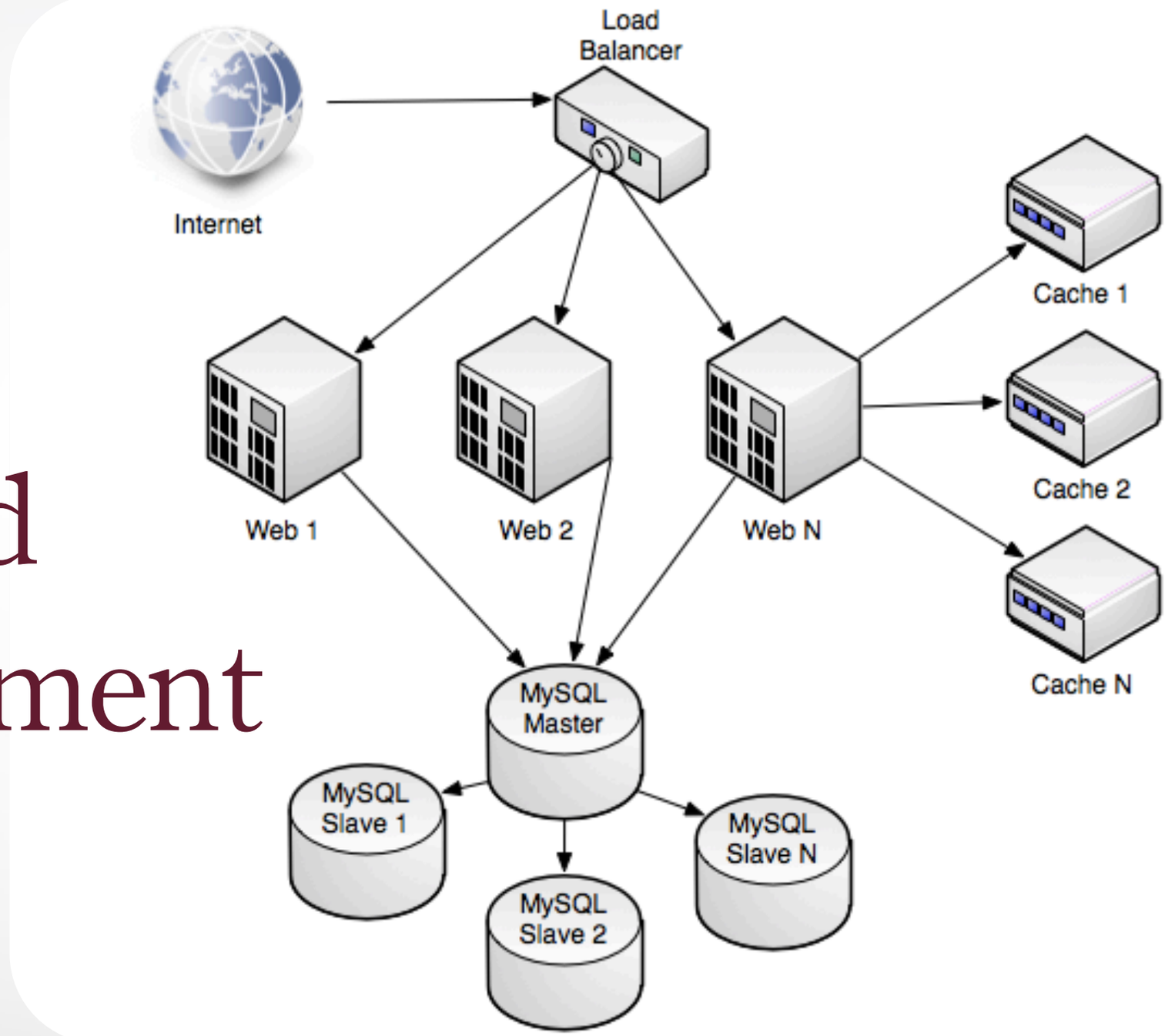
Be Sure You've Tried Everything Else

- Memcached
- Boost Module
- Load balanced web servers
- MySQL Master/Slave replicate
- Turning Views into Custom Queries

More Things To Try...

- Moar memory!
- Move .htacess to vhost config
- Apache tunes
- MySQL tunes
- Replace search with Apache Solr
- Optimizing PHP (custom compile)
- Apache Drupal module
- Replace Apache with nginx
- Switched to 3rd party services for comments
- Replace contrib modules with custom development

Typical Balanced Environment



Types of sharding

Partitioning

- Horizontal
- Divides something into two parts
- Unshuffle
- Reduced index size
- Hard to do

Federation

- Vertical
- A set of things
- Uses logical divisions
- Split up across physically different machines

Horizontal Partitioning

Scaling your application's performance

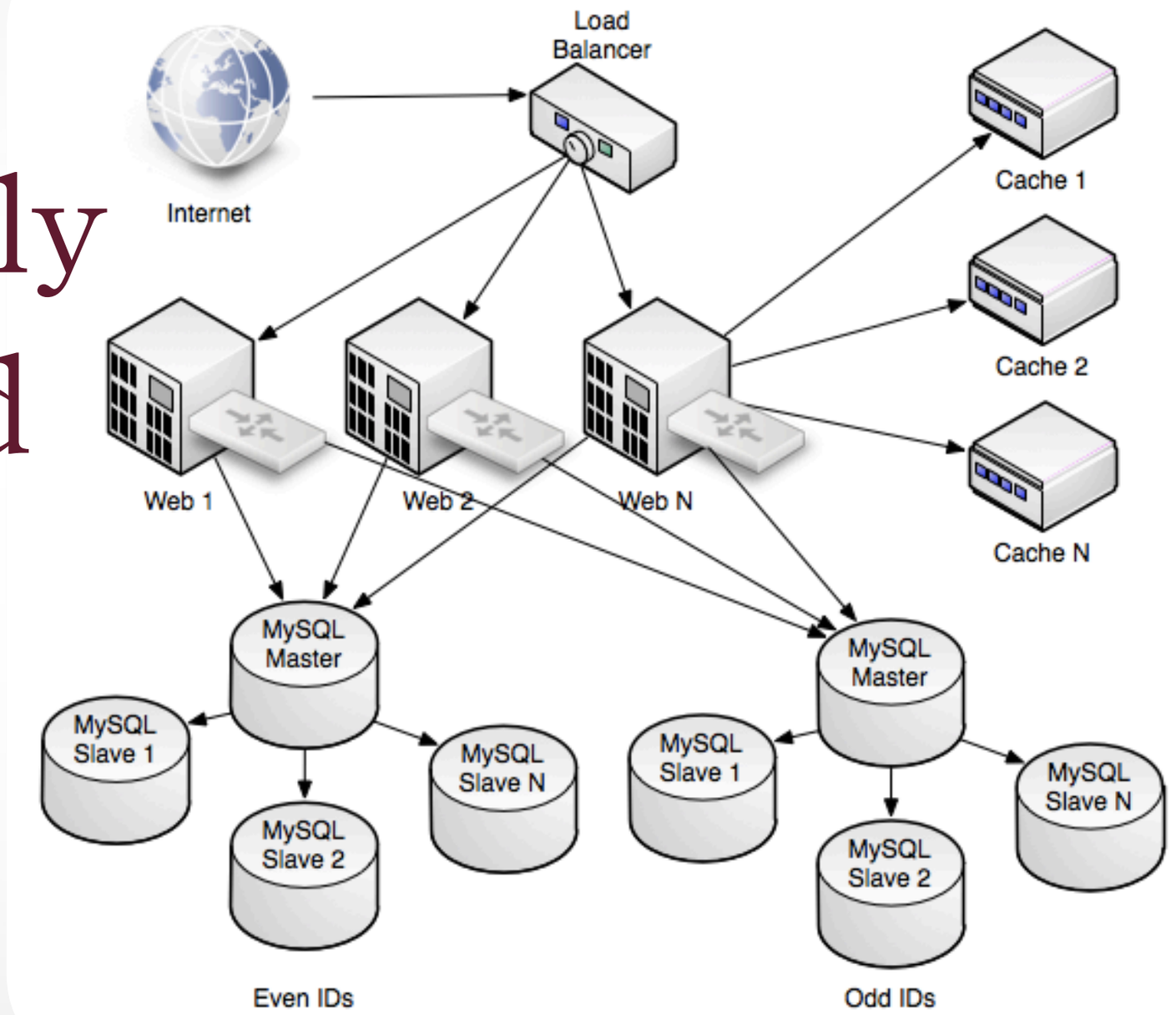
Distributed data load

This is the Shard of Last Resort

Even/Odd Partitions

- This is not Master/Master replication
- Rows are divided between physical databases
- Will require custom database API to properly achieve split rows
- Applies to node loads, entity loads, etc
- Achieved by auto_increment by N with different starting offsets and application distributes writes in round-robin fashion and via keyed mechanisms to distribute reads and reassemble data

Horizontally Partitioned Databases



Federation

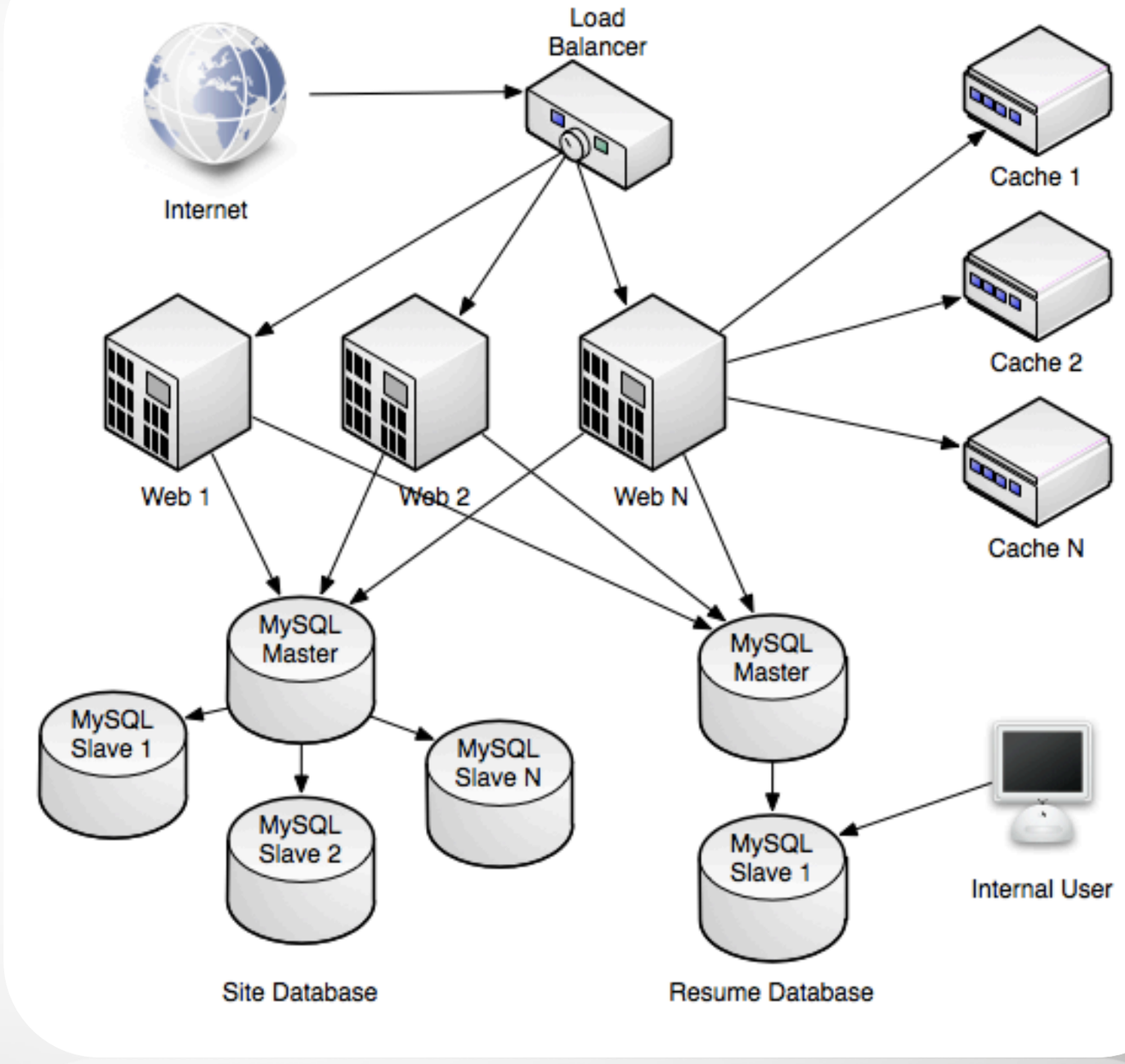
Vertically partitioning data by logical affiliation

Sharding for shared application data

Manageability – distributing data sets

Security - Allows for exposing certain bits of data to other applications without exposing all

Vertically Scaled Databases



Application Sharding

Not just sharding data

Shard the components of your site

Sample Use Cases

Collecting resumes within your existing site

Building an ideation tool

Sharding Resume Data

- Accepting resumes for a large corporation
- Users submit resume via Webform
- Submit and process data into separate database
- Resume data is processed by internal HR software to evaluate potential employees

Sharding Schemas

Same physical database, different schemas

Uses database prefixing in settings.php

~ or ~

Different physical databases

Uses db_set_active to switch db connections

Database Prefixes

- Handled in settings.php
- Uses MySQL's dot separator to target different schemas
- Requires that the MySQL user used by Drupal has proper permissions
- Ex: db_1.users and db_2.users

Database Prefixes

Drupal 6

```
$db_prefix = array (
   'default'         => '',
   'users'           => 'shared_.',
   'sessions'        => 'shared_.',
   'role'            => 'shared_.',
   'authmap'         => 'shared_.',
   'users_roles'     => 'shared_.',
   'profile_fields' => 'shared_.',
   'profile_values' => 'shared_.',
);
```

Database Prefixes

Drupal 7

```
$databases = array (
 'default' => array (
  'default' => array (
   'prefix' => array(
     'default' => '',
     'users'          => 'shared_.',
     'sessions'       => 'shared_.',
     'role'           => 'shared_.',
     'authmap'        => 'shared_.',
     'users_roles'    => 'shared_.',
    ),
  ),
 ),
);
```

Database Prefixes

Tips, Tricks, and Caveats

Can share user data between Drupal and Drupal 7 with table alters and strict prevention of Drupal 7 logins or user saves

Should log in with the lower version of Drupal

Different Physical Databases

- Set up additional connections in *settings.php*
- Change connections using *db_set_active()*
- Use *db_set_active()* to switch back when done
- **Watch for schema caching and watchdog errors**

Different Databases

Drupal 6

```
$db_url = array (
  'default' => 'mysql://user:pass@host1/db1',
  'second'  => 'mysql://user:pass@host2/db2',
  'third'   => 'mysql://user:pass@host3/db3',
);
```

Database Prefixes

Drupal 7

```
$other_database = array (
  'database' => 'databasename',
  'username' => 'username',
  'password' => 'password',
  'host'     => 'localhost',
  'driver'   => 'mysql',
);

Database::addConnectionInfo('moduleKey', 'default',
$other_database);
db_set_active('moduleKey');
// Execute queries
db_set_active();
```

Switching Databases

```php
$schema = drupal_get_schema('table_name');
db_set_active('database_key');

// Execute queries
Drupal_write_record('table_name', $data);
db_set_active();
```

Saving Data in Another Database

- Hook_install_schema()
- drupal_write_record()
- Keeps web site database smaller
- Can keep sensitive data offsite
- Partitioned tables can limit/protect your web site database from internal users

Saving Data in Another Database

- Resume data is submitted via form
- Form's _submit function accepts final data
  - Schema loads table definition
  - Connects to the HR instance of MySQL
  - Writes new record
  - Uploads any files to private file space
  - Switches database back
- HR Director can query new resumes

Using MongoDB

MongoDB is a NoSQL database

"Schema-less" – data schema defined in code

Fast

Document-based

Simpler to scale vertically than MySQL

MongoUK

10gen Conference in London, UK

September 19, 2011

10gen.com/conferences/mongouk-sept-2011

MongoDB and Drupal

drupal.org/project/mongodb

7.x allows for field storage, cache, sessions, and blocks to be stored in MongoDB

Allows for connections to your own collections

MongoDB Data

- Four levels of objects
  - Connection
  - Database (schema)
  - Collection
  - Cursor (query results)
- Non-relational database
- Collections tend to be denormalized

MongoDB Documents

```
Resumes.Resume: {
   first_name: "John",
   last_name: "Smith",
   title: "Web Developer",
   address: {
      city: "London",
      country: "UK"
   },
   skills: [ 'PHP', 'Drupal', 'MySQL' ],
   ssn: 123456789,
}
```

Querying MongoDB Documents

```
$applicant = $applicants->find (
  array (
    'username' => 'Smith',
    'ssn': 1,
  ),
  array (
  'first_name' => 1,
  'last_name' => 1,
  ),
);
```

MongoDB Sharing via REST

- Simple REST – included as part of MongoDB
- Sleepy Mongoose – REST interface for MongoDB (Python)
- MongoDB REST (Node.js)

Ideation
REST Interface

Get a list of all idea documents

```
http://127.0.0.1:28017/ideation/ideas/
```

Get all comments for a specific idea

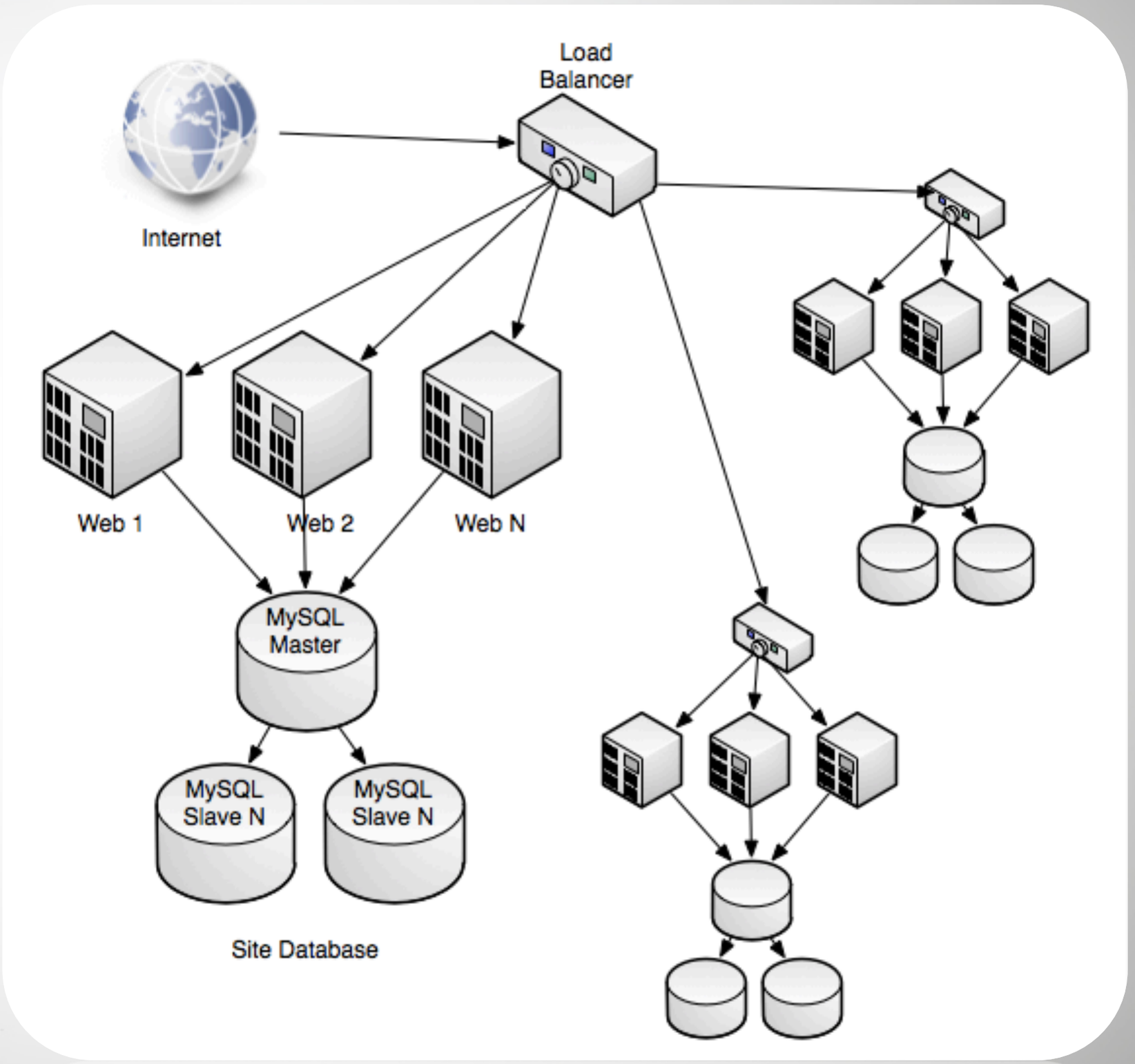
```
http://127.0.0.1:28017/ideation/comments/…
    …?filter__id=4a8acf6e7fbadc242de5b4f3…
    …&limit=10&offset=20
```

Will likely need a dedicated MongoDB REST inteface

Applications on Separate Web Tiers

- Application sharding *is* data sharding
- Separate Drupal instances
- Use mod_proxy as a pass-through
- Can used multiple load-balanced environments

Proxied Web Clusters



Questions?

Contact

thagler@phase2technology
@phase2tech
703-548-6050
d.o: tobby
Slides: agileapproach.com