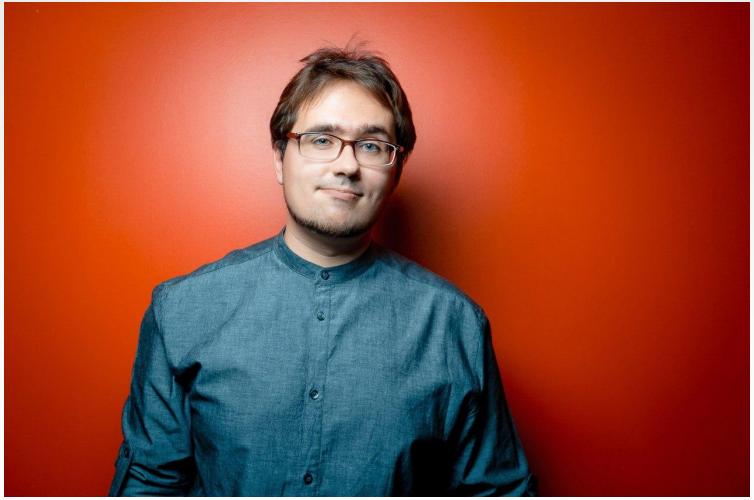
Mom, I so wish Hibernate for my NoSQL database…

Speaker : Alexey Zinoviev

About



- I am a scientist. The area of my interests includes machine learning, traffic jams prediction, BigData algorythms.
- But I'm a programmer, so I'm interested in NoSQL databases, Java, Android, Hadoop, Spark.

Introduction

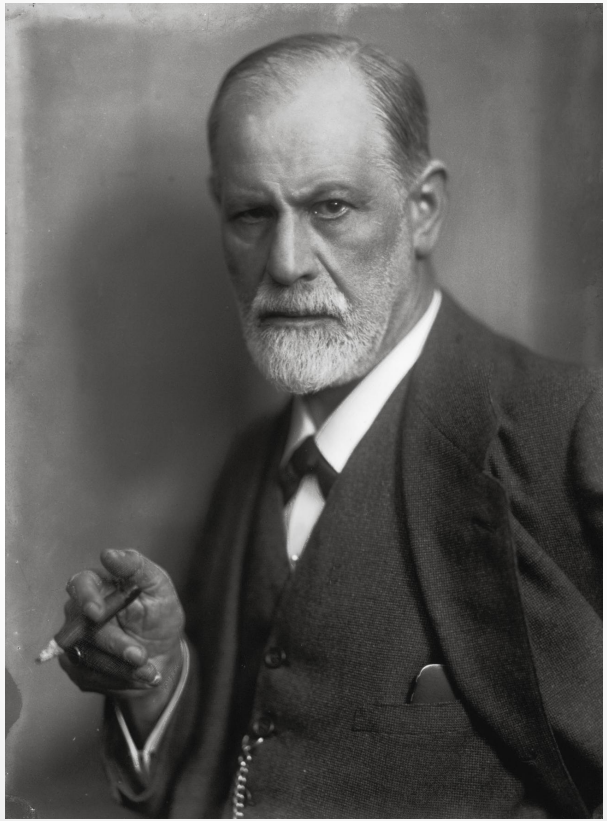
The Good Old Days

One of these fine days…

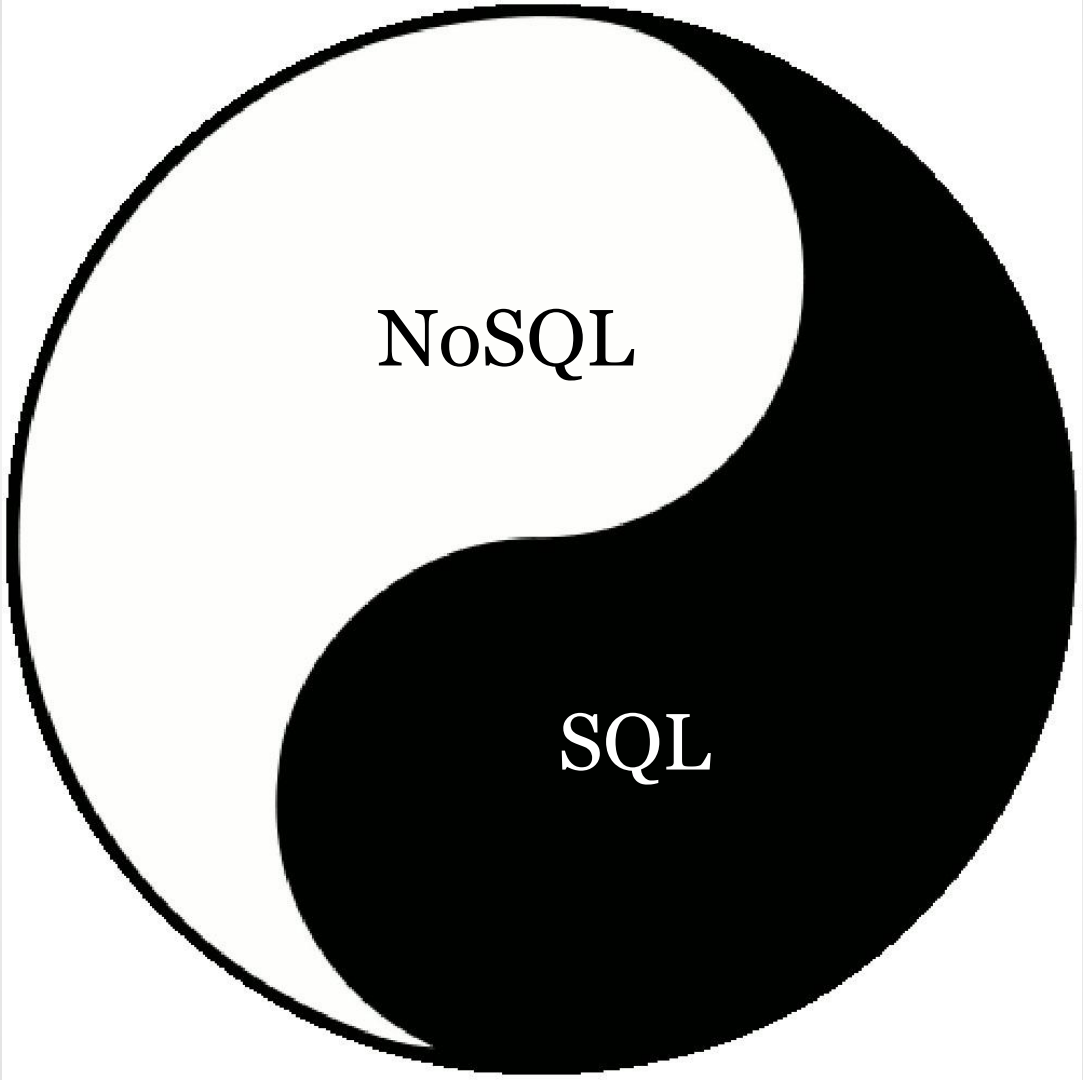
We have a NoSQL job for you, son!

You like SQL and HATE nontraditional data

Let’s talk about it, Java-boy…

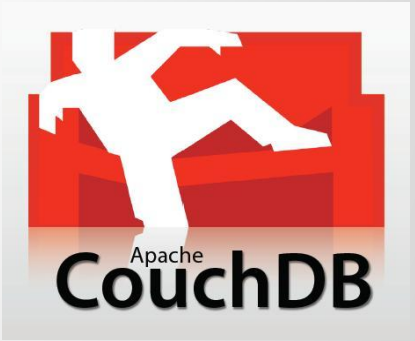


NoSQL

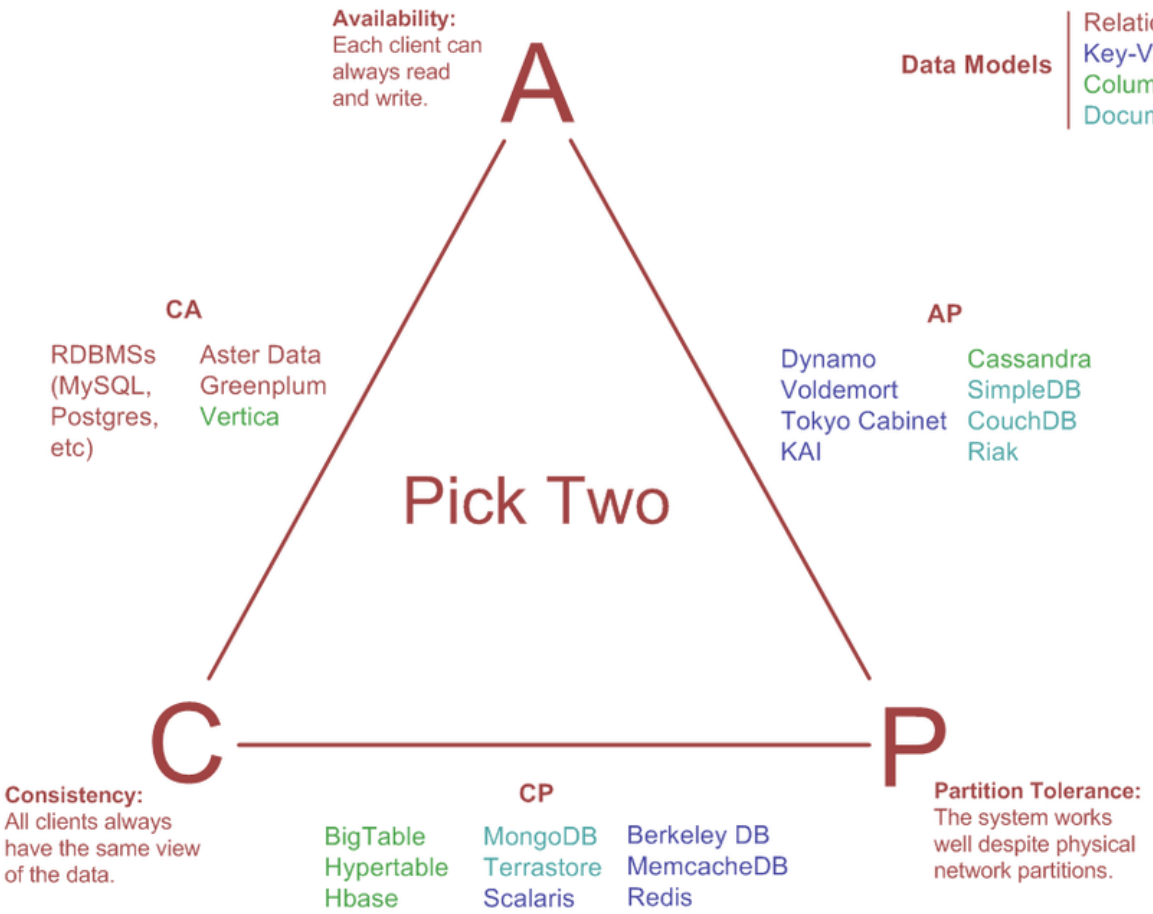

NoSQL
SQL

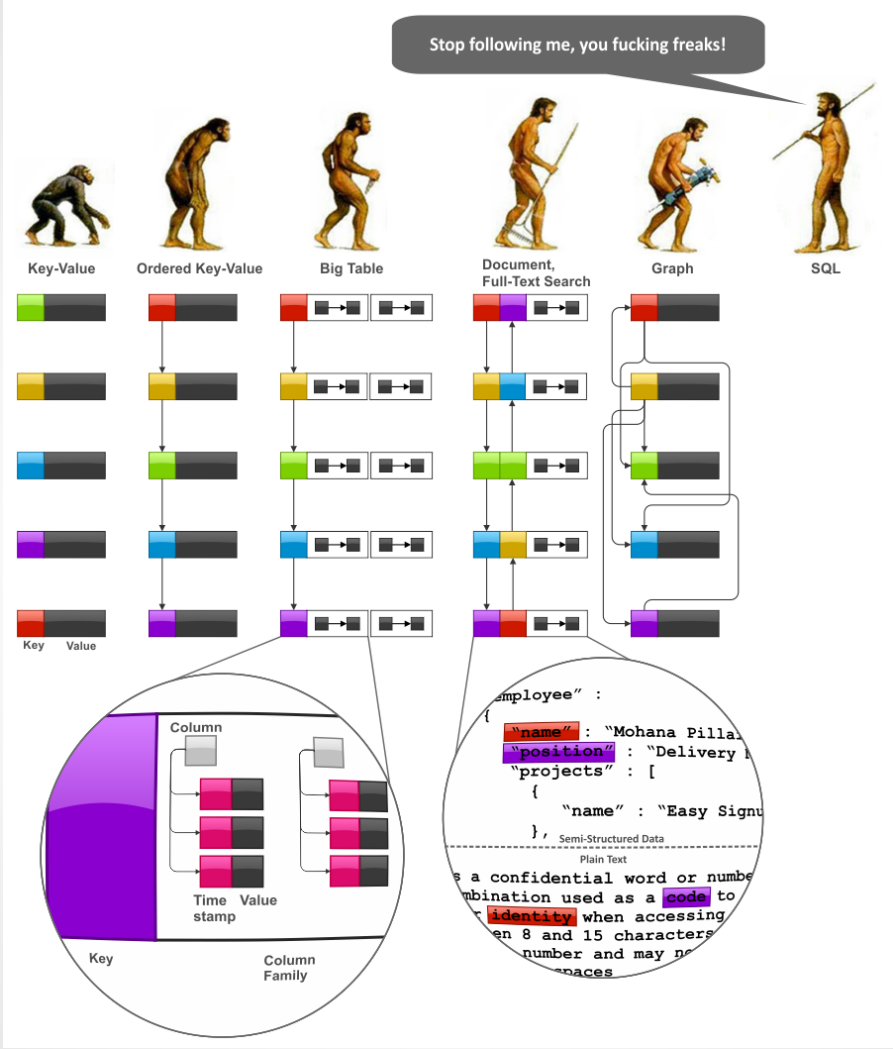
The Database Market

Have your ever heard about...



Visual Guide to NoSQL Systems




Stop following me, you fucking freaks!
Key-Value
Ordered Key-Value
Big Table
Document, Full-Text Search
Graph
SQL
Key
Value
Column
Time stamp
Value
Key
Column Family
"name" : "Mohana Pilla
"position" : "Delivery
"projects" : [
"name" : "Easy Signu
Semi-Structured Data
Plain Text
a confidential word or numbe
bination used as a code to
identity when accessing
en 8 and 15 characters
number and may n
paces

Flower varieties I

Data Model	Performance	Scalability	Flexibility	Complexity	Functionality
Key–value Stores	high	high	high	none	variable (none)
Column Store	high	high	moderate	low	minimal
Document Store	high	variable (high)	high	low	variable (low)
Graph Database	variable	variable	high	high	graph theory
Relational Database	variable	variable	low	moderate	relational algebra

Flower varieties II

Database	Data model	Query API	Data storage system
Cassandra	Column Family	Thrift	Memtable/SSTable
CouchDB	Documents	Map/Reduce	Append-only-B-tree
Hbase	Column Family	Thrift, REST	Memtable/SSTable on HDFS
MongoDB	Documents	Cursor	B-tree
Neo4j	Edges/Verticies	Graph	On-disk linked lists
Riak	Key/Value	Nested hashes, REST	Hash

Flower varieties III

Database	Secondary indexes	MapReduce	Free form queries
Cassandra	yes	no	CQL (no joins)
CouchDB	yes	JavaScript	Views
Hbase	no	Hadoop	weak support
MongoDB	yes	JavaScript	Full stack without joins
Neo4j	yes (with Lucene)	no	Search, graph operations
Riak	yes	JavaScript, Erlang	weak support, Lucene

But.. how about @ and JPA and ..

Cassandra World

Hector

- CRUD + column iteration, Partial support of JPA standart
- Consistency Level can be set per Column Family and per operation type (Read, Write)
- Based on Thrift RPC protocol (1 response per 1 request)
- Mapping a Collection (POJO property) to columns
- Inheritance through ‘single table’
- Custom converters to/from byte[]

```java
@Entity
@DiscriminatorValue("table_desk")
public class Desk extends BasicTable {

    @Column(name = "desk_type")
    private String deskType;

    @Column(name = "drawerList")
    private List<Drawer> drawerList = new ArrayList<Drawer

    public List<String> getDrawerList() {
       return drawerList;
    }
```

```java
@Entity
@Table(name="Furniture")
@Inheritance
@DiscriminatorColumn(name = "type", discriminatorType = DiscriminatorType.STRING)
public abstract class Furniture {
  @Id
  private int id;

  @Column(name="material")
  private String material;

  @Column(name="color")
  private String color;

  //  getters/setters required, but not showing to conserve space

}
```

```java
package com.mycompany.furniture;

// imports omitted

@Entity
@DiscriminatorValue("chair")
public class Chair extends Furniture {

  @Column(name="recliner")
  private boolean recliner;

  @Column(name="arms")
  private boolean arms;
```

DataStax Java Driver

- DataStax developed a new protocol that doesn't have RPC limitations (Asynchronous I/O)
- Low-level API with simple mapping
- Works with CQL3
- QueryBuilder reminds CriteriaAPI
- Accessor-annotated interfaces

```
public class AsynchronousExample extends SimpleClient {
    public AsynchronousExample() {
    }

    public ResultSetFuture getRows() {
        Query query = QueryBuilder.select().all().from("simplex", "songs");
        return getSession().executeAsync(query);
    }

    public static void main(String[] args) {
        AsynchronousExample client = new AsynchronousExample();
        client.connect("127.0.0.1");
        client.createSchema();
        client.loadData();
        ResultSetFuture results = client.getRows();
        for (Row row : results.getUninterruptibly()) {
            System.out.printf( "%s: %s / %s\n",
                    row.getString("artist"),
                    row.getString("title"),
                    row.getString("album") );

        }
        client.dropSchema("simplex");
        client.close();

    }
}
```


@Accessor
public interface UserAccessor {
@Query("SELECT * FROM complex.users WHERE id = :id")
User getUserNamed(@Param("userId") UUID id);
@Query("SELECT * FROM complex.users WHERE id = ?")
User getOnePosition(UUID userId);
@Query("UPDATE complex.users SET addresses[:name]=:a
ResultSet addAddress(@Param("id") UUID id, @Param("
@Query("SELECT * FROM complex.users")
public Result<User> getAll();
@Query("SELECT * FROM complex.users")
public ListenableFuture<Result<User>> getAllAsyn

Do you want more adventures?

Other Cassandra's OM

- Achilles : well documented and provides transactions
- Astyanax : connection pool, thread safety and pagination
- Pelops : old project, good bycicle
- PlayORM : strange but powerful thing
- Easy-Cassandra : simple annotations + CRUD
- Thrift as low level API

Mongo World

Morphia

- Integrated with Spring, Guice and other DI frameworks
- Lifecycle Method Annotations (@PrePersist, @PostLoad)
- Built on top of Mongo Java Driver
- More better than old-style BSON-object quering
- Fluent Query API :

```
ds.createQuery(MyEntity.class).filter("foo >",12).order("date, -foo");
```

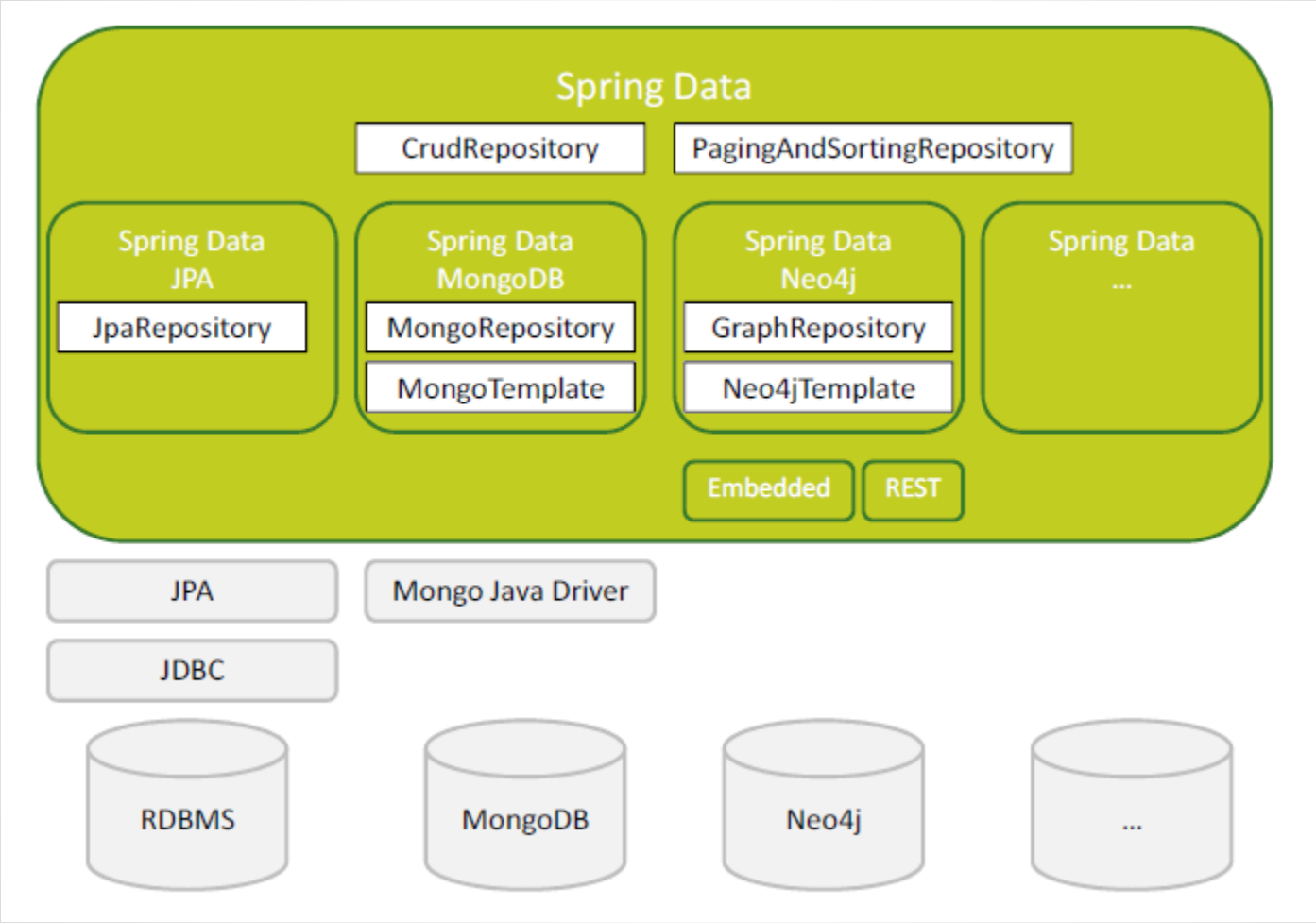
```java
@Entity("employees")
class Employee {

  @Id ObjectId id;

  Address address;

  Key<Employee> manager;

  @Reference List<Employee> underlings = new ArrayList<

  @Property("left") Date endDate;

  //fields can be indexed for better performance
  @Indexed boolean active = false;

  //fields can loaded, but not saved
  @NotSaved String readButNotStored;

  //Lifecycle methods -- Pre/PostLoad, Pre/PostPersist.
  @PostLoad void postLoad(DBObject dbObj) { ... }
```

```
@EntityListeners(BackAccountWatcher.class)
Public class BankAccount {
   @Id String id;
   Date lastUpdated = new Date();
}

class BankAccountWatcher{

   @PrePersist void prePersist(BankAccount act) {act.lastUpdated

}
```

.. or

```
class BankAccount {
   @Id String id;
   Date lastUpdated = new Date();

   @PrePersist void prePersist() {lastUpdated = new Date();}
}
```

```
DBCollection table = db.getCollection("user");

BasicDBObject document = new BasicDBObject();
document.put("name", "alex");
document.put("age", 27);
document.put("createdDate", new Date());
table.insert(document);


BasicDBObject query = new BasicDBObject();
query.put("name", "alex");

BasicDBObject newDocument = new BasicDBObject();
newDocument.put("name", "alex-updated");

BasicDBObject updateObj = new BasicDBObject();
updateObj.put("$set", newDocument);

table.update(query, updateObj);
```


Spring Data
CrudRepository
PagingAndSortingRepository
Spring Data JPA
JpaRepository
Spring Data MongoDB
MongoRepository
MongoTemplate
Spring Data Neo4j
GraphRepository
Neo4jTemplate
Spring Data ...
Embedded
REST
JPA
Mongo Java Driver
JDBC
RDBMS
MongoDB
Neo4j
...

Spring Data MongoDB

- Templating : connection configs, collection lifecycle (create, drop), Map/Reduce + Aggregation
- Mapping: @Document, @Index, @Field
- Repository support: geospatial queries, queries derived from method signatures (at runtime)
- Paging, sorting, CRUD operations

```
public class Order {
    @Id private String id;
    private Date date;
    @Field("custInfo") private String customerInfo;
    List<Item> items;
}

public class Item {
    private int quantity;
    private double price;
    @Field("desc") private String description;
}


public interface OrderRepository extends MongoRepository

List<Order> findByItemsQuantity(int quantity);

List<Order> findByItemsPriceGreaterThan(double price);

}
```

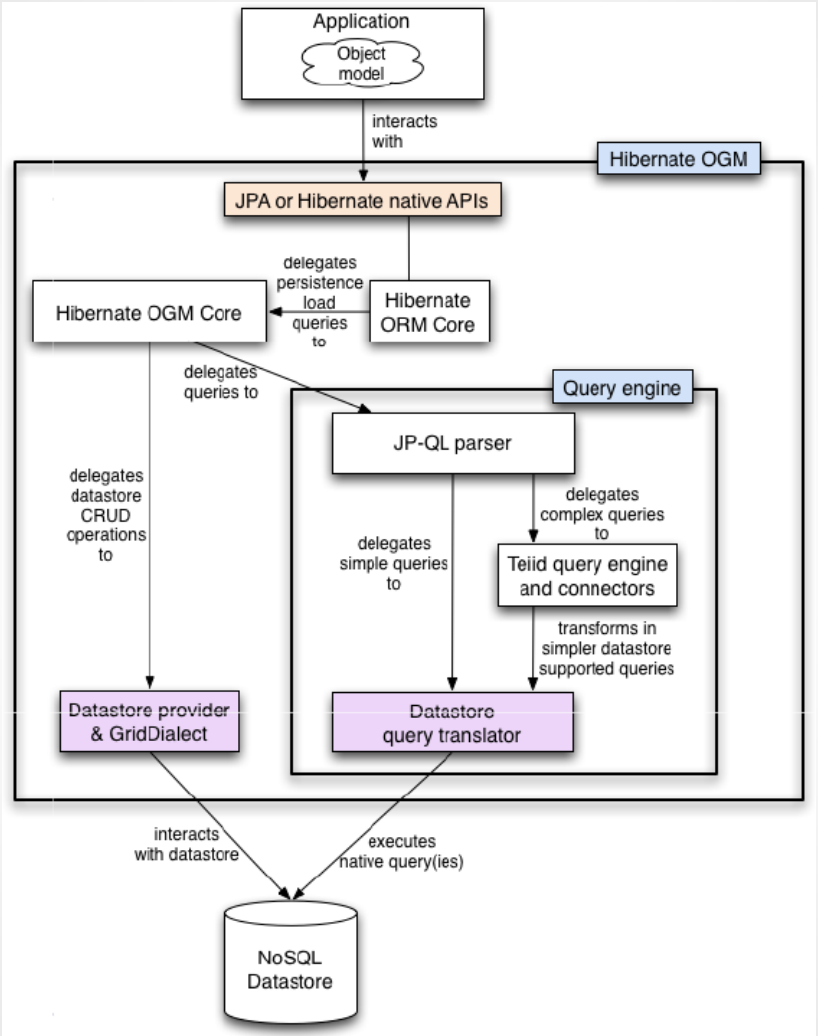
Mongo - mongo

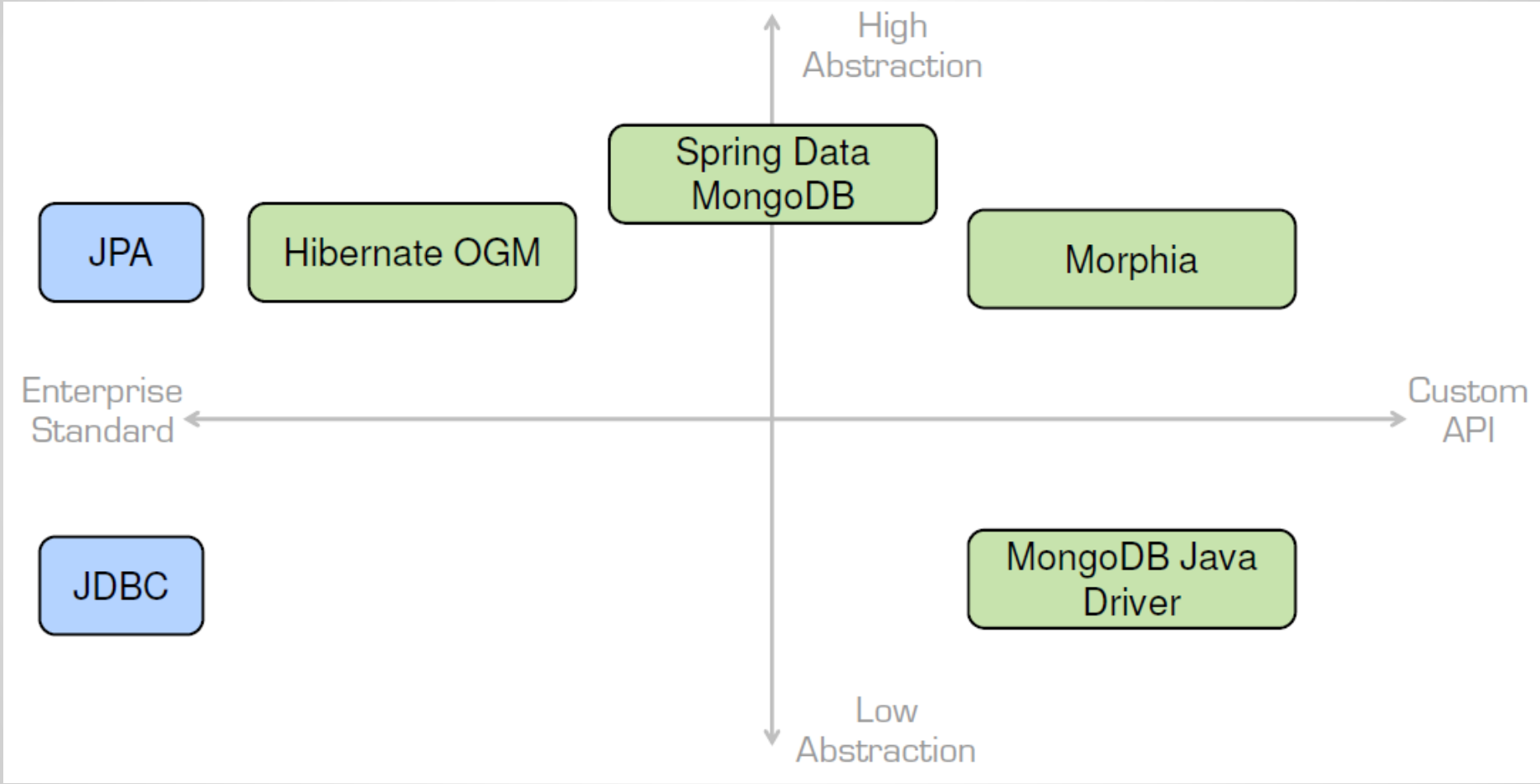
- Jongo : mongo - shell queries in Java-code
- EclipseLink : different support of different NoSQL databases
- MJORM : Google Code, XML mapping + MQL (SQL syntax for Mongo data extracting)
- DataNucleus : support many Js as JDO, JPA

Hibernate in NoSQL world

Hibernate OGM

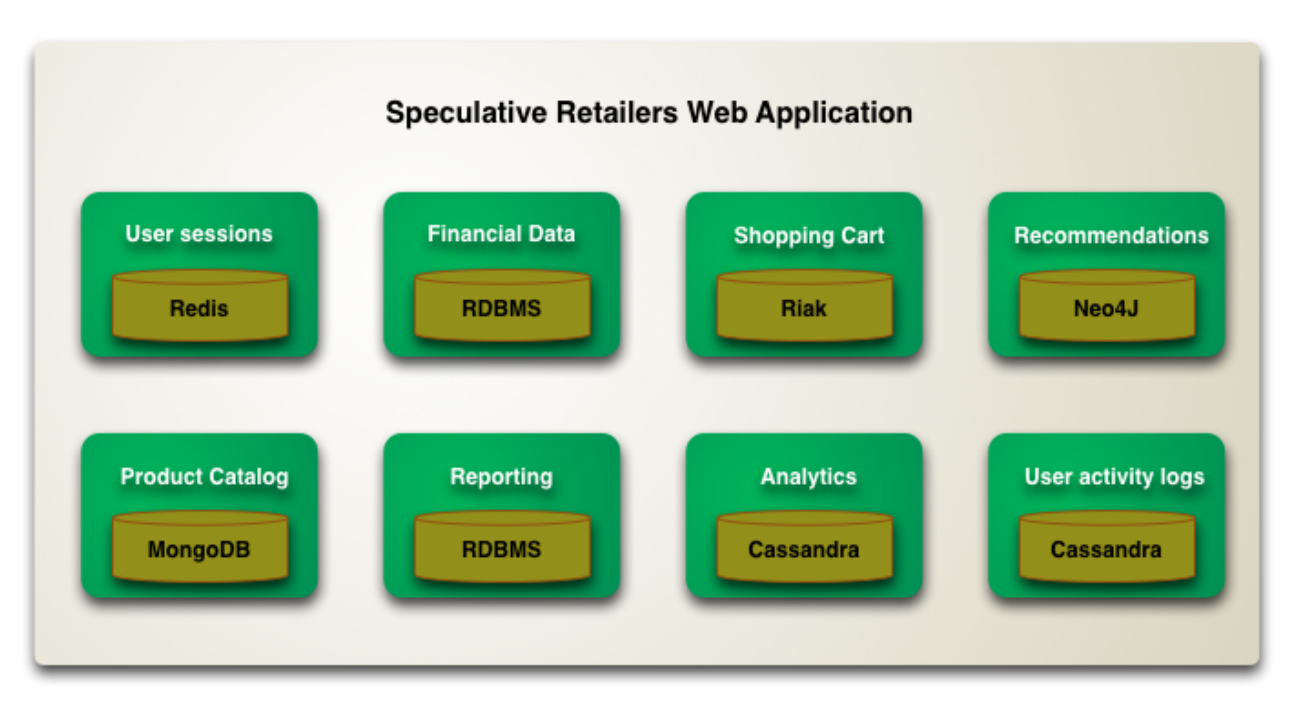
- Java Persistence (JPA) support for NoSQL solutions
- JP-QL queries are converted in native backend queries
- Hibernate Search as indexing engine and use full-text queries
- You can call flush(), commit() and demarcate transactions
- It supports only MongoDB, Neo4j, Infinispan, Ehcache


Application
Object model
interacts with
Hibernate OGM
JPA or Hibernate native APIs
delegates persistence load queries to
Hibernate OGM Core
Hibernate ORM Core
delegates queries to
Query engine
JP-QL parser
delegates datastore CRUD operations to
delegates simple queries to
delegates complex queries to
Teiid query engine and connectors
transforms in simpler datastore supported queries
Datastore provider & GridDialect
Datastore query translator
interacts with datastore
executes native query(ies)
NoSQL Datastore


High
Abstraction
Spring Data
MongoDB
JPA
Hibernate OGM
Morphia
Enterprise
Standard
Custom
API
JDBC
MongoDB Java
Driver
Low
Abstraction

Polyglot Persistance

Polyglot Persistance



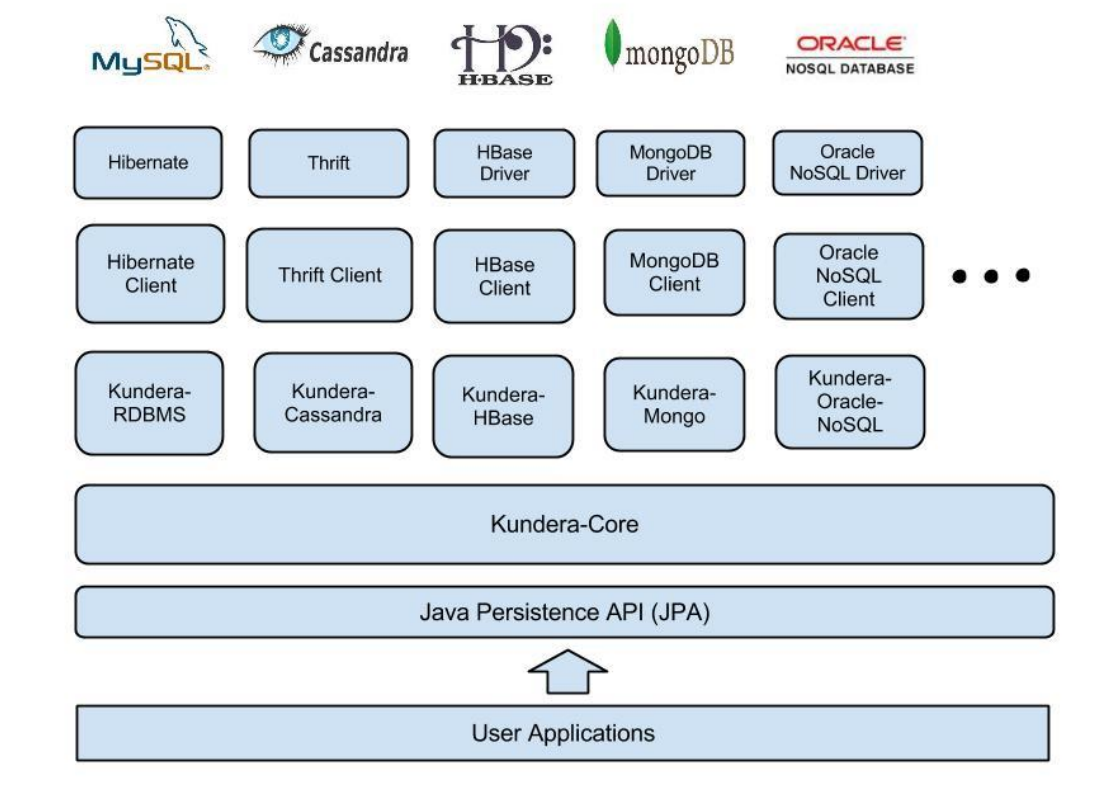
Polyglot Persistance

- Redis: Rapid access for reads and writes. No need to be durable
- RBDMS: Needs transactional updates and has tabular structure.
- Riak: Needs high availability across multiple locations. Can merge inconsistent writes
- Neo4j: Rapidly traverse links between friends and ratings.
- MongoDB: Lots of reads, infrequent writes. Powerful aggregation mechanism.
- Cassandra: Large-scale analytics on large cluster. High volume of writes on multiple nodes

Kundera : Polyglot approach

- Atomicity guarantee and Transaction management
- Strictly JPA 2.1 compatible
- It supports Cassandra, Mongo, Hbase, Redis, Neo4j and etc
- @Embedded and @ElementCollection for ColumnFamily and nested documents
- OneToMany, OneToOne, ManyToMany relationships
- Not full JPQL support for different database

Kundera's holes



If you developing a project for...

- a U.S. company - Morphia or Hector
- a transnational company with history of merging - Kundera
- a Deutsche Bank - Hibernate OGM or SpringData
- a Russian company, or one in in the former U.S.S.R. - native drivers is the best approach (you will spend so many time)
- ... just to play - Jongo, Easy - Cassandra and EclipseLink

Your Country Needs You!

- Know your cases and data!
- Choose right database!
- Choose right framework!
- In Soviet Russia backends are waiting you!

Your questions?