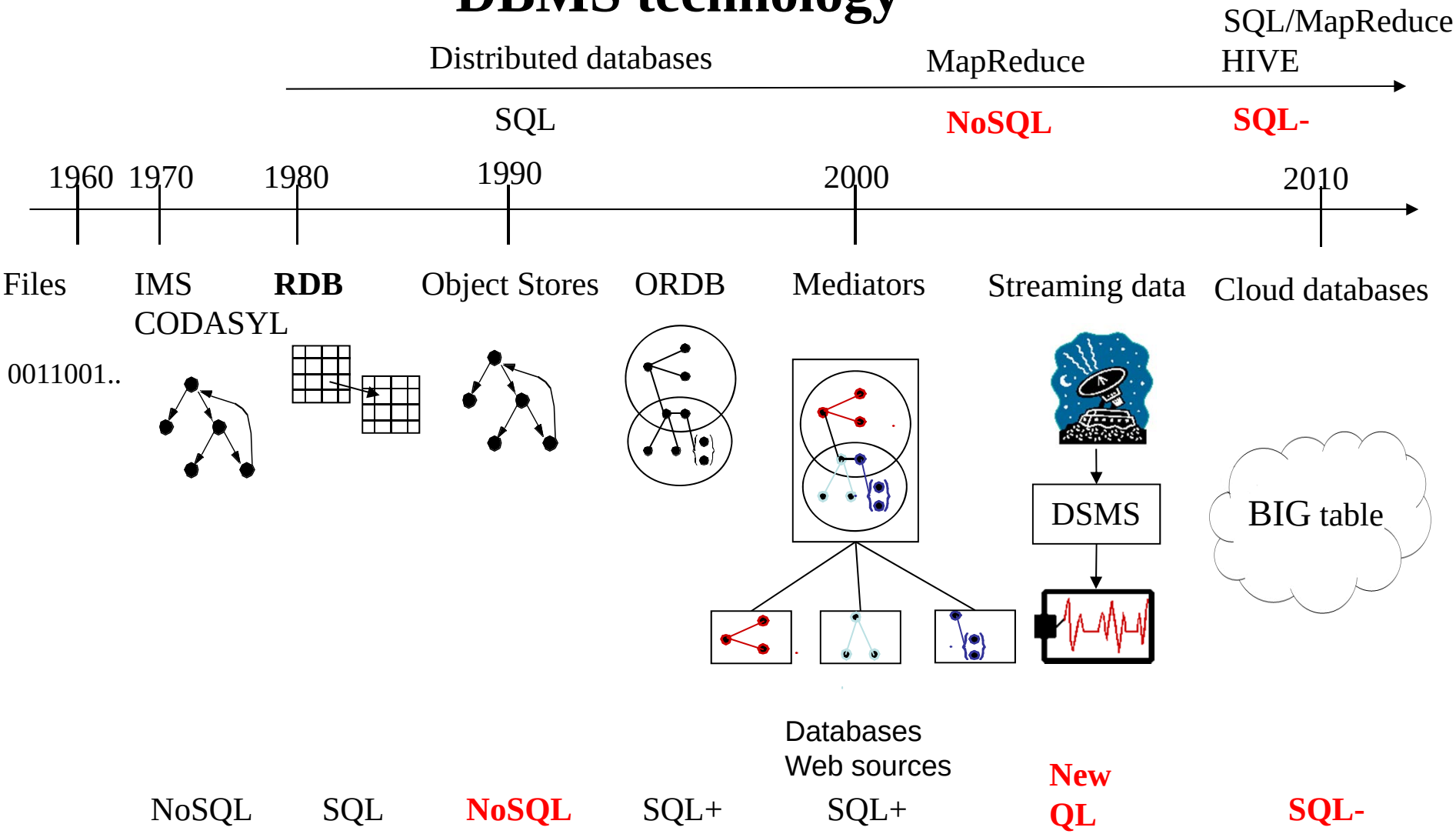
Introduction to NoSQL Databases

Tore Risch
Information Technology
Uppsala University
2013-03-05

Tore Risch
Uppsala University, Sweden

UDBL

Evolution of DBMS technology



UDBL

Tore Risch
Uppsala University, Sweden

Kinds of DBMS support

	Simple Data	Complex data
Query (SQL)	Relational DBMSs	Object-Relational DBMSs
No Query (NoSQL)	File systems (scalable) Storage managers	Object Stores

UDBL

Tore Risch
Uppsala University, Sweden

Classification of Modern Database applications

Complex Queries SQL, **New QL**	Business operations Business analytics Personal db	Multimedia search Custom Datatypes **Data streams** **Web analytics**
Simple Queries **SQL-**	**E-business**	**Web search**
No Queries **NoSQL**	Text, data logs Simple computations **Web documents**	CAD system Complex computations **Web surfing**
	Simple Data	Complex data

UDBL

Tore Risch
Uppsala University, Sweden

Kind of Database support

	Simple Data	Complex data
Complex Queries SQL, **New QL**	Relational DBMS	Object-Relational DBMS **Data Stream Mgmt. Syst.** **Virtuoso, Neo4J, Amos II**
Simple Queries **SQL-**	**Google App Engine (GQL)** **Amazon SimpleDB** **Microsoft Azure**	**Google search engine** **Facebook HIVE**
No Queries **NoSQL**	File systems **Content Mgmt. Syst.**	Object Stores **Google BigTable** **Yahoo Hadoop** **MongoDB,CouchDB**

What is a NoSQL Database?

- A key/value store
  Basic index manager, no complete query language
  - E.g. Google BigTable, Amazon Dynamo
- A web document database
  For web documents, not for small business transactions
  - E.g. MongoDB, CouchDB
- A DBMS with a limited query language
  Provides for high volume small business transactions
  - Sometimes called *cloud databases*
  - E.g. Google App Engine, Microsoft Azure, Amazon SimpleDB, Facebook HIVE

What is a NoSQL Database?

- A DBMS where mapreduce is used instead of queries
  Manual programs to iterate over entire data sets
  - E.g. Hadoop, MongoDB, CouchDB, Dynamo
- A mapreduce engine with a limited query language on top:
  HIVE on top of Hadoop provides HIVEQL
  - Provides non-procedural data analythics (select from groupby) without detailed programming
  - Executed in batch as paralle Hadoop jobs
- A DBMS with a new query language for new applications
  - Streambase, Virtuoso, Neo4J, Amos II
- Other non-relational databases
  - Including Object Stores

NoSQL Characteristics

- Highly distributed and parallel architectures
  - Typically runs on data centers
  - This is similar to parallel databases!
- Highly scalable systems by compromised consistency
  - No 2-phase commit as in distributed databases
  - *Eventual* consistency
    - Or perhaps never consistency
    - Similar options available in modern DBMSs too
  - Puts burden on programmer to handle consistency!
    - Race conditions
    - Recovery

    $\Rightarrow$ Customized implementation of transacations
  - Mainly suitable for applications not needing consistency

NoSQL Characteristics

- New query languages for new applications
  - SQL- for cloud databases
    - Most simple web applications do not need full SQL
    - Simple SQL permits high scalability
    - Familiar model
    - Full SQL too complex for new systems
  - Graph query languages
    - SPARQL for RDF
      - RDF data model
      - For searching linked data (http://linkeddata.org/)
  - Stream query languages
    - CQL
      - Variant of SQL for streams
    - SCSQL (UU)
      - Functional parallel data stream query language

MapReduce

- Parallel *batch* processing using mapreduce
  - Many NoSQL databases uses mapreduce for parallel batch processing of data stored in data centers
  - Highly scalable implementation of
    - parallell batch processing
    - of same (e.g. Java) program
    - over large amounts of data stored in different files
    - Based on a scalable file system (e.g. HDFS)
- The mapreduce function:
  - Applies a (costly) user function *mapper* producing key/value pairs in parallel on many nodes accessing files in a cluster
  - Applies a user *aggregate function* on the key/value pairs produced by the *mapper*
  - Very similar to GROUP BY in SQL
  - Read reference article on MapReduce

Mapreduce code

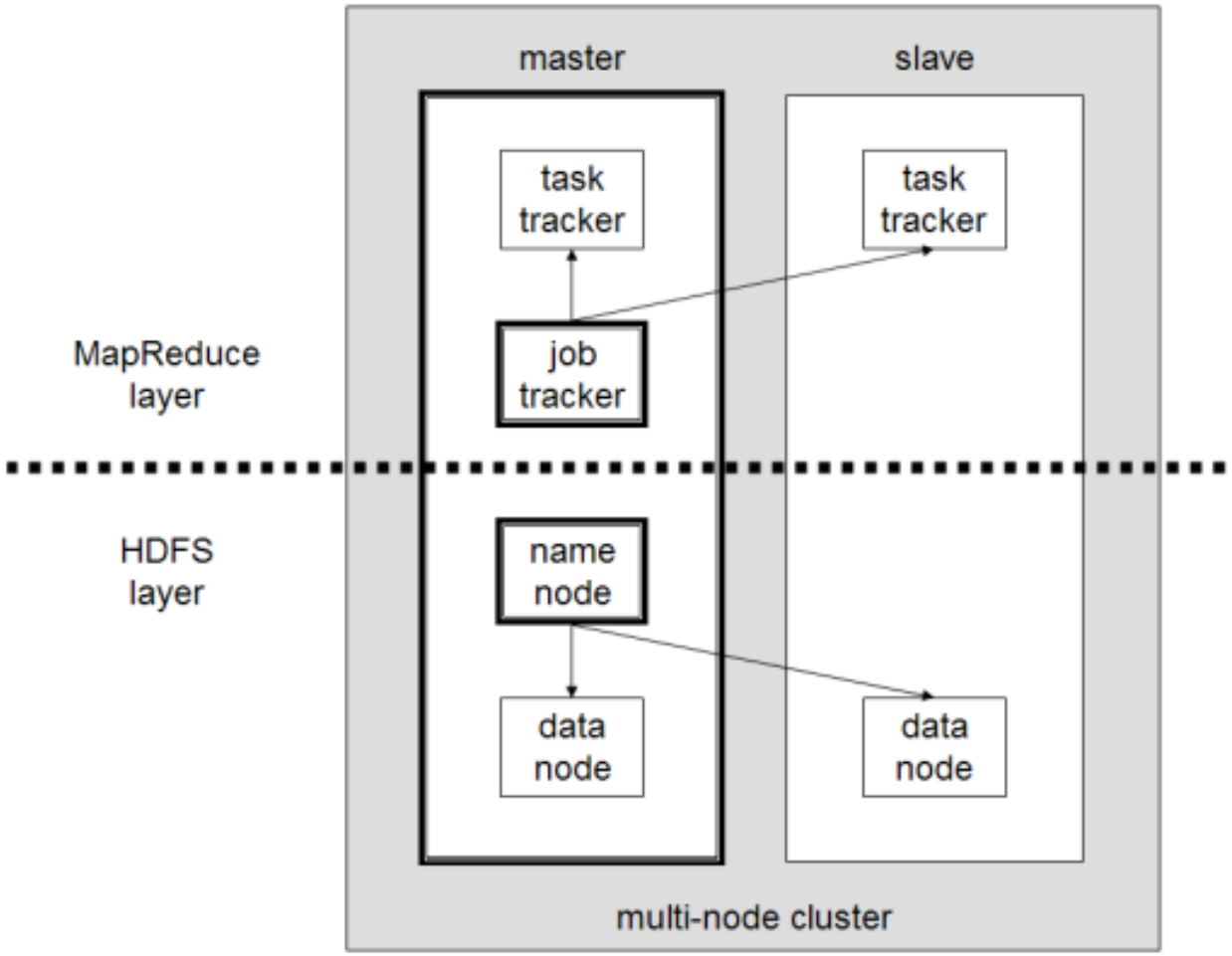
```
function map(String name, String document):
// name: document name, i.e. HDFS file contents
// document: document contents, parsed HDFS file tokens
// Can make own parser as preprocessor

for each word w in document:
        emit (w, 1)

function reduce(String word, Iterator partialCounts):
// word: a word
// partialCounts: a list of aggregated partial counts

sum = 0;
 for each pc in partialCounts:
    sum += ParseInt(pc);
    emit (word, sum)
```

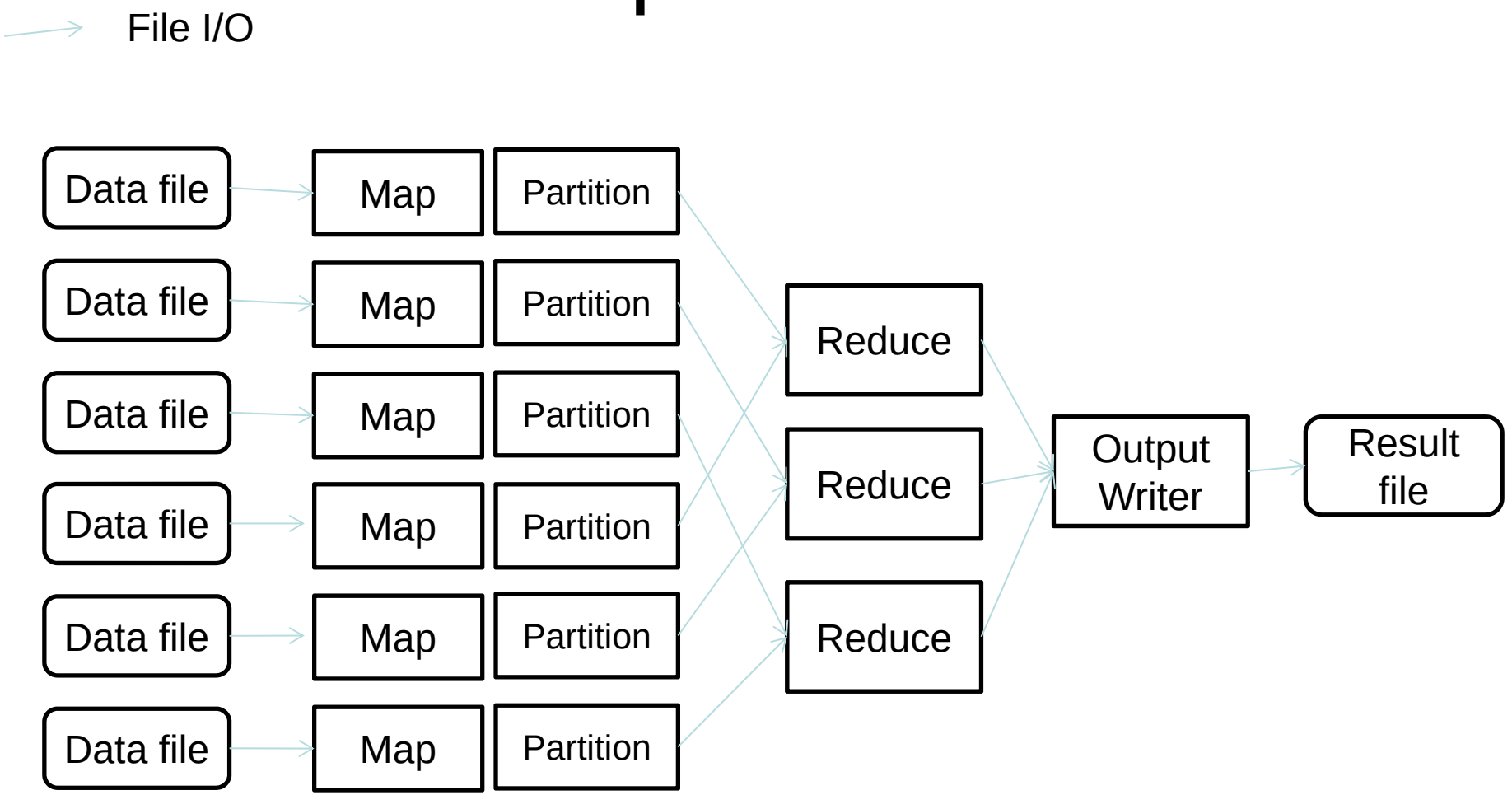
Mapreduce manager architecture



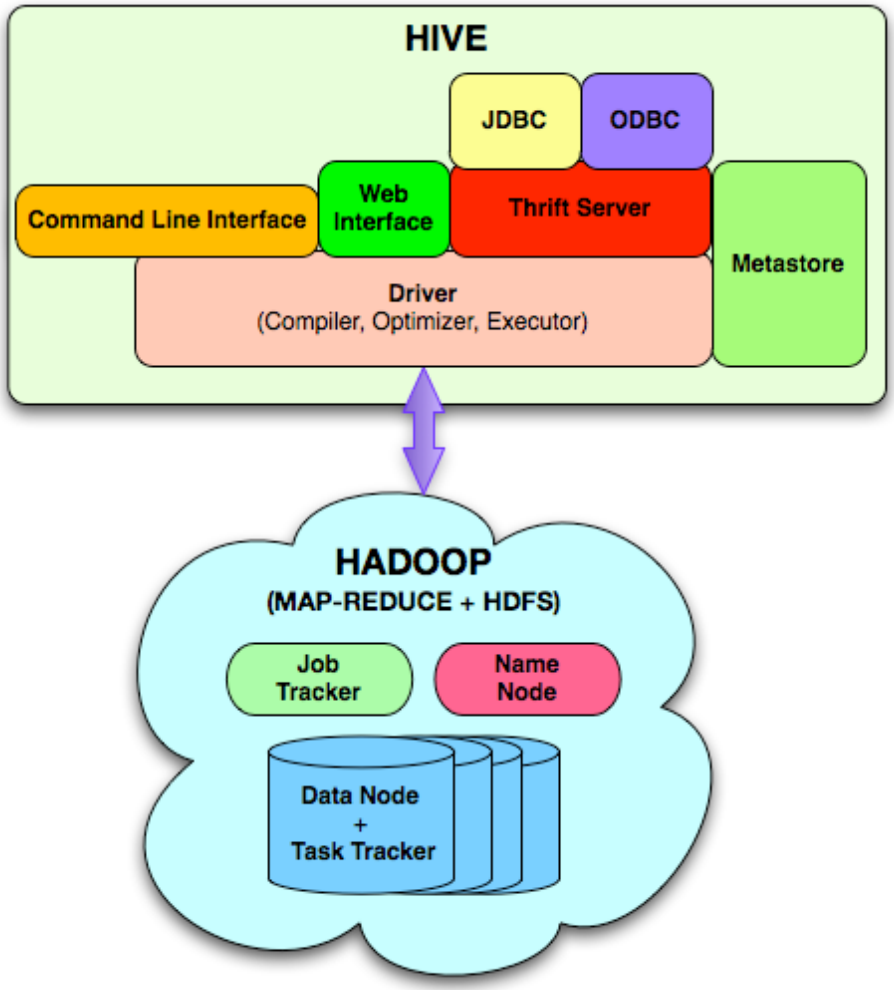
Mapreduce stages

- Input reader
  - System component that reads files from scalable file system (e.g. HDFS) and sends to map functions applied in parallell
- Map function
  - Applied in parallel on many different files
  - Parsers input file data from HDFS
  - Does some (expensive) computation
  - Emits key value pairs as result
  - Result stored by MapReduce system as file
- Partion function (optional)
  - Partitions output key/value pairs fro map function into groups of key/value pairs to be reduced in parallel
  - Usually hash partitioning
- Reduce function
  - Iterates over set of key/value pairs to produced a reduced set of key value pairs stored in the file system
  - C.f. aggregate functions

Mapreduce


File I/O
Data file
Data file
Data file
Data file
Data file
Data file
Map
Map
Map
Map
Map
Map
Partition
Partition
Partition
Partition
Partition
Partition
Reduce
Reduce
Reduce
Output Writer
Result file

Hive architecture



Wordcount in Hive

```
FROM (
MAP doctext USING 'python wc_mapper.py'
   AS (word, cnt)
FROM docs
CLUSTER BY word
)
REDUCE word, cnt USING
   'pythonwc_reduce.py';
```

Architecture

- Metastore: stores system catalog
- Driver: manages life cycle of HiveQL query, session handles and session statistics
- Query compiler: Compiles HiveQL into a distributed execution plan of map/reduce tasks
- Execution engines: The component executes the execution plan in proper dependency order; interacts with Hadoop
- HiveServer: provides JDBC/ODBC drivers and other interfaces for integrating other applications.
- Client API components: CLI, web interface, jdbc/odbc API
- Extensibility interface include SerDe (parse/print), User Defined Functions and User Defined Aggregate Functions.