Scaling Hibernate

Emmanuel Bernard - Max Ross


JBoss®
a division of Red Hat


Google™

Google™

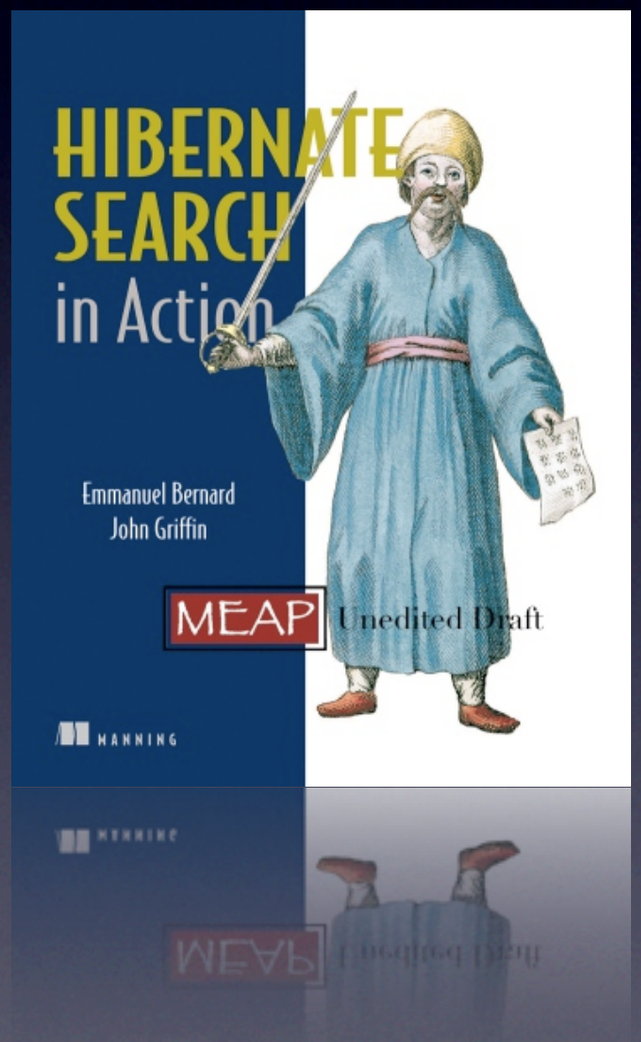
JBoss®
a division of Red Hat

Emmanuel Bernard

Hibernate Search in Action

blog.emmanuelbernard.com

twitter.com/emmanuelbernard



Max Ross

Google App Engine

Hibernate Shards

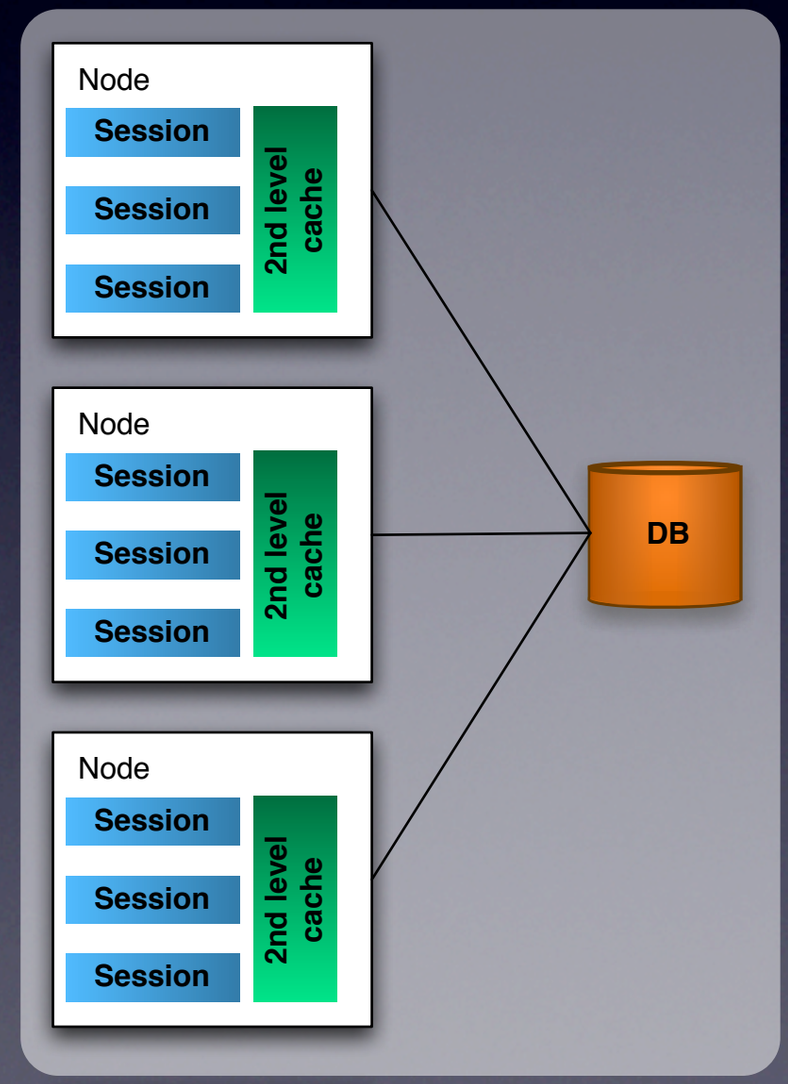
What is scalability?

What is scalability?

- Users
- Resource
- Data
- Uptime

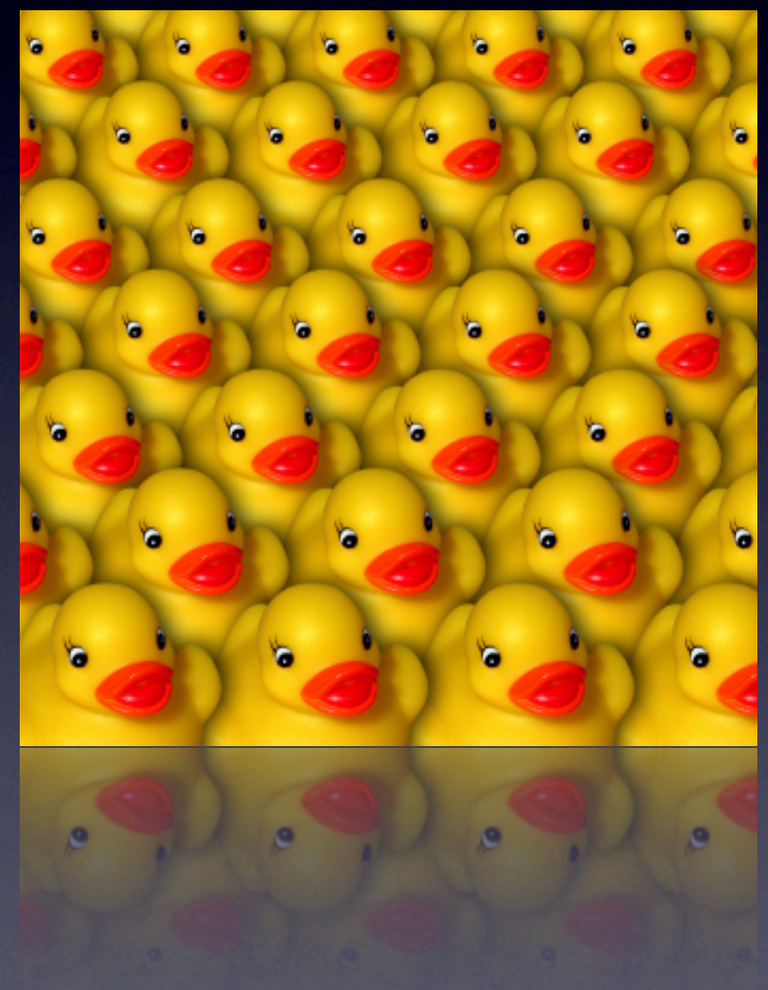
How does Hibernate stand?

- Limitations?
- SQL optimizations
- 2nd level cache
- Conversation



Changes in mass

- Bulk insert / update / delete
- Stateless session



To Googolzillions and beyond

Googolzillion things?
Who are you?

- Social network
- SaaS

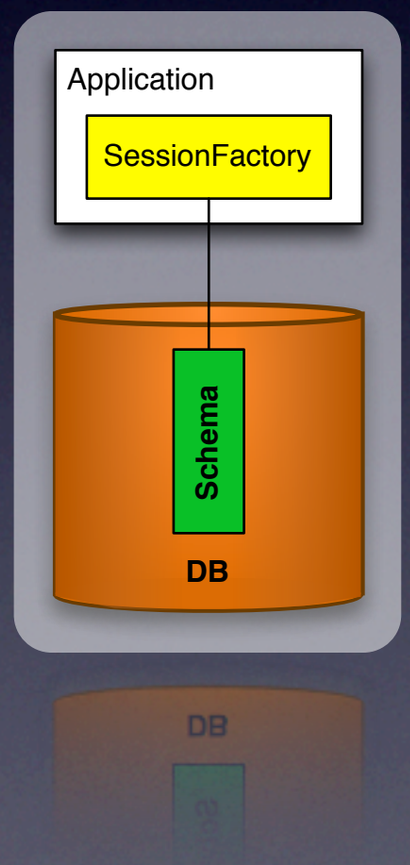
Problem

- Same data model
- Too much load
- Too much data
- Too many lawyers

Separating customer data

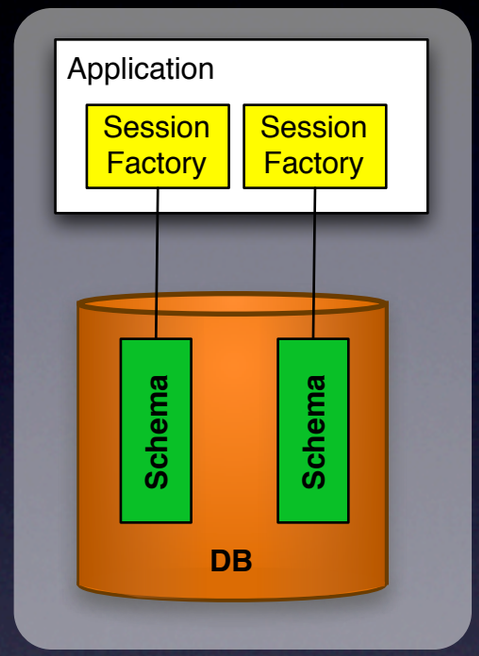
Logical separation

- All customers share tables
- Manual or Hibernate Filter

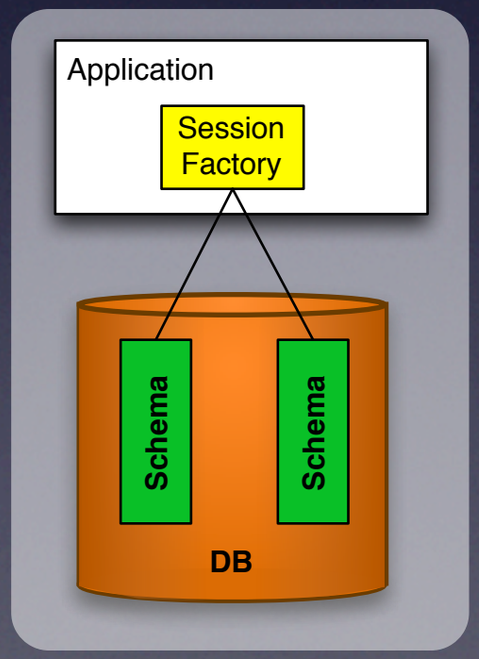


One user per schema

- One SessionFactory per schema



- Rewrite SQL



Use database security

- Map JAAS credentials to DB credentials
- One connection (pool) per user

Oracle security

- Oracle VPD
- Application defines active user

Storing in multiple databases

SessionFactory == DB

- Same schema across DBs
- Expensive in RAM
- Data isolated

Sharing state across `SessionFactorys` is probably doable

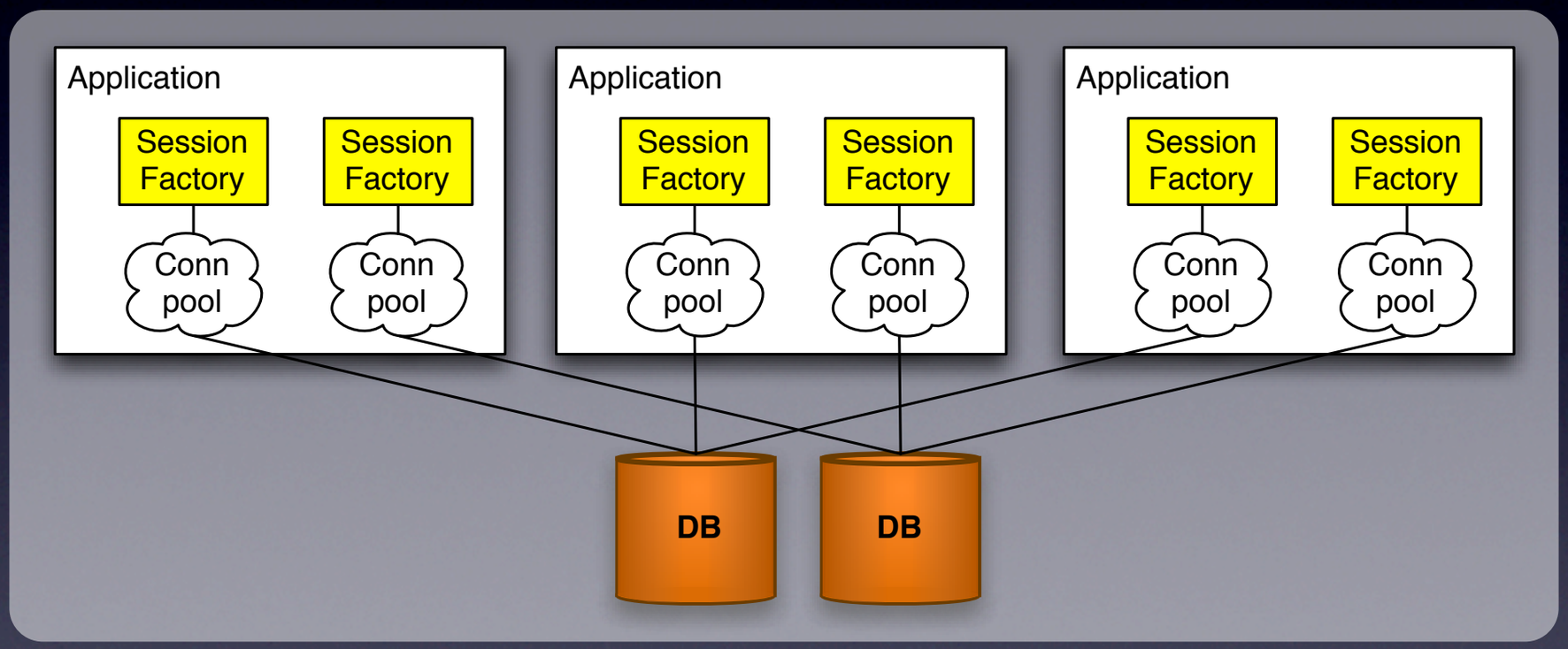
How many customer per DB?

- One
- One per schema
- Several per schema

- Dispatch customer to the right SessionFactory

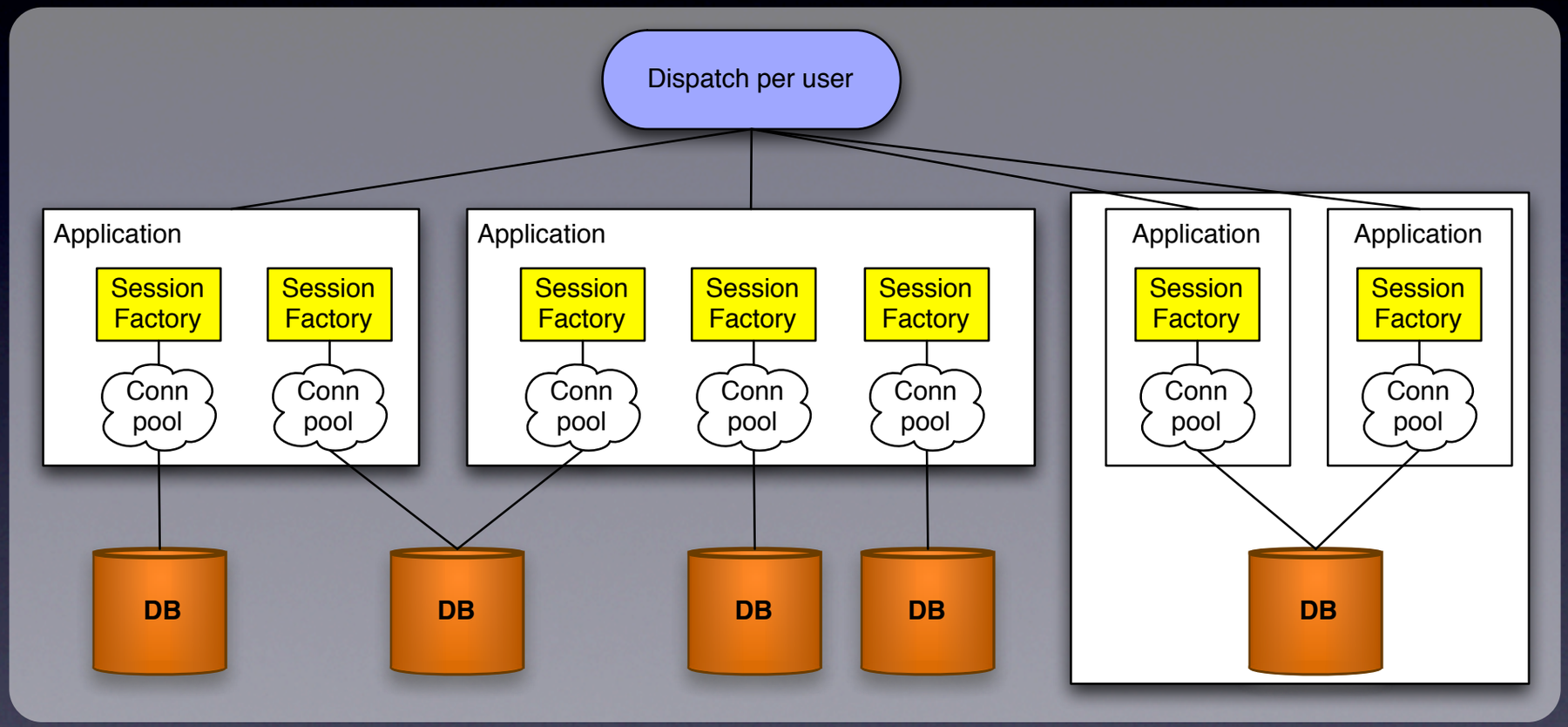
Adjusting the application layer

Homogeneous nodes



- Memory
- Too many connections
- Slow to start

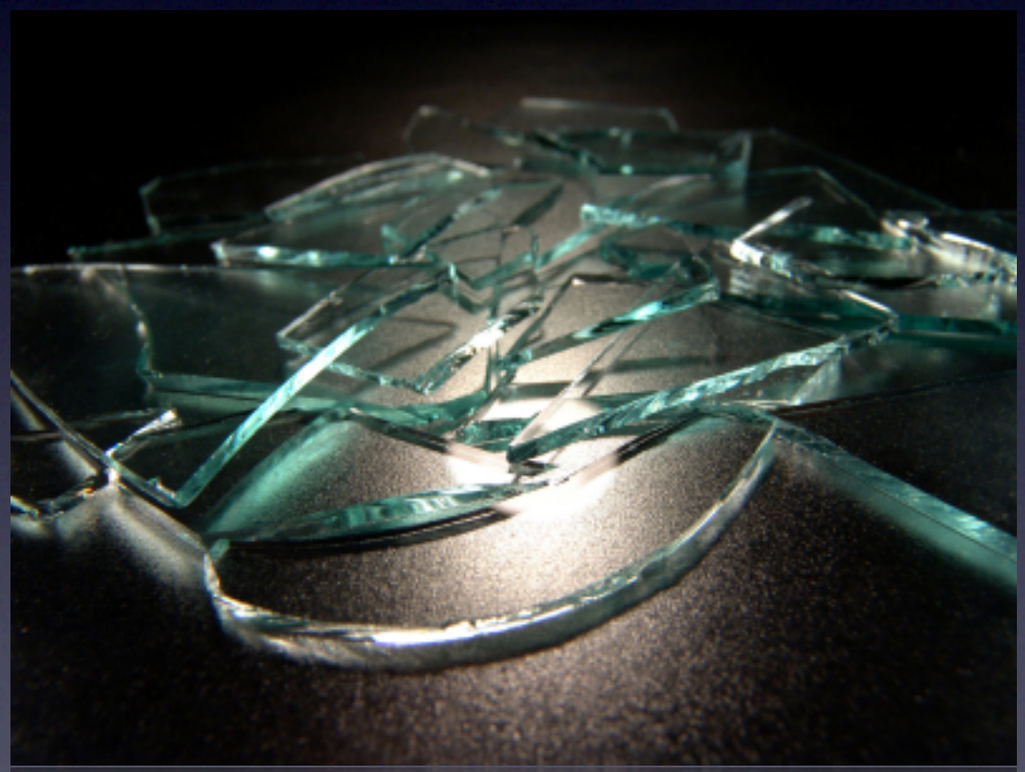
Specialized nodes



- Load balancing rules
- Easy scalability
- Efficient resource-wise

What if you need to query all your data?

Hibernate Shards

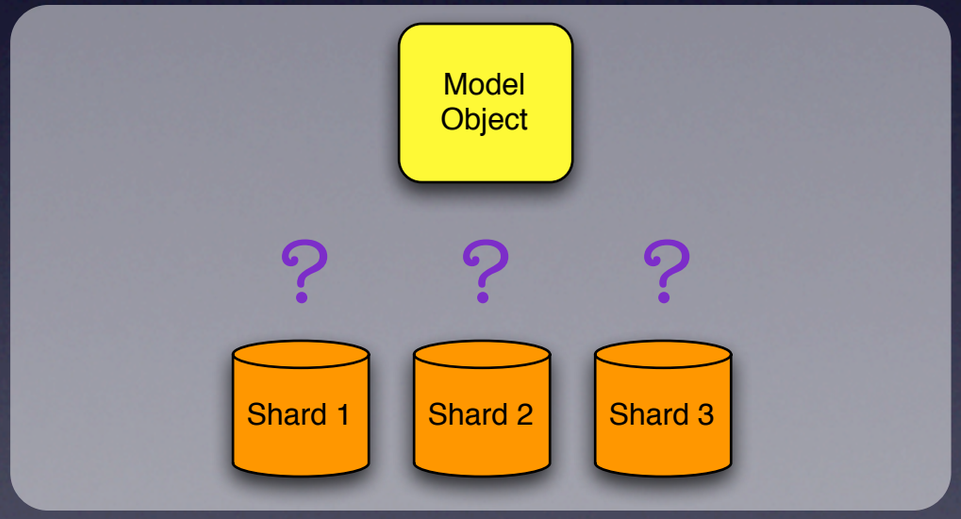


Simplified Horizontal Partitioning

- Separates app logic from federation logic
- Standard Hibernate API
- Unified view of your data

Shard Strategy

- Federation logic is application specific
  - Selection
  - Resolution
  - Access



Shard Selection

- On which shard do we create the record?
  - Round robin
  - Capacity based
  - Attribute based
  - Performance based

Shard Resolution

- On which shard do we find the record?
  - Exhaustive search
  - Map ID ranges to shards
  - Distributed cache

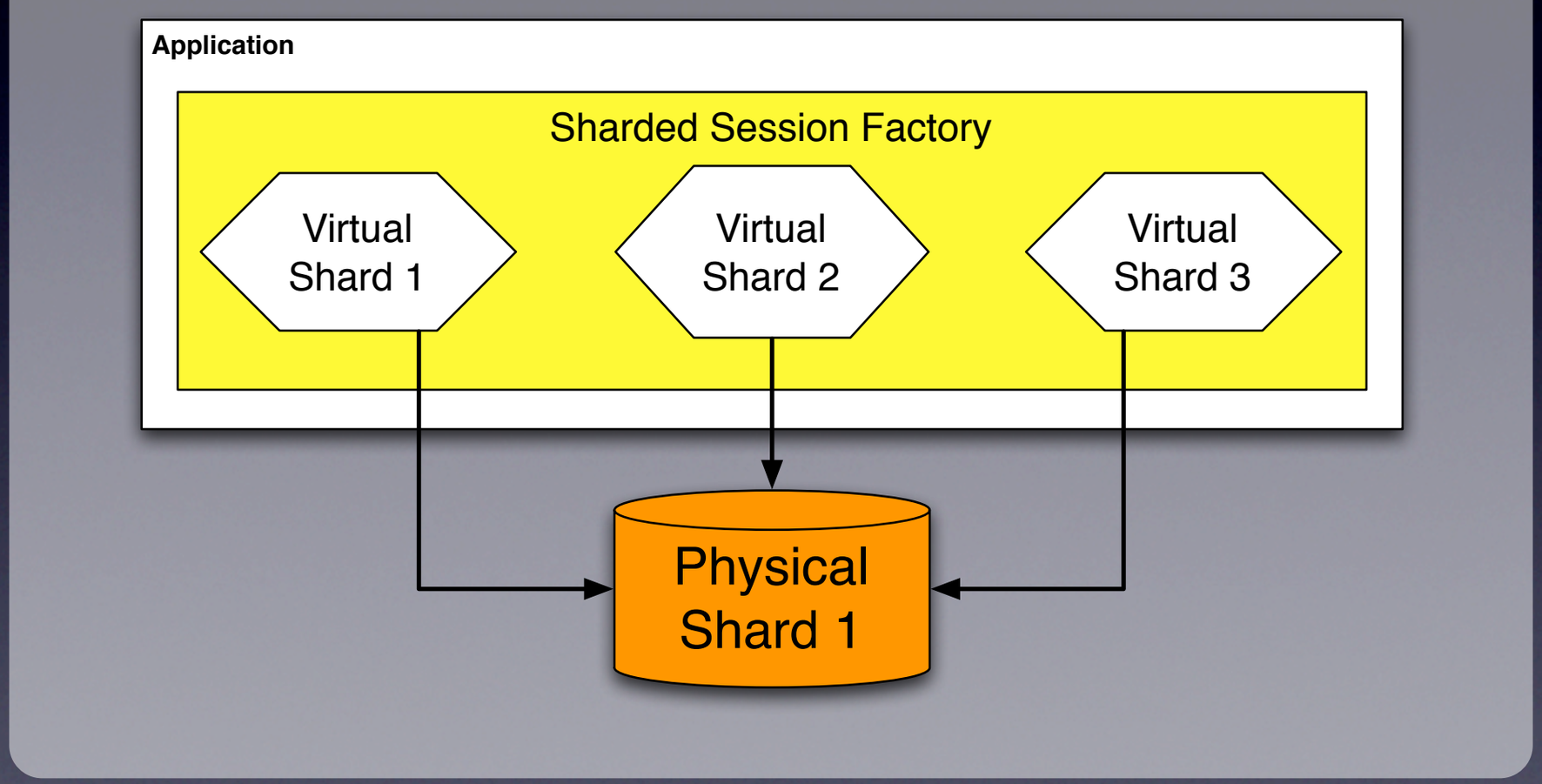
Shard Access

- How do we apply operations across shards?
  - Serially
  - In parallel (bring your own thread pool)
  - Hybrid

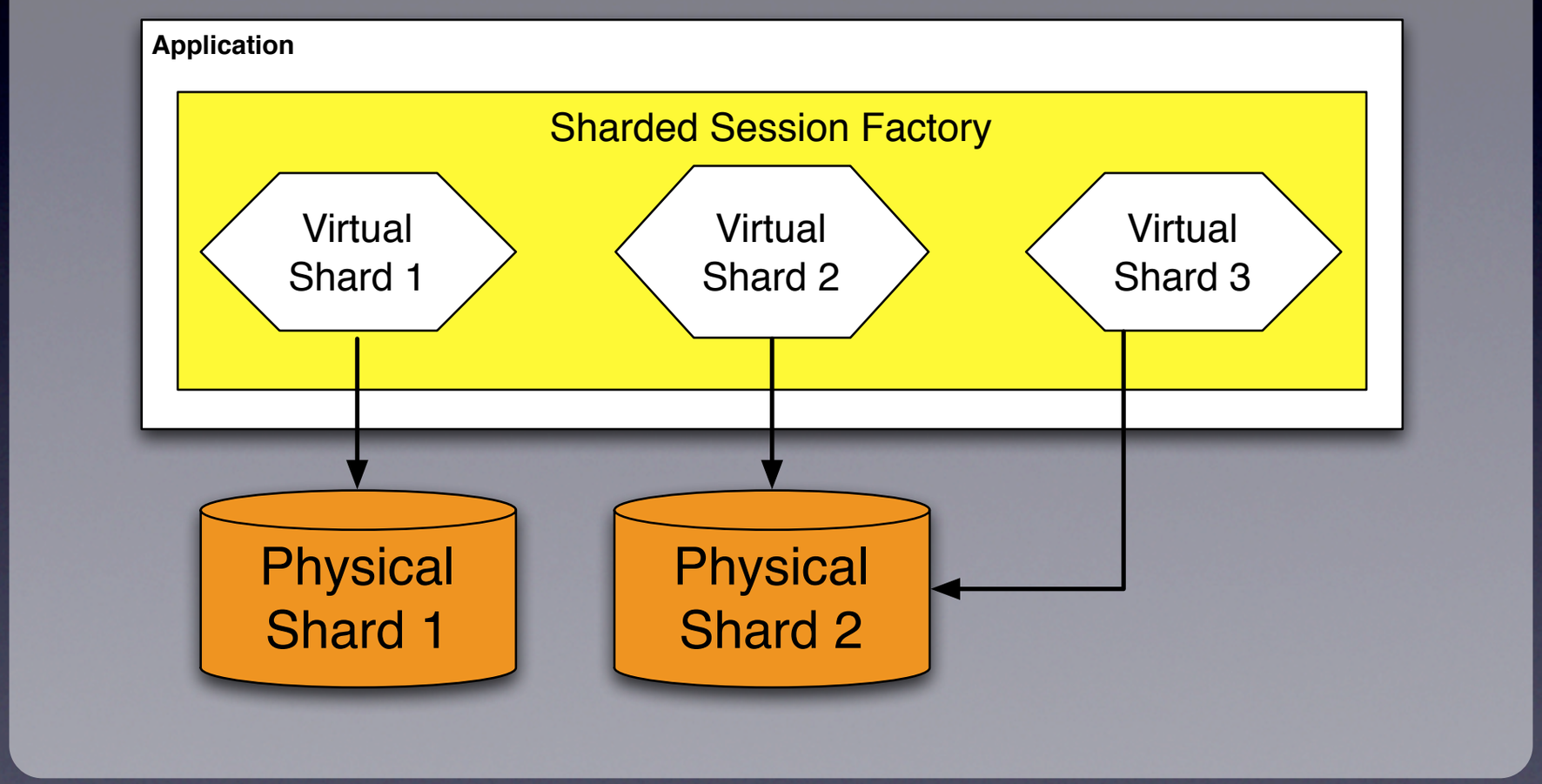
Writing the app is the easy part

- Operational challenges/risks are amplified
- Virtual shards can help

Virtual Shards



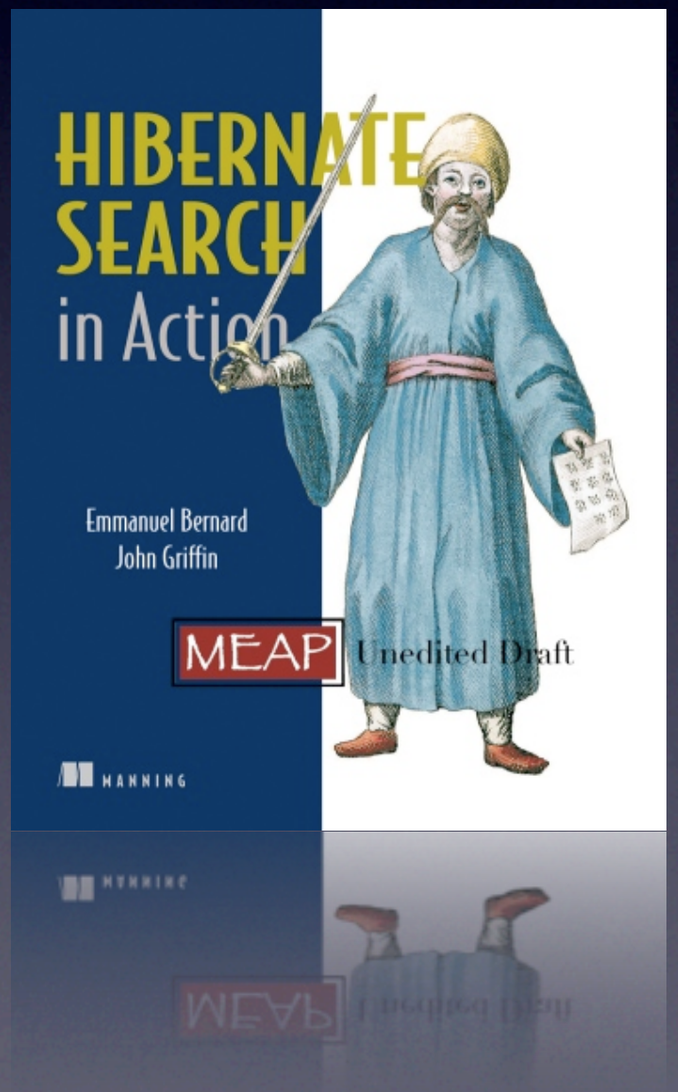
Virtual Shards



Coming Soon

- Static Data
- Full-fledged ShardedQuery
- JPA

Hibernate Search



Full-text search your domain objects

- Hibernate + Lucene
- Same programmatic model
- Index synchronized

Human queries

- Data set
- Word centric
- Typos / Synonyms
- Relevance

SQL underperforms

- Wildcard
  - Table/Index full scan
- Multiple joins
- Relevance?

Customer

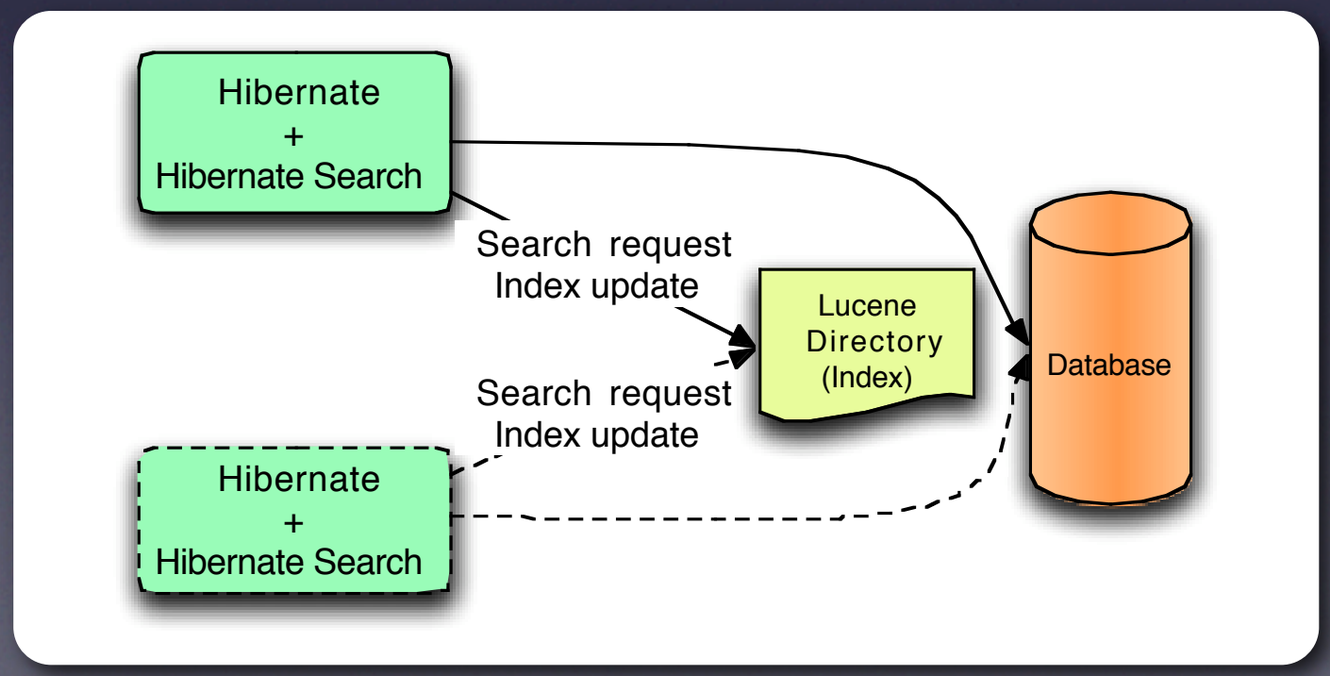
DBA

Full-text search

- Move load away from the DB
- Replace or complement searches

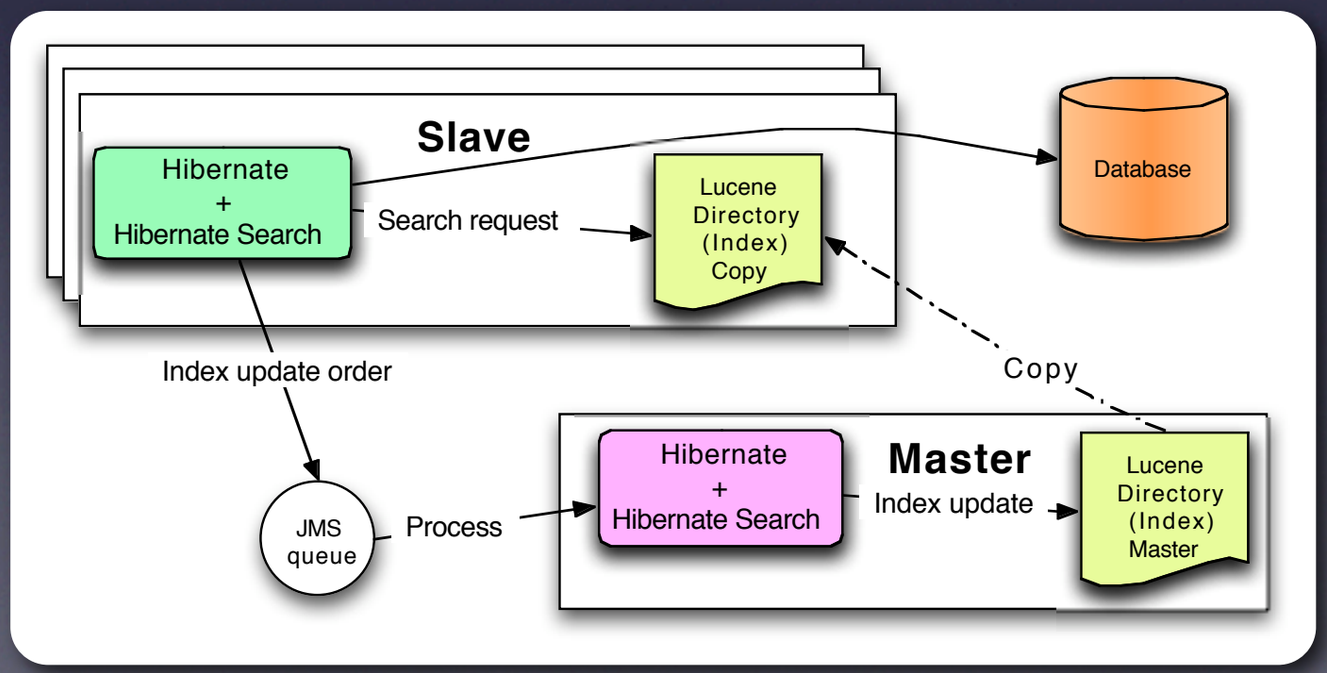
Scalability
Symmetric cluster

- Distributed lock
- Immediate visibility
- Affects front end



Scalability
Asymmetric cluster

- Search local / change sent to master
- Asynchronous indexing (delay)
- No front end extra cost / good scalability



Scalabilities (sic)

- Hibernate a good citizen
- Isolating customer data
- Deal with multiple databases
- Hibernate Shards
- Hibernate Search

Q&A

- For more infos
  - Hibernate Search in Action
  - Java Persistence with Hibernate
  - Max's podcasts
    - http://google-code-updates.blogspot.com/2007/08/google-developer-podcast-episode-six.html
    - http://www.javaworld.com/podcasts/jtech/2008/072408jtech.html
  - hibernate.org