Insight into

DBMS:

Design and Implementation

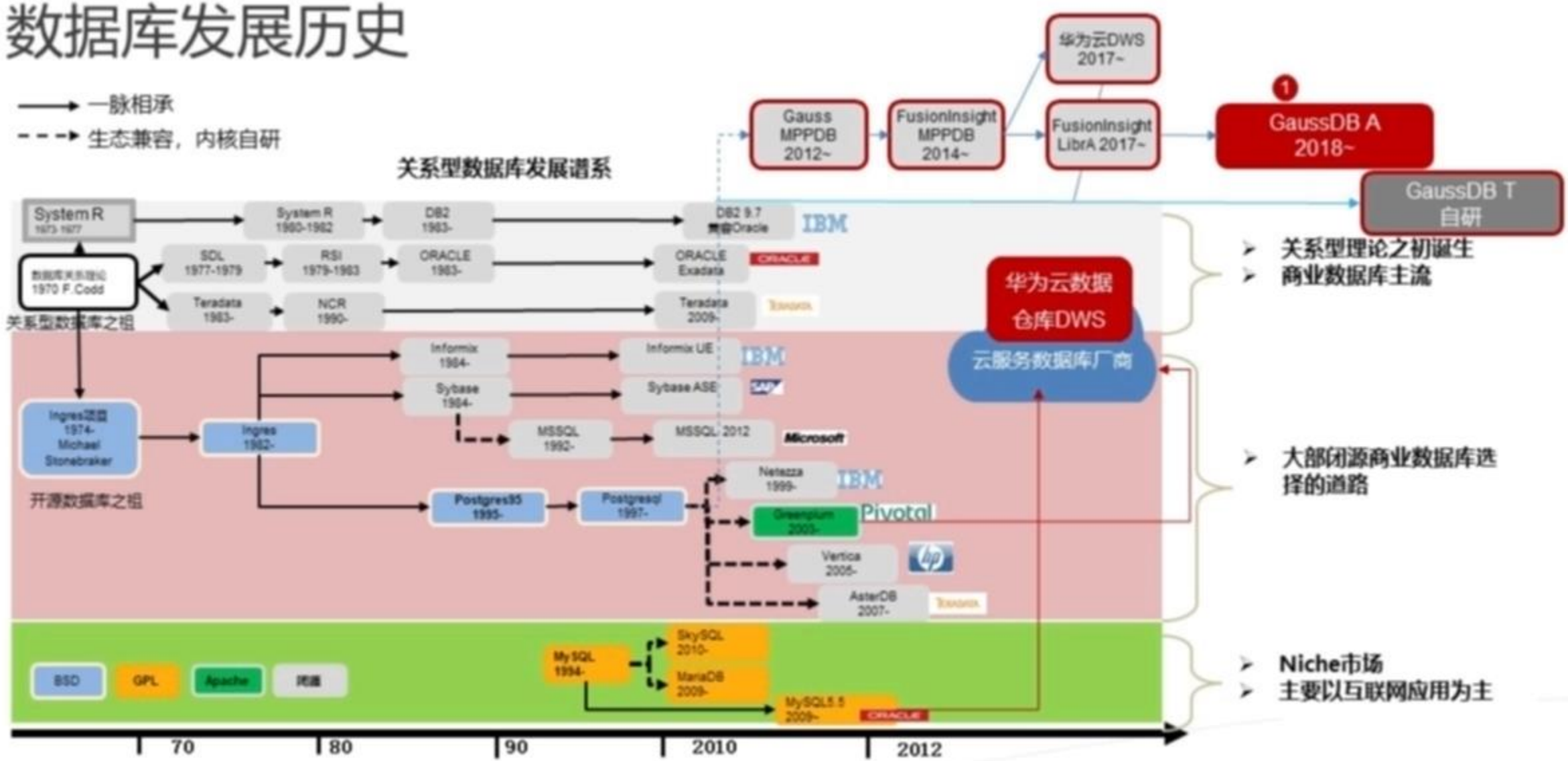
Chapter 6 A: Go deeper and wider
File, DBMS, ERP/MRP, DW/DM, Big Data

孔令波
mlinking@126.com
+86 15010255486

Outline of the chapters covered

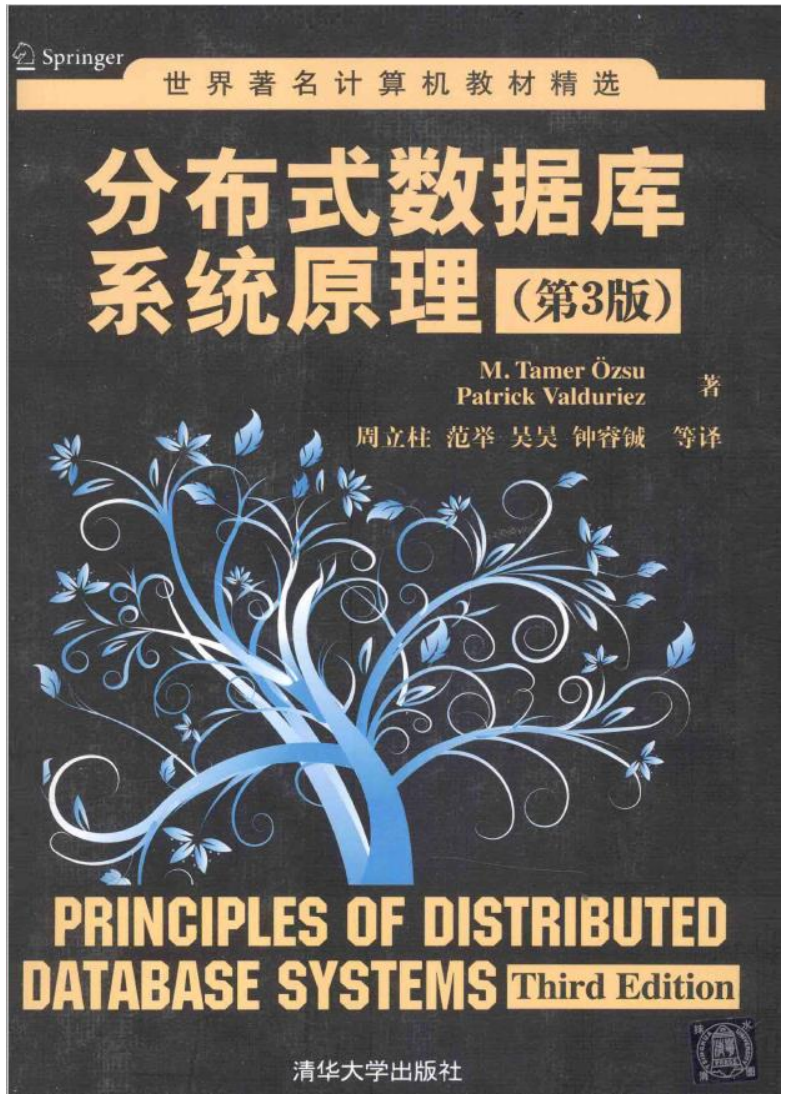
- **Introduction**
- **Overview of the projects**
- **Demonstration of Development environment**
  - Watch and practice by yourself
- **My understanding about (R)DBMS**
  - History and D&I
- **SQL translation with 2 conversions**
  - SQL → RA (Relational Algebra)
  - RA → Sequence of File operations
- **Transaction control**
- **Deeper**
  - File, (R)DBMS, ERP, DW, Big Data (No SQL, SQL again)
  - SQL on MPP and Hadoop (Greenplum, **HAWQ**)

数据库发展历史



1 注：改名原因为华为自研数据库成为独立品牌GaussDB，其中T表示OLTP，A表示OLAP数据库

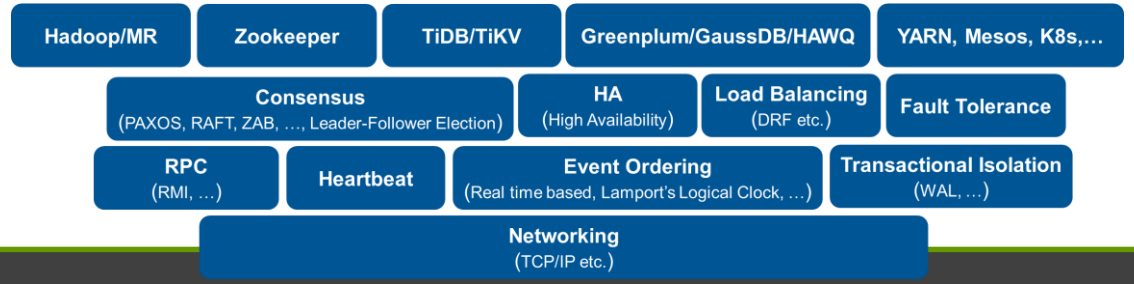
2021年7月7日21:49:57 以后再翻翻吧



- 分布式数据库系统原理
- *厄兹叙 (M.Tamer Ozsu) / Patrick Valduriez 著，周立柱 / 范举 译*
- **2014**

觉得还是应该从 那个层次结构的方式来介绍 Distributed DBMS
2021年7月7日21:50:50

在草稿之上还有 Reading Torlerance, SAUCR(?)的信息，课后查一下吧
2021年5月26日12:57:28

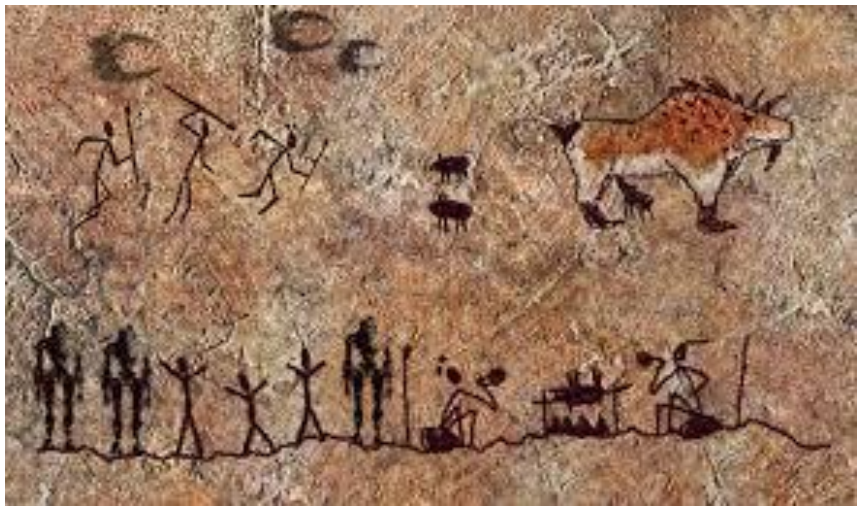


Chapter 5: Go deeper and wider

- **Overview about the evolution of Data Management & Analytics**
  - Evolution of DM
    - Before IT
    - In IT : File system, RDBMS, ERP, DW, Big Data
  - Hints –
    - Integrate DM with DA (Data Analytics)
      - ✓ Data – diverse (from structured to unstructured) and huge (from single to distributed)
- **SQL again**
- **Final (if you want to share)**

Kernel is Data management – storage + processing

- **By that, a long history**
  - Early human records-clay tablets, hieroglyphics, cave paintings, paper records of family histories, treaties, inventories, and so on



cave painting

Hieroglyphics [ˌhaiərəˈglifiks]
hieroglyphist [ˌhaiərəˈglifist]
n. 象形文字研究者,书写象形文字者

Piles of paper files

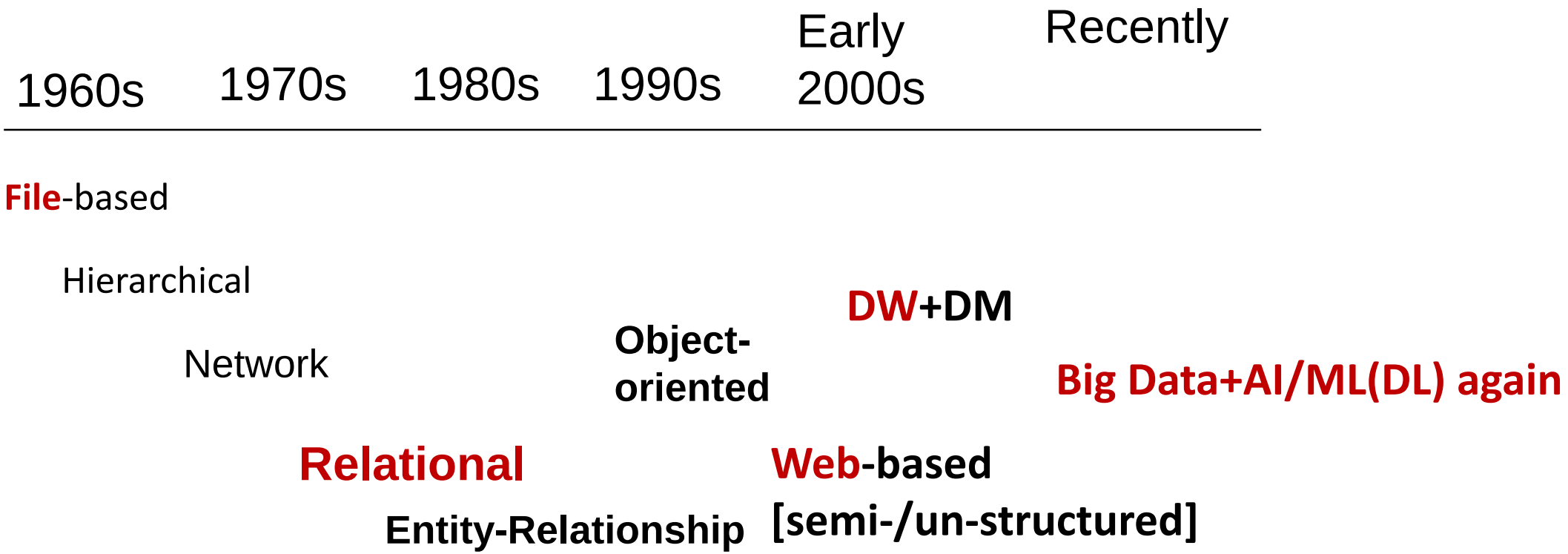
All previous data processing is **MANKIND** ☺ - Read, Write, Create, Compose, Print, …

Chapter 8: Go deeper and wider

- **Overview about the evolution of Data Management & Analytics**
  - Evolution of DM
    - Before IT
    - In IT : File system, RDBMS, ERP, DW, Big Data
  - Hints –
    - Integrate DM with DA (Data Analytics)
    - Integrate DM with DA (Data Analytics)
      - ✓ Data – diverse (from structured to unstructured) and huge (from single to distributed)
- **SQL again**
- **Final (if you want to share)**

Evolution of Data Models

- **Timeline**



Old way to manage data – Simple File System

- Each file was used by its own application programs
- Each file was owned by individual or department who commissioned its creation

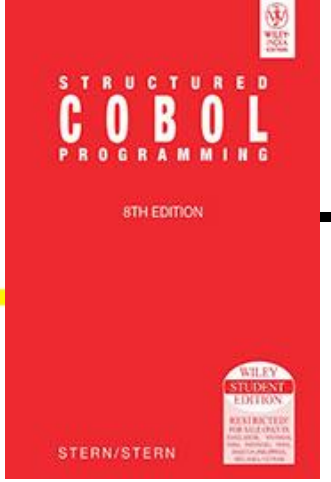
Historical Perspective (1960-)

- Companies began automating their back-office bookkeeping in the 1960s.
- COBOL and its record-oriented file model were the work-horses of this effort.
- Typical work-cycle:

1. a batch of transactions was applied to the old-tape-master
2. a new-tape-master produced
3. printout for the next business day.

- **CO**mmon **B**usiness-**O**riented **L**anguage (COBOL 2002 standard)

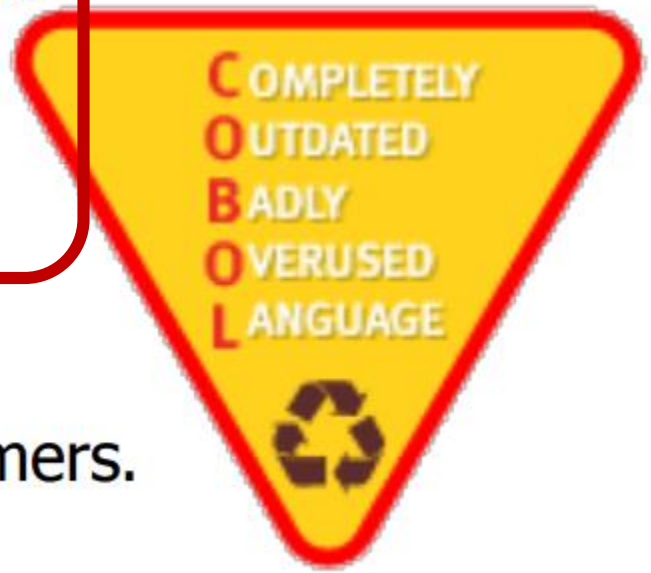
COBOL



A quote by Prof. dr. E.W. Dijkstra,
18 June 1975:

cripple ['kripl]
vt.使跛, 受伤致残

"The use of COBOL cripples the mind; its teaching should, therefore, be regarded as a criminal offence."



But: In 2012 still vacancies available for COBOL programmers.

COBOL Code (just an example!)

```
01  LOAN-WORK-AREA.
        03  LW-LOAN-ERROR-FLAG          PIC  9(01)       COMP.
        03  LW-LOAN-AMT                 PIC  9(06)V9(02) COMP.
        03  LW-INT-RATE                        PIC  9(02)V9(02) COMP.
        03  LW-NBR-PMTS                 PIC  9(03)       COMP.
        03  LW-PMT-AMT                  PIC  9(06)V9(02) COMP.
        03  LW-INT-PMT                  PIC  9(01)V9(12) COMP.
        03  LW-TOTAL-PMTS               PIC  9(06)V9(02) COMP.
        03  LW-TOTAL-INT                PIC  9(06)V9(02) COMP.
*
    004000-COMPUTE-PAYMENT.
*
        MOVE 0 TO LW-LOAN-ERROR-FLAG.

        IF (LW-LOAN-AMT ZERO)
           OR
          (LW-INT-RATE ZERO)
           OR
          (LW-NBR-PMTS ZERO)
           MOVE 1 TO LW-LOAN-ERROR-FLAG
           GO TO 004000-EXIT.

        COMPUTE LW-INT-PMT = LW-INT-RATE / 1200
           ON SIZE ERROR
              MOVE 1 TO LW-LOAN-ERROR-FLAG
              GO TO 004000-EXIT.
```


Before RA – network, hierarchical model, …

Historical Perspective (1970's)

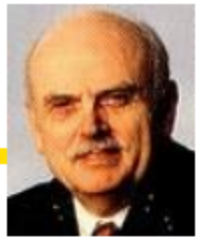
- Transition from handling transactions in daily batches to systems that managed an on-line database that could capture transactions as they happened.
- At first these systems were ad hoc
- Late in the 60's, "network" and "hierarchical" database products emerged.

Data Base Task Group

- A network data model standard (DBTG) was defined, which formed the basis for most commercial systems during the 1970's.
- In 1980 DBTG-based Cullinet was the leading software company.

http://liacs.leidenuniv.nl/~bakkerem2/dbdm/Leiden-DBDM-02-Databases-1x1.pdf

The "relational" data model



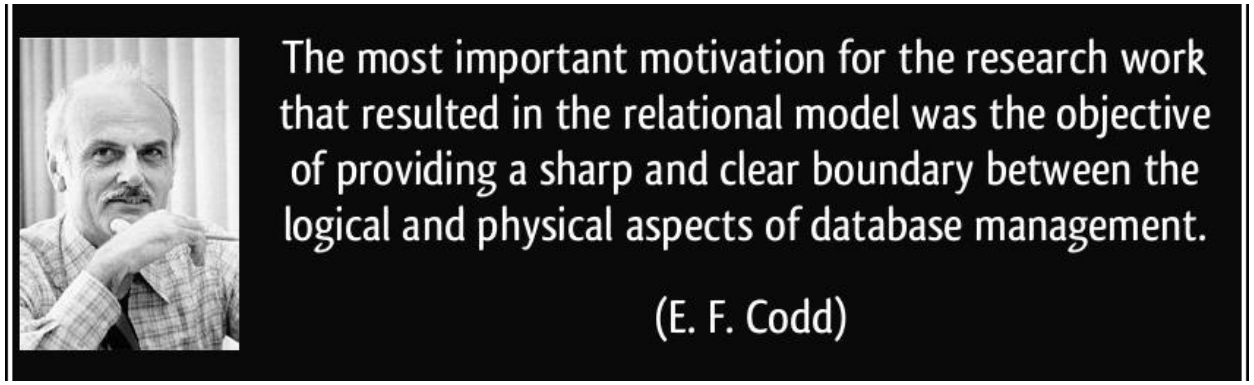
The "relational" data model, by Ted Codd in his landmark 1970 article *"A Relational Model of Data for Large Shared Data Banks",* was a major advance over DBTG.

- The relational model unified data and metadata => only one form of data representation.
- A non-procedural data access language based on algebra or logic.
- The data model is easier to visualize and understand than the pointers-and-records-based DBTG model.
- Programs written in terms of the "abstract model" of the data, rather than the actual database design => programs insensitive to changes in the database design.

For more friendly management

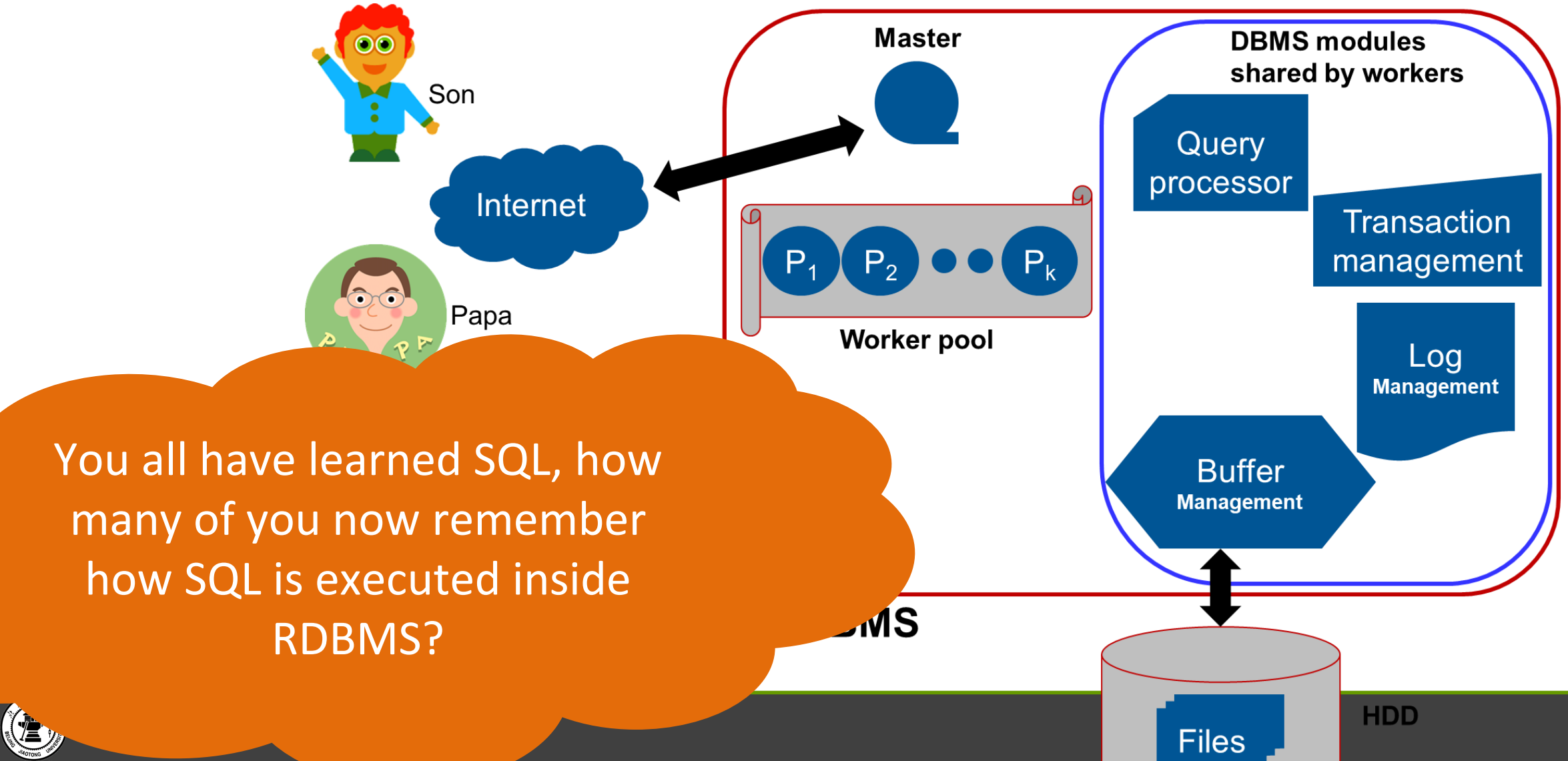
Born: August 23, 1923
Died: April 18, 2003 (aged 79)

1970 - E.F. Codd and the Relational Model



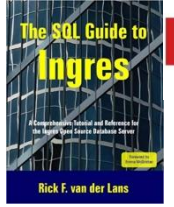
- I prefer to use to the top goal and 2 roles to indicate the benefit of DBMS
  - Top goal:
    - ✓ Support the **data access** of **many users** **concurrently**
  - 2 roles:
    1. Concurrent data management by many users
    2. Provide friendly/flexible interaction for common users

Sketch of internal modules of modern RDBMS



You all have learned SQL, how many of you now remember how SQL is executed inside RDBMS?

Old RDBMSs – Ingres, System R …



ORACLE DATABASE

IBM DB2

Informix

The "relational" data model success

- Both industry and university research communities embraced the relational data model and extended it during the 1970s.
- It was shown that a high-level relational database query language could give performance comparable to the best record-oriented database systems. (!)
- This research produced a generation of systems and people that formed the basis for IBM's DB2, Ingres, Sybase, Oracle, Informix and others.

INGRES

Ingres at UC Berkeley in 1972 (1/2)

Inspired by Codd's work on the relational database model, (Stonebraker, Rowe, Wong, and others) a project that resulted in:

- the design and build of a relational database system
- the query language (QUEL)
- relational optimization techniques
- a language binding technique
- storage strategies
- pioneering work on distributed databases

Ingres at UC Berkeley in 1972 (2/2)

The academic system evolved into Ingres from Computer Associates.

Nowadays: PostgreSQL; also the basis for a new object-relational system.

PostgreSQL
the world's most advanced open source database

Further work on:

- distributed databases
- database inference
- active databases (automatic responsing)
- extensible databases.

IBM: System R (1/2)

- Codd's ideas were inspired by the problems with the DBTG network data model and with IBM's product based on this model (IMS).

controversial [ˌkɔntrəˈvəːʃəl]
adj. 有争议的, 引起争议的

- Codd's relational model was very controversial:
  - too simplistic
  - could never give good performance.

- IBM Research chartered a 10-person effort to prototype a relational system => a prototype, System R (evolved into the DB2 product)

http://liacs.leidenuniv.nl/~bakkerem2/dbdm/Leiden-DBDM-02-Databases-1x1.pdf

IBM: System R (2/2)

- Defined the fundamentals on:
  - query optimization,
  - data independence (views),
  - transactions (logging and locking), and
  - security (the grant-revoke model).

- SQL from System R became more or less the standard.

- The System R group further research:
  - distributed databases (project R*) and
  - object-oriented extensible databases (project Starburst).

http://liacs.leidenuniv.nl/~bakkerem2/dbdm/Leiden-DBDM-02-Databases-1x1.pdf

USA funded database research period 1970 - 1996:

- Projects at UCLA => Teradata

TERADATA
THE BEST DECISION POSSIBLE™

- Projects at CCA (SDD-1, Daplex, Multibase, and HiPAC):
  - distributed database technology
  - object-oriented database technology
- Projects at Stanford:
  - deductive database technology
  - data integration technology
  - query optimization technology.
- Projects at CMU:
  - general transaction models
  - => Transarc Corporation.

http://liacs.leidenuniv.nl/~bakkerem2/dbdm/Leiden-DBDM-02-Databases-1x1.pdf

Extending RDBMS – distributed, OO, 3-tier applications, GIS, Temporal DBMS, …

The database research agenda of the **1980's**
Extending Relational Databases

- geographically distributed databases
- parallel data access.
- Theoretical work on distributed databases led to prototypes which in turn led to products.
  - *Note: Today, all the major database systems offer the ability to distribute and replicate data among nodes of a computer network.*
- Execution of each of the relational data operators in parallel => hundred-fold and thousand-fold speedups.
  - *Note: The results of this research appear nowadays in the products of several major database companies.* Especially beneficial for data warehousing and decision support systems; effective application in the area of OLTP is challenging.

http://liacs.leidenuniv.nl/~bakkerem2/dbdm/Leiden-DBDM-02-Databases-1x1.pdf

Application architecture

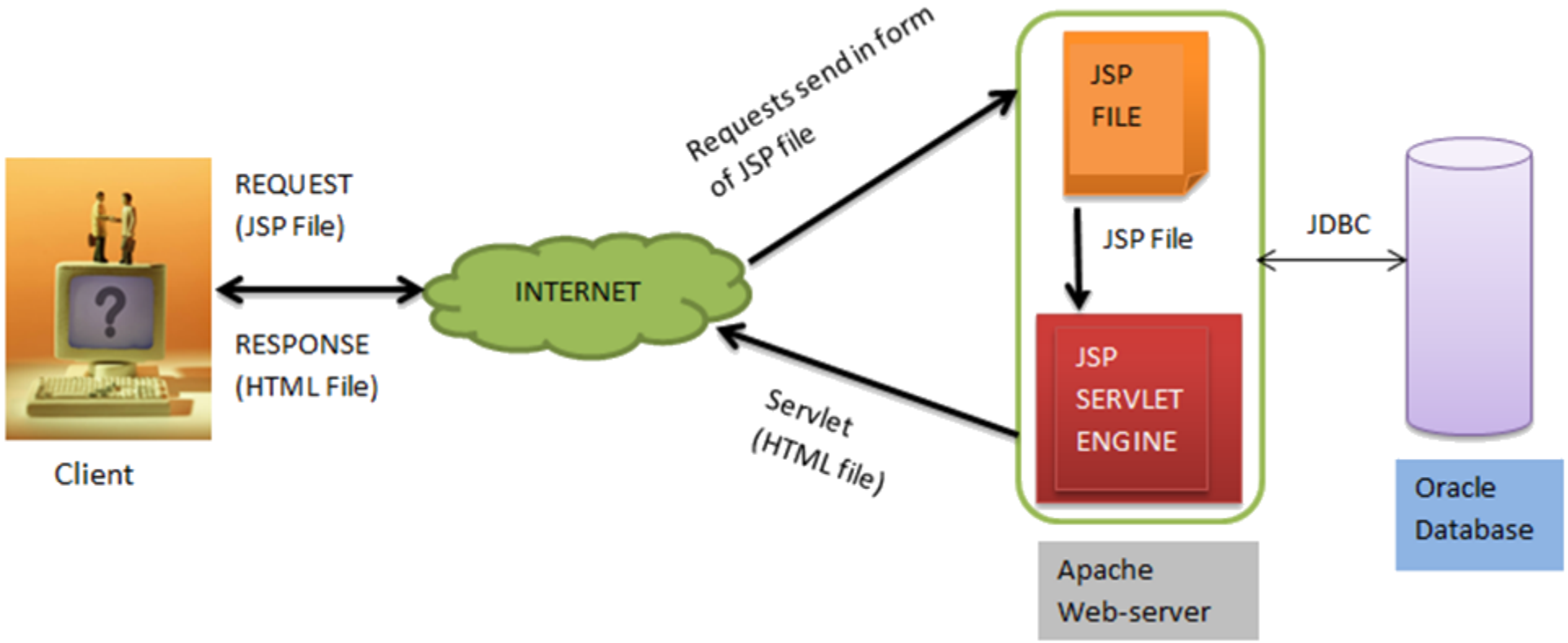
- C/S access to DBs
  - Client/Server architecture is a general model for systems where a service is provided by one system (the server) to another (the client)
- Web-based access to DBs
  - Browser/Server architecture is a general model, in which Web server serves pages to browsers (clients) and can access database(s) ← You've tried JSP/Tomcat or Flask/Mango…

Many types of data are adopted into DBMS

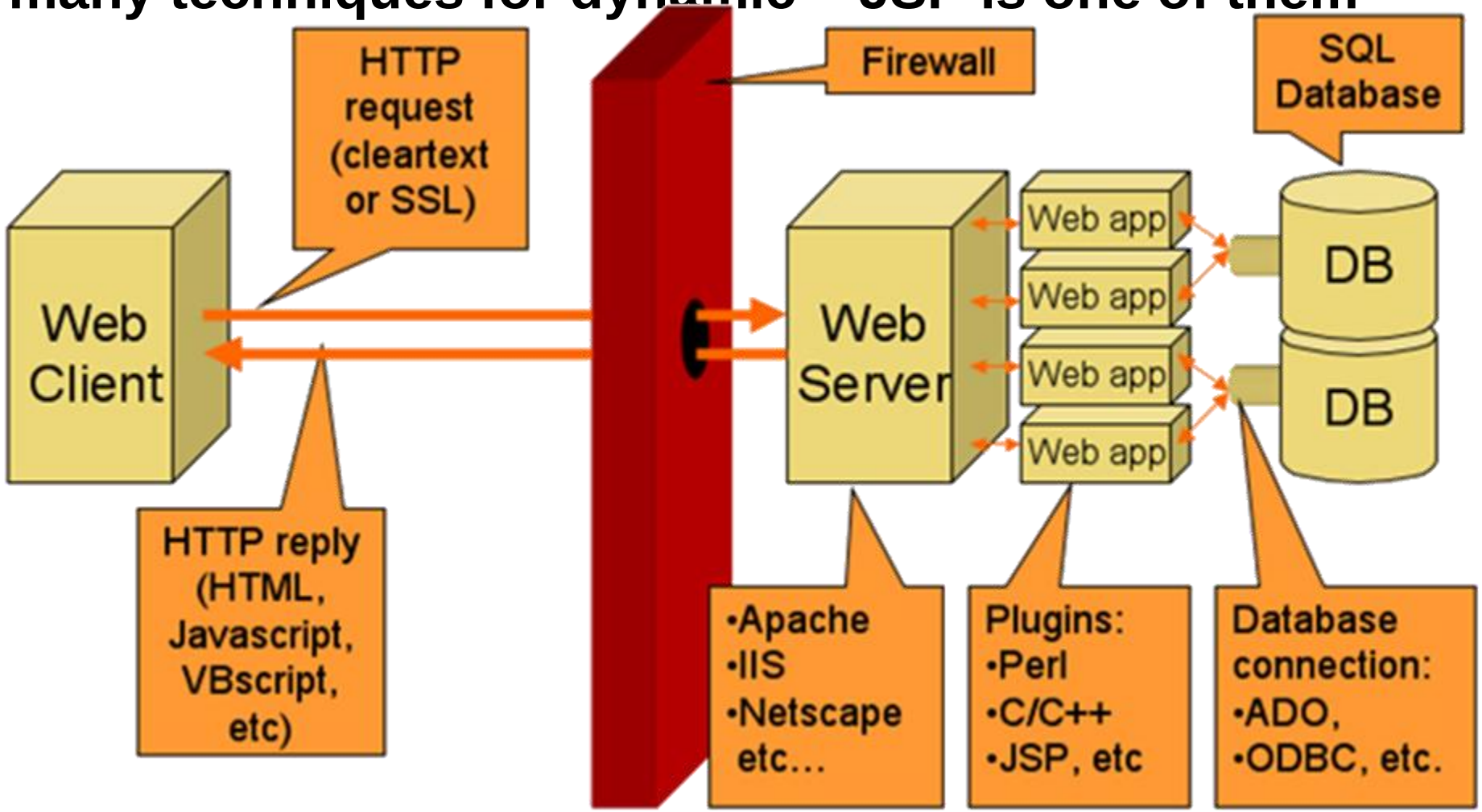
- Object Oriented DBs
- Semi-structured Data and XML: The research topic during my Ph.D at PKU
- Multimedia DBs
- Temporal DBs
- Logic DBs ……

Want to manage different data types as many as possible~~~

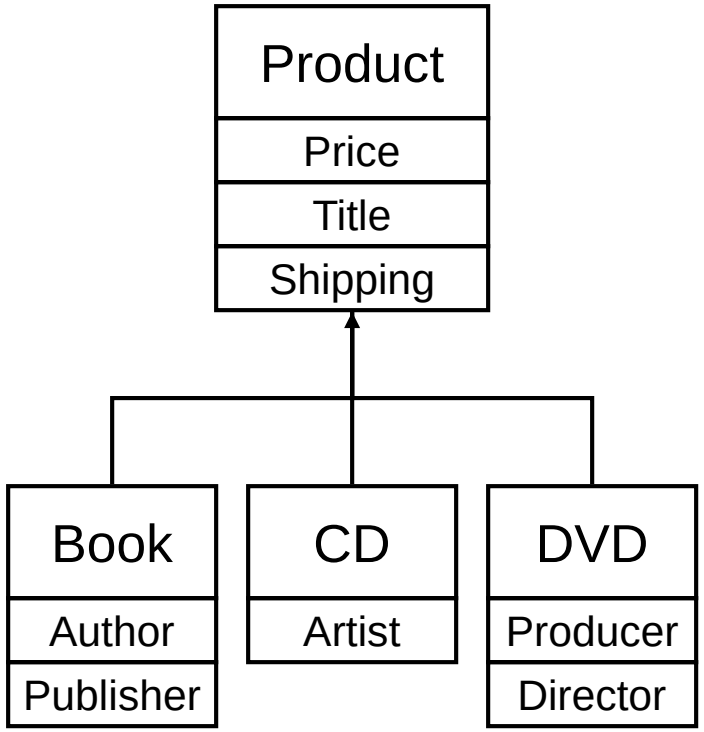
Web-based Databases



- Now many techniques for dynamic – JSP is one of them



OODB Example



- **Product is abstract**
  - You cannot make a Product directly
  - You can, however, make a Book, CD, or DVD, and these are Products

Advantages

- Good integration with Java, C++, etc.
- Can store complex information
- Fast to recover whole objects
- Has the advantages of the (familiar) object paradigm

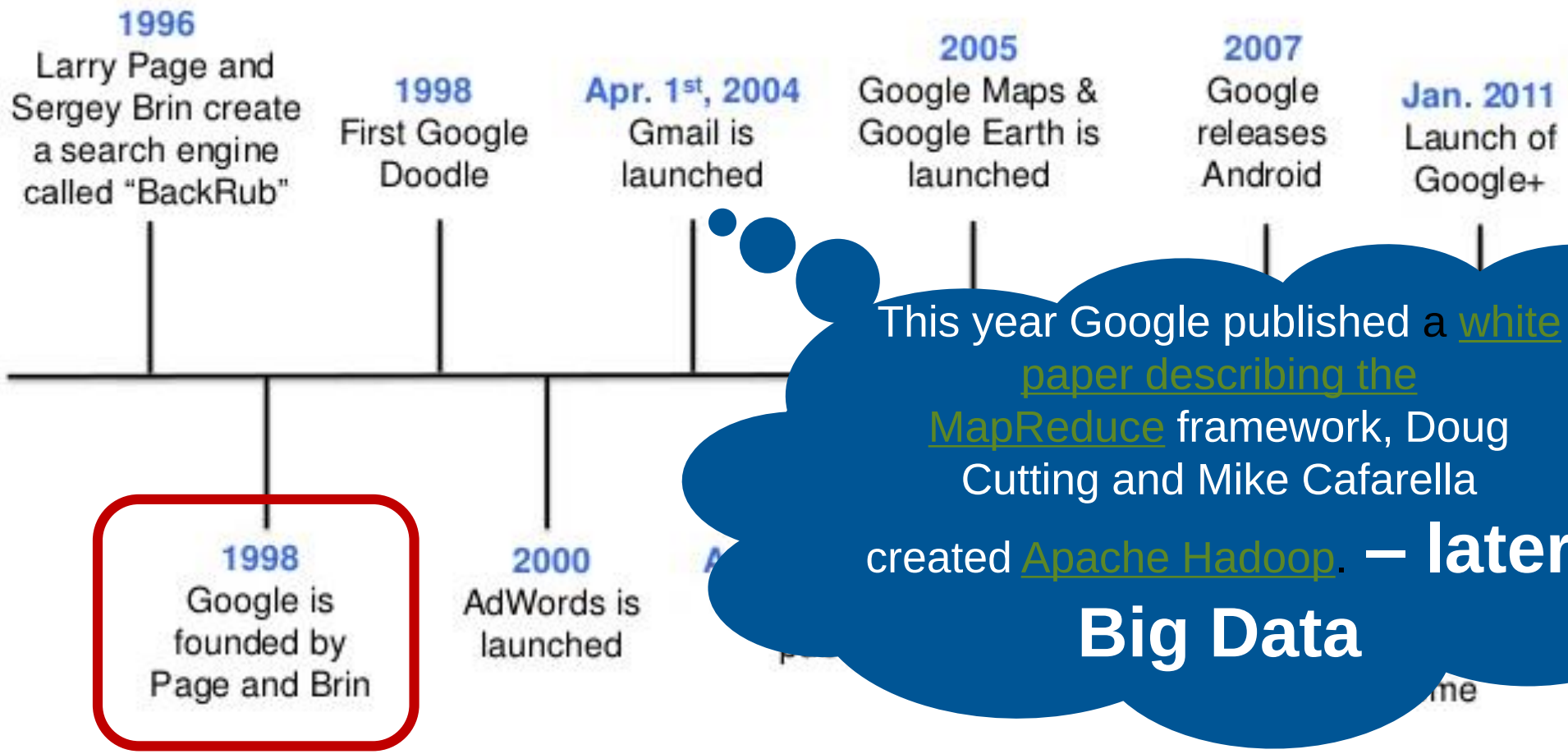
Disadvantages

- There is no underlying theory to match the relational model
- Can be more complex and less efficient
- OODB queries tend to be procedural, unlike SQL

- Multimedia DBs
  - Store Images, Music and audio, Video and animation, Full texts of books, Web pages
- Querying Multimedia DBs
  - Metadata searches
    - ✓ Information about the multimedia data (metadata) is stored
    - ✓ This can be kept in a standard relational database and queried normally
    - ✓ Limited by the amount of metadata available
  - Content searches
    - ✓ The multimedia data is searched directly
    - ✓ Potential for much more flexible search
    - ✓ Depends on the type of data being used
    - ✓ Often difficult to determine what the ‘correct’ results are

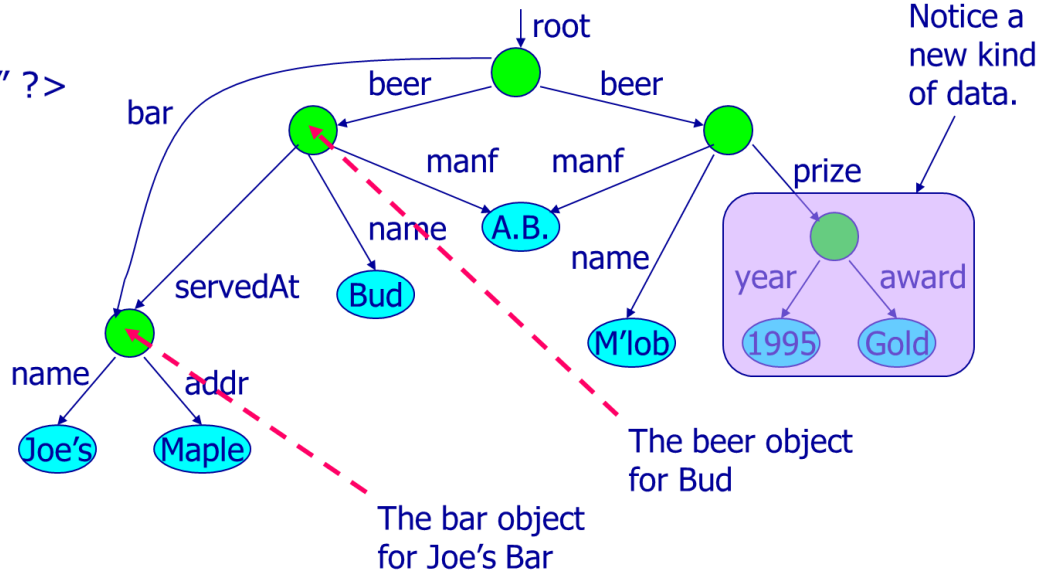
Google triggers other researches …

- **CBIR, XML, …**



- Semi-structured Data and XML
  - XML oriented keyword search – The research topic during my Ph.D at PKU
  - And Graphs of Semi-structured Data

```
<? XML VERSION = "1.0" STANDALONE = "yes" ?>
<BARS>
   <BAR><NAME>Joe's Bar</NAME>
        <BEER><NAME>Bud</NAME>
              <PRICE>2.50</PRICE></BEER>
        <BEER><NAME>Miller</NAME>
              <PRICE>3.00</PRICE></BEER>
   </BAR>
   <BAR> …
</BARS>
```



XPATH and XQUERY

- **XPATH is a language for describing paths in XML documents.**
  - Really think of the semistructured data graph and *its* paths.
  - Why do we need path description language: can't get at the data using just Relation.Attribute expressions.

- **XQUERY is a full query language for XML documents with power similar to OQL (Object Query Language, query language for object-oriented databases).**

Content-Based Retrieval

QBIC™ (Query By Image Content) from IBM - searches for images having similar colour or layout

'Modern' Databases

http://wwwqbic.almaden.ibm.com/cgi-bin/stamps-demo

Content-Based Retrieval

- **Image retrieval is hard**
  - It is often not clear when two images are ‘similar’
  - Image interpretation is unsolved and expensive
  - Different people expect different things

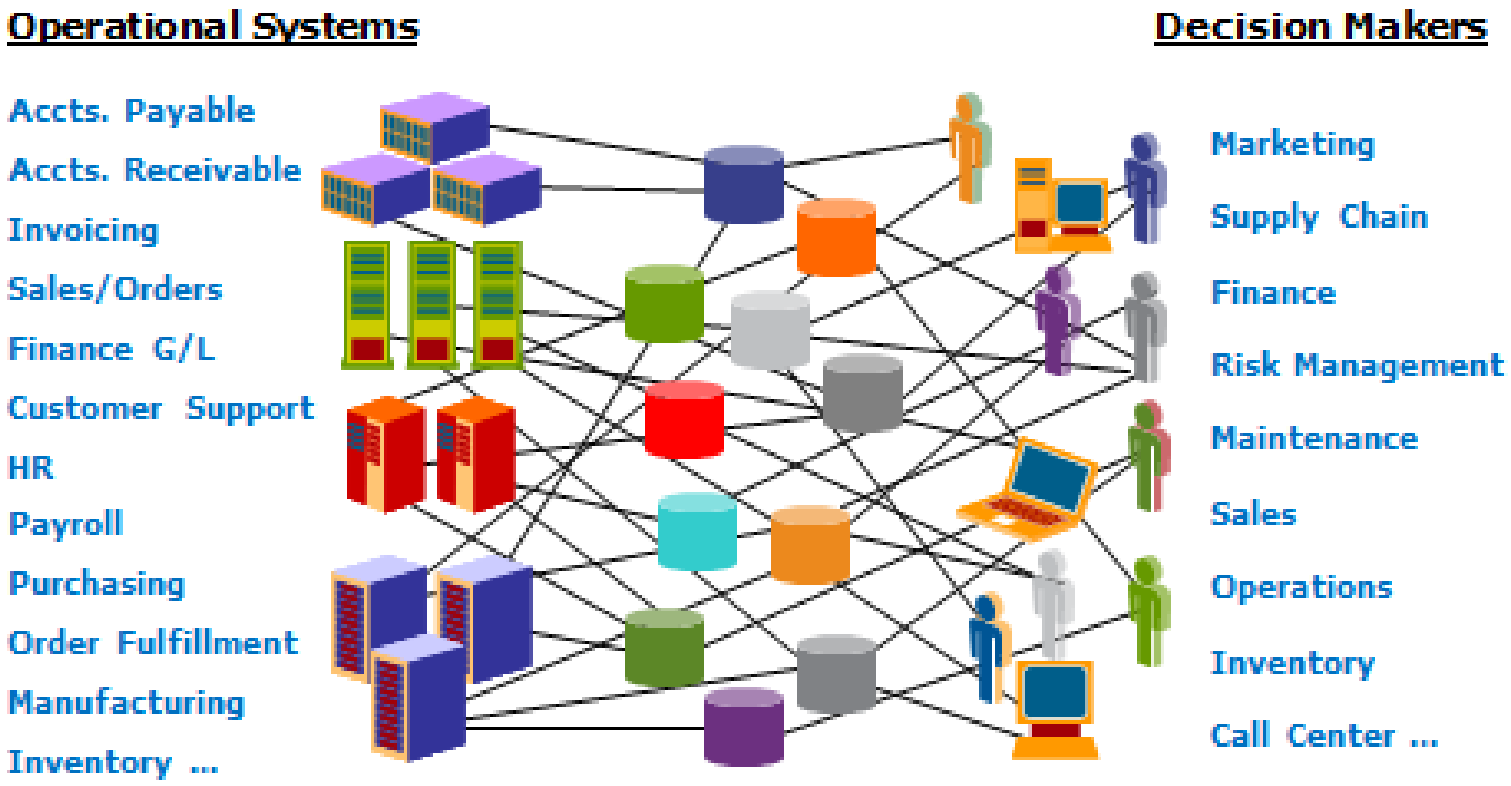
- **Do we look for?**
  - Images of roses
  - Images of red things?
  - Images of flowers?
  - Images of red flowers?
  - Images of red roses?

Chapter 5: Go deeper and wider

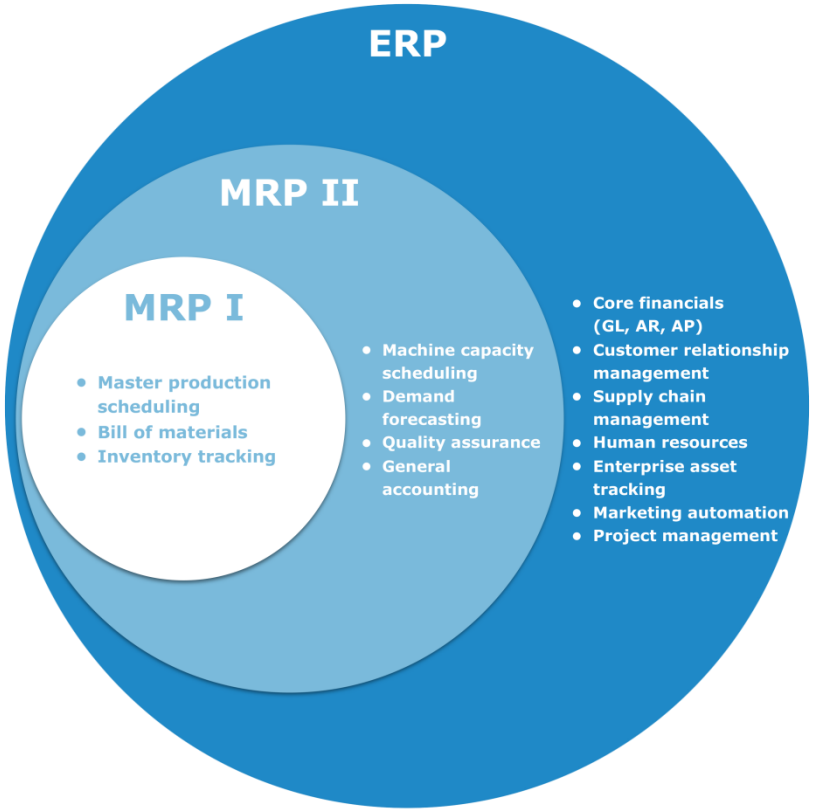
- **Overview about the evolution of Data Management & Analytics**
  - Evolution of DM
    - Before IT
    - In IT : File system, RDBMS, **ERP**, **DW**, Big Data
  - Hints –
    - Integrate DM with DA (Data Analytics)
      - ✓ Data – diverse (from structured to unstructured) and huge (from single to distributed)
- **SQL again**
- **Final (if you want to share)**

1 Integration of the Data and Analytics

- **DBMS: Transaction data are stored in many scattered databases**
  - It's easy to do IDUQ (insert, delete, update, query) – OLTP style



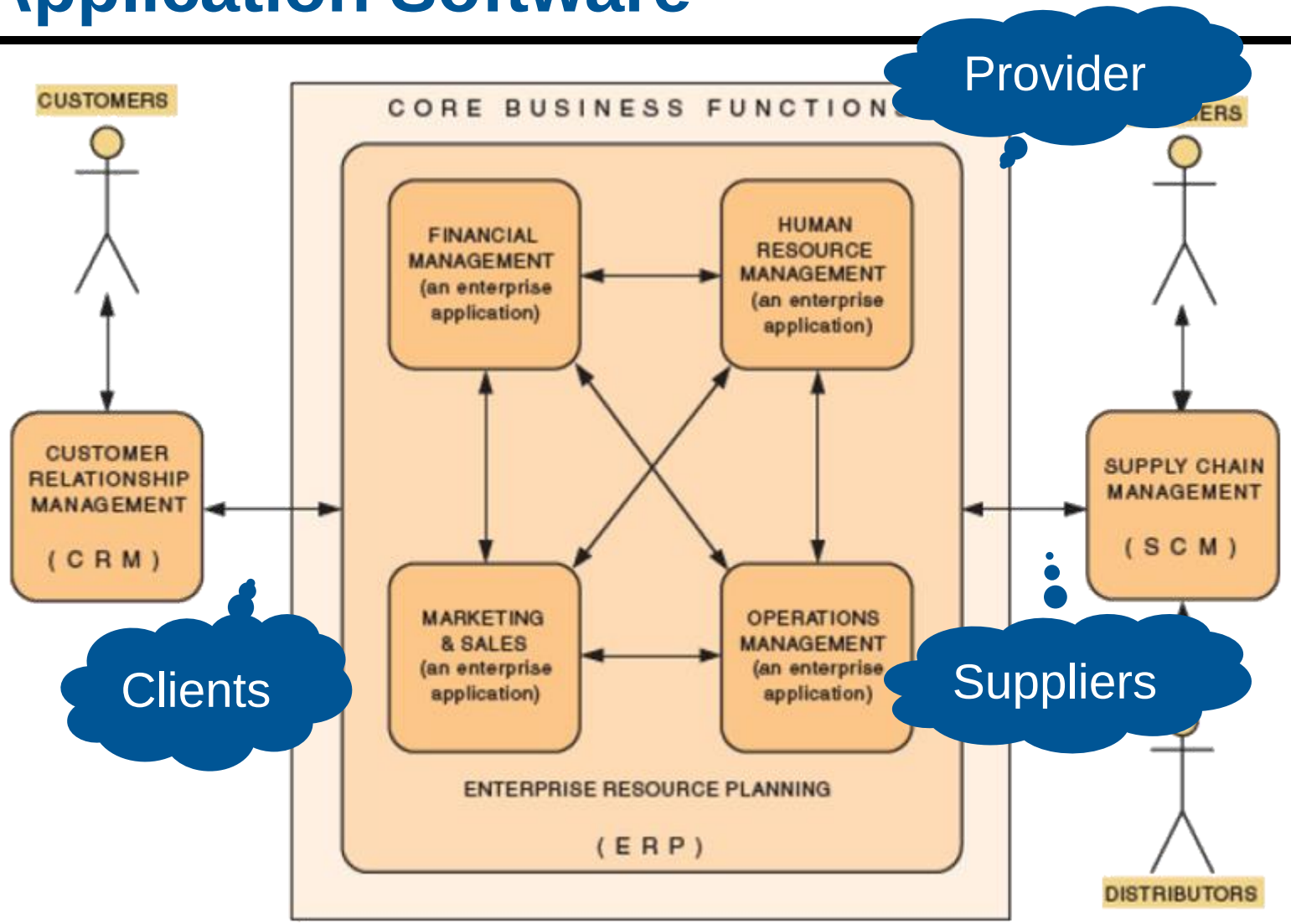
From MRP to ERP



- **MRP I/II: Manufacturing Resource Planning**
  - MRP I was some of the first business software to be widely adopted during the 1970s.
    - Manufacturers sought these systems in order to improve efficiency and accuracy when it came to basic processes such as production scheduling and inventory management.
  - By the 1980s, manufacturers realized they needed software that could also tie into their accounting systems and forecast inventory requirements - MRP II
- **ERP: Enterprise resource planning**
  - Business analysis
  - **ERP** systems are regarded as the successors to MRP II software. ERP suites include applications that are well outside the scope of manufacturing.

http://www.softwareadvice.com/resources/mrp-vs-mrp-ii/

Great applications are constructed – Enterprise Application Software



https://www.voortman.net/en/company/news/448-mrp-erp-or-mis

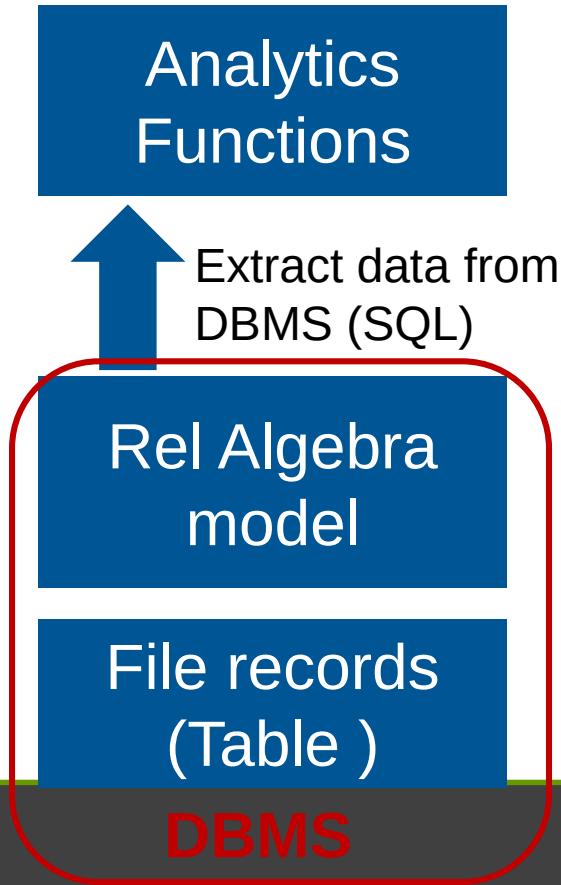
- **SAP, TeraData etc.**

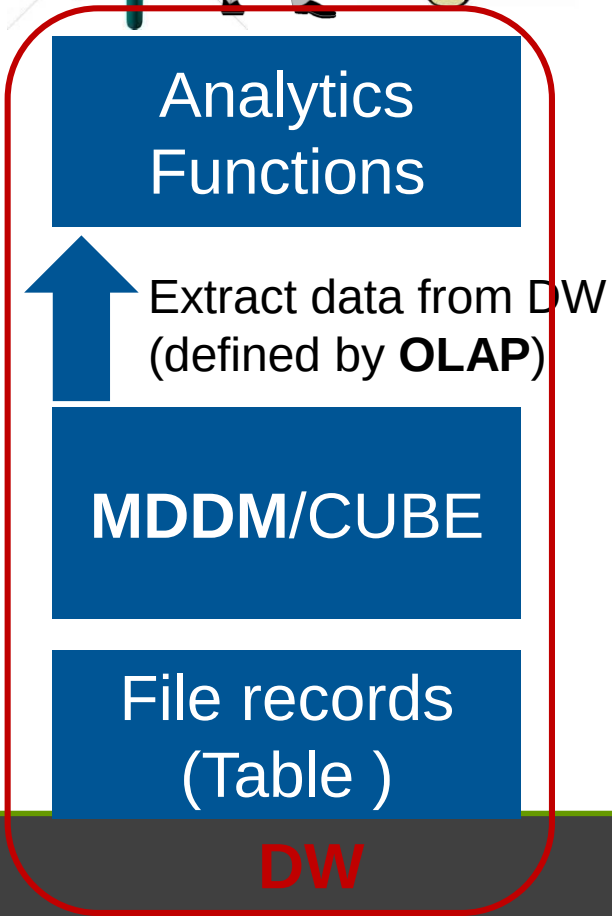
TERADATA

THE BEST DECISION POSSIBLE™

But, analytics is still separated

- + **Integrate** Data analytics into RDBMS was tried in 1990s
- + **diverse** data



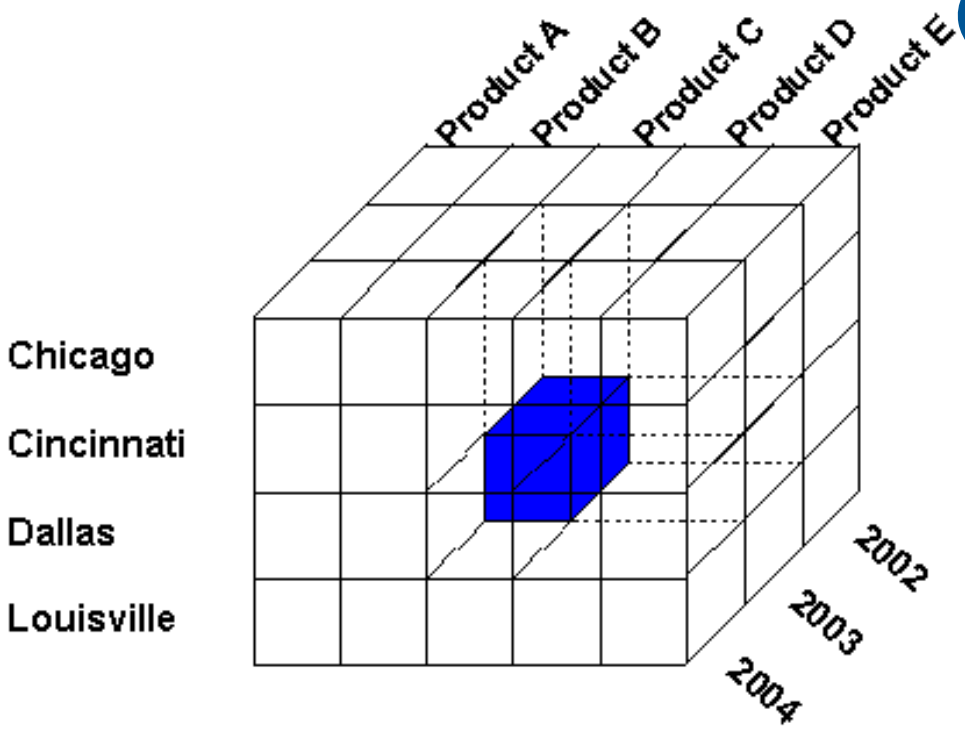


Statistic result is stored in advace

http://docs.oracle.com/html/B13915_04/i_olap_chapter.htm

- **Define the data with the multi-dimensional view**
  - A multidimensional data is optimized for a… …ata. Such data sources are sometimes calle… analytical processing (OLAP) data so…

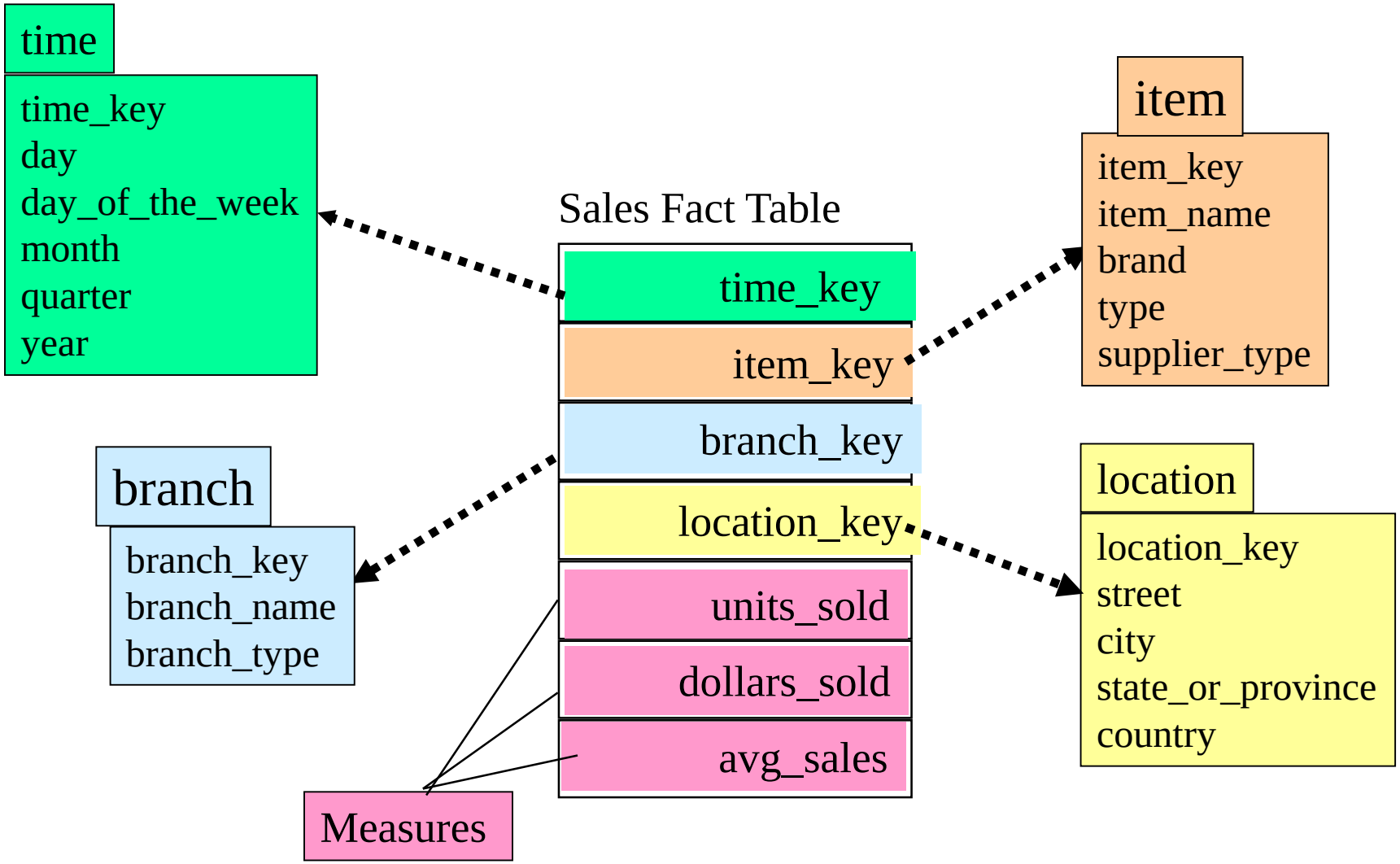
Some cumulative information is stored in advanced and will be updated always



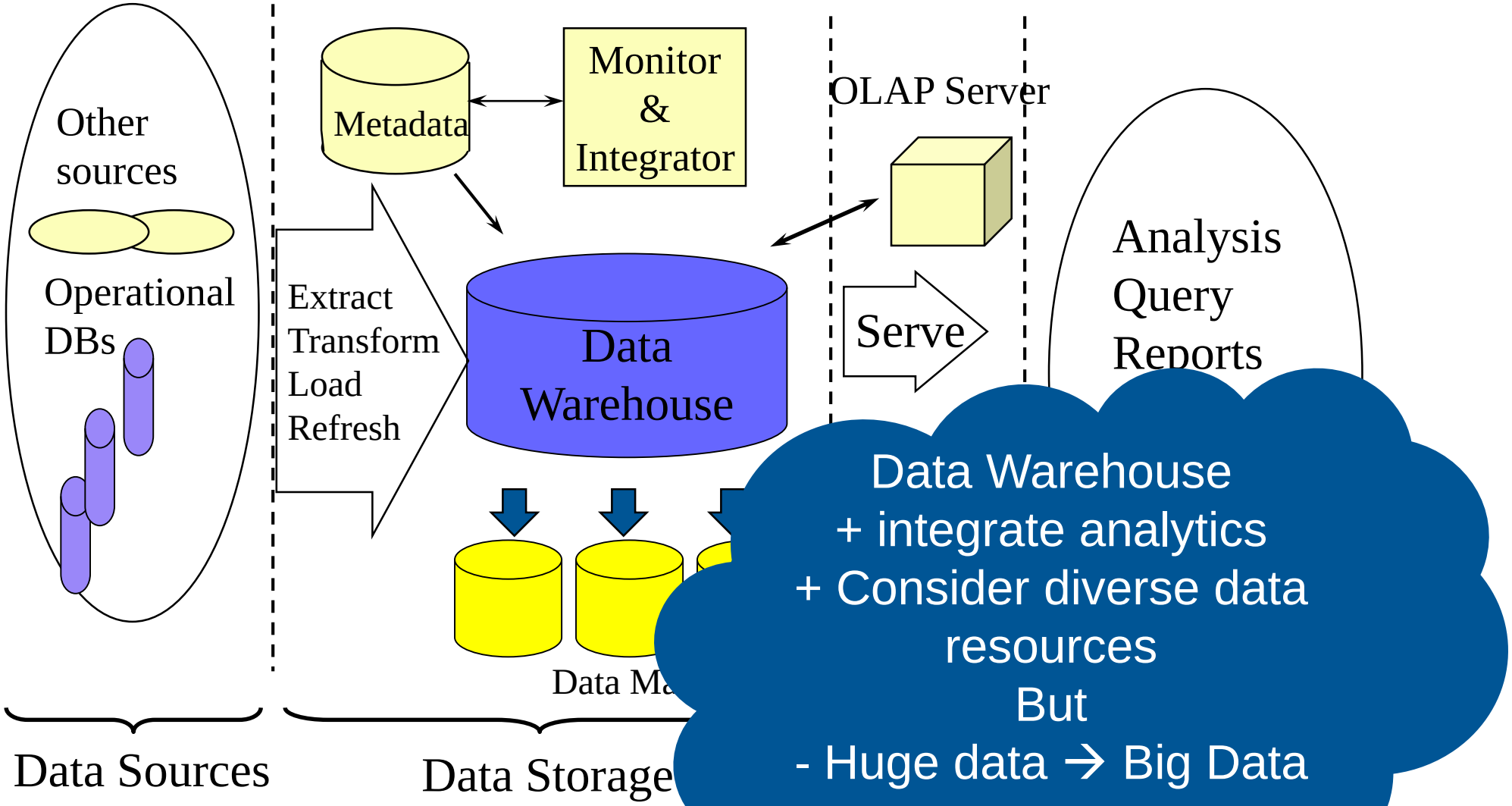
Page Items: Product All Pro… Time 2000

	Sales		
	All C…nels	Online	Retail
Worldwide	116,857,329	75,635,960	41,221,369
Americas	34,267,589	22,106,466	12,161,123
Asia	20,816,479	13,429,064	7,387,415
Australia	9,665,096	6,235,097	3,429,999
Europe	52,108,165	33,865,333	18,242,832

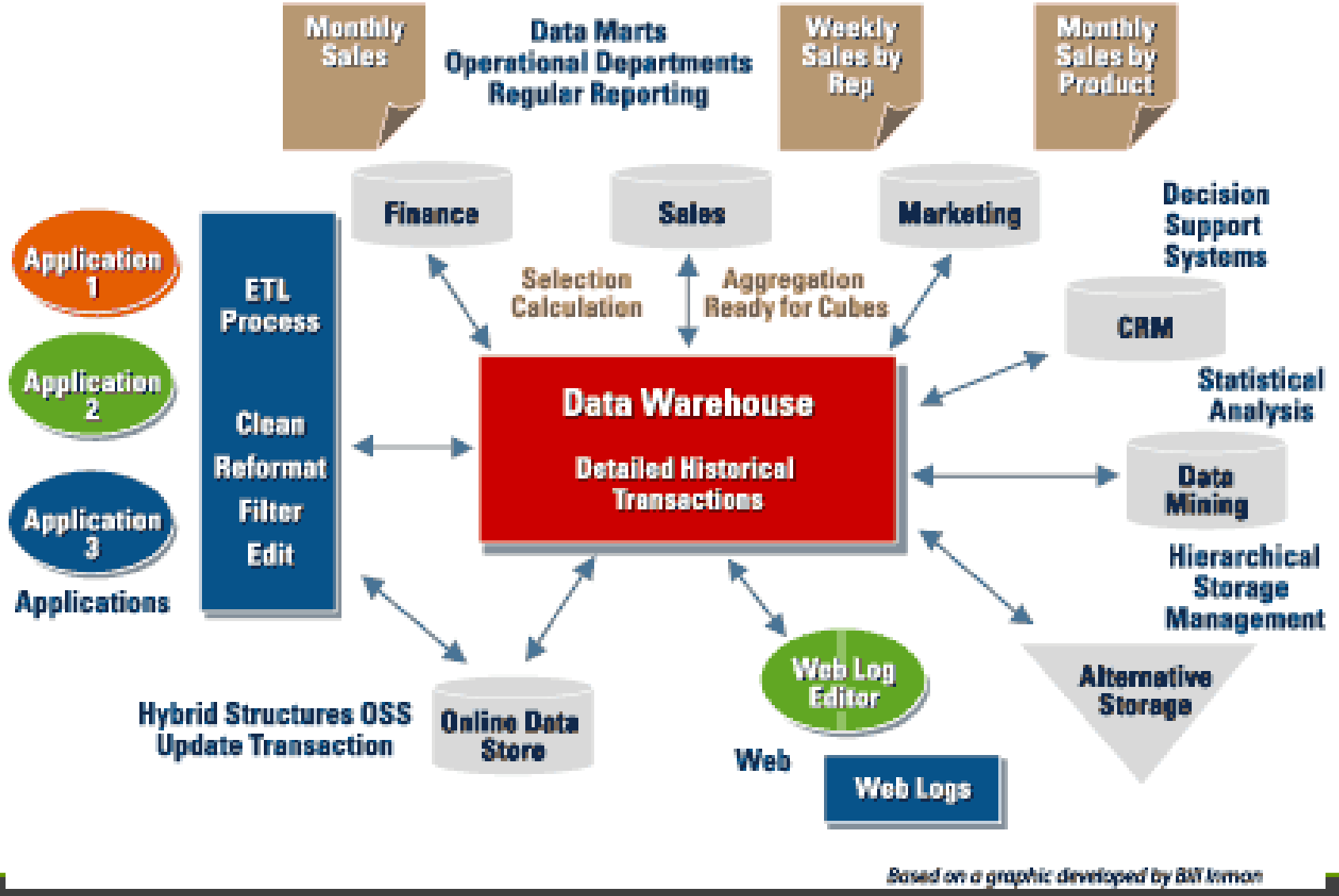
MDDM of Star Schema (based on RDBMS)



Data Warehouse
integrate analytics with data management



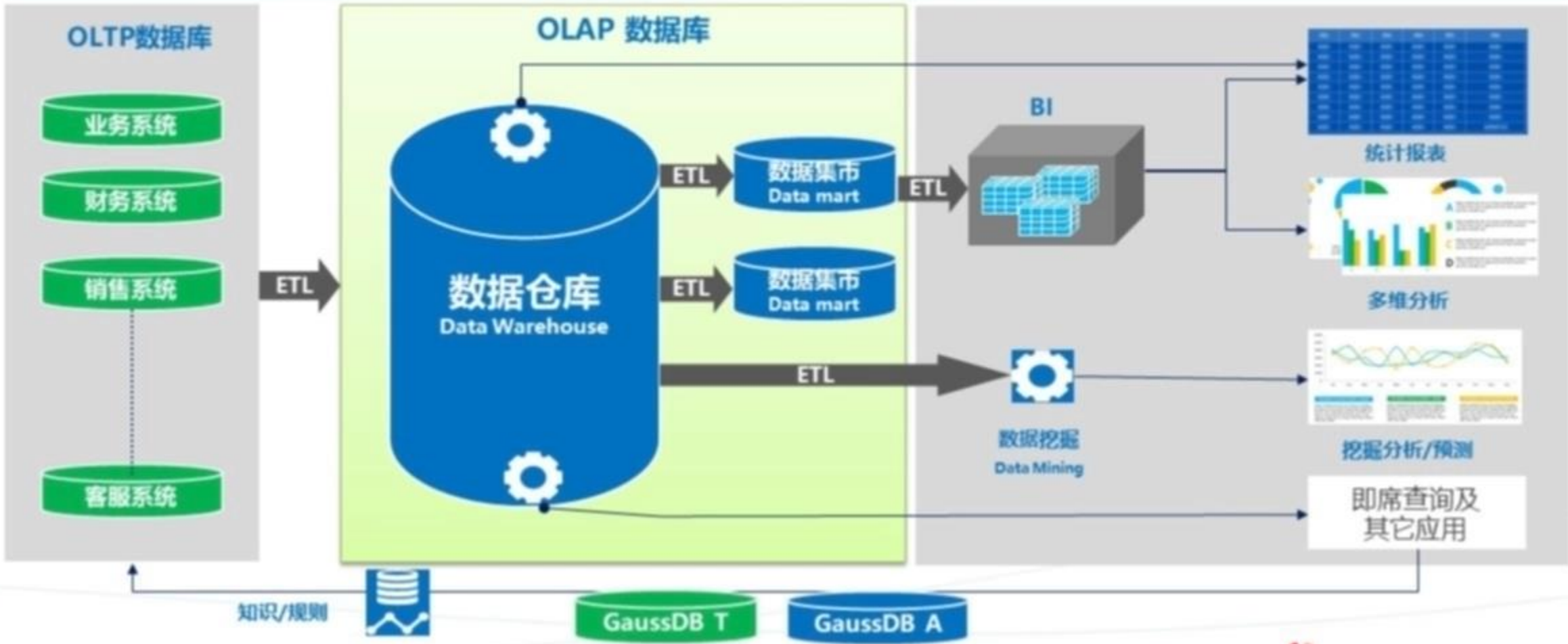
Data Warehouse tries to integrate (Later)



http://download.101com.com/pub/tdwi/images/ww20/Syncsort-dia1.gif

典型的企业OLTP和OLAP数据库

- 联机事务处理(OLTP)： 存储/查询业务应用中活动的数据以支撑日常的业务活动；
- 联机分析处理(OLAP)：存储历史数据以支撑复杂的分析操作，侧重决策支持；

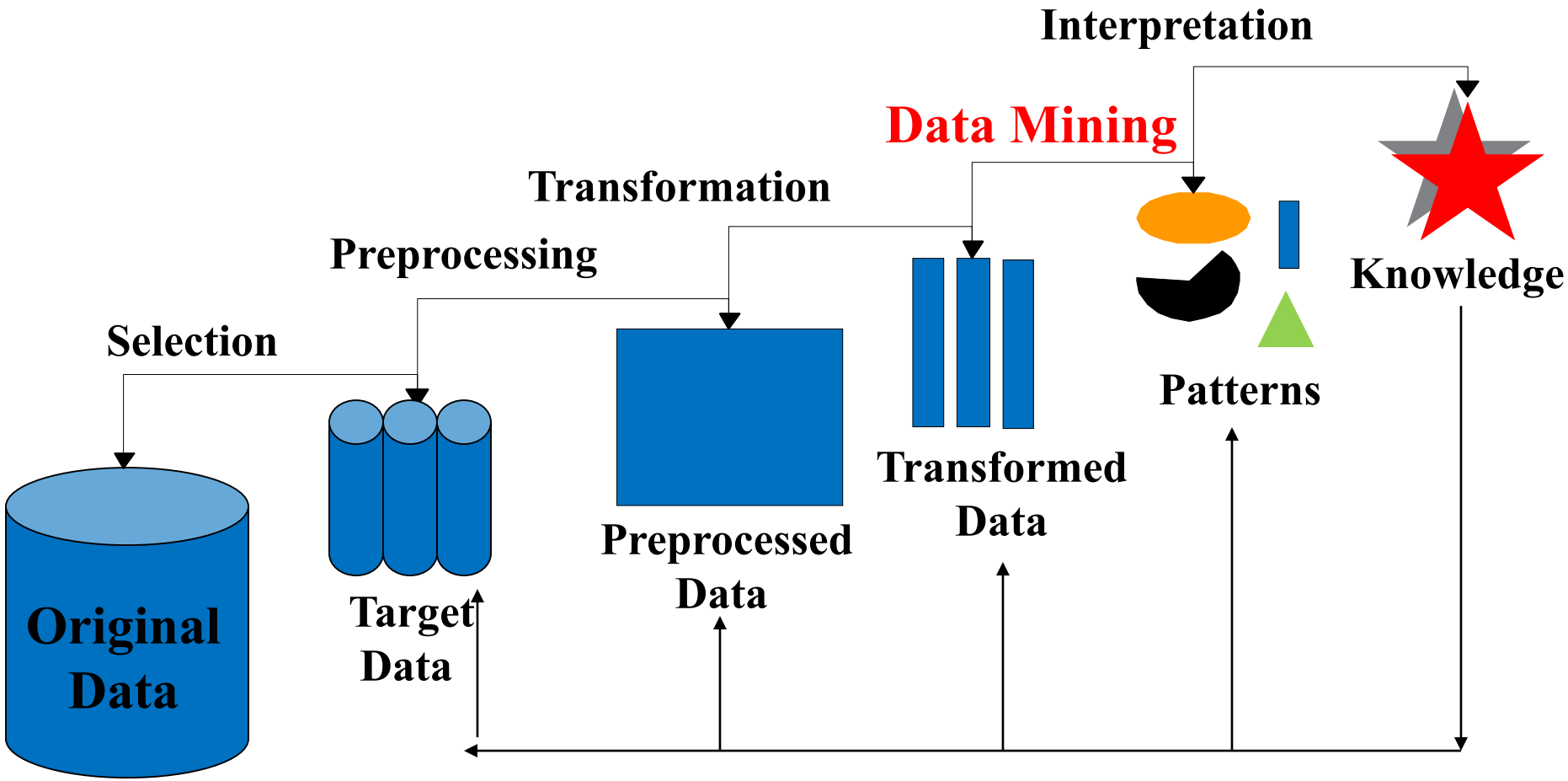


Data Mining

- **Tools that:**
  - Proactively and automatically search the data
  - uncover problems or opportunities hidden in data relationships
  - form computer models based on their findings, and then
  - use the models to predict business behavior
- **A methodology designed to perform knowledge discovery expeditions over the database data with only minimal end-user intervention during the discovery phase**

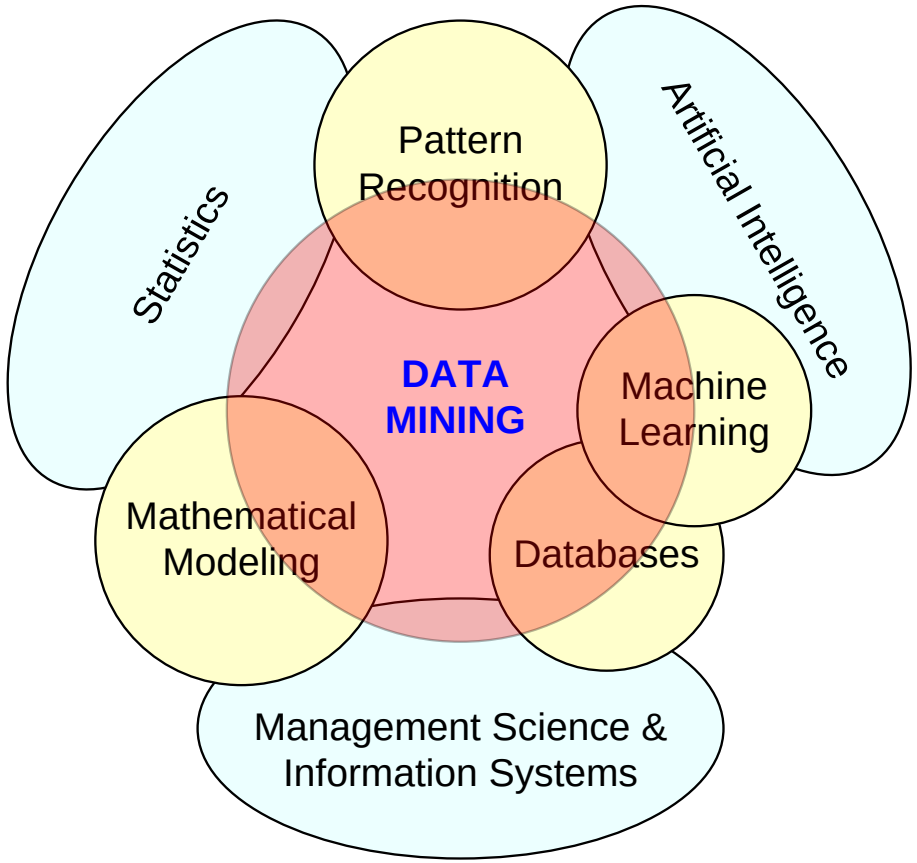
More popular model for Data Mining- KDD

- KDD: Knowledge Discovery from Data

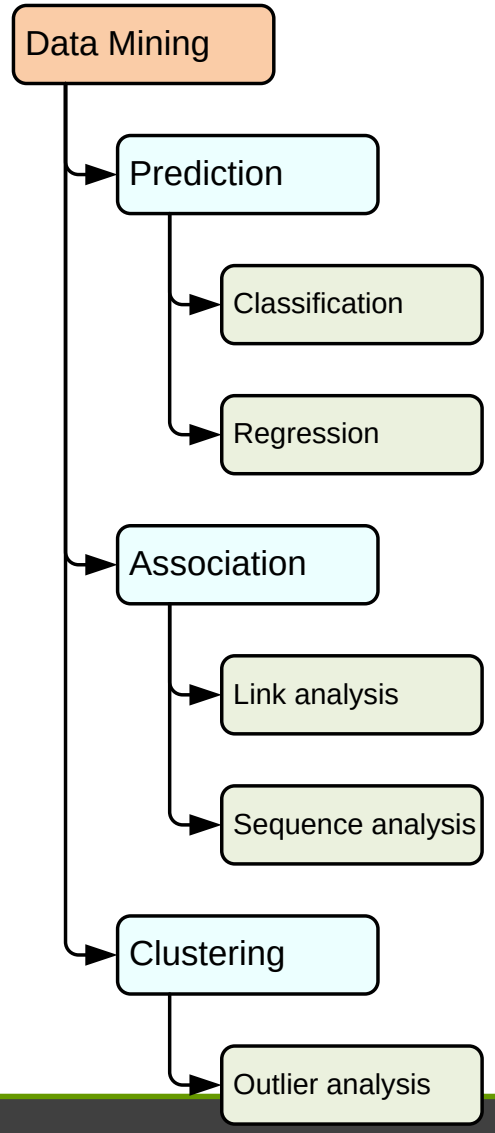


Origins of Data Mining

- **Draws ideas from AI, machine learning, pattern recognition, statistics, and database systems**
- **Traditional Techniques may be unsuitable due to**
  - Enormity of data
  - High dimensionality of data
  - Heterogeneous, distributed nature of data

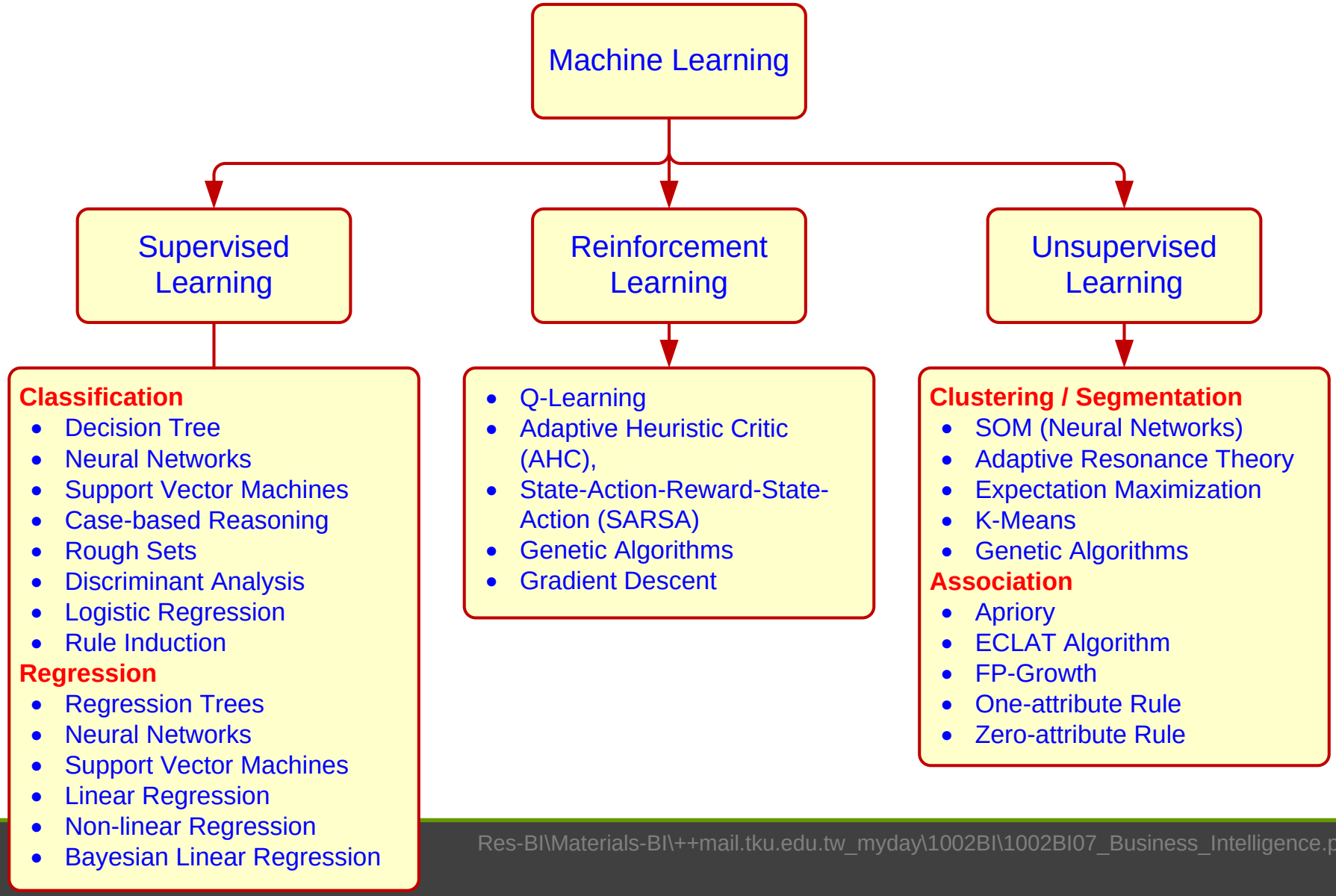


A Taxonomy for Data Mining Tasks



Learning Method	Popular Algorithms
Supervised	Classification and Regression Trees, ANN, SVM, Genetic Algorithms
Supervised	Decision trees, ANN/MLP, SVM, Rough sets, Genetic Algorithms
Supervised	Linear/Nonlinear Regression, Regression trees, ANN/MLP, SVM
Unsupervised	Apriory, OneR, ZeroR, Eclat
Unsupervised	Expectation Maximization, Apriory Algorithm, Graph-based Matching
Unsupervised	Apriory Algorithm, FP-Growth technique
Unsupervised	K-means, ANN/SOM
Unsupervised	K-means, Expectation Maximization (EM)

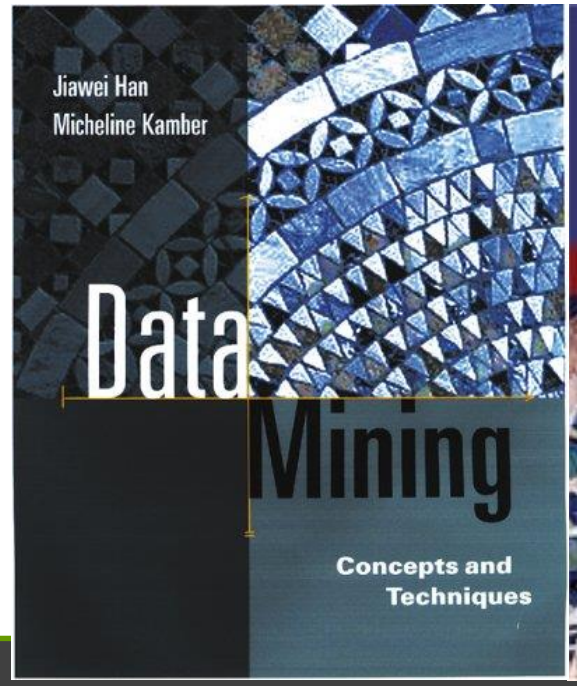
Machine Learning Methods

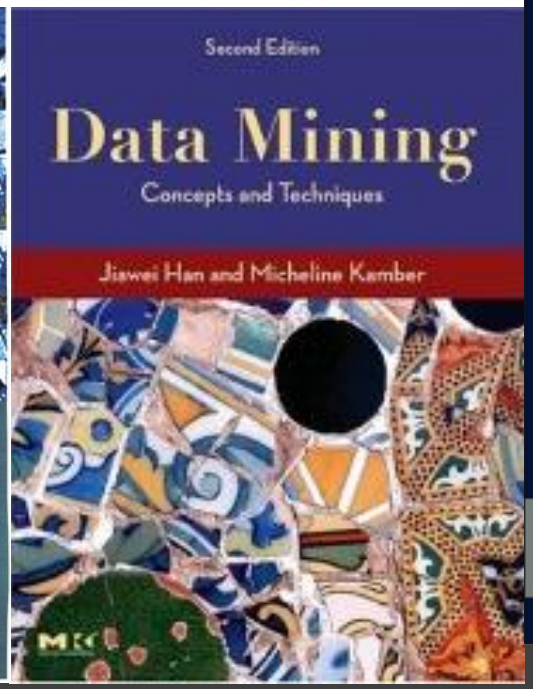


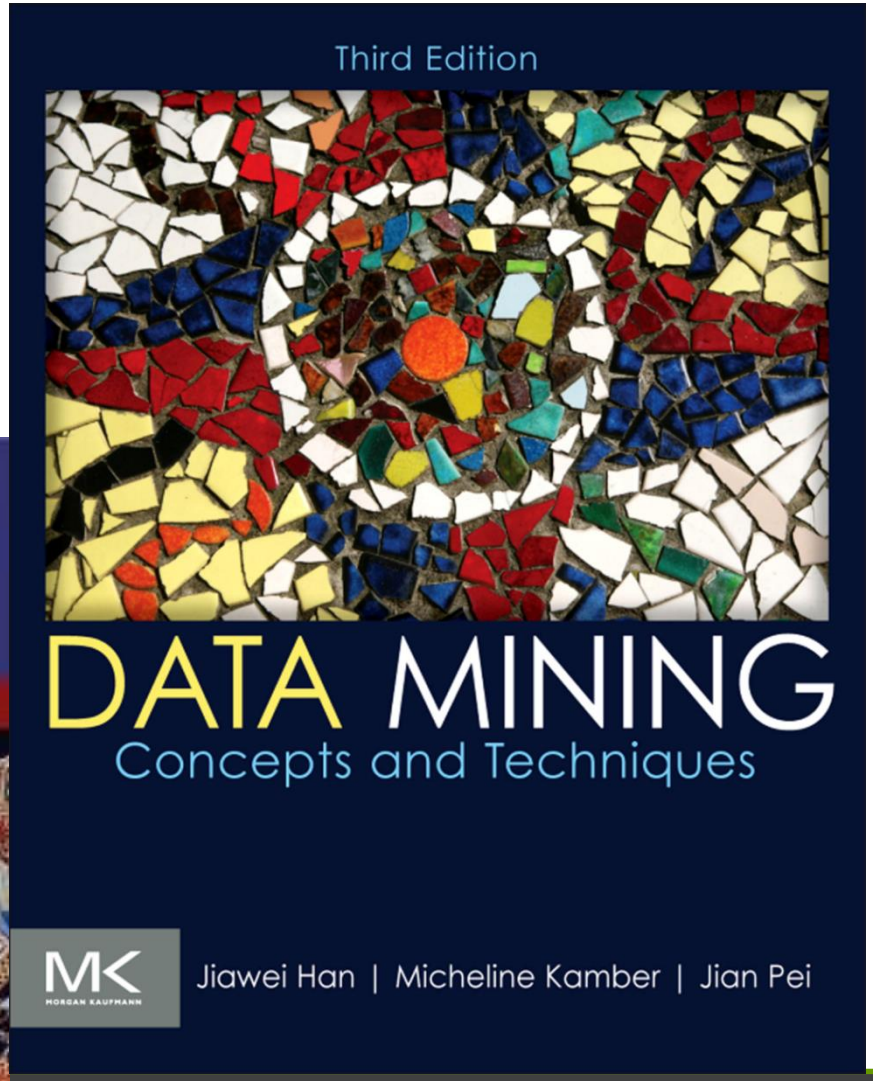
Additional reference

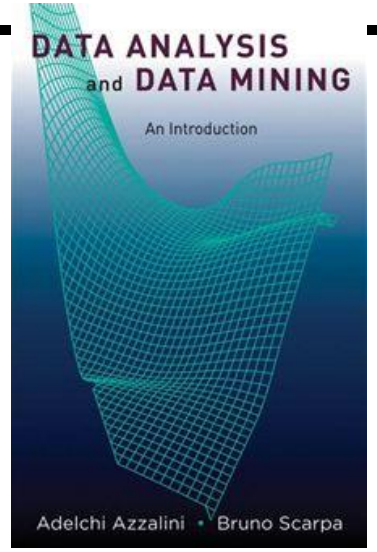
http://www.cs.uiuc.edu/homes/hanj/

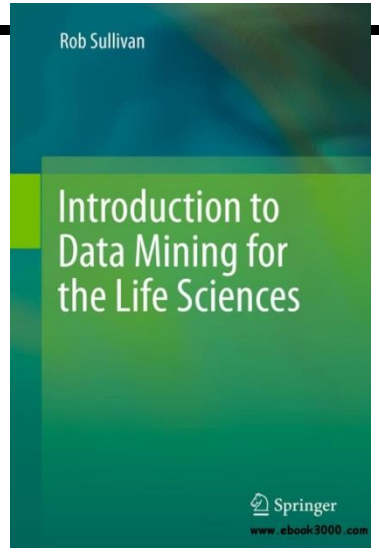
- **Data Mining: Concepts & Techniques**
  - By Jiawei Han and Micheline Kamber
  - 3rd edition

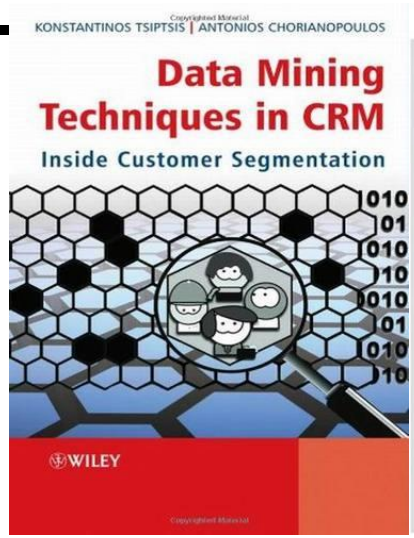


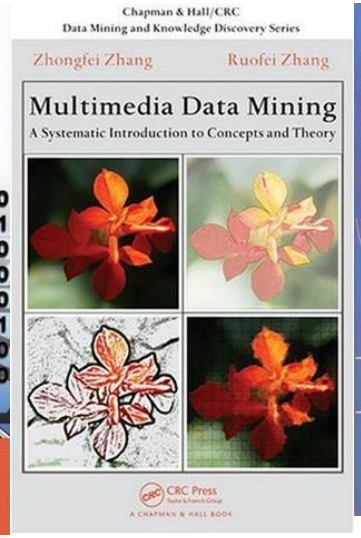


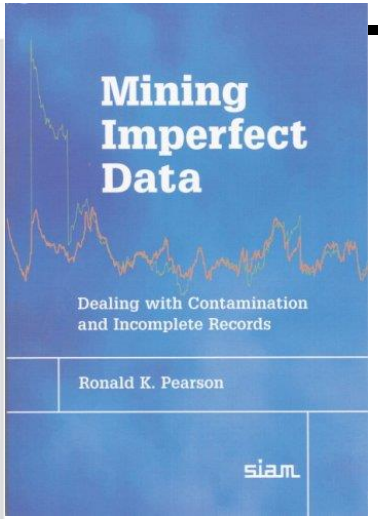



DATA ANALYSIS
and DATA MINING
An Introduction
Adelchi Azzalini • Bruno Scarpa

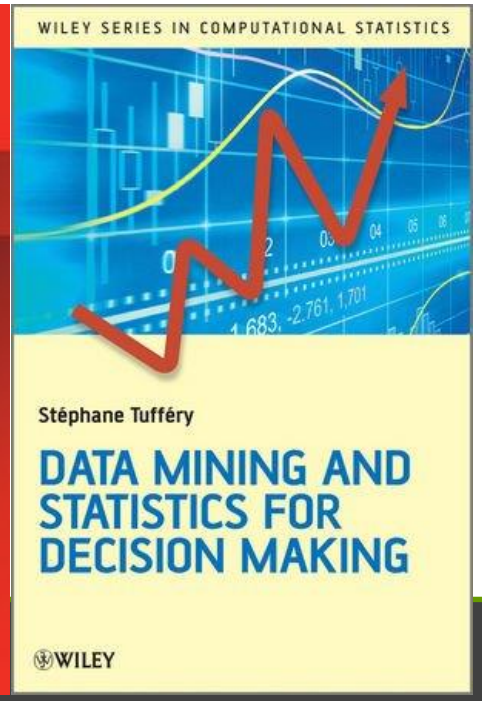

Rob Sullivan
Introduction to
Data Mining for
the Life Sciences
Springer
www.ebook3000.com

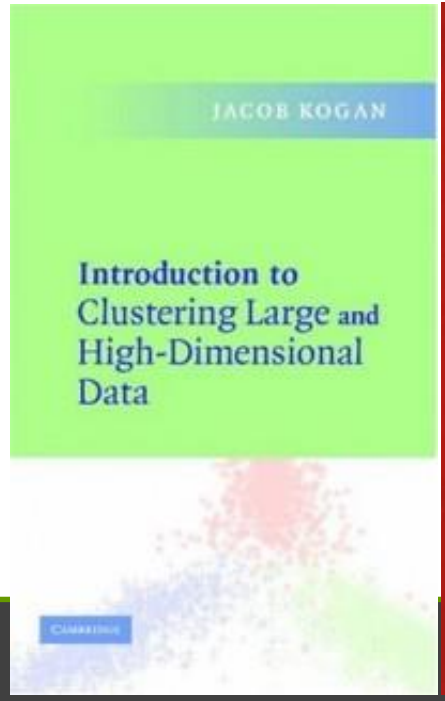

KONSTANTINOS TSIPTSIS | ANTONIOS CHORIANOPOULOS
Data Mining
Techniques in CRM
Inside Customer Segmentation
WILEY

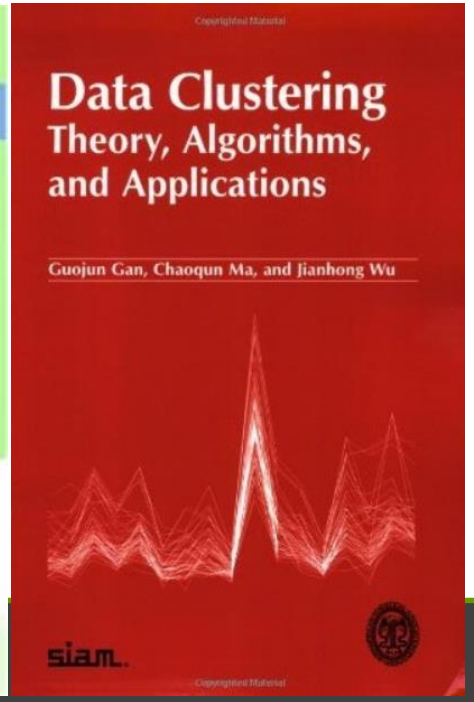

Chapman & Hall/CRC
Data Mining and Knowledge Discovery Series
Zhongfei Zhang
Ruofei Zhang
Multimedia Data Mining
A Systematic Introduction to Concepts and Theory
CRC Press


Mining
Imperfect
Data
Dealing with Contamination
and Incomplete Records
Ronald K. Pearson
siam


LECTURE NOTES IN ELECTRICAL ENGINEERING | 25
Sio-Iong Ao
Data Mining
and Applications
in Genomics
Springer


WILEY SERIES IN COMPUTATIONAL STATISTICS
Stéphane Tufféry
DATA MINING AND
STATISTICS FOR
DECISION MAKING
WILEY


JACOB KOGAN
Introduction to
Clustering Large and
High-Dimensional
Data


Data Clustering
Theory, Algorithms,
and Applications
Guojun Gan, Chaoqun Ma, and Jianhong Wu
siam.

Chapter 5: Go deeper and wider

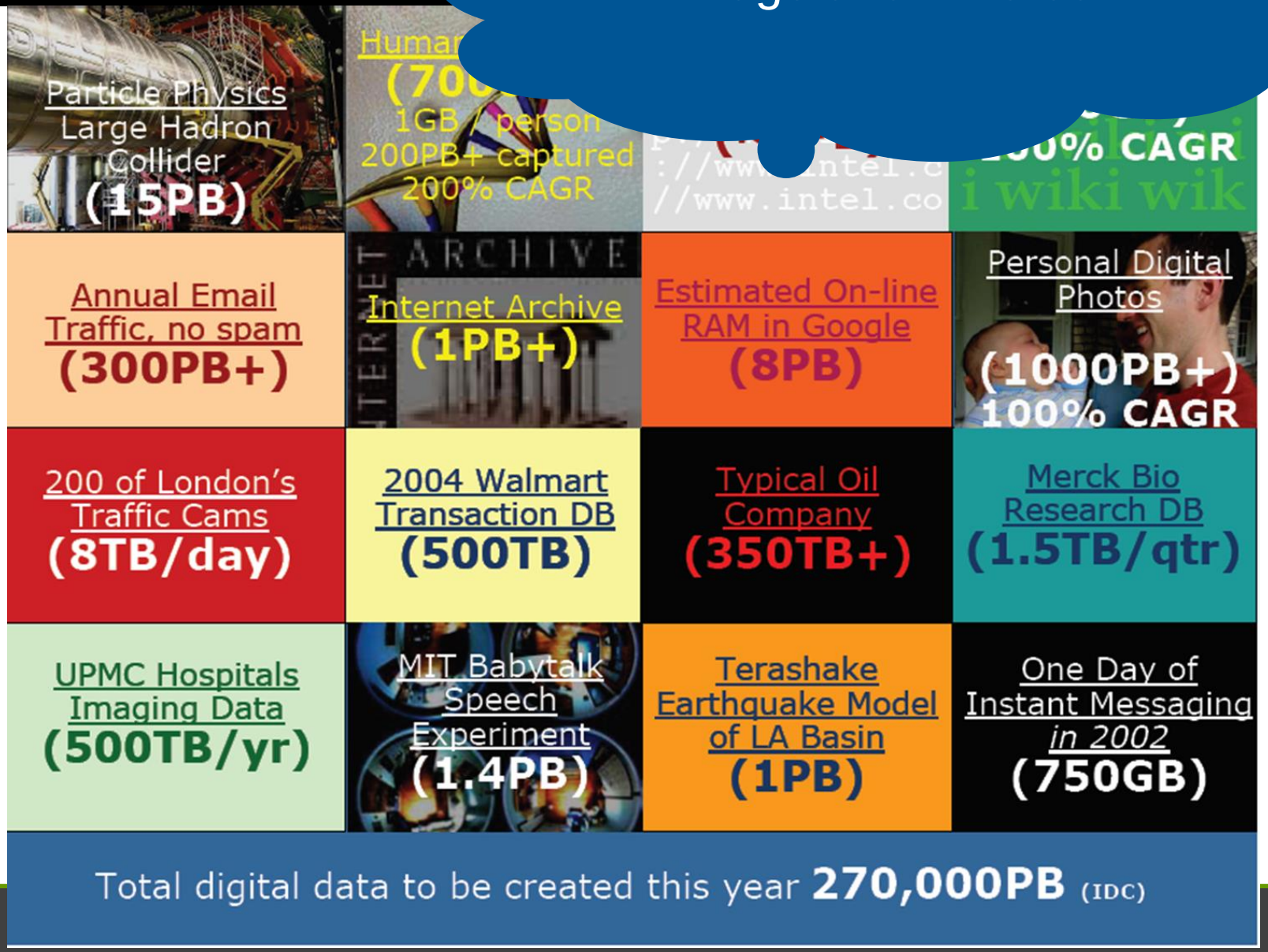
- **Overview about the evolution of Data Management & Analytics**
  - Evolution of DM
    - Before IT
    - In IT : File system, RDBMS, **ERP**, **DW**, **Big Data**
  - Hints –
    - Integrate DM with DA (Data Analytics)
      - ✓ Data – diverse (from structured to unstructured) and huge (from single to distributed)
- **SQL again**
- **Final (if you want to share)**

Big Data is coming

- **Ambitious to manage huge and diverse data – 3Vs: Volume, Variety, Velocity**
  - 2.5 quintillion bytes of data are generated every day!
    - A quintillion is 10^{18}
  - coming from many quarters, like Social media sites, Sensors, Digital photos, Business transactions, Location-based data, Web data, e-commerce, Bank/Credit Card, …
  - For information: Google processes 20 PB a day (2008)
    - http://www.worldwidewebsize.com/
      - ✓ "The Indexed Web contains **at least 9.18 billion pages** (Sunday, 09 December, 2012)."
  - For science: NASA & Hubble scope

Here comes Big Data

Huge and Diverse



Fundamental ideas to understand BD:
Distributed for huge +
Redundancy for data safty

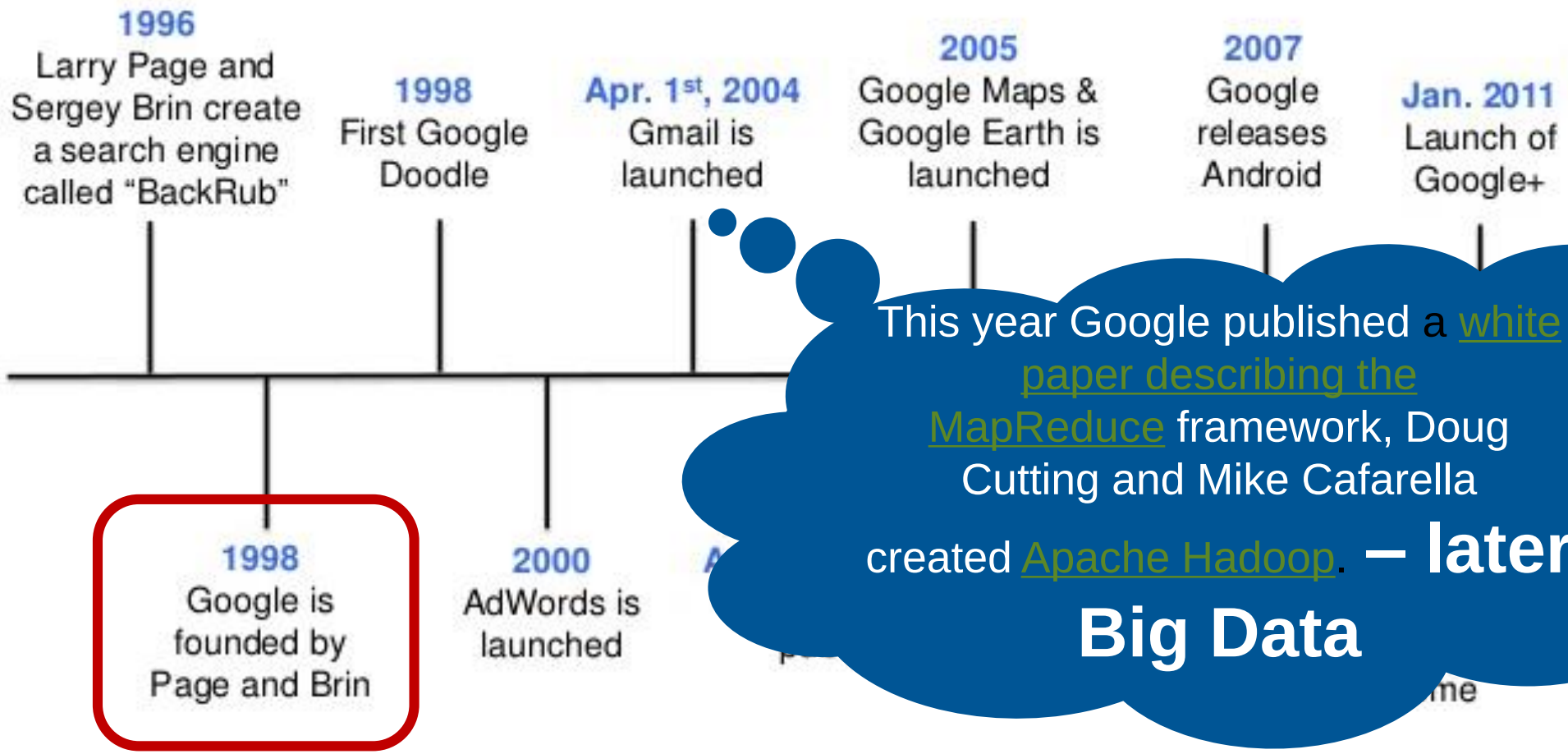
- **Making decisions!**
  - For governance: Edward Snowden

Are you pleased with the response speed?

- Found about 307,000,000 items (using 0.28 second)

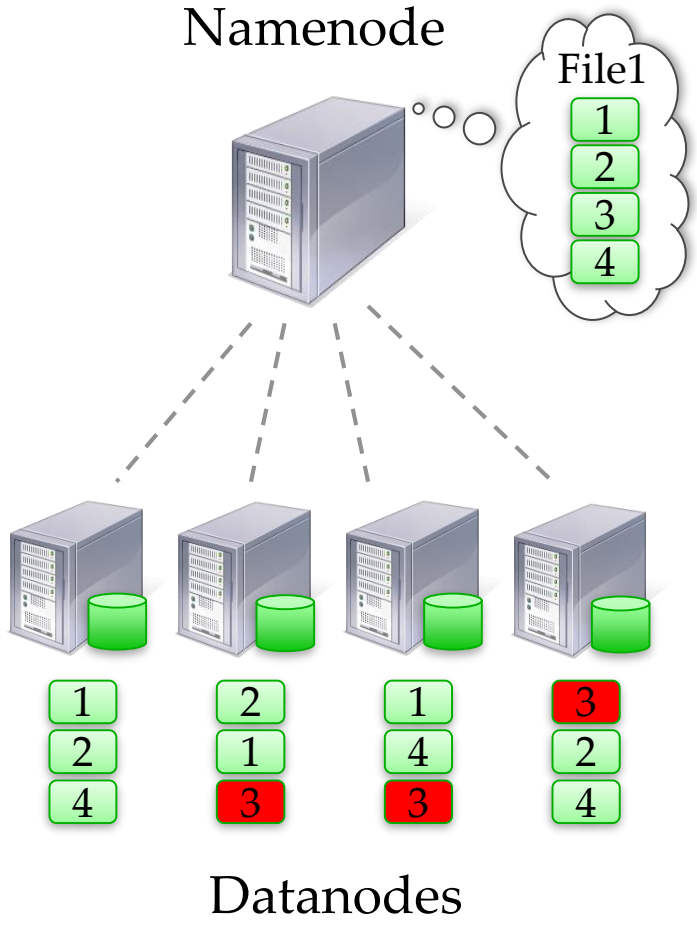
Google triggers other searches …

- CBIR, XML, …



HDFS/Hadoop with Cluster system

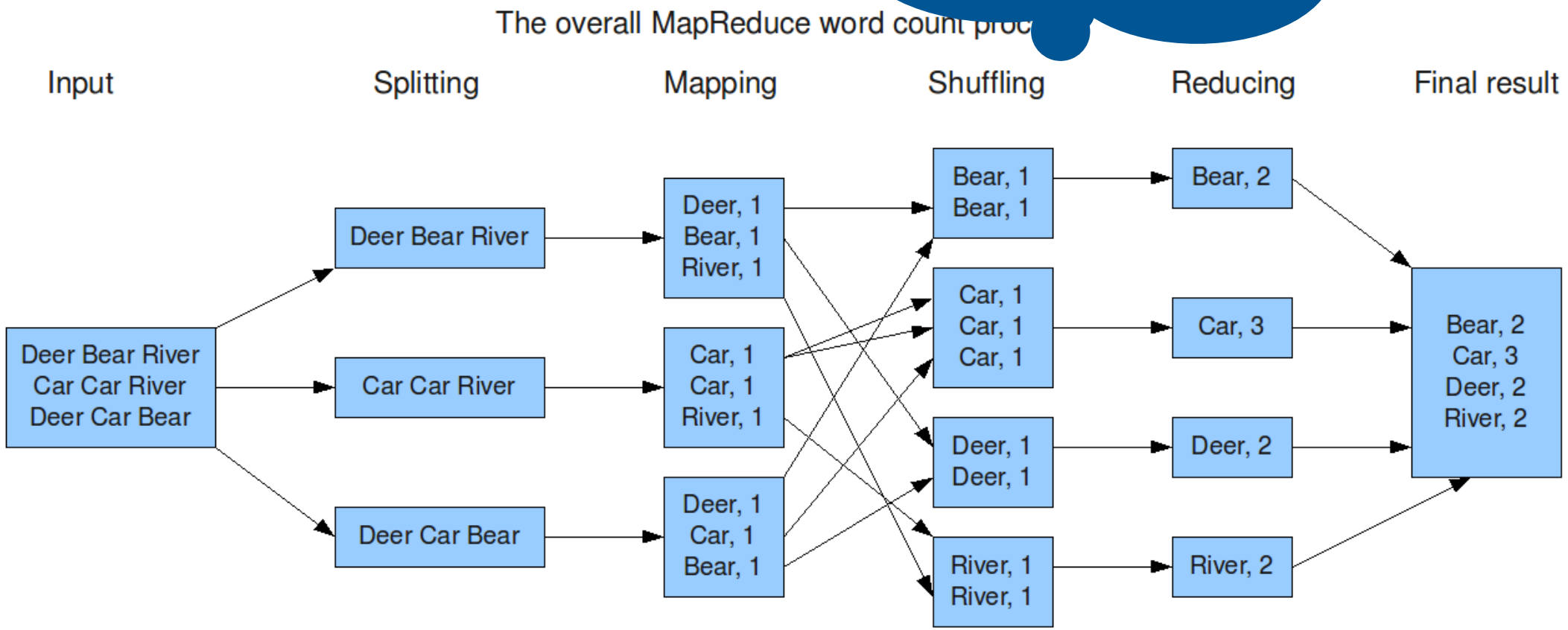
- **Hadoop Distributed file system (HDFS)**
  - Single namespace for entire cluster
    - Namenode stores metadata (file names, locations, etc)
  - Replicates data 3x for fault-tolerance
    - Files split into 128MB blocks
    - Blocks replicated across several datanodes (often 3)
  - Optimized for large files, sequential reads
  - Files are append-only



http://www.cs.berkeley.edu/~demmel/cs267_Spr12/Lectures/Zaharia_Cloud_2012.ppt

Map/Reduce - YARN

M/R is too naïve to manipulate data – word counting. We need more flexible query for analytics

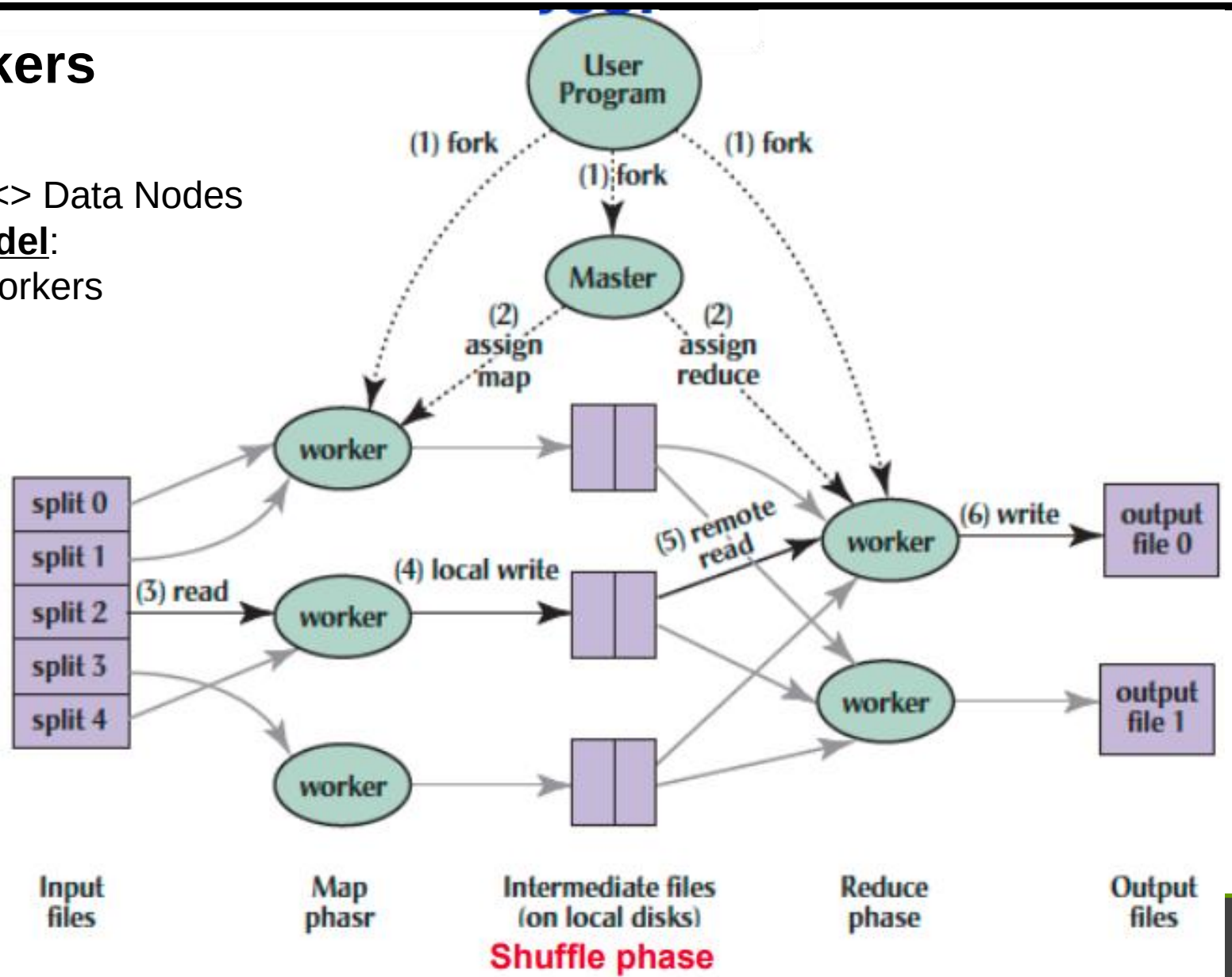


Computing model

Master – Workers

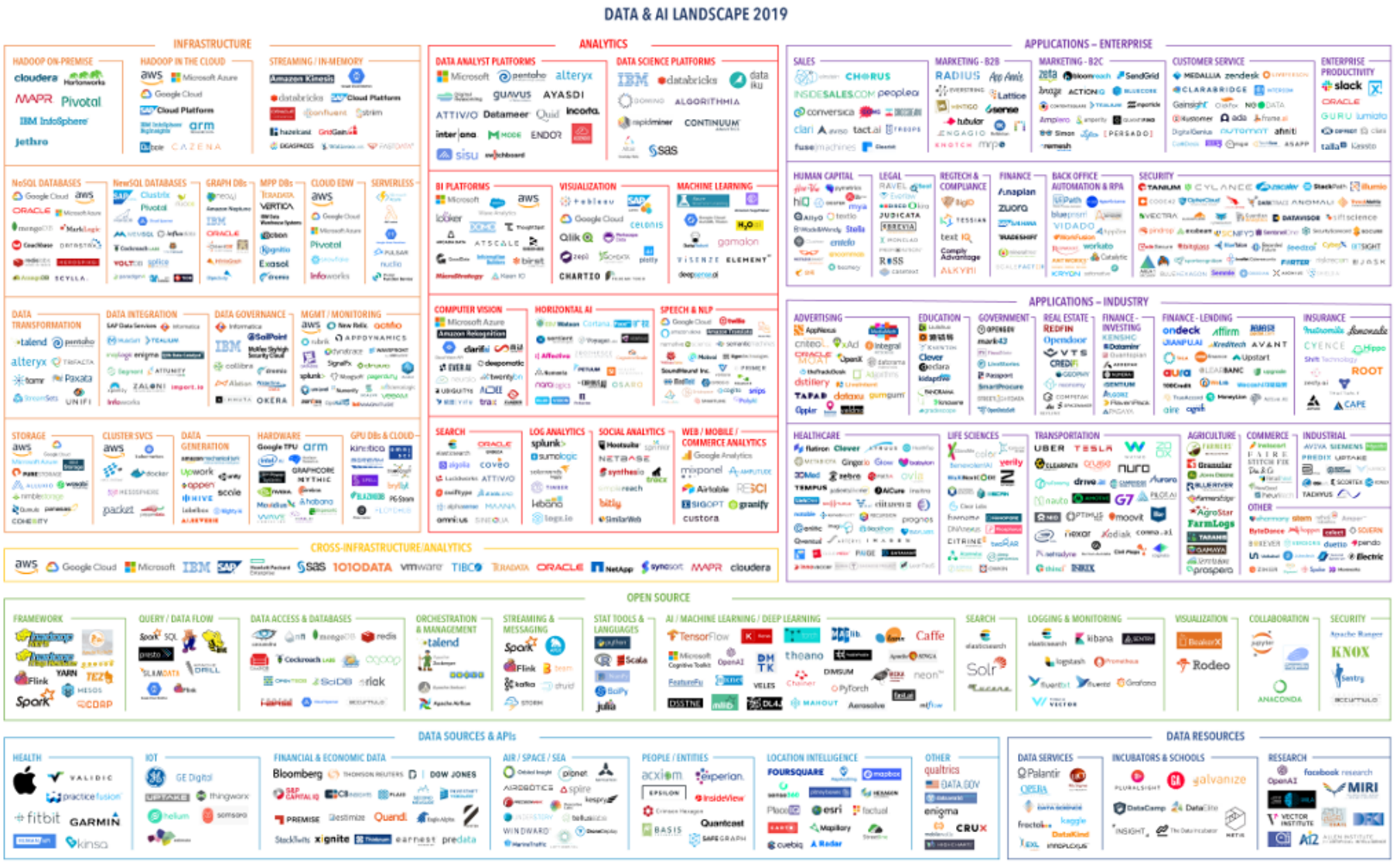
Data Storage:
Name node <> Data Nodes

Computing Model:
Master <> Workers

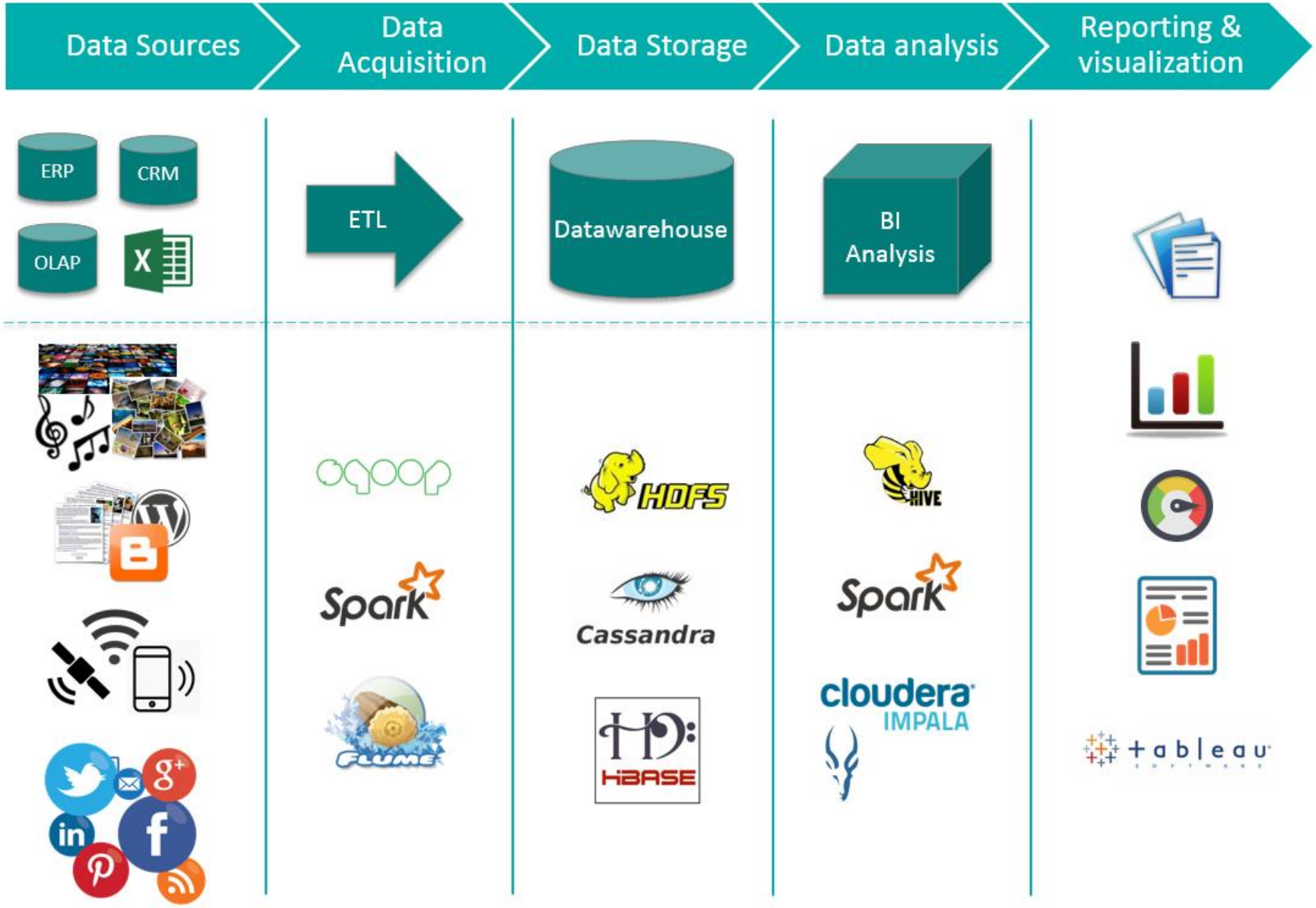


http://mattturck.com/bigdata2017/

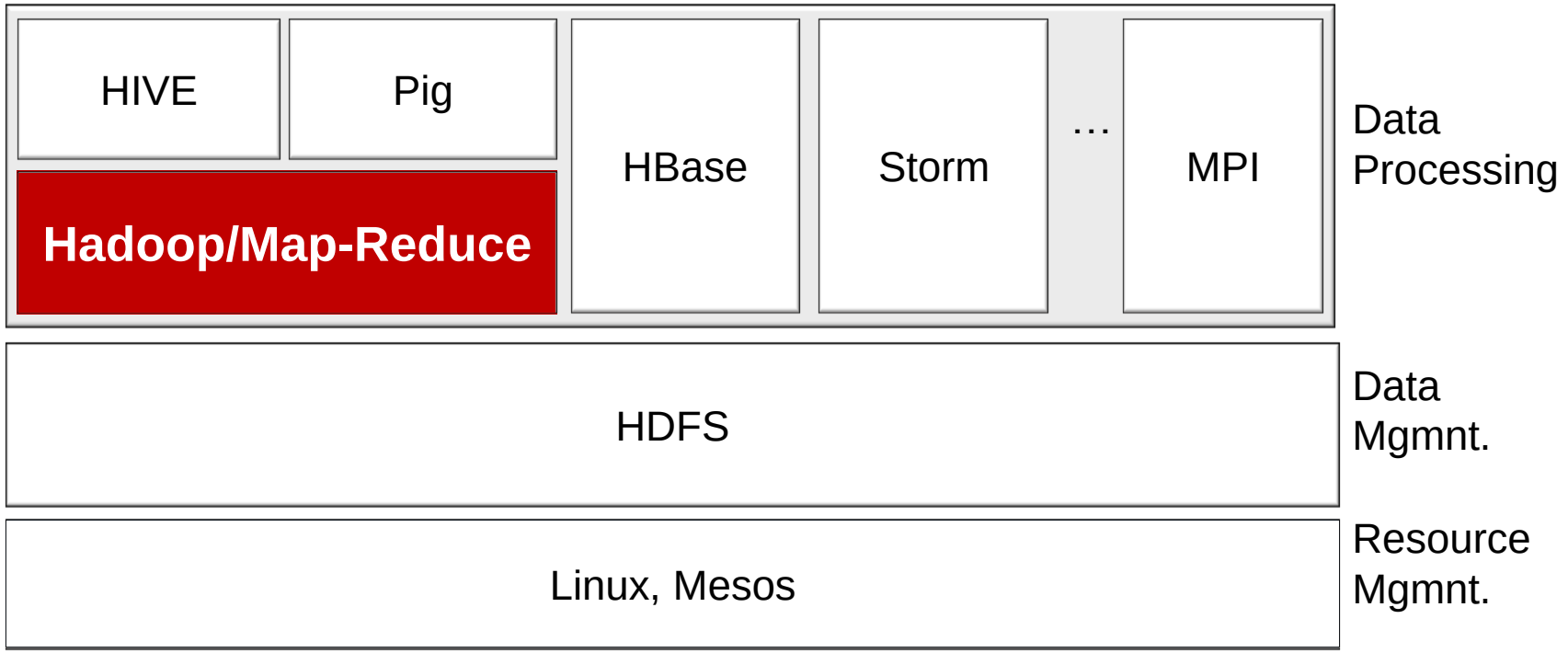
The 2019 Big Data Landscape



Integrating Data + Analytics is also carried in BD

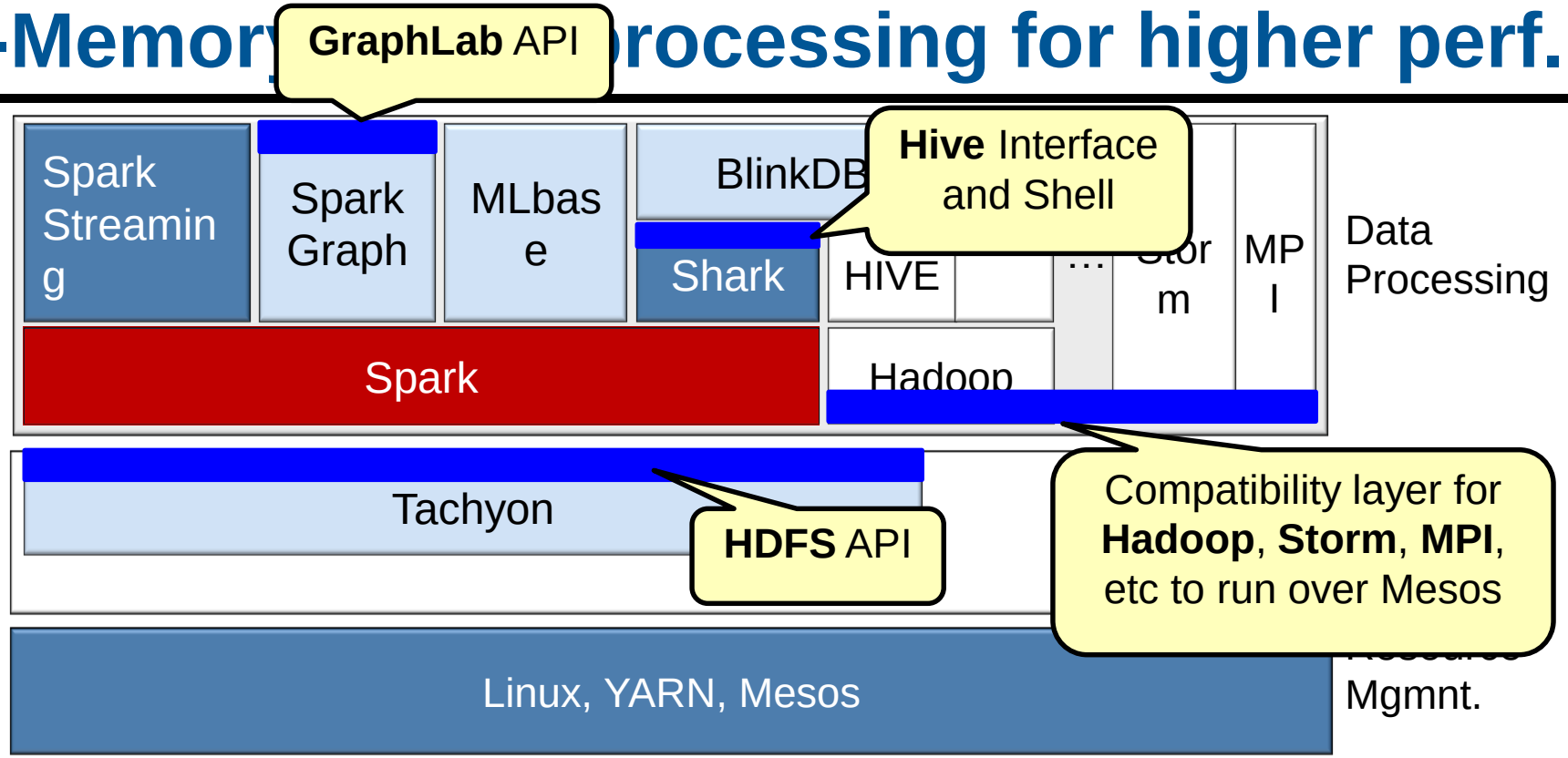


1st generation



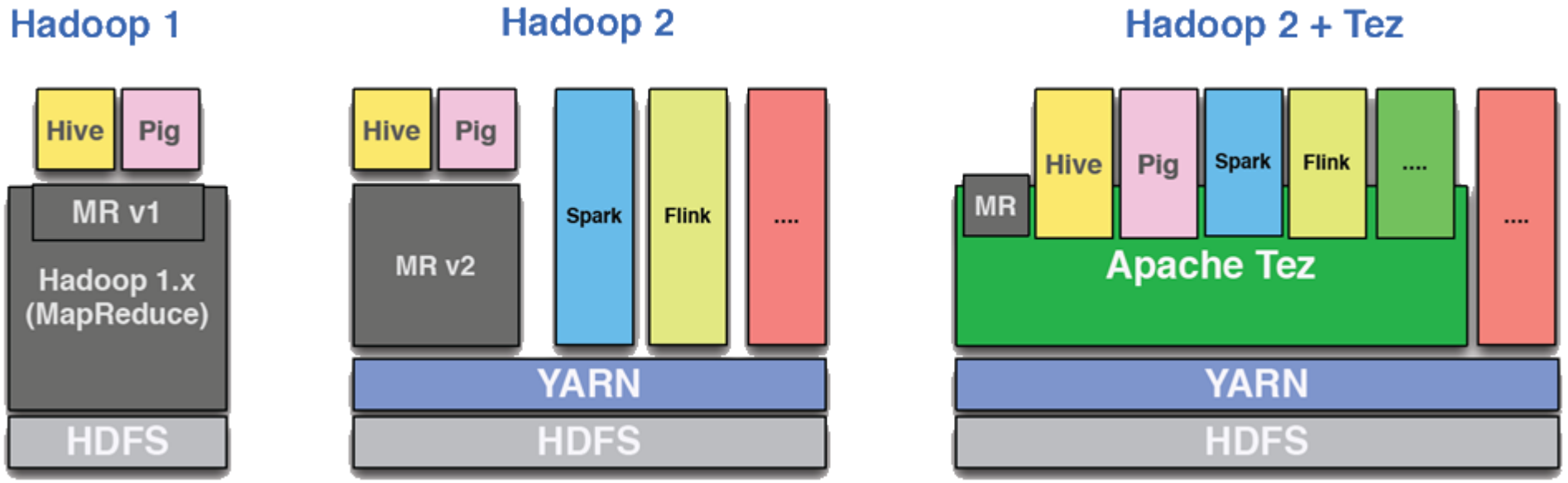
- **The data processing is based on M/R model, which required intermediate result be kept into/out of disks – low performance**

2nd generation – Spark – In-Memory processing for higher perf.



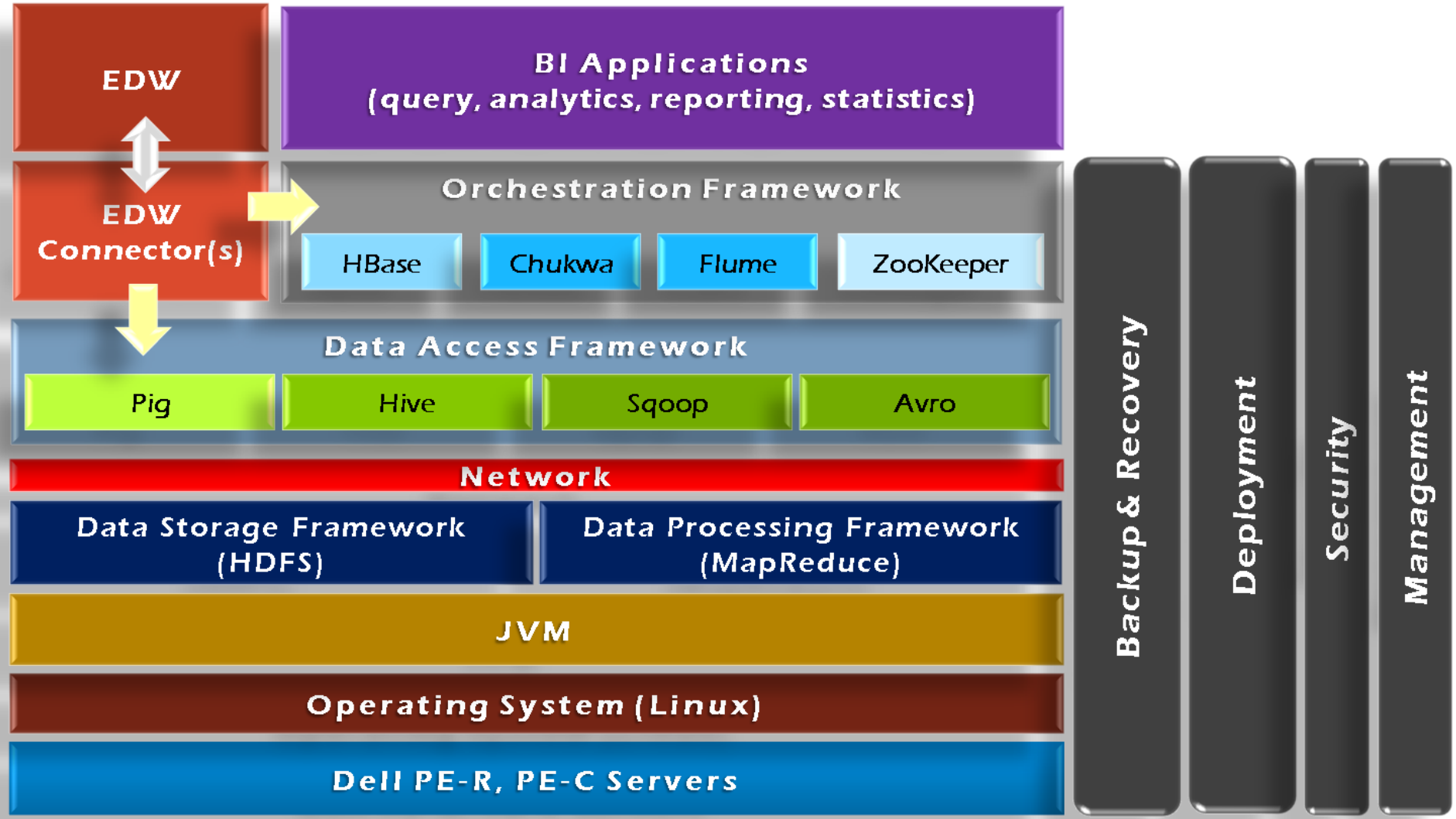
- The data processing is based on Spark model, in which intermediate result is kept in memory – higher performance
- Shark → Spark SQL, and SQL is re-found effective to represent business logics for data analytics!
- Spark is further extended to Support existing interfaces now

- **Recently I believe there should be the 4th classic software**
  - Big Data
    - Hadoop/MapReduce, HIVE, Spark, HAWQ, etc.



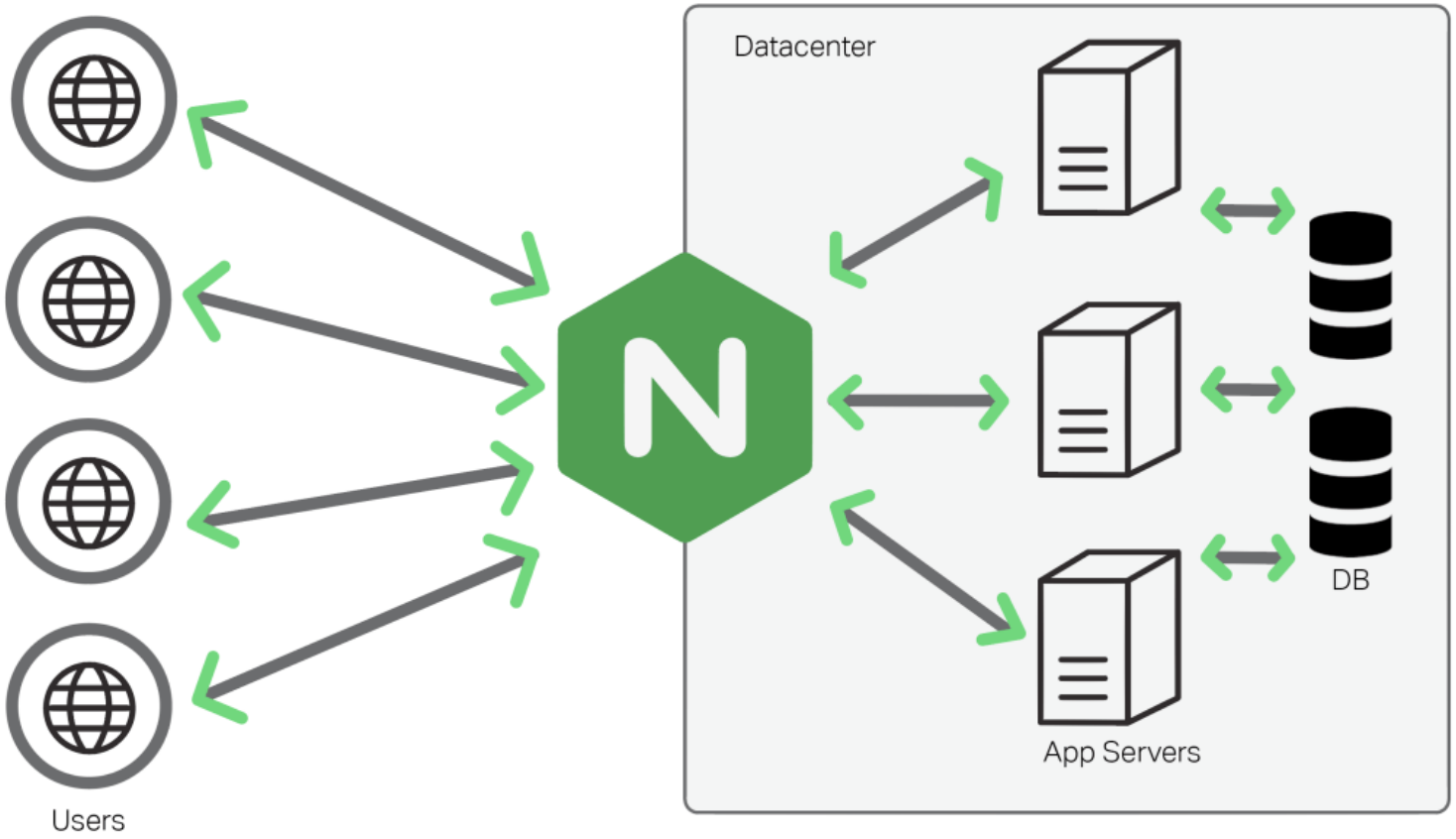
https://stackoverflow.com/questions/39411888/why-would-someone-run-spark-flink-on-tez

Hadoop Framework Tools



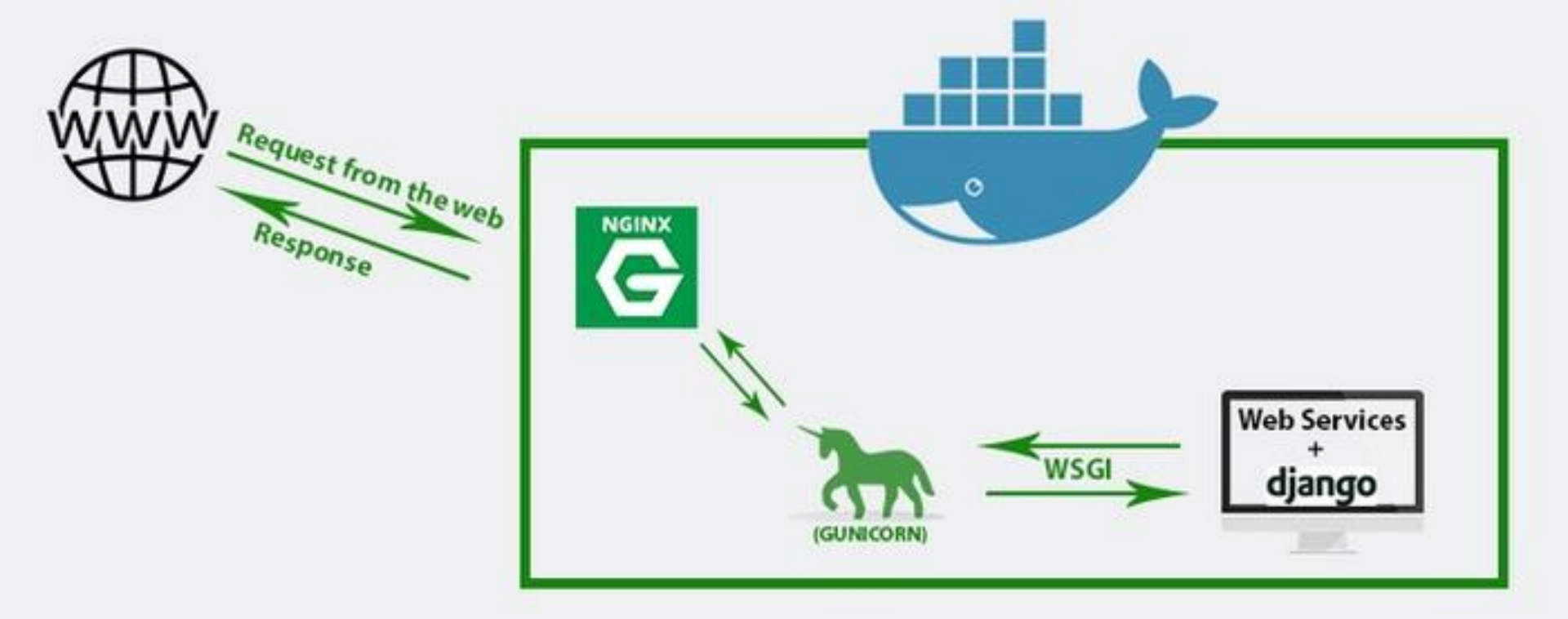
Application architecture becomes more complex
In Big Data age

- + Nginx



Application architecture becomes more complex
In Big Data age

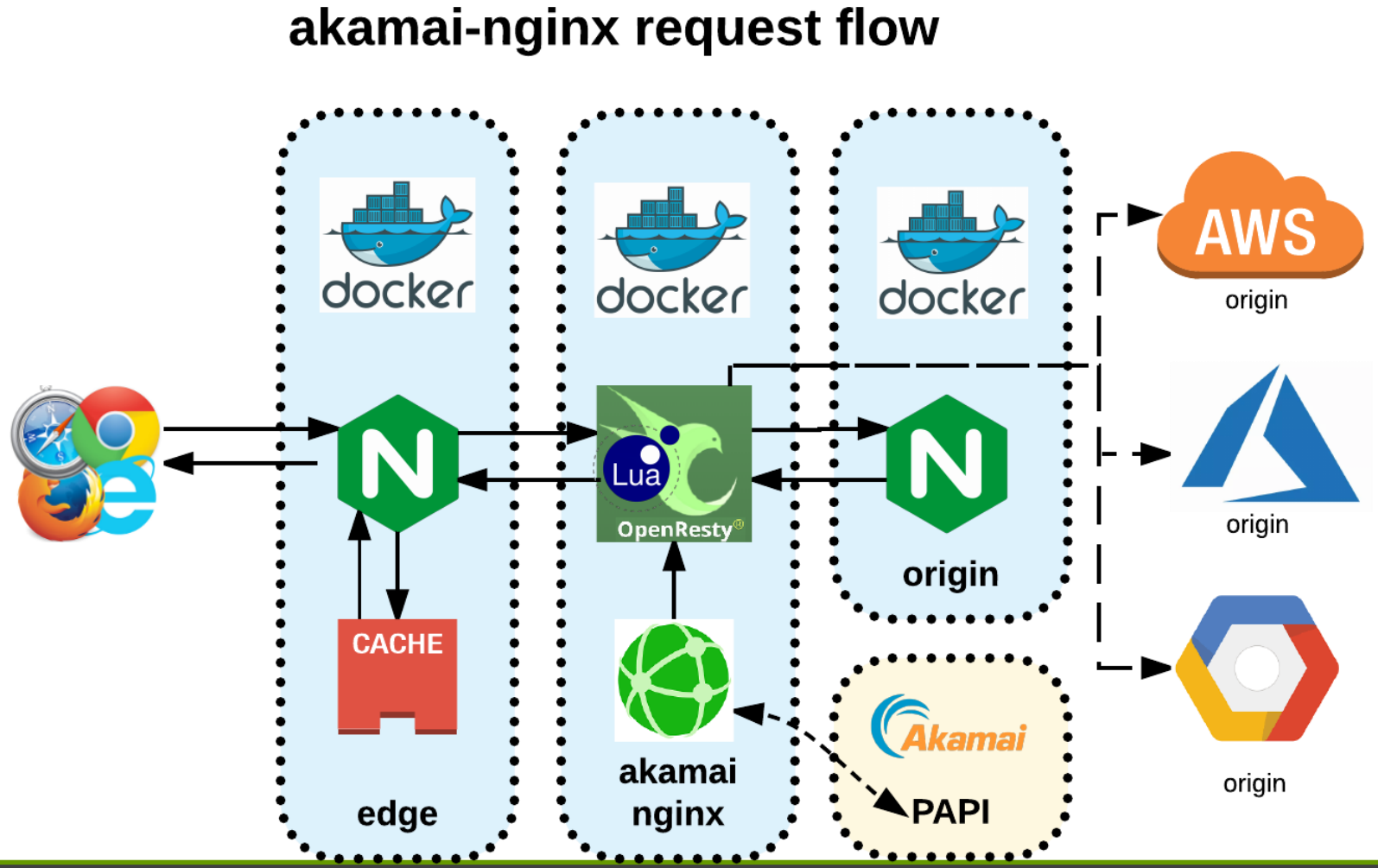
- **Linux + Docker, Nginx, Django etc.**



https://www.codementor.io/samueljames/nginx-setting-up-a-simple-proxy-server-using-docker-and-python-django-f7hy4e6jv

Application architecture becomes more complex
In Big Data age

- **Linux** + Docker, Nginx, Lua etc.



Akamai PAPI is called once, to generate equivalent LUA config

https://github.com/wyvern8/akamai-nginx

Many Cloud platforms

Chapter 5: Go deeper and wider

- **Overview about the evolution of Data Management & Analytics**
- **SQL again**
  - SQL → NoSQL(HIVEQL/Spark SQL) → NewSQL
- **Final (if you want to share)**

You've known something about SQL, right?

- **I hope you really are ☺**
  - SQL (like other computer languages) is formally defined in a notation called "Backus-Naur Form"
- **Subset of SQL may be**

<query specification> ::=
SELECT [<set quantifier>] <select list> <table expression>

<select list> ::=
<asterisk>
| <select sublist> [{ <comma> <select sublist> }...]

<select sublist> ::= <derived column> | <qualifier> <period> <asterisk>

<derived column> ::= <value expression> [<as clause>]

<as clause> ::= [AS] <column name>

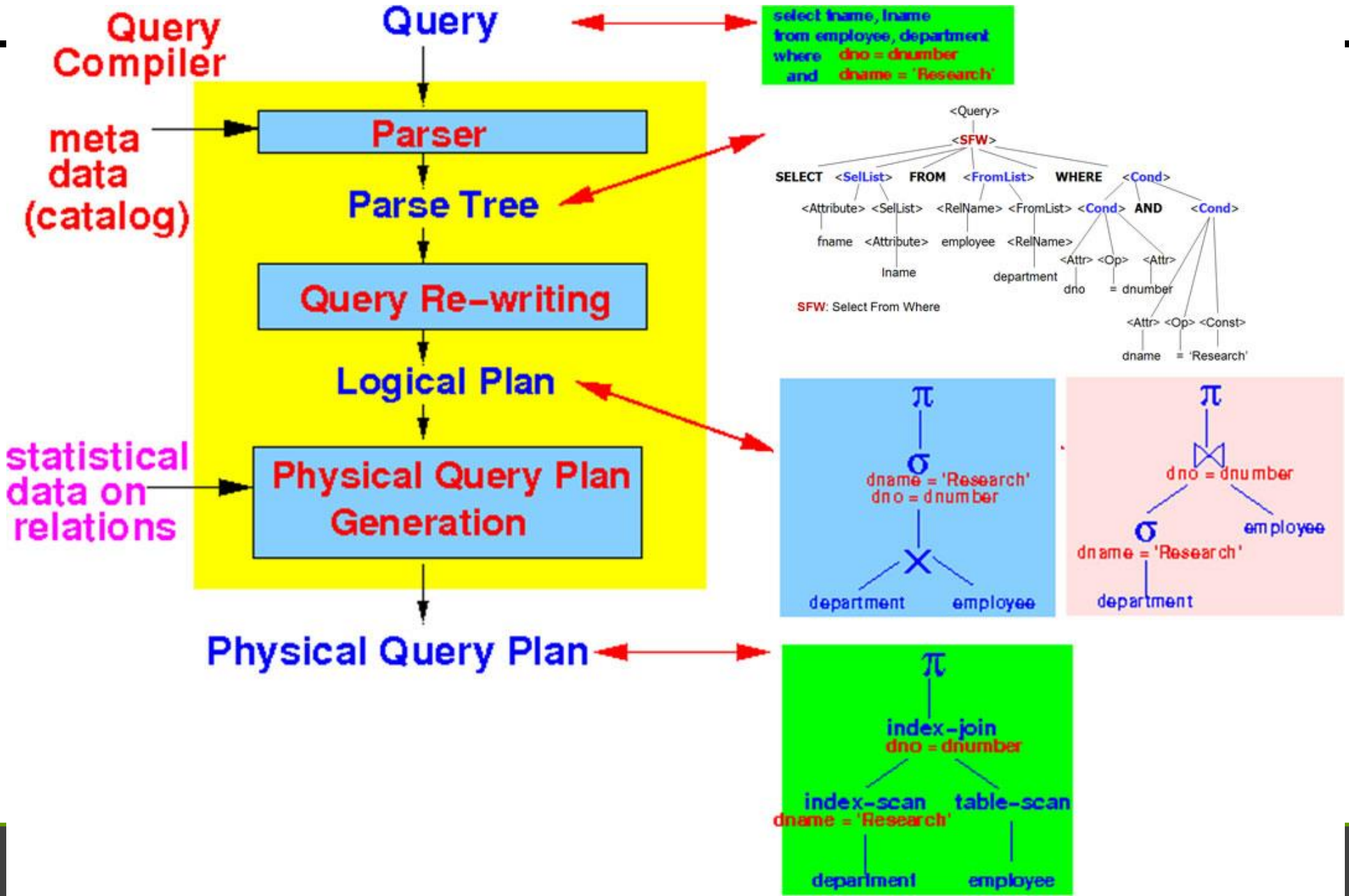
<table expression> ::=
<from clause>
[<where clause>]
[<group by clause>]
[<having clause>]

<from clause> ::= FROM <table reference> [{ <comma> <table reference> }...]

How is SQL translated and executed?

- **Math Expression → AST**
  - Helpful to understand SQL processing
- **SQL Processing**
  - SQL → Parse Tree [Math → AST]
  - Parse Tree → Canonical RA
    - Canonical RA is converted into equivalent RAs (for Algebraic optimization)
  - RA → PQP (Physical Query Plan) [Math → AST]
    - RA could be mapped to many PQPs
    - With help of catalog information, physical optimization
  - The optimal PQP is executed
    - Materialize or Pipline

In short


Query
Query Compiler
meta data (catalog)
Parser
Parse Tree
Query Re-writing
Logical Plan
statistical data on relations
Physical Query Plan Generation
Physical Query Plan
select fname, lname
from employee, department
where dno = dnumber
and dname = 'Research'
<Query>
<SFW>
SELECT <SelList> FROM <FromList> WHERE <Cond>
<Attribute> <SelList> <RelName> <FromList> <Cond> AND <Cond>
fname <Attribute> employee <RelName>
lname department
<Attr> <Op> <Attr>
dno = dnumber
SFW: Select From Where
<Attr> <Op> <Const>
dname = 'Research'
π
σ
dname = 'Research'
dno = dnumber
department
employee
π
dno = dnumber
σ
employee
dname = 'Research'
department
π
index-join
dno = dnumber
index-scan
dname = 'Research'
table-scan
department
employee

NoSQL - Not Only SQL

- **Stands for Not Only SQL**
  - Class of non-relational data storage systems
  - Usually do not require a fixed table schema nor do they use the concept of joins
- **All NoSQL offerings relax one or more of the ACID properties (will talk about the CAP theorem)**
  - Key features (advantages):
    - non-relational, data are replicated to multiple, nodes (so, identical & fault-tolerant) and can be partitioned, cheap, easy to implement (open-source), massive write performance, fast key-value access

https://www.mytechlogy.com/IT-blogs/8362/from-sql-to-nosql-the-continued-evolution-of-databases/#.WTa6NtJ9600

How is data stored with NoSQL?

There are a variety of types:

- Column Store – Each storage block contains data from only one column
- Document Store – stores documents made up of tagged elements
- Key-Value Store – Hash table of keys
  - **Dictionary**

Other Non-SQL Databases

- XML Databases, Graph Databases, Object Oriented Databases, …

CouchDB JSON Example – dictionary

```
{
  "_id": "guid goes here",
  "_rev": "314159",

  "type": "abstract",

  "author": "Keith W. Hare"

  "title": "SQL Standard and NoSQL Databases",

  "body": "NoSQL databases (either no-SQL or Not Only SQL)
           are currently a hot topic in some parts of
           computing.",
  "creation_timestamp": "2011/05/10 13:30:00 +0004"
}
```

MongoDB to documents (JSON):

```
{
        _id: ObjectId("51156a1e056d6f966f268f81"),
        type: "Article",
        author: "Derick Rethans",
        title: "Introduction to Document Databases with MongoDB",
        date: ISODate("2013-04-24T16:26:31.911Z"),
        body: "This arti…"
},
{
        _id: ObjectId("51156a1e056d6f966f268f82"),
        type: "Book",
        author: "Derick Rethans",
        title: "php|architect's Guide to Date and Time Programming with PHP",
        isbn: "978-0-9738621-5-7"
}
```

Typical NoSQL API

Basic API access:

- get(key) -- Extract the value given a key
- put(key, value) -- Create or update the value given its key
- delete(key) -- Remove the key and its associated value
- execute(key, operation, parameters) -- Invoke an operation to the value (given its key) which is a special data structure (e.g. List, Set, Map .... etc).

Map Reduce is a typical NoSQL “query”

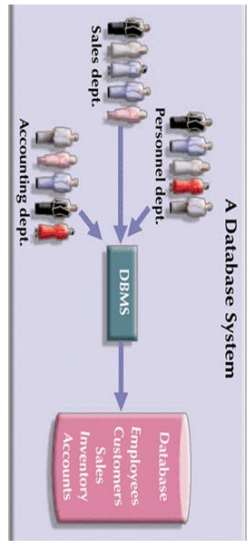
- **Technique for indexing and searching large data volumes**
- **Two Phases, Map and Reduce**
  - Map
    - Extract sets of Key-Value pairs from underlying data
    - Potentially in Parallel on multiple machines
  - Reduce
    - Merge and sort sets of Key-Value pairs
    - Results may be useful for other searches

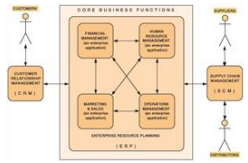

Redis
HBASE
Cassandra
mongoDB
riak
CouchDB
relax
Tokyo Cabinet
Scalaris
Project Voldemort
Neo4j
the graph database
membase

Who is using them?

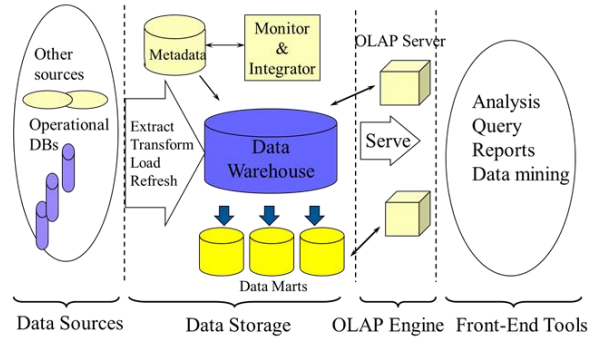
File based DM

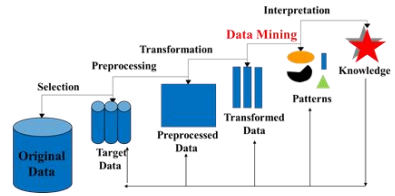
(R)DBMS



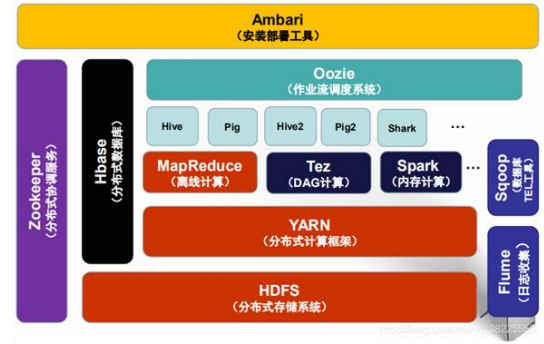


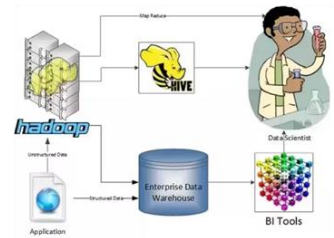
Data Warehouse/DM

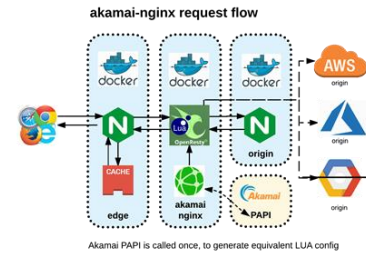




Big Data







Akamai PAPI is called once, to generate equivalent LUA config

NoSQL – NewSQL

Redis - 95
Memcached - 33
DynamoDB - 16
Rick - 13

MongoDB - 279
CouchDB - 28
Couchbase - 24
DynamoDB – 15
MarkLogic - 11

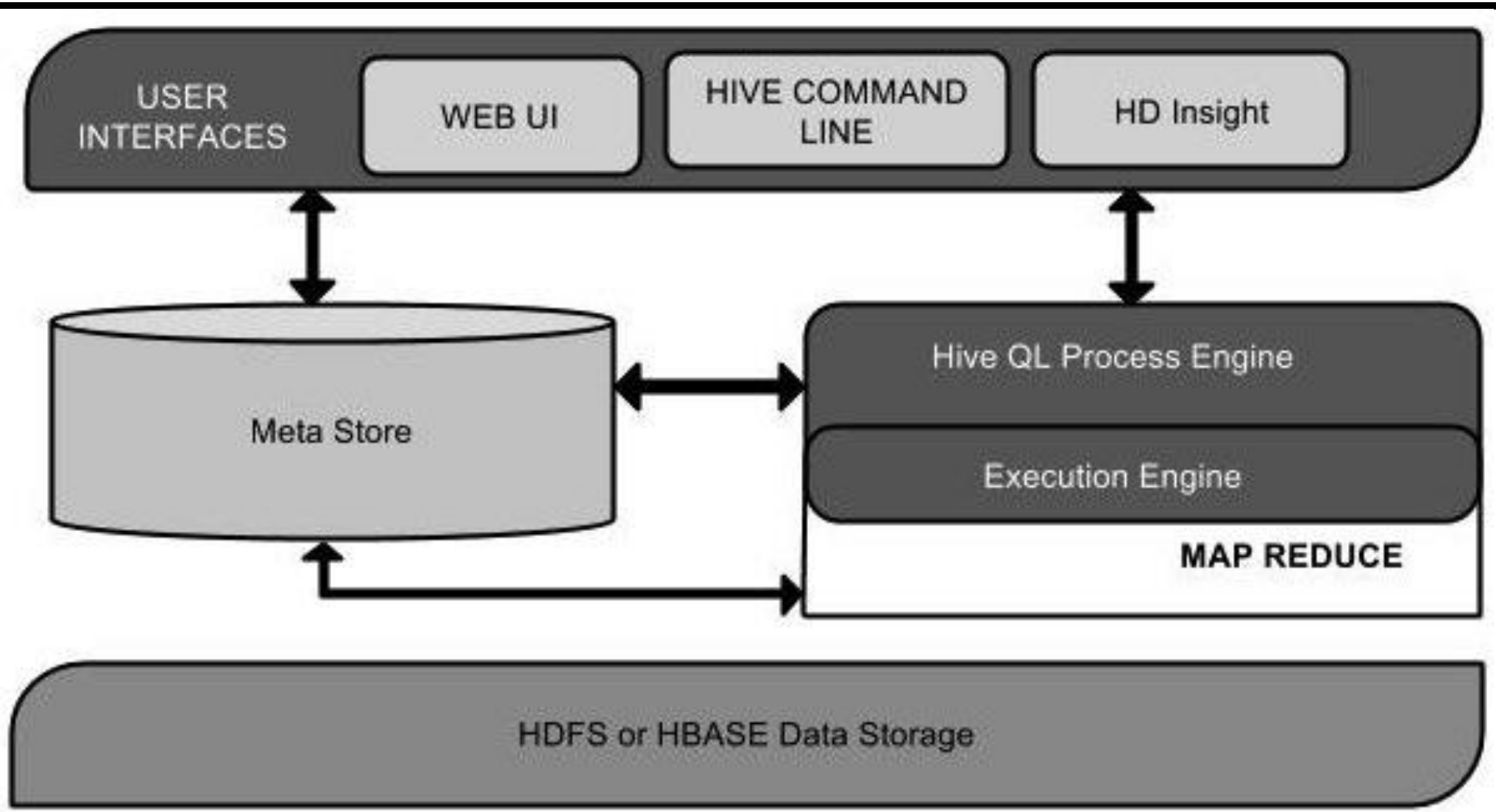
Cassandra - 109
HBase - 62

Neo4j - 30
OrientDB - 4
Titan – 3
Giraph - 1

Solr - 81
Elasticsearch - 70
Splunk – 41

Hive and HiveQL
- SQL is re-found useful in Big Data

- **Hive is a DW tool on Big Data**
  - Developed by Facebook and a top-level Apache project
  - A data warehousing infrastructure based on Hadoop
    - like Hadoop, is designed for batch processing of large datasets
    - Immediately makes data on a cluster available to non-Java programmers via SQL like queries
  - Built on **HiveQL** (HQL), a SQL-like query language
    - Interprets HiveQL and generates MapReduce jobs that run on the cluster
    - Enables easy data summarization, ad-hoc reporting and querying, and analysis of large volumes of data


USER INTERFACES
WEB UI
HIVE COMMAND LINE
HD Insight
Meta Store
Hive QL Process Engine
Execution Engine
MAP REDUCE
HDFS or HBASE Data Storage

HiveQL

- **HQL provides the basic SQL-like operations:**
  - Select columns using SELECT
  - Filter rows using WHERE
  - JOIN between tables
  - Evaluate aggregates using GROUP BY
  - Store query results into another table
  - Download results to a local directory (i.e., export from HDFS)
  - Manage tables and queries with CREATE, DROP, and ALTER

HiveQL

Create a Complex Table

```
CREATE TABLE employees  (
    (name STRING,
    salary FLOAT,
    subordinates ARRAY<STRING>,
    deductions MAP<STRING, FLOAT>,
    address STRUCT<street:STRING,
                   city:STRING,
                   state:STRING,
                   zip:INT>)
  ROW FORMAT DELIMITED
  FIELDS TERMINATED BY '\t'
  STORED AS TEXTFILE;
```

Sample Select Clauses

- Select from a single table

```
SELECT *
FROM sales
WHERE amount > 10 AND
                                region = "US";
```

- Select from a partitioned table

```
SELECT page_views.*
FROM page_views
WHERE  page_views.date >= '2008-03-01' AND
       page_views.date <= '2008-03-31'
```

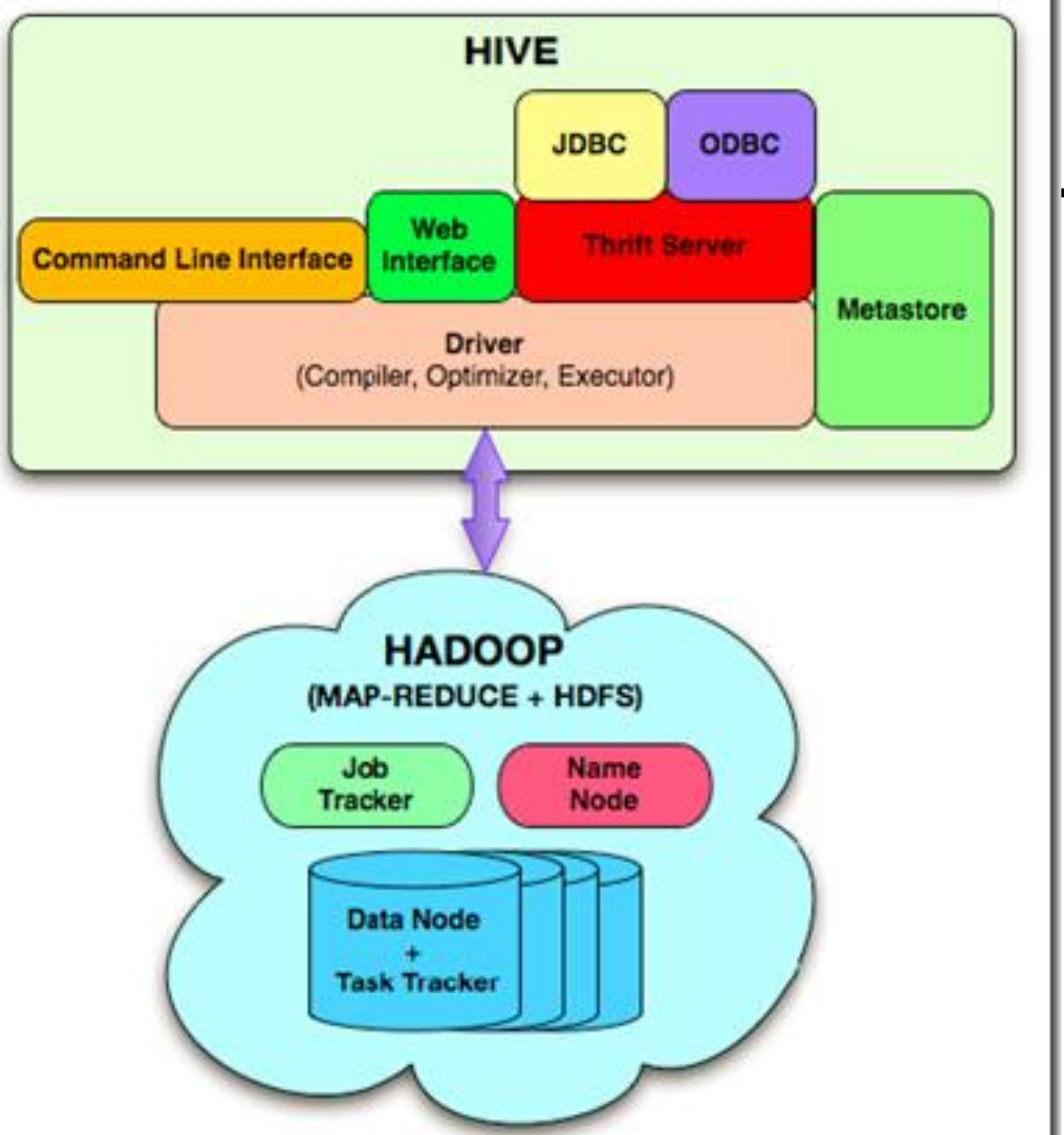
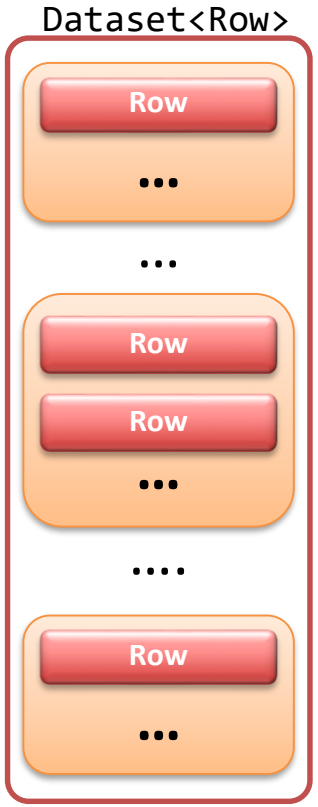


Fig. 1: Hive System Architecture

https://jasperpeilee.wordpress.com/tag/hive/

SQL is re-found useful in Big Data – in Spark

- Return top 20 frequent words




```
case class Word(text: String, n: Long)
val wordsFreq = freq.map {
    case (text, count) => Word(text, count)
}.toDF() // Dataset<Word>
wordsFreq.createOrReplaceTempView("wordsFreq")


//Dataset<Row>
val topWords = sparkSession.sql("select text, n
                                 from wordsFreq
                                 order by n desc
                                 limit 20")
topWords.collect().foreach(println)
```

Everyone could write SQL, right? ☺

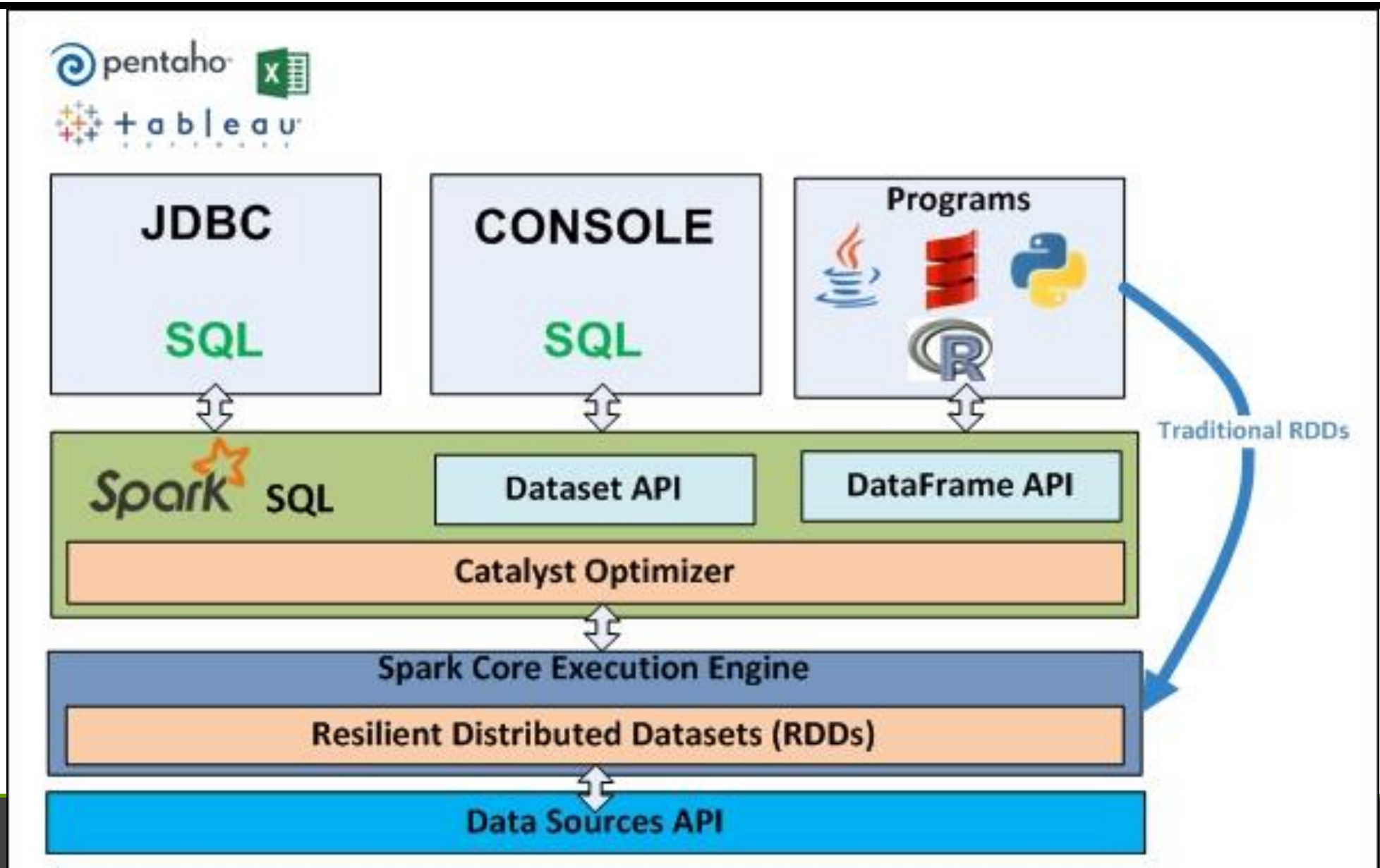
```
from pyspark.sql import Row

sc = spark.sparkContext

# Load a text file and convert each line to a Row.

lines = sc.textFile("examples/src/main/resources/people.txt")

parts = lines.map(lambda l: l.split(","))

people = parts.map(lambda p: Row(name=p[0], age=int(p[1])))

# Infer the schema, and register the DataFrame as a table.

schemaPeople = spark.createDataFrame(people)

# SQL can be run over DataFrames that have been registered as a table.

teenagers = spark.sql("SELECT name FROM people WHERE age >= 13 AND

# The results of SQL queries a

# rdd returns the content as an :class

teenNames = teenagers.rdd.map(lambda

for name in teenNames:

    print(name)
```

Everyone could write SQL, right? ☺

How to process SQL is a basic question to dive into the design and implementation of Big Data

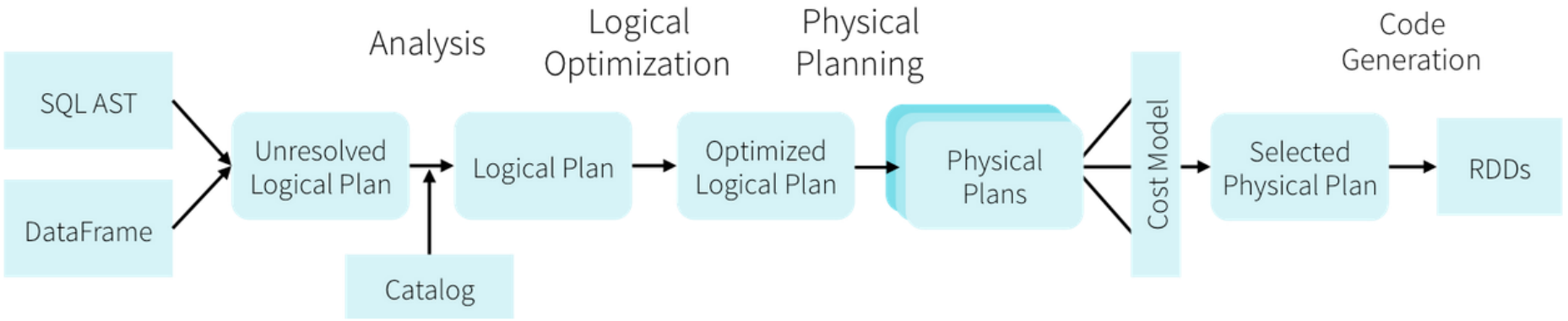
Architecture of Spark SQL

https://www.packtpub.com/mapt/book/big-data-and-business-intelligence/9781785884696/4/ch04lvl1sec26/Architecture+of+Spark+SQL

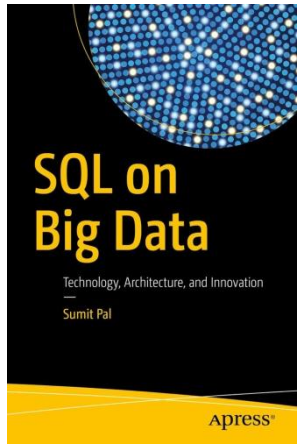


Challenges of SQL in Big Data

- Share similar flowchart



- But, Data are scattered in different nodes
  - Reducing the amount of data to be shuffled is a major challenge
  - How to optimize?
  - How to carry out the execution efficiently?
  - How to process SQL queries on streaming data?
  - …

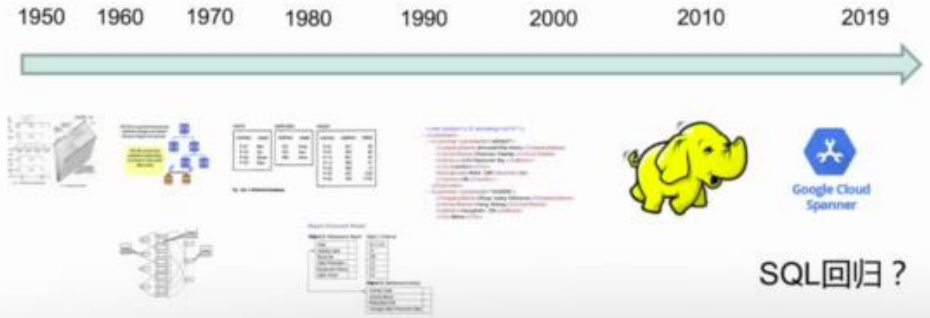


HAWQ


专访偶数科技创始人兼CEO常
雷：一个创业者的野蛮生长
2017-12-29


偶数科技获红杉中国、红点中国
投资
2017-10-31

SQL （分布式数据库）回归？



MR被其创造者Google 在2011年就放弃了

Cloud Spanner 的文章！

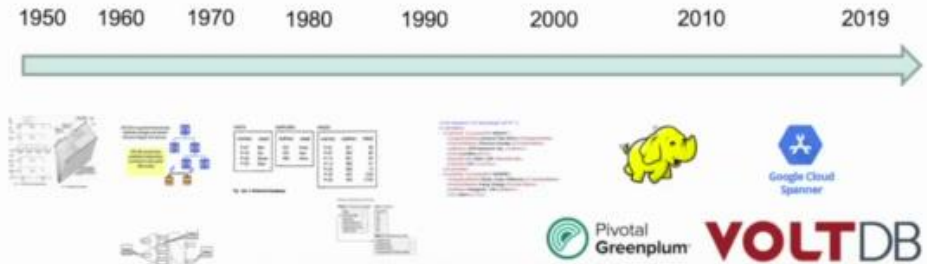
2008：MapReduce 是技术大倒退

作为一种数据处理模式，MapReduce 是一种大倒退。

- 模式（Schema）很有价值
- 实现糟糕，不支持 Access Method
- 高级访问语言很有价值

Google 内部2011年放弃MR，2015年公开放弃

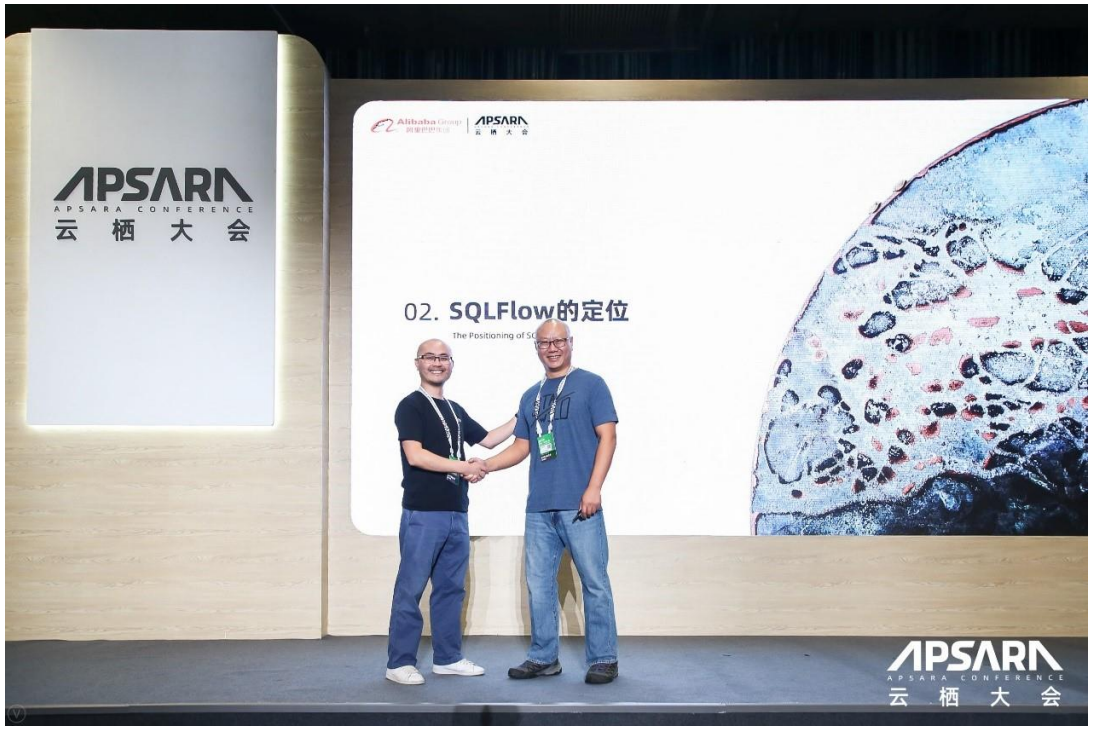
SQL （分布式数据库）从未离开



2019

王益 在蚂蚁金服 推出 SQLFlow + ElasticDL

https://zhuanlan.zhihu.com/p/87682668
https://developer.aliyun.com/article/719553 [原文]

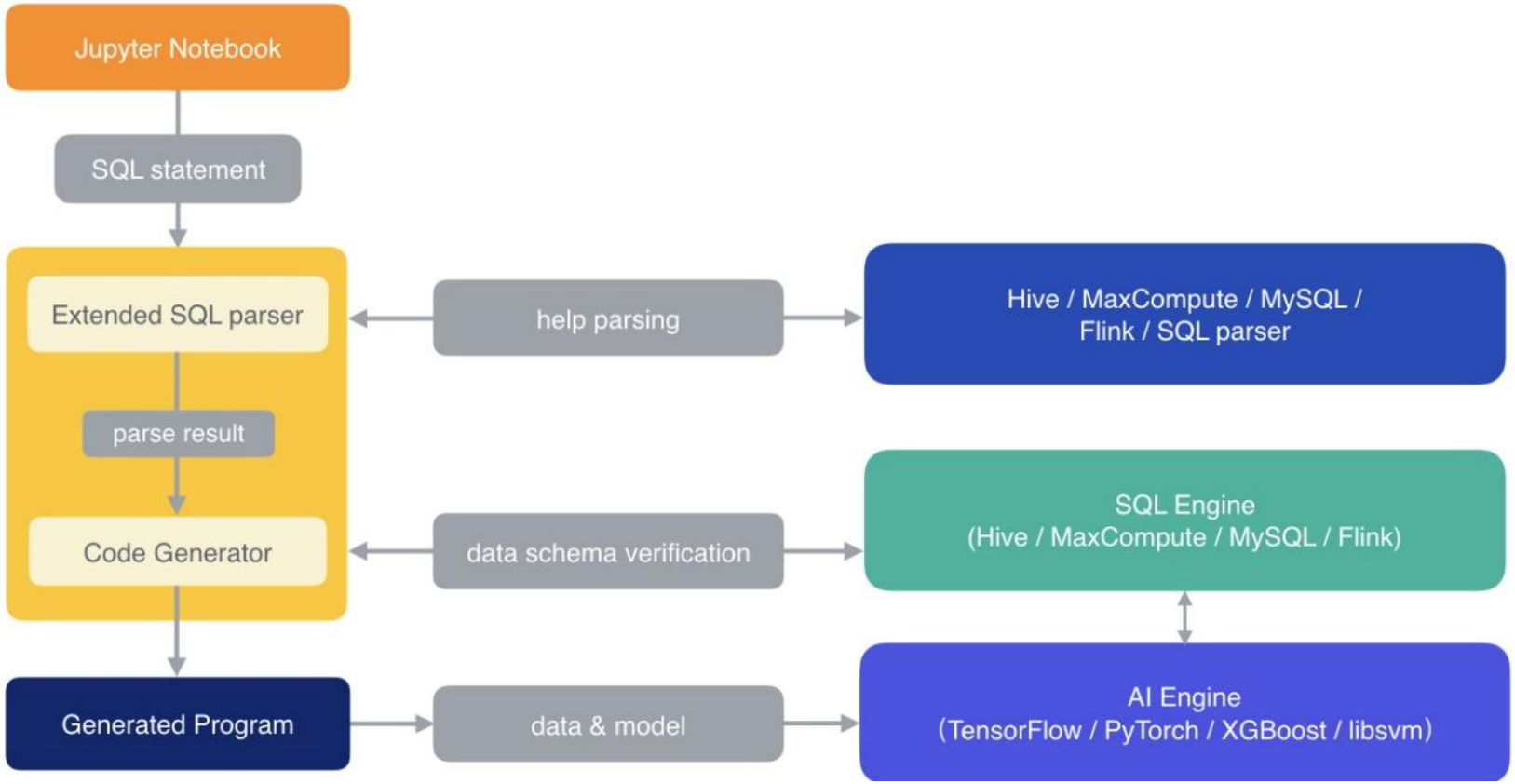


项目官网：https://sqlflow.org
GitHub地址：https://github.com/sql-machine-learning/sqlflow
您也可以使用docker，运行文章中的汽车价格预测模型：docker run -p 8888:8888sqlflow/sqlflow:didi

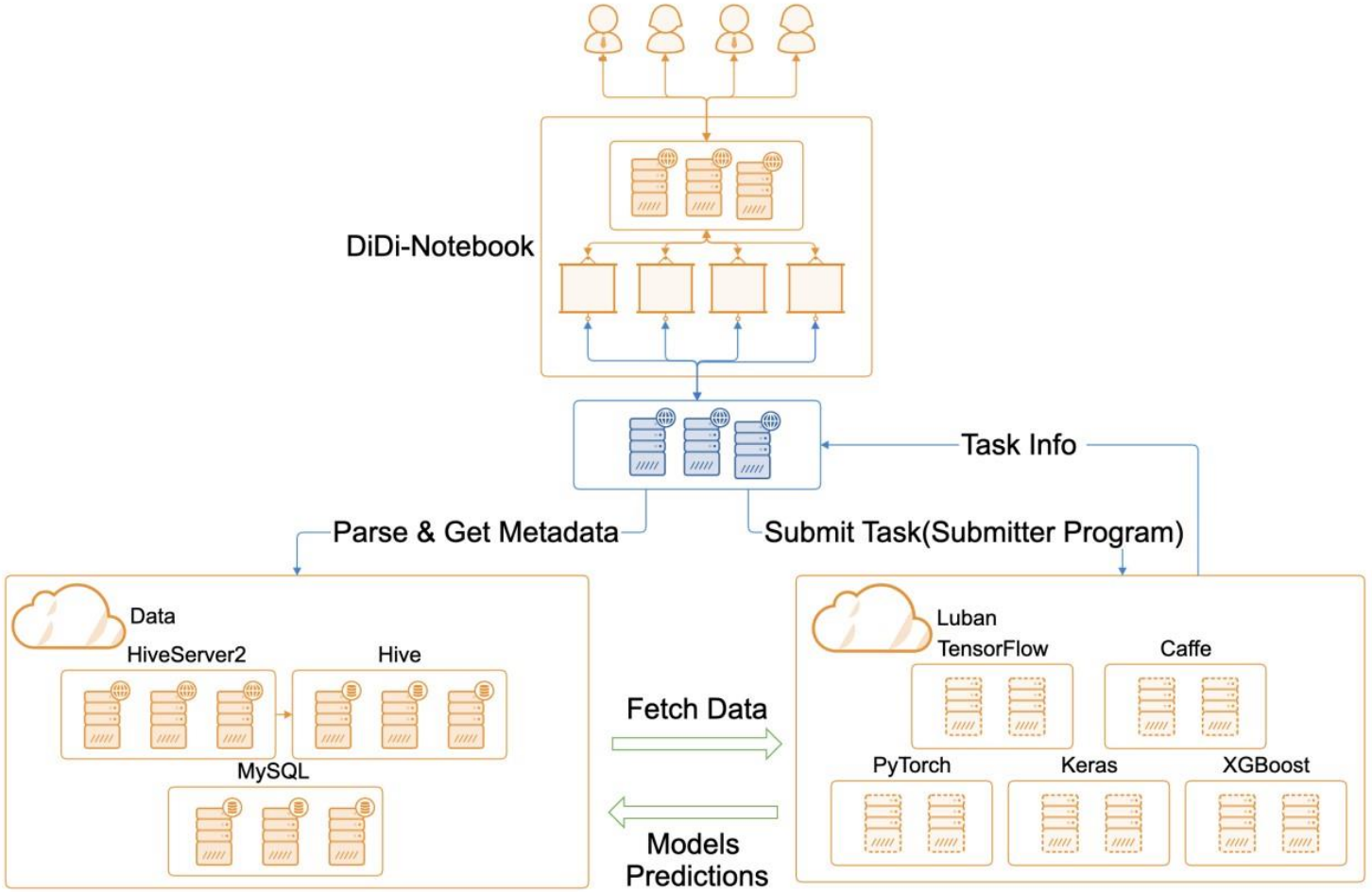
SQLFlow

https://developer.aliyun.com/article/719553 [原文]

SQLFlow的技术架构



SQLFlow 和滴滴数据的整合逻辑



SQLova: English to SQL translation

Table:

Player	Country	Points	Winnings ($)
Steve Stricker	United States	9000	1260000
K.J. Choi	South Korea	**5400**	756000
Rory Sabbatini	South Africa	3400	4760000
Mark Calcavecchia	United States	2067	289333
Ernie Els	South Africa	2067	289333

Question: What is the points of South Korea player?
SQL: `SELECT` Points `WHERE` Country = South Korea
Answer: 5400

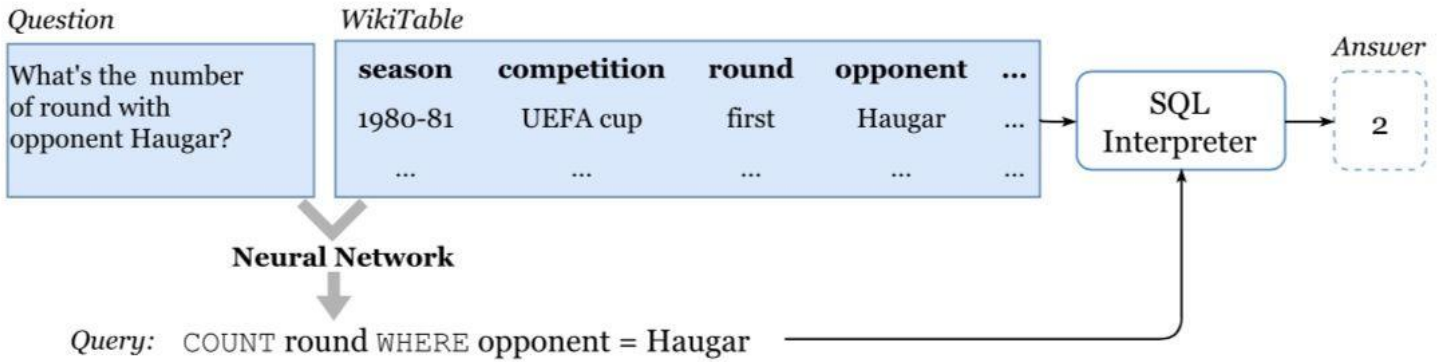


Figure 1: Answering a table question by synthesizing a query and executing it on the provided table.

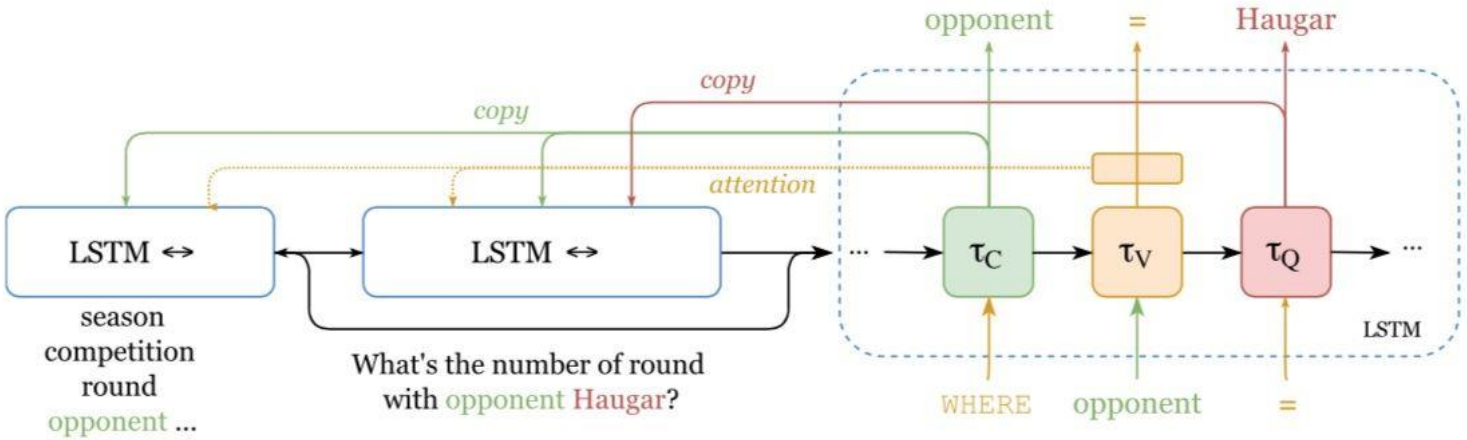


Figure 2: Model overview for the example in Figure 1. The model encodes table columns as well as the user question with a bidirectional LSTM and then decodes the hidden state with a typed LSTM, where the decoding action for each cell is statically determined.

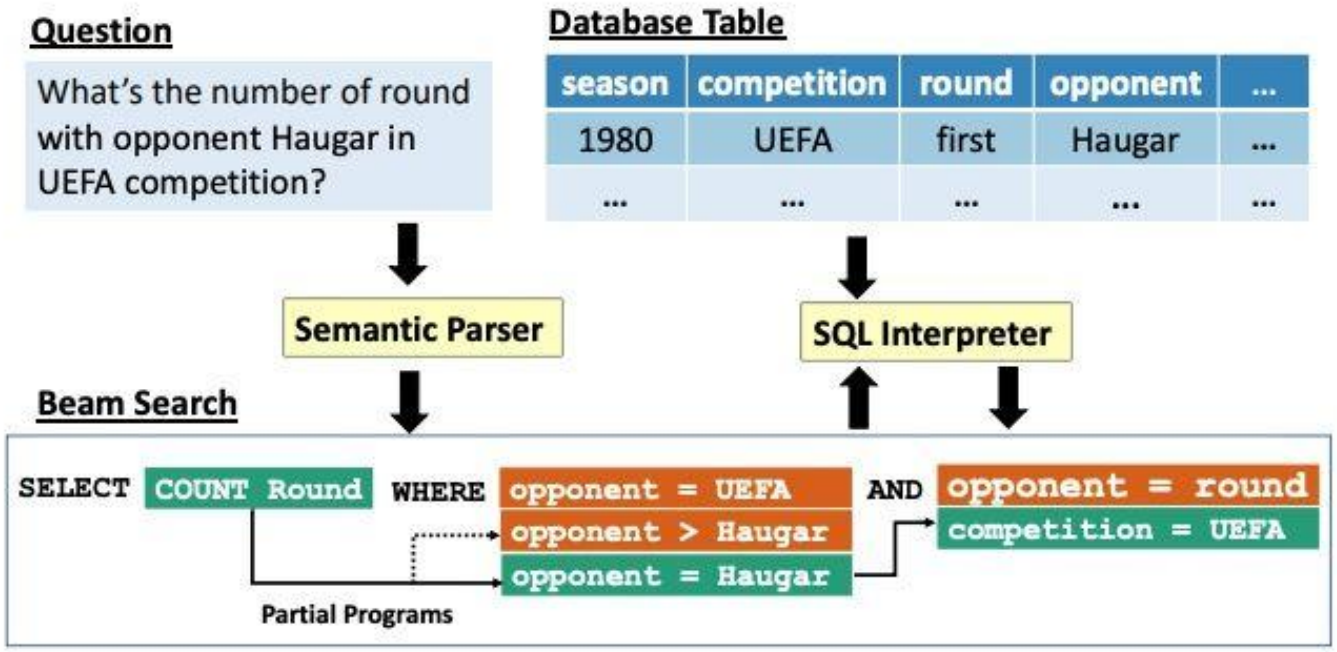


Figure 1: An execution-guided decoder evaluates partially generated queries at appropriate timesteps and then excludes those candidates that cannot be completed to a correct SQL query (red background). Here, "`opponent > Haugar`" would yield a runtime error, whereas "`opponent = UEFA`" would yield an empty result.

$$[\mathrm{CLS}],\ T_{n,1},\ \cdots\ T_{n,L},\ [\mathrm{SEP}],\ T_{h_1,1},\ T_{h_1,2},\ \cdots,\ [\mathrm{SEP}],\ \cdots,\ [\mathrm{SEP}],\ T_{h_{N_h},1},\ \cdots,\ T_{h_{N_h},M_{N_h}},\ [\mathrm{SEP}]$$

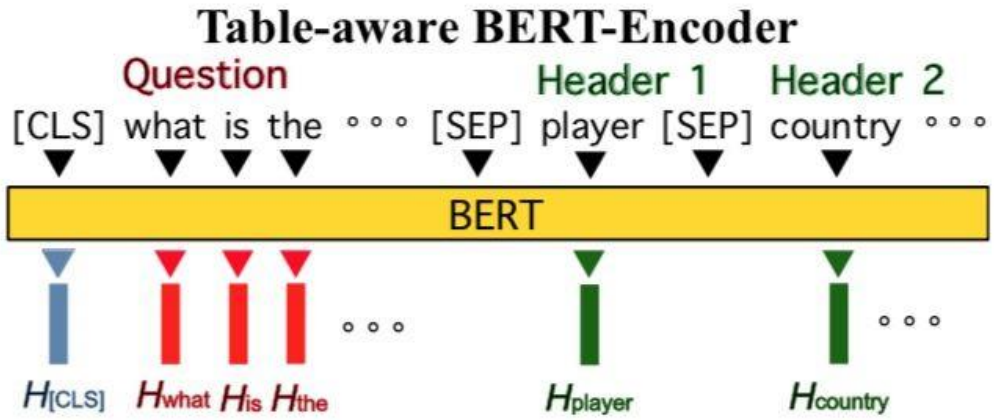
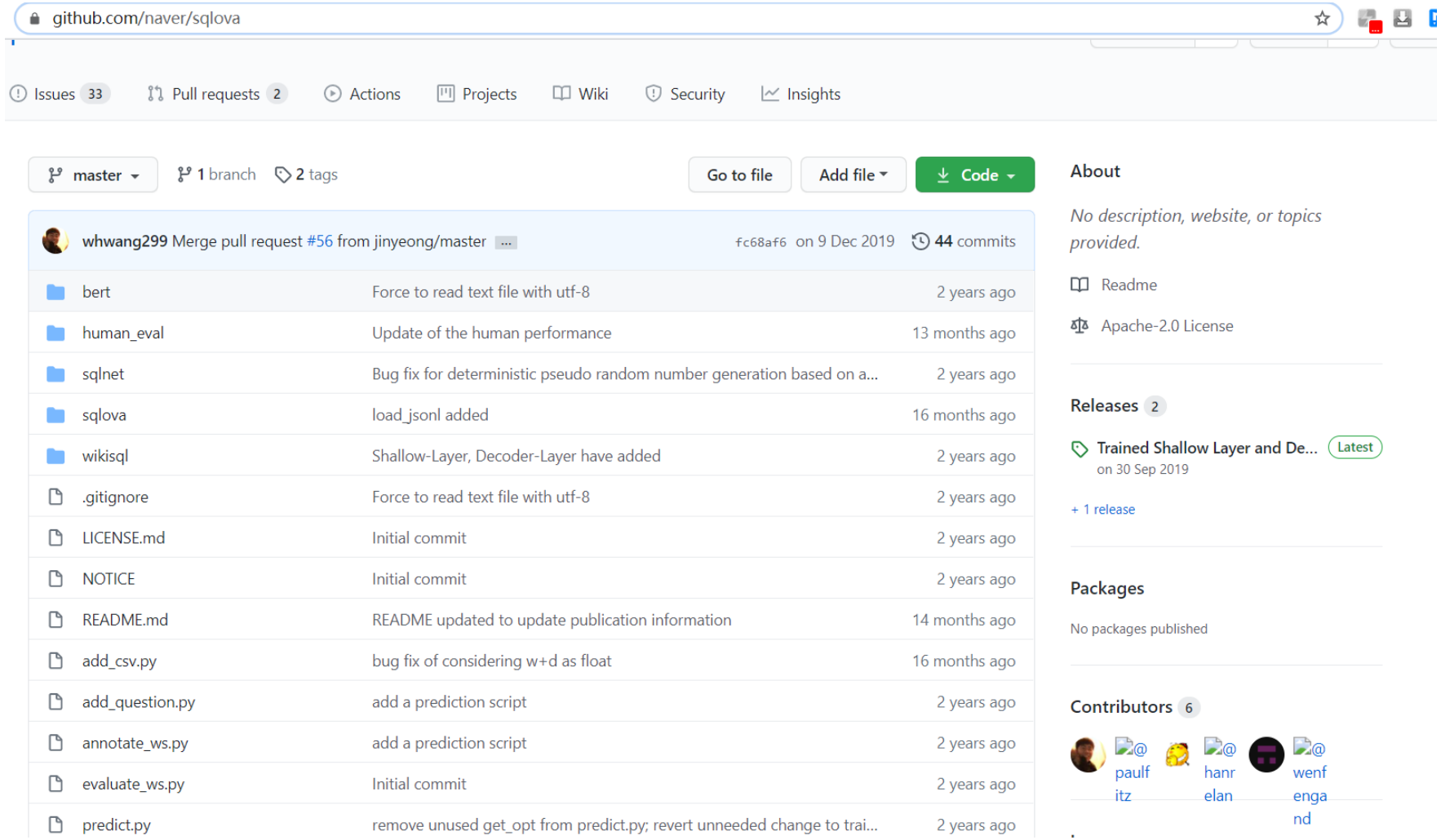


Figure 1: (A) The scheme of input encoding process by table-aware BERT. Final output vectors are represented by colored bars: light blue for `[CLS]` output, red for question words, and green for tokens from table headers.

https://github.com/naver/sqlova



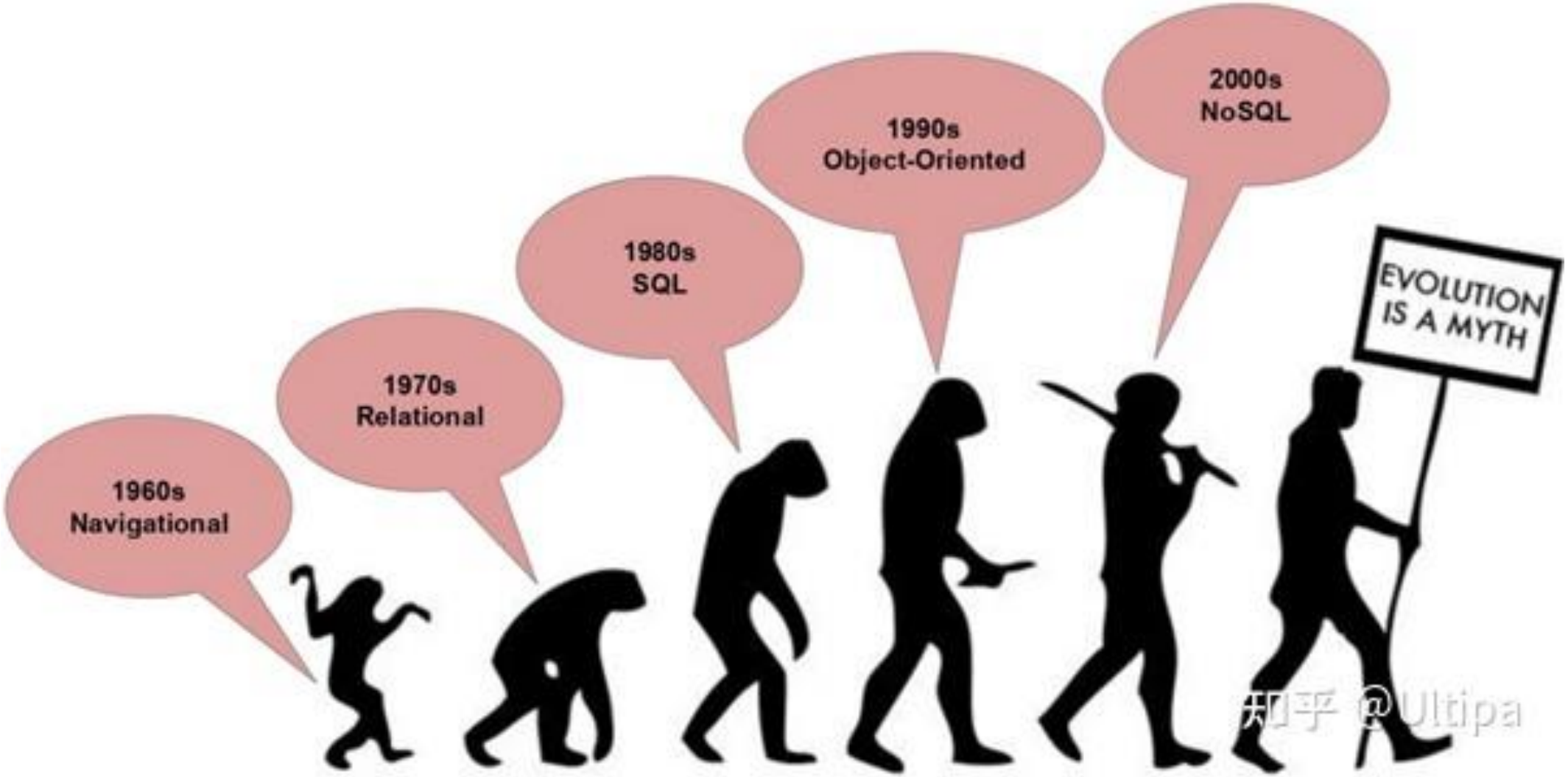
	Dev			Test		
	Acc_{lf}	Acc_{qm}	Acc_{ex}	Acc_{lf}	Acc_{qm}	Acc_{ex}
Content Insensitive						
Dong and Lapata (2016)	23.3%	-	37.0%	23.4%	-	35.9%
Augmented Pointer Network (Zhong et al., 2017)	44.1%	-	53.8%	42.8%	-	52.8%
Seq2SQL (Zhong et al., 2017)	49.5%	-	60.8%	48.3%	-	59.4%
SQLNet (Xu et al., 2017)	-	63.2%	69.8%	-	61.3%	68.0%
TypeSQL w/o type-awareness (ours)	-	66.5%	72.8%	-	64.9%	71.7%
TypeSQL (ours)	-	**68.0%**	**74.5%**	-	**66.7%**	**73.5%**
Content Sensitive						
Wang et al. (2017a)	59.6%	-	65.2%	59.5%	-	65.1%
TypeSQL+TC (ours)	-	**79.2%**	**85.5%**	-	**75.4%**	**82.6%**

Table 1: Overall results on WikiSQL. Acc_{lf}, Acc_{qm}, and Acc_{ex} denote the accuracies of exact string, canonical representation, and execute result matches between the synthesized SQL with the ground truth respectively. The top six results are content-insensitive, which means only the question and table schema are used as inputs. The bottom two are content-sensitive, where the models use the question, the table schema, and the content of databases.

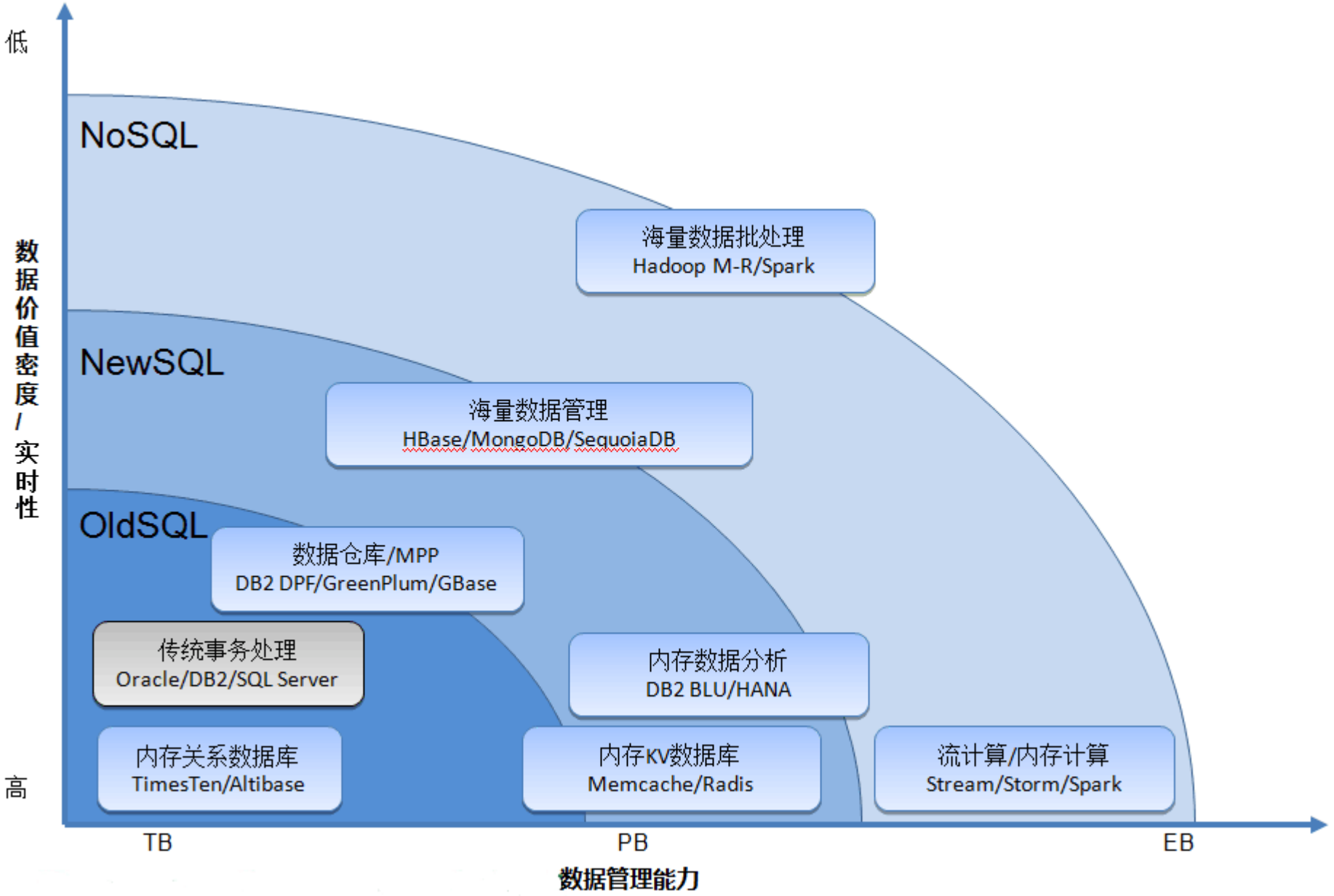
	Dev			Test		
	Acc_{agg}	Acc_{sel}	Acc_{where}	Acc_{agg}	Acc_{sel}	Acc_{where}
Seq2SQL (Zhong et al., 2017)	90.0%	89.6%	62.1%	90.1%	88.9%	60.2%
SQLNet (Xu et al., 2017)	90.1%	91.5%	74.1%	90.3%	90.9%	71.9%
TypeSQL (ours)	90.3%	**93.1%**	**78.5%**	90.5%	**92.2%**	**77.8%**
TypeSQL+TC (ours)	90.3%	**93.5%**	**92.8%**	90.5%	**92.1%**	**87.9%**

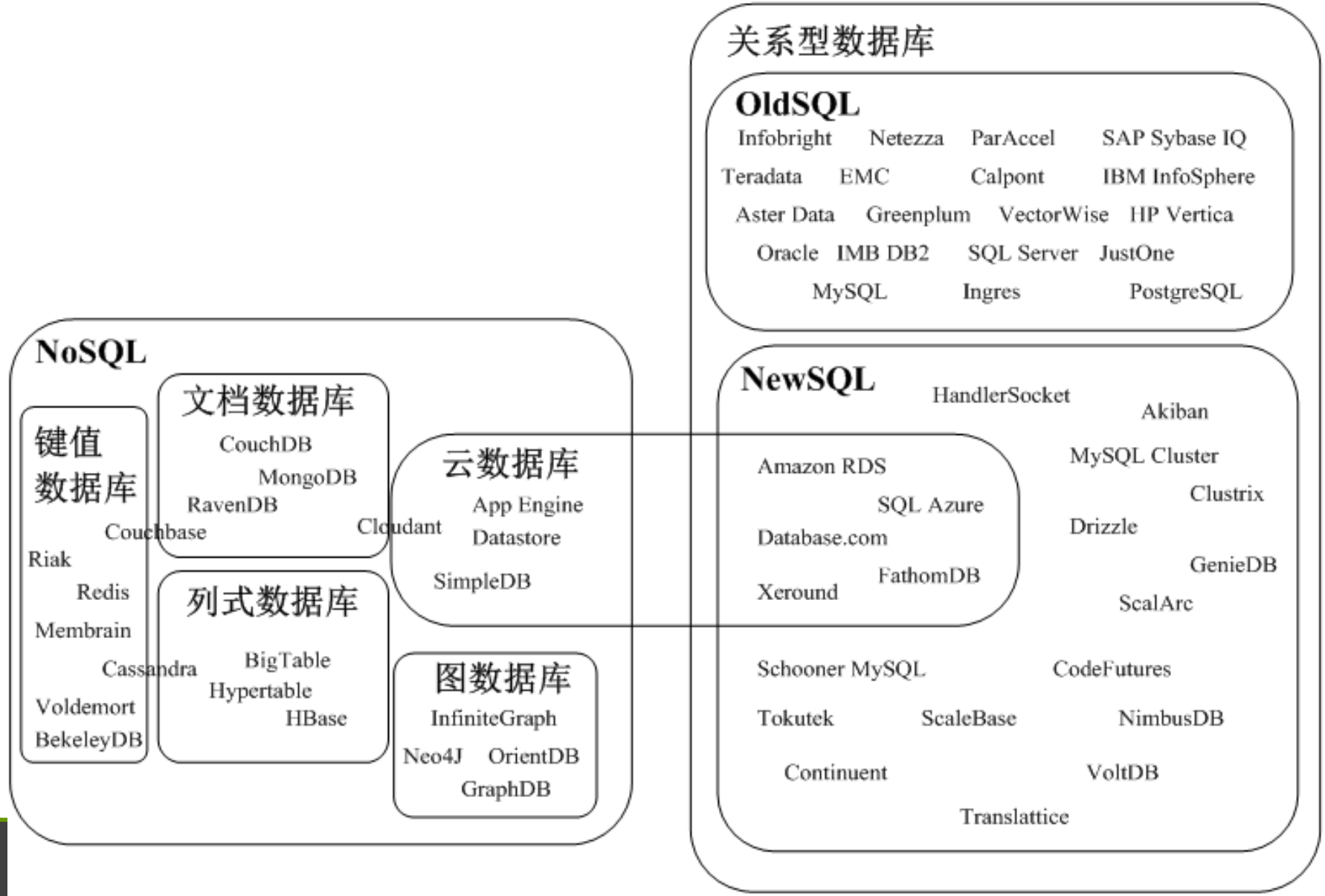
Table 2: Breakdown results on WikiSQL. Acc_{agg}, Acc_{sel}, and Acc_{where} are the accuracies of canonical representation matches on AGGREGATOR, SELECT COLUMN, and WHERE clauses between the synthesized SQL and the ground truth respectively.

NewSQL?


1960s
Navigational
1970s
Relational
1980s
SQL
1990s
Object-Oriented
2000s
NoSQL
EVOLUTION
IS A MYTH
知乎 @Ultipa

- **SQL(Structured Query Language)：数据库，指关系型数据库。**
  - 好处来源于它的统一性和易用性，缺点是面对大量的数据时，他的性能会随着数据库的增大而急剧下降。
  - 主要代表：SQL Server、Oracle、MySQL、PostgreSQL。
- **NoSQL(Not Only SQL)：泛指非关系型数据库。**
  - 以放宽ACID原则为代价，NoSQL采取的是最终一致性原则，而不是像关系型数据库那样地严格遵守着ACID的原则，这意味着如果在特定时间段内没有特定数据项的更新，则最终对其所有的访问都将返回最后更新的值。 这就是这样的系统通常被描述为提供基本保证的原因（基本可用，软状态，最终一致性） — 而不是ACID。
  - 主要代表：MongoDB、Redis、CouchDB。
- **NewSQL：对各种新的可扩展/高性能数据库的简称。**
  - NewSQL选择汲取了SQL和NewSQL的优点，希望将ACID和可扩展性以及高性能结合，但是目前而言，不适用于所有的场景。
  - 主要代表：Clustrix、GenieDB。


低
数据价值密度/实时性
高
NoSQL
NewSQL
OldSQL
海量数据批处理
Hadoop M-R/Spark
海量数据管理
HBase/MongoDB/SequoiaDB
数据仓库/MPP
DB2 DPF/GreenPlum/GBase
传统事务处理
Oracle/DB2/SQL Server
内存数据分析
DB2 BLU/HANA
内存关系数据库
TimesTen/Altibase
内存KV数据库
Memcache/Radis
流计算/内存计算
Stream/Storm/Spark
TB
PB
EB
数据管理能力


关系型数据库
OldSQL
Infobright
Netezza
ParAccel
SAP Sybase IQ
Teradata
EMC
Calpont
IBM InfoSphere
Aster Data
Greenplum
VectorWise
HP Vertica
Oracle
IMB DB2
SQL Server
JustOne
MySQL
Ingres
PostgreSQL
NewSQL
HandlerSocket
Akiban
Amazon RDS
MySQL Cluster
SQL Azure
Clustrix
Database.com
Drizzle
FathomDB
GenieDB
Xeround
ScalArc
Schooner MySQL
CodeFutures
Tokutek
ScaleBase
NimbusDB
Continuent
VoltDB
Translattice
NoSQL
键值数据库
Couchbase
Riak
Redis
Membrain
Cassandra
Voldemort
BekeleyDB
文档数据库
CouchDB
MongoDB
RavenDB
Cloudant
列式数据库
BigTable
Hypertable
HBase
云数据库
App Engine
Datastore
SimpleDB
图数据库
InfiniteGraph
Neo4J
OrientDB
GraphDB

Chapter 5: Go deeper and wider

- **Overview about the evolution of Data Management & Analytics**
- **SQL again**
- **Final (if you want share)**
  - How much do you know now about D&I of DBMS and OS?

Other models like **H**ierarchical, **N**etwork

1961 Charles Bachman at GE – IDS (Integrated Data Store: 集成数据存储) – 1st NDBMS (Network), also 1st DBMS

1968 Vern Watts at IBM – IMS (Information Management System: 信息管理系统) – 1st HDBMS (Hierarchical)

1970 Edgar F. Codd at IBM published paper “A Relational Model of Data for Large Shared Data Banks” – Father of RDBMS

OLTP

1970, E. F. Codd
Relational Algebra

- System R (1974-1978@IBM)
  prototype of RDBMS –
  1st try of SQL
- **Ingres (1973-1985@UCB)**
  INteractive Graphics
  REtrieval System
- Oracle 2.0 (1978 1st SQL based
  RDBMS)
- DB2 (1983)
  Database2 for MVS
- PostgreSQL (**1996**)
  1989 Postgres 1
- SQL Server (1988)
  MS+Sybase+Ashton-Tate
- MySQL 1.0 (1996)
- …

OLAP/DW(DM)

1980s, DB + Analytics

- 1981 NCR
  1st Analytics oriented DB
- 1983 Teradata
  1st DSS
- 1986 IBM
  1st keyword
  Data Warehouse

IBM Informix

2005 Pivotal

Big Data

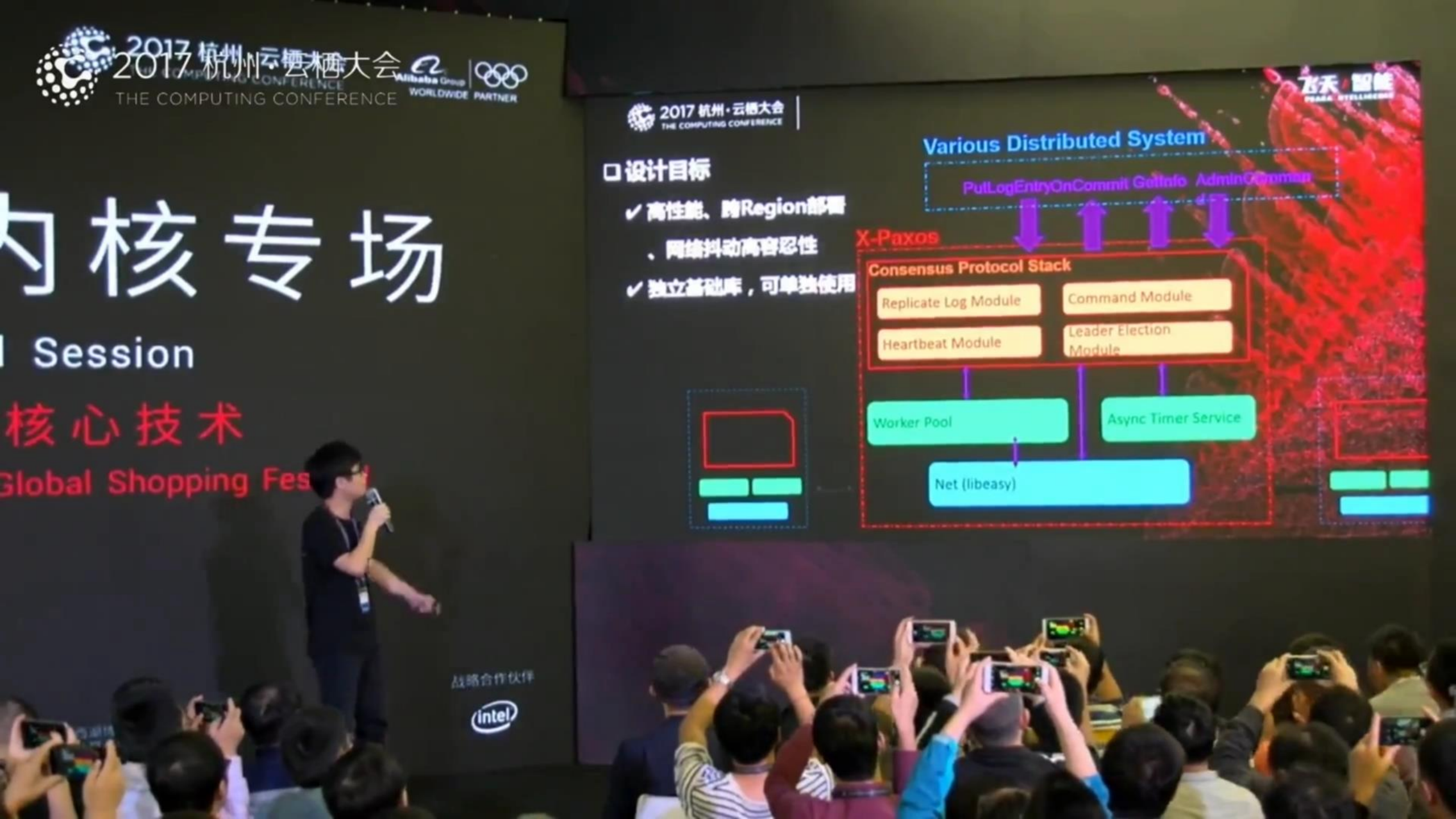
2000s Big Data

- 2004 Google
  published 3 papers

1998, Carlo Strozzi used N
to name his lightweig
Strozzi NoSQL open-source relati
which is distinct from the aro
general concept of NoSQL d

NoSQL

中国不能，也不会缺席！


2017 杭州·云栖大会
THE COMPUTING CONFERENCE
Alibaba Group
WORLDWIDE PARTNER
内核专场
Session
核心技术
Global Shopping Fes
战略合作伙伴
intel
2017 杭州·云栖大会
THE COMPUTING CONFERENCE
飞天 智能
□设计目标
✔高性能、跨Region部署
、网络抖动高容忍性
✔独立基础库，可单独使用
Various Distributed System
PullLogEntryOnCommit GetInfo AdminCommand
X-Paxos
Consensus Protocol Stack
Replicate Log Module
Command Module
Heartbeat Module
Leader Election Module
Worker Pool
Async Timer Service
Net (libeasy)

2017 杭州·云栖大会
THE COMPUTING CONFERENCE
Alibaba Group WORLDWIDE PARTNER

内核专场

Session

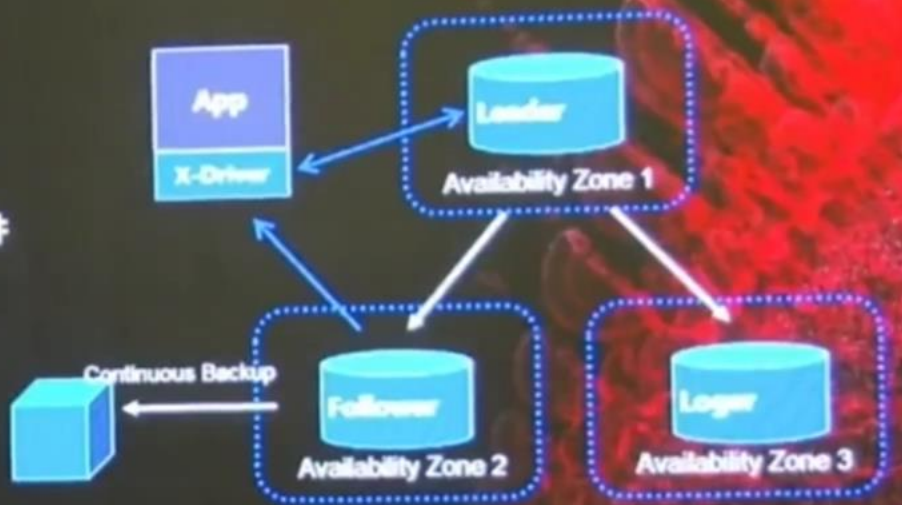
核心技术

Global Shopping Festival

2017 杭州·云栖大会
THE COMPUTING CONFERENCE

X-Cluster部署案例——Multi-AZ

- 数据强一致
  - ✓ 单AZ不可用数据0丢失
  - ✓ 单AZ不可用秒级切换
  - ✓ 切换自封闭，无第三方组件
- 0成本增加
  - ✓ 相对主备模式0成本增加
- 持续备份
  - ✓ RPO < 1s



2017 杭州·云栖大会
THE COMPUTING CONFERENCE

Alibaba Group WORLDWIDE PARTNER

内核专场

Session

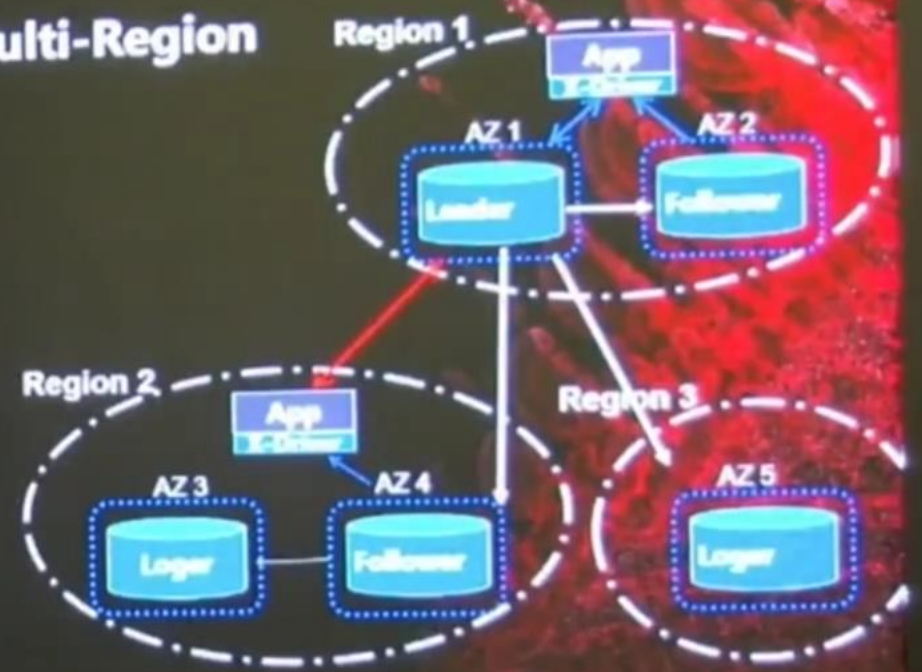
核心技术

Global Shopping Festival

2017 杭州·云栖大会
THE COMPUTING CONFERENCE

X-Cluster部署案例——Multi-Region

- 真正Region级的强一致能力
  - ✓ 单个Region不可用0数据丢失
- 高性能
  - 跨Region强同步下依然保持高性能
- 灵活的切换策略
  - 优先切换同Region
  - 定制跨Region切换顺序
- 高伸缩性
  - 可无限制的扩充Region/AZ的节点
  - 可自由的调节Region/AZ内节点



intel


2017 杭州·云栖大会
E COMPUTING CONFERENCE
2017 杭州·云栖大会
飞天 / 智能
2017 杭州·云栖大会
企业高可用架构专场
High-availability Enterprise Architecture Session
让系统在双十一峰值流量下依然稳如磐石
Making the System Stable under the Peak Flow of 11.11
企业高可用架构专场
High-availability Enterprise Architecture Session
让系统在双十一峰值流量下依然稳如磐石
Making the System Stable under the Peak Flow of 11.11


HDC.2019
HDC.2019


2020 TECHO PARK
[TECHO]
开发者大会
Tencent 腾讯 | 腾讯云
Tencent 腾讯 | 腾讯云
腾讯云·未来开发者云梯计划


Bai du 百度
股票代码 Stock Code：9888.HK
热烈庆祝百度集团股份有限公司香港联合交易所成功上市
Congratul
on HKEX
9888.HK
爱这时代
星辰大海
Bai du 百度
Bai du 百度


ByteDance
字节跳动

全球超算500强中国上榜数量蝉联第一

全球超算500强榜单每半年发布一次
在德国法兰克福举行的国际超级计算大会当地时间6月17日发布了最新榜单

中国境内有219台超算上榜

中国超算上榜数量蝉联第一

这是2017年11月以来，中国超算上榜数量连续第四次位居第一

2 美国　3 日本　4 法国　5 英国　6 德国

中国企业也继续保持上榜数量优势

在此次榜单上

全球超算制造商前三位

联想	浪潮	中科曙光
173台	71台	63台

新华社发（李栋制图）

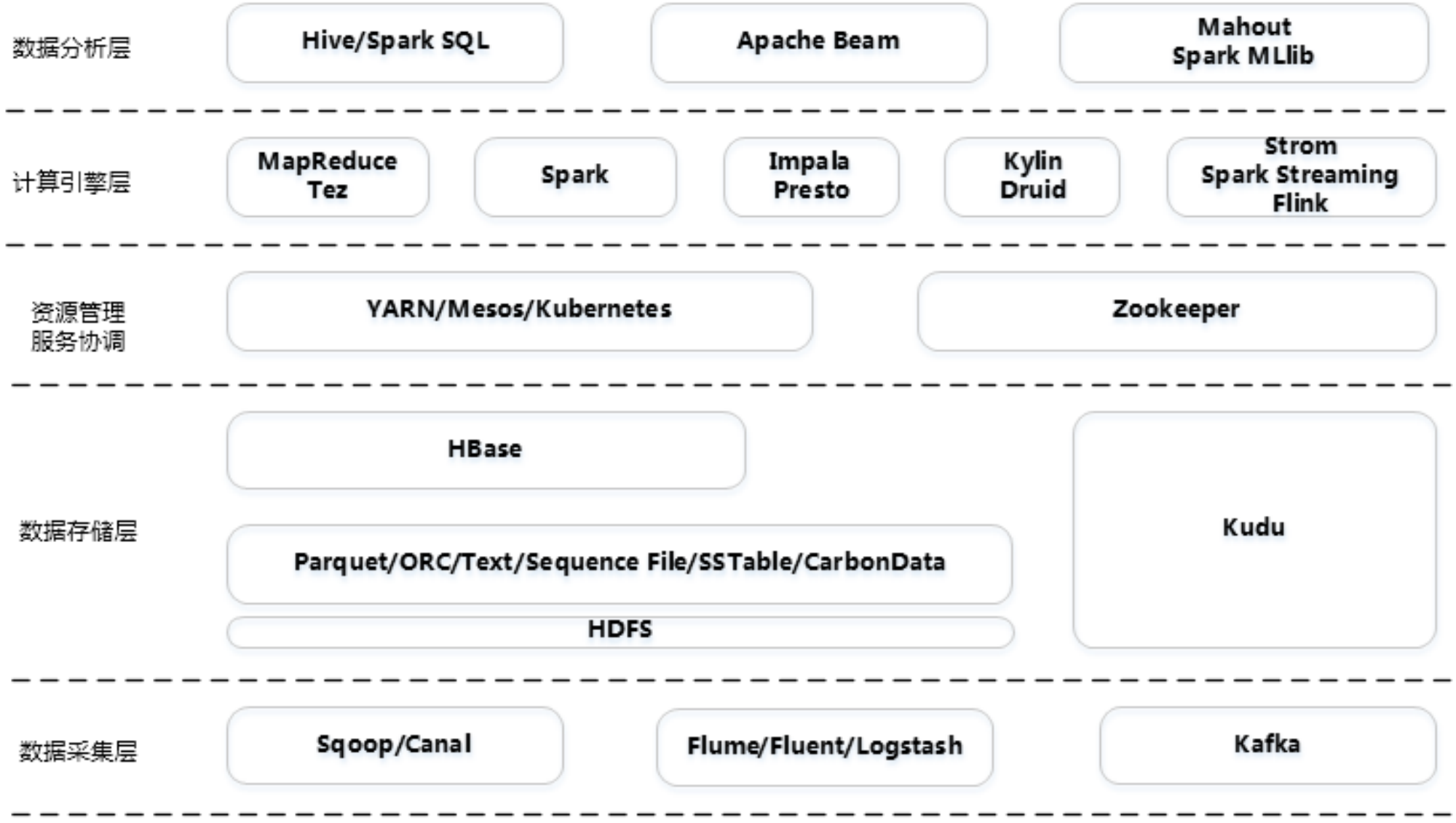

TFIAS
首届超级计算创新应用天府论坛
The 2020 Tianfu Forum on Innovative Application of Supercomputing
TFIAS

Unix、Windows、BSD Based等操作系统在2017年11月的榜单中首次尽数消失，TOP500所有系统都采用开源的Linux系统。

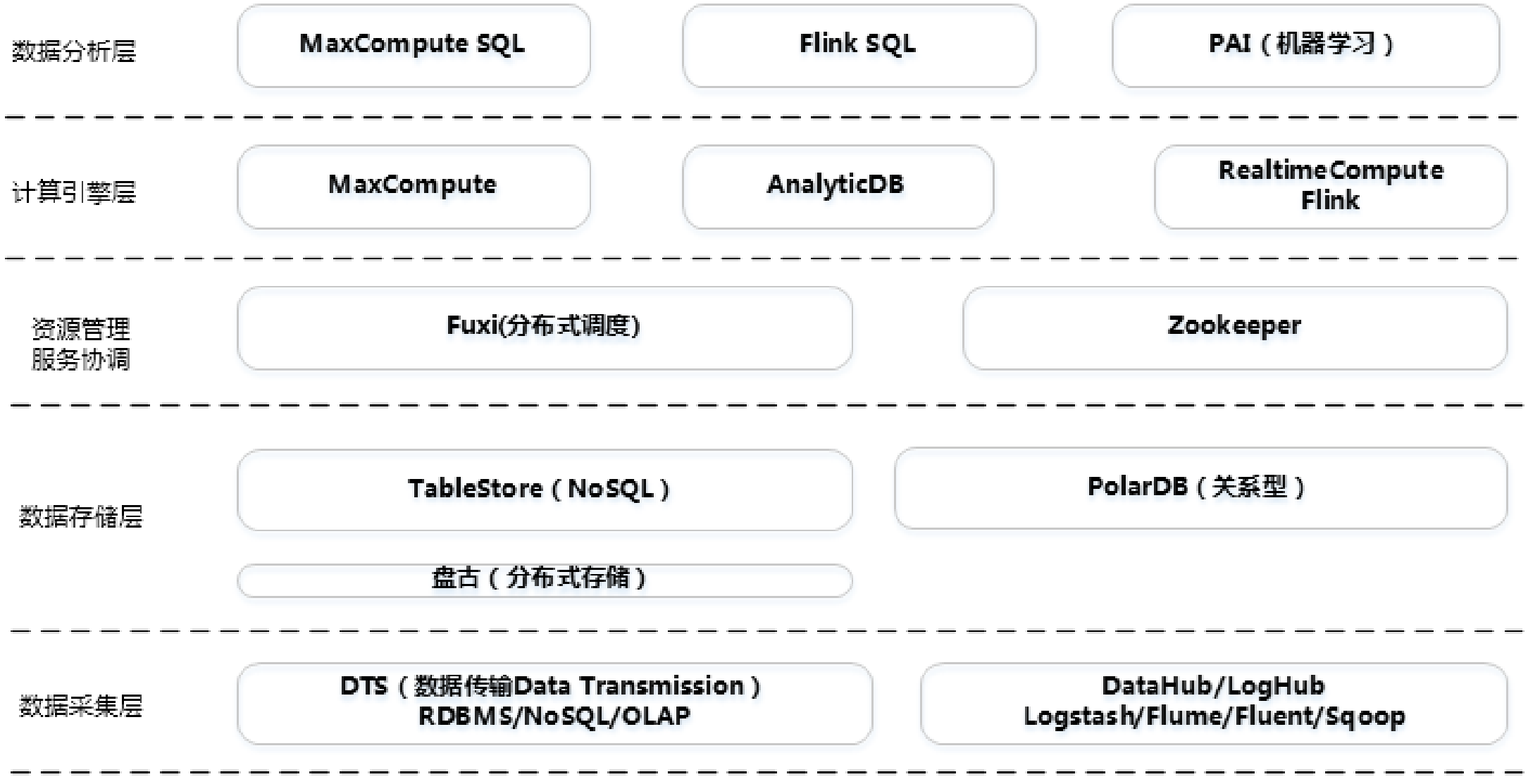
谷歌技术栈


数据分析层
FlumeJava
Tenzing
SQL
计算引擎层
MapReduce
Dremel
Pregel
Percolator
MillWheel
资源管理
服务协调
Borg/Omega
Chubby
MegaStore
Spanner
数据存储层
BigTable
GFS
数据采集层

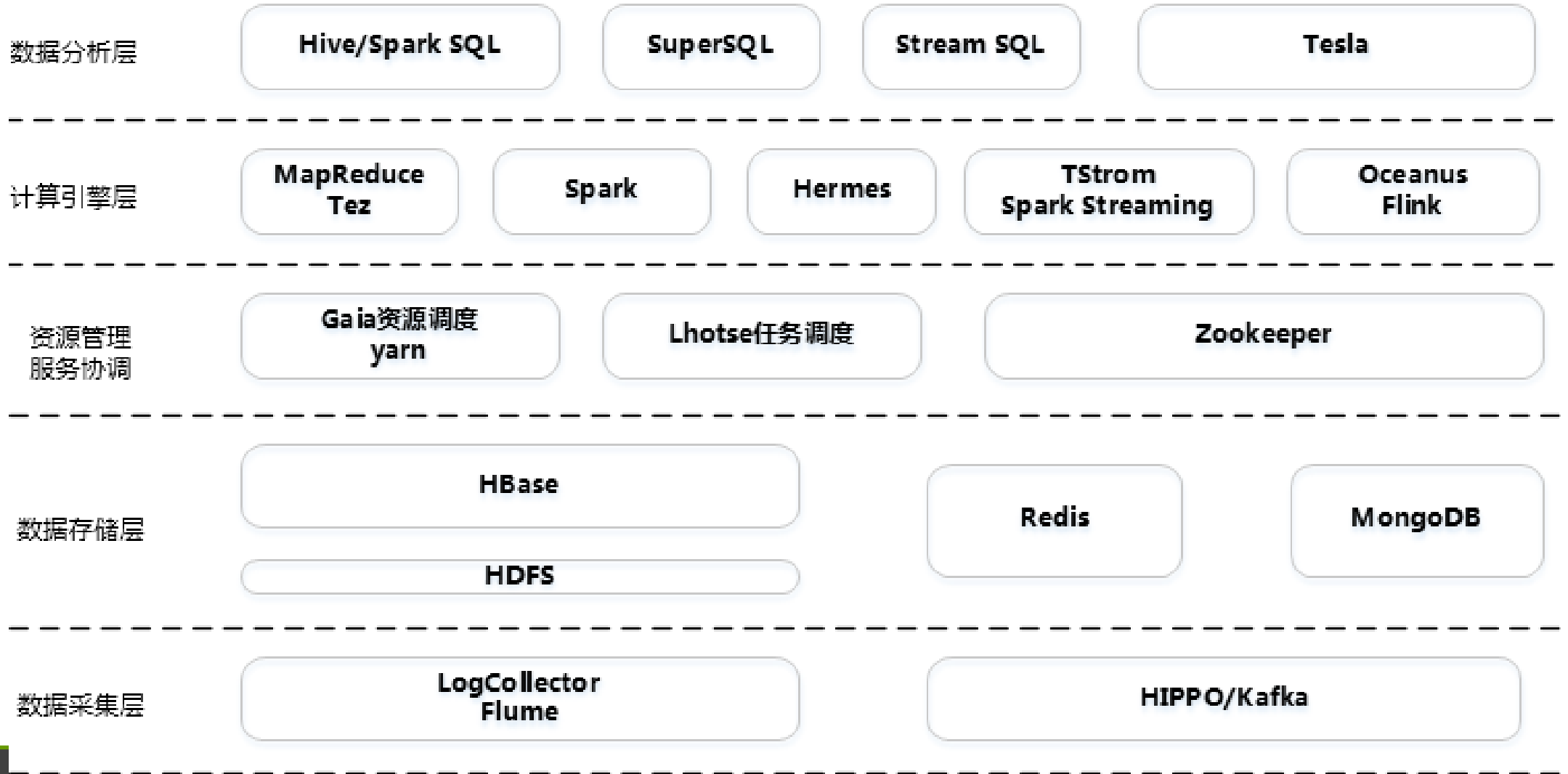
开源技术栈


数据分析层
Hive/Spark SQL
Apache Beam
Mahout
Spark MLlib
计算引擎层
MapReduce
Tez
Spark
Impala
Presto
Kylin
Druid
Strom
Spark Streaming
Flink
资源管理
服务协调
YARN/Mesos/Kubernetes
Zookeeper
数据存储层
HBase
Parquet/ORC/Text/Sequence File/SSTable/CarbonData
HDFS
Kudu
数据采集层
Sqoop/Canal
Flume/Fluent/Logstash
Kafka

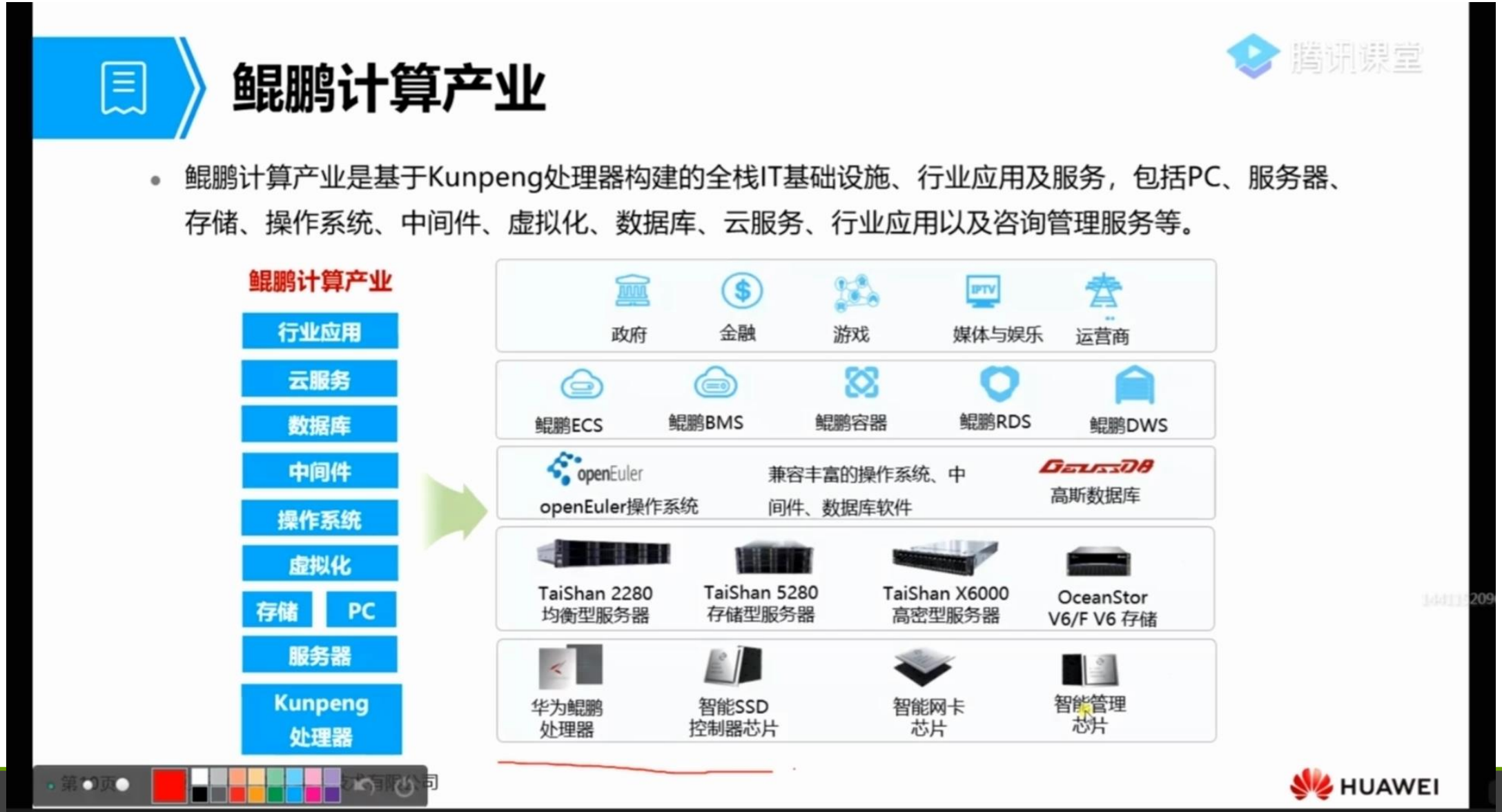
阿里技术栈

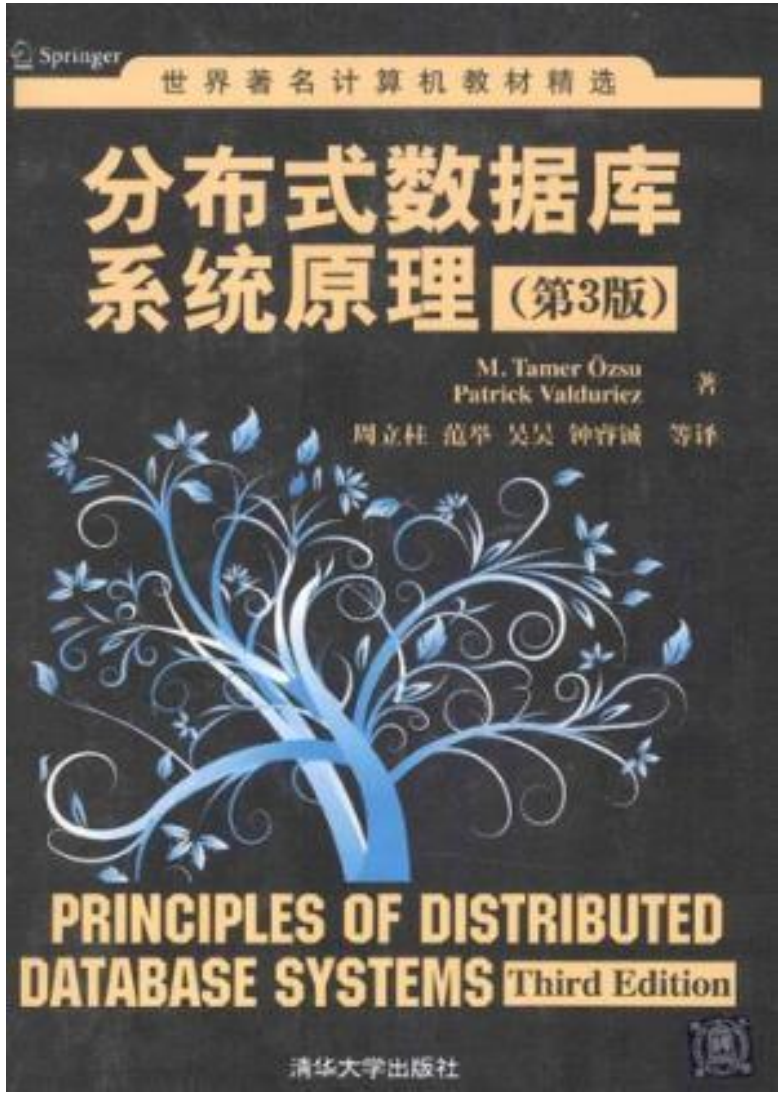

数据分析层
MaxCompute SQL
Flink SQL
PAI（机器学习）
计算引擎层
MaxCompute
AnalyticDB
RealtimeCompute
Flink
资源管理
服务协调
Fuxi(分布式调度)
Zookeeper
数据存储层
TableStore（NoSQL）
PolarDB（关系型）
盘古（分布式存储）
数据采集层
DTS（数据传输Data Transmission）
RDBMS/NoSQL/OLAP
DataHub/LogHub
Logstash/Flume/Fluent/Sqoop

腾讯技术栈

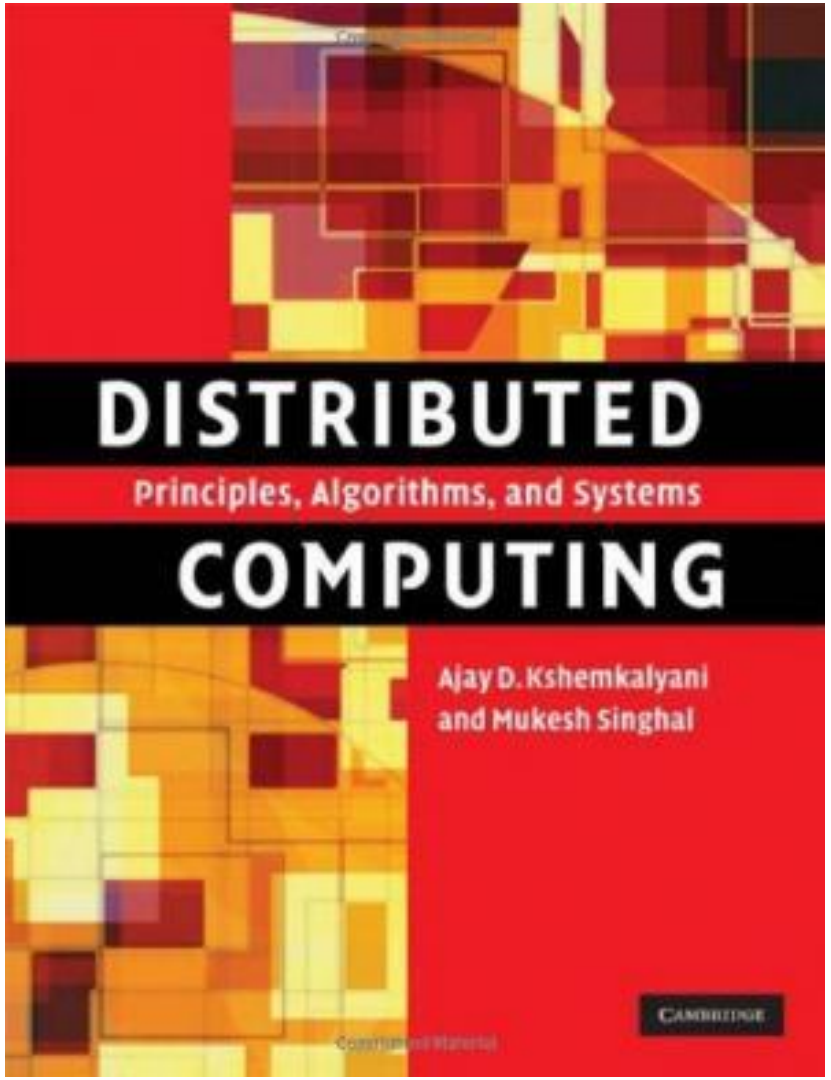

数据分析层
Hive/Spark SQL
SuperSQL
Stream SQL
Tesla
计算引擎层
MapReduce
Tez
Spark
Hermes
TStrom
Spark Streaming
Oceanus
Flink
资源管理
服务协调
Gaia资源调度
yarn
Lhotse任务调度
Zookeeper
数据存储层
HBase
HDFS
Redis
MongoDB
数据采集层
LogCollector
Flume
HIPPO/Kafka

华为技术栈？ 呵呵
– “为了卖硬件而不得不制作软件”，而且作业也很好！

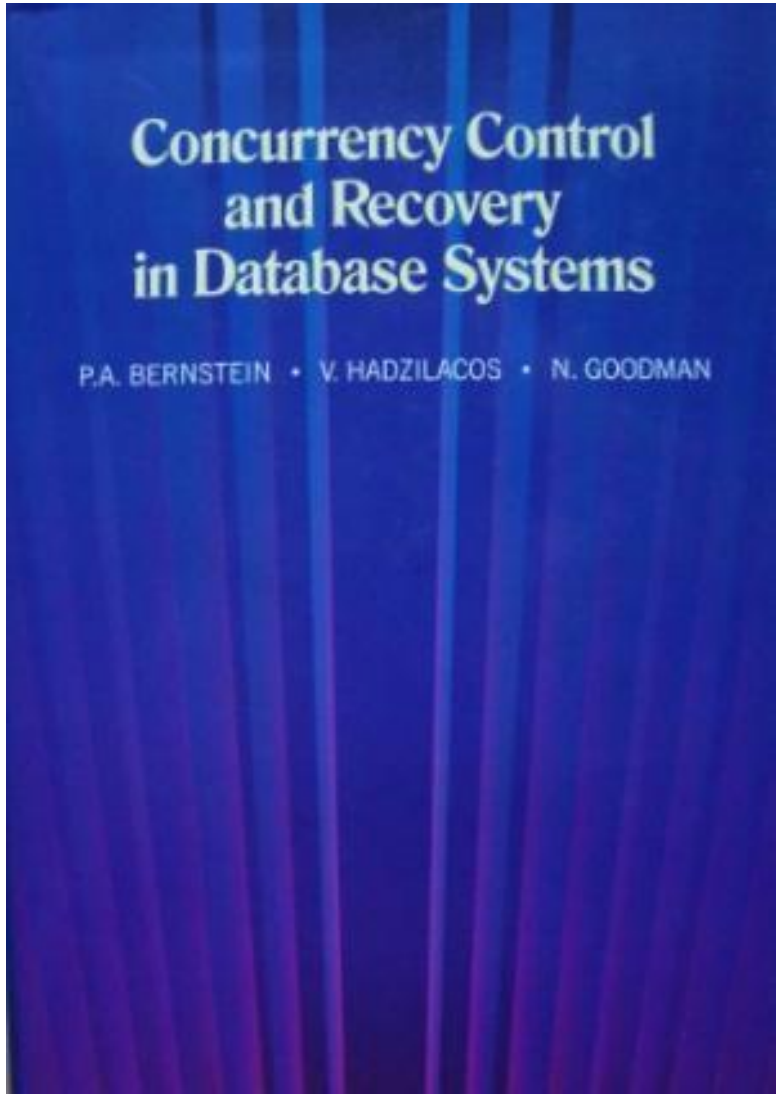




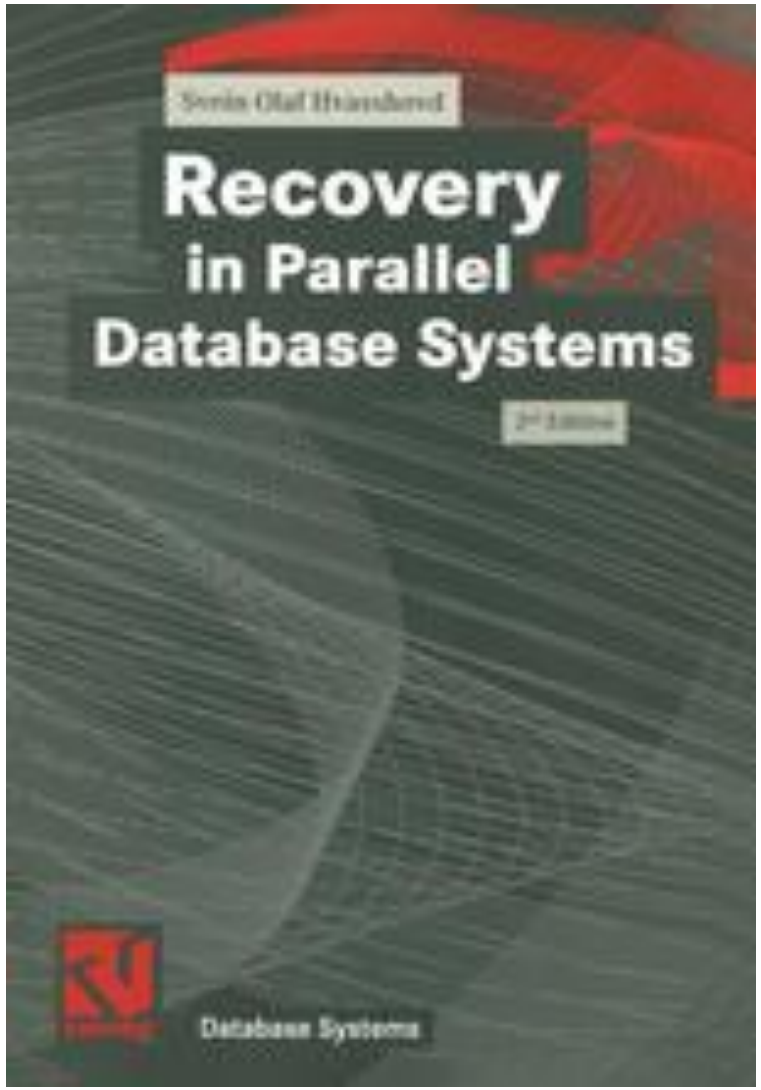
- 分布式数据库系统原理
- *厄兹叙 (M.Tamer Ozsu) / Patrick Valduriez 著，周立柱 / 范举 译*
- 2014
- 清华大学出版社



- Distributed Computing: Principles, Algorithms, and Systems
- *Ajay D. Kshemkalyani*, *Mukesh Singhal*
- **2008**
- **中文版 2012**



- Concurrency Control and Recovery in Database Systems
- *Philip A. Bernstein*, *Vassos Hadzilacos*, *Nathan Goodman*
- **1987**



- Recovery in Parallel Database Systems
- *Svein-Olaf Hvasshovd (auth.)*
- **1999**