Post-relational Databases

- Five genres of post-relational databases
- The Object-oriented database
- MongoDB: a document database
- Data science, big data and data analytics

The impedance mismatch problem

- Combination of SQL with a host language
  - mix of declarative and procedural programming paradigms
  - two completely different data models
  - different set of data types
- Interfacing with SQL is not straightforward
  - data has to be converted between host language and SQL due to the impedance mismatch
  - ~30% of the code and effort is used for this conversion!
- The problem gets even worse if we would like to use an object-oriented host language
  - Solution: post-relational or NoSQL databases

Post-relational databases

- Post-relational, or NoSQL (not only SQL) databases refer to databases intended to support an alternative logical data model to the transaction processing oriented relational model. When data is modeled and stored in a format that differs substantially from the relational model, then the database can be characterized as a post-relational database.
- Driven by Web2.0 applications - Web search, social networking sites and E-commerce.
  - Google, Amazon, Facebook, Twitter.
- The end of the assumption that SQL and ACID are the tools to solve all our data management problems.

NoSQL databases: big Web site databases

Data management requirements for the Big sites:

- Ability to handle unlimited data: the amount of data generated by 1 billion users of Facebook or Google.
- Real time data stream processing needs: real-time feeds and machine learning.

The data volume is so huge that the relational technology can not handle. Big Web sites use some form of NoSQL databases for data storage. Examples:

Hadoop HBase, MongoDB, Cassandra,
Amazon's DynamoDB, Google's BigTable

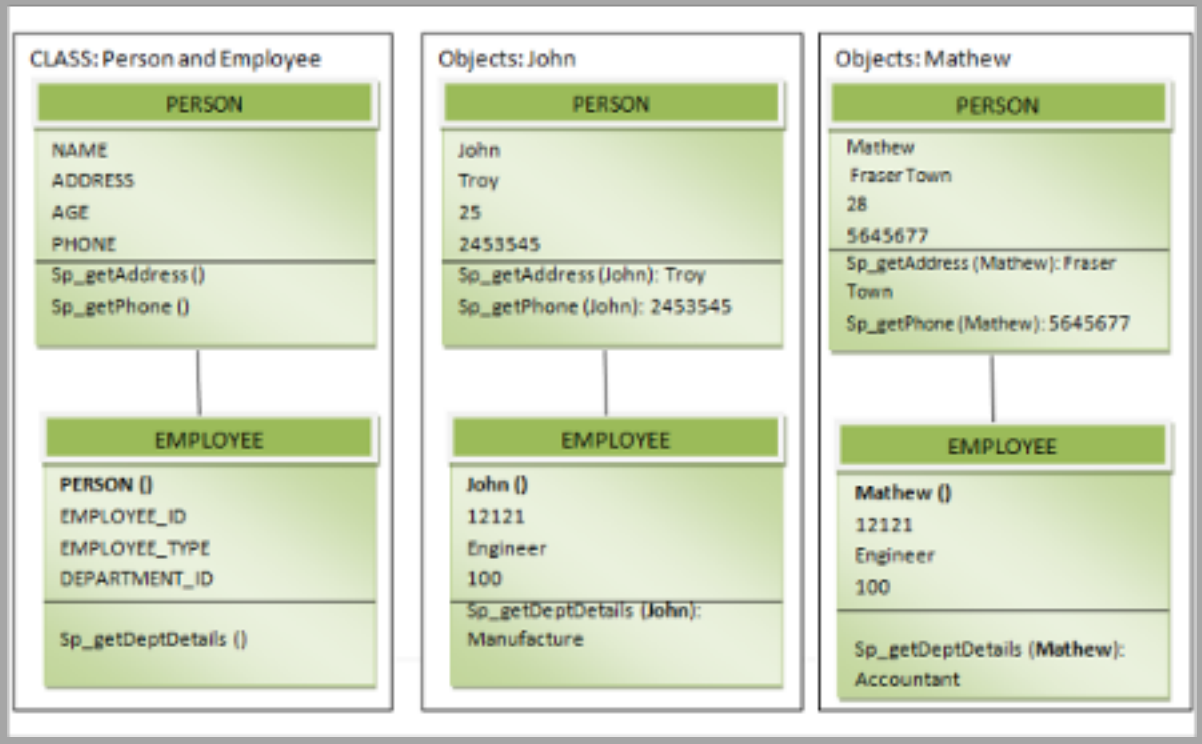
Semistructured and unstructured data can not be easily stored and processed in the relational format.

Five genres of post-relational databases

- Object-oriented or object-relational databases
  - db4o, ObjectStore, Objectivity, Versant, ...
- Column stores:
  - BigTable (Google), HBase (Apache Hadoop)
- Document stores:
  - CouchDB, MongoDB, ...
- Key-value (tuple) stores:
  - Membase, Redis, ...
- Graph databases:
  - Neo4j, …

Post-relational 1: Object-oriented databases

- ODBMSs use object data model, the same as object-oriented programming languages -- no object-relational impedance mismatch (owing to the uniform model).



Post-relational 2: Column stores

- A column-oriented database serializes all of the values of a column together. In a column-oriented system, the primary key is the data, mapped from the row ID. Efficient for column-based queries.

Row-oriented system

RowId	EmpId	Lastname	Firstname	Salary
001	10	Smith	Joe	40000
002	12	Jones	Mary	50000
003	11	Johnson	Cathy	44000
004	22	Jones	Bob	55000

Column-oriented system

```
10:001,12:002,11:003,22:004;
Smith:001,Jones:002,Johnson:003,Jones:004;
Joe:001,Mary:002,Cathy:003,Bob:004;
40000:001,50000:002,44000:003,55000:004;
```

Post-relational 3: Document stores

The JASON document data model: Each document is a data record of different format (number and size of fields)

```
{
        first_name: “Paul”,
        surname: “Miller”,
        city: “London”,
        location: [45.123,47.232],
        cars: [
           { model: “Bentley”,
              year: 1973,
              value: 100000, ….},
           { model: “Rolls Royce”,
              year: 1965,
              value: 330000, ….},
        ]
}
```

Post-relational 4: Key-value stores

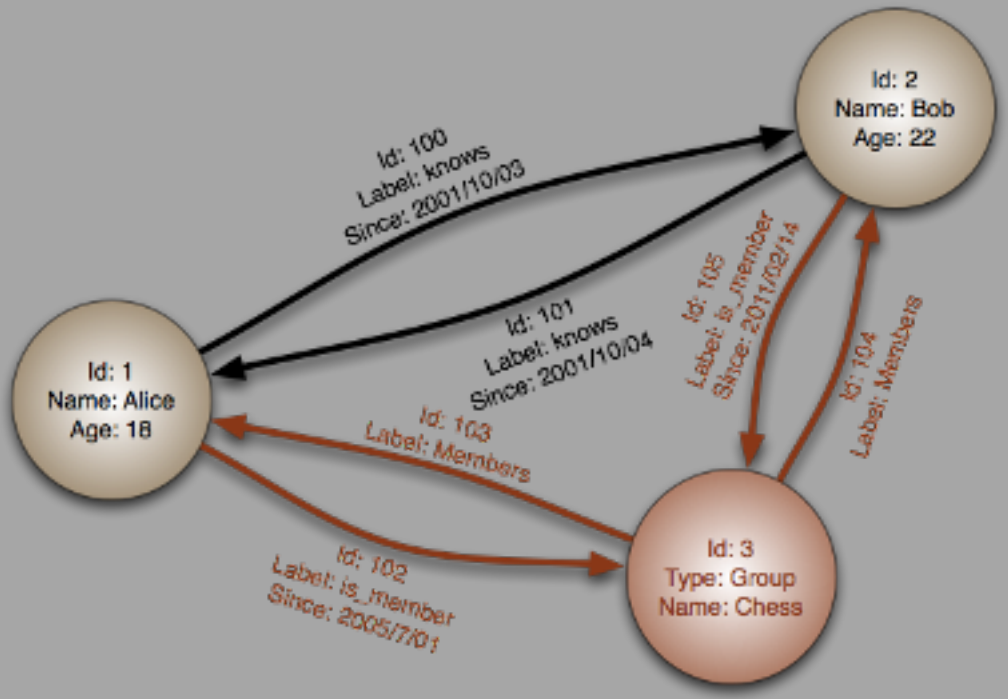
- A data object or record, identified by a key, contains data in different fields. No database schema is pre-defined.
- The key-value pair is used to keep data for the key.The key is the unique identifier for value.The key and value can be anything.

The list contains the stock ticker, whether its a "buy" or "sell" order, the number of shares, and the price.

Key	Value
123456789	APPL, Buy, 100, 84.47
234567890	CERN, Sell, 50, 52.78
345678901	JAZZ, Buy, 235, 145.06
456789012	AVGO, Buy, 300, 124.50

Post-relational 5: Graph databases

A graph database uses nodes and edges and properties to store data. The underlying storage engine uses the relational database or NoSQL technologies such as key-value store or document-oriented database.



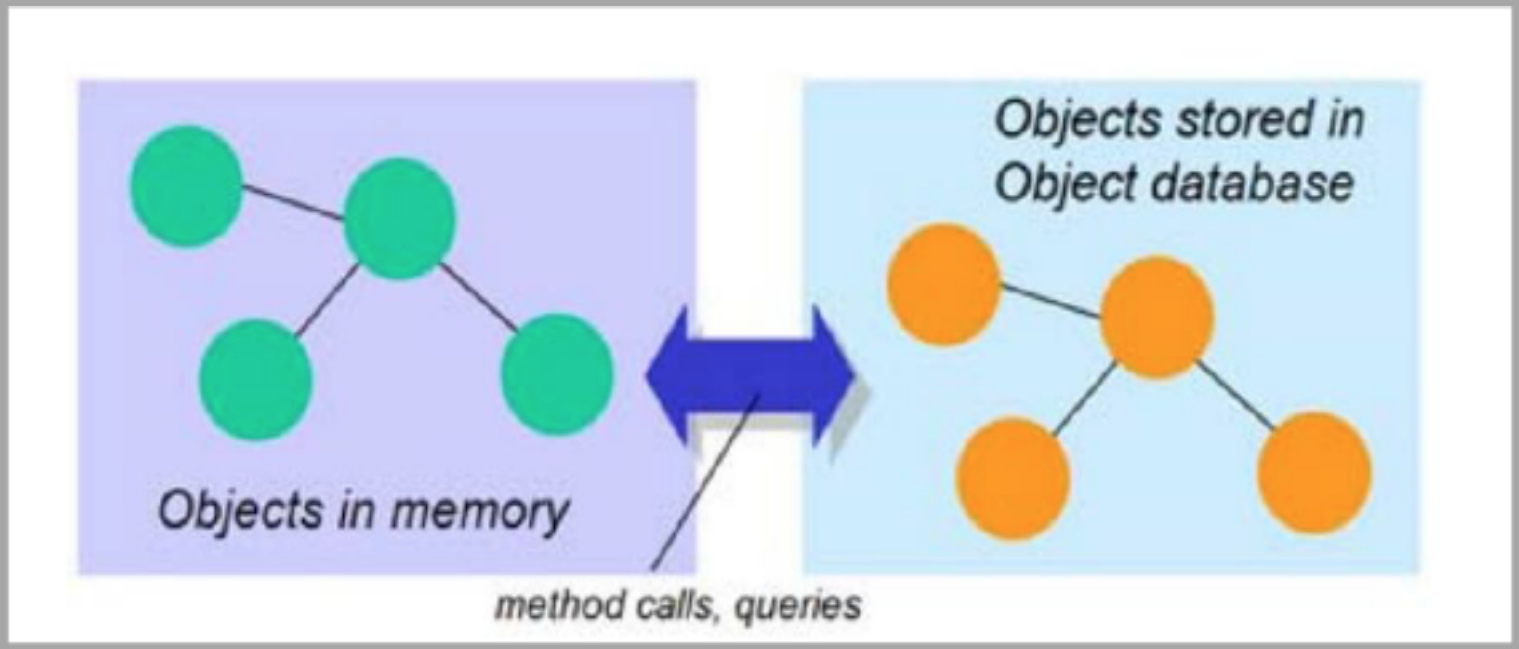
Post-relational databases: insides

We use two examples to explain in details the post-relational databases:

- The object-oriented database (1990s)
- The document database MongoDB (2009)

The object data model

The seamless interaction between in-memory data objects and persistent data objects.



Object-oriented databases

- An object database combines the features of an object-oriented language and a DBMS
  - treat data as objects
    - object identity
    - attributes and methods
    - relationships between objects
  - extensible type hierarchy
    - inheritance, overloading and overriding as well as customised types
  - declarative query language

Object-Relational databases

- The object-relational data model extends the relational data model: complex data types, object-oriented features, and extended version of SQL to deal with the richer type system.
- Complex data types
  - new collection types including multisets and arrays
  - attributes can no longer just contain atomic values (1NF) but also collections
  - nest and unnest operations for collection type attributes
  - ER concepts such as composite attributes or multivalued attributes can be directly represented in the object-relational data model
- Type inheritance for inheriting attributes of user-defined types

Object-Oriented Databases

- ODL (Object Description Language), like CREATE TABLE part of SQL, defines persistent classes, whose objects are stored permanently in the database.
- OQL (Object Query Language), simulate SQL in an OO paradigm.

Object-oriented databases ...

- Everything is an object. Objects are defined by static properties— attributes, and also dynamic properties – methods. Values for attributes can have structures, rather than being an atomic type. Values returned from methods are computed.
- Objects of the same kind form a class.
- A subclass inherits attributes and methods from its superclasses, and only explicitly declares attributes and methods specific to the subclass.

Object-orientation: Example

```
Class Staff {
        attribute string empNo;
        attribute Struct sname
                {string firstname, string lastname} name;
         attribute Struct Addr
                {string street, string city} address;
        attribute date DOB;
        method integer Age() {calculate age from DOB}
};
Class Academic: Staff {
        attribute string specialty
};
```

MongoDB: a document database

MongoDB is an example of a free and open source document-oriented database

- Documents = JSON objects
- Store data as BSON (Binary JSON)
- Easy to access related information
- Flexible indexing capability
- Easy to adapt to common coding practices

MongoDB: a document database

RDBMS	MongoDB
Database	Database
Table	Collection
Row	Document
Index	Index
JOIN	Embedded Document or Reference

How is data stored?

RDBMS	MongoDB
In tables with rows and columns. Each record is a row. **Impedance mismatch**: In application, data stored as objects => mapping object representation of data to tabular representation might slow down development	Each record is a file written in JSON format Each document has a rich structure: i.e. can contain sub-documents or arrays of sub-documents **Advantage**: JSON documents align with the structure of objects in modern programming languages Data is stored in a much more natural and intuitive way
RDBMS optimizes data for storage efficiency (in the past when storage was more expensive)	MongoDB's document model is optimized for how the application accesses data (developer time and speed to market are now more expensive than storage)

References among data objects

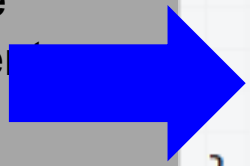
RDBMS	MongoDB
Enforce foreign key constraints at data level	Does not enforce foreign key constraints at data level => can be enforced at application level.

MongoDB has two options to store related data:

- Embedded Document
- Reference: i.e. create a reference between separate documents (see next slides)

Embedded Document Model: One-to-One Relationship

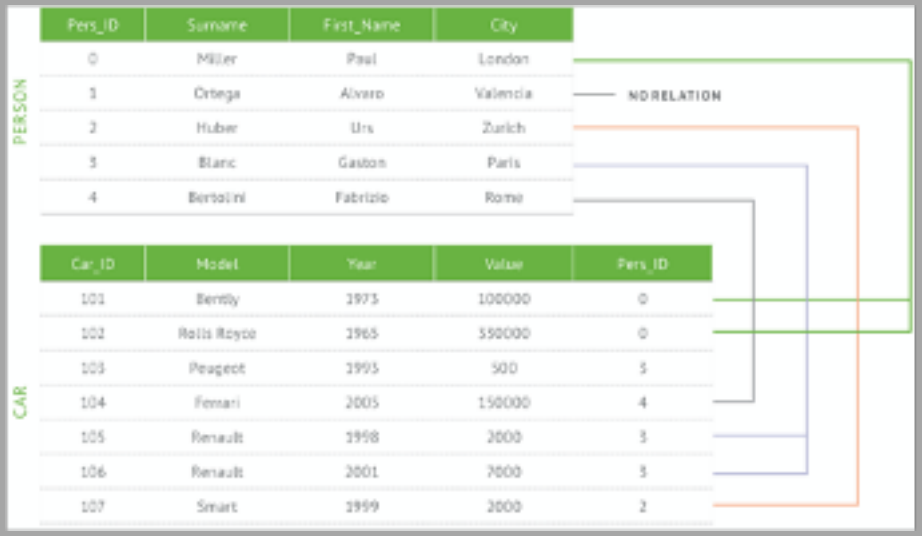
Embedded Document Model: One sub-document inside another document



```
{
    _id: "joe",
    name: "Joe Bookreader",
    address: {
                street: "123 Fake Street",
                city: "Faketon",
                state: "MA",
                zip: "12345"
             }
}
```

Embedded Document Model: One to Many Relationship

RDBMS	MongoDB



Embedded Document Model: an array of sub-documents inside another document

```
first_name: "Paul",
surname: "Miller",
city: "London",
location: [45.123,47.232],
cars: [
   { model: "Bentley",
        year: 1973,
        value: 100000, ….},
   { model: "Rolls Royce",
        year: 1965,
        value: 330000, ….},
]
```

Document Reference Model: One to Many Relationship

```
{
    _id: 123456789,
    title: "MongoDB: The Definitive Guide",
    author: [ "Kristina Chodorow", "Mike Dirolf" ],
    published_date: ISODate("2010-09-24"),
    pages: 216,
    language: "English",
    publisher_id: "oreilly"
}
```

```
{
    _id: 234567890,
    title: "50 Tips and Tricks for MongoDB Developer",
    author: "Kristina Chodorow",
    published_date: ISODate("2011-05-06"),
    pages: 68,
    language: "English",
    publisher_id: "oreilly"
}
```

```
{
    _id: "oreilly",
    name: "O'Reilly Media",
    founded: 1980,
    location: "CA"
}
```

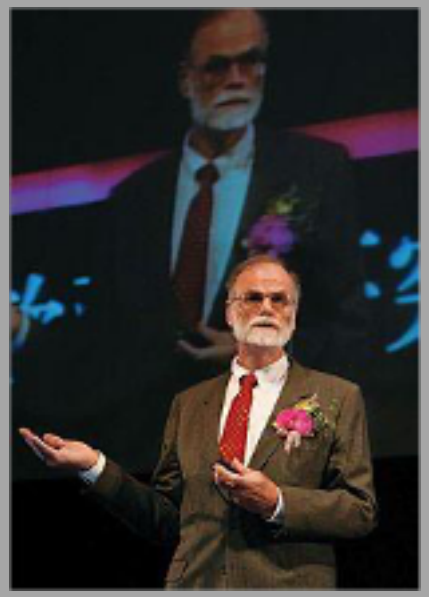
The MongoDB query language

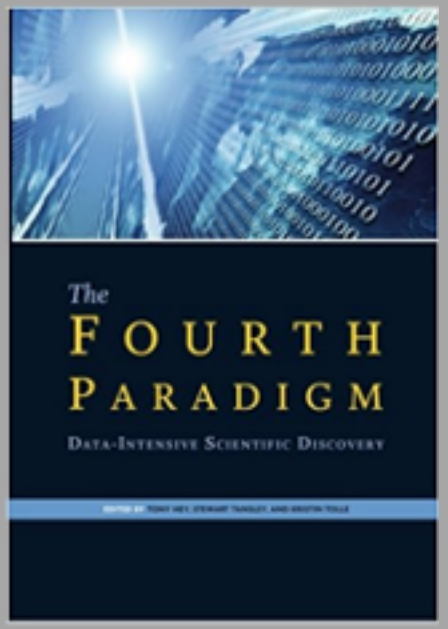
MongoDB has a Javascript flavour query language.

```
db.movies.insert({
    "title":"Toy Story",
    "Length":120,
    "cast":["Tom Hanks", "Tim Allen", "Don Rickles"],
    "Classification":["animation","comedy","adventure"],
    "direct":"John Lasseter"
})

    db.movies.find({title:"Toy Story"})

    db.movies.find({title:/Toy/})
```

Data Science

- Data science, also known as data-driven science, is an interdisciplinary field about scientific methods, processes, and systems to extract knowledge or insights from data in various forms, either structured or unstructured.
- Turing award winner Jim Gray imagined data science as a "fourth paradigm" of science -- empirical, theoretical, computational and now data-driven)



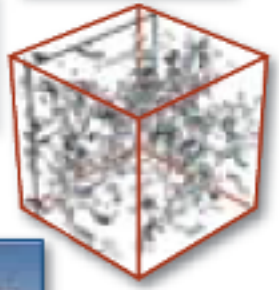


The four paradigms of science

Science Paradigms

- Thousand years ago:
  science was **empirical**
  describing natural phenomena
- Last few hundred years:
  theoretical branch
  using models, generalizations
- Last few decades:
  a **computational** branch
  simulating complex phenomena
- Today: **data exploration** (eScience)
  unify theory, experiment, and simulation
  - Data captured by instruments or generated by simulator
  - Processed by software
  - Information/knowledge stored in computer
  - Scientist analyzes database / files using data management and statistics

$$\left(\frac{\dot{a}}{a}\right)^2 = \frac{4\pi G p}{3} - \mathrm{K}\frac{c^2}{a^2}$$



Source: Hey, Tony, et al. (ed.) The fourth paradigm: data-intensive scientific discovery. Redmond, WA: Microsoft Research, 2009.

Data science, big data and data analytics

- Data science = big data + data analytics
- Big data:
  - Structured and unstructured data
  - Data streams, linked data and voluminous data
  - Cloud computing and big data processing
- Data analytics
  - Statistical analysis and patterns
  - Machine learning, data mining and predictive modelling
  - Visualization

Further readings

- Redmond, Eric, and Jim R. Wilson. Seven databases in seven weeks: a guide to modern databases and the NoSQL movement. Pragmatic Bookshelf, 2012.
- Chang, Fay, et al. "Bigtable: A distributed storage system for structured data." ACM Transactions on Computer Systems (TOCS) 26.2 (2008): 4.
- DeCandia, Giuseppe, et al. "Dynamo: amazon's highly available key-value store." ACM SIGOPS operating systems review 41.6 (2007): 205-220.
- Armstrong, Timothy G., et al. "LinkBench: a database benchmark based on the Facebook social graph." Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data. ACM, 2013.
- Hey, Tony, et al. (ed.) The fourth paradigm: data-intensive scientific discovery. Redmond, WA: Microsoft Research, 2009.