Parallele Datenverarbeitung
Pig, Hive & SystemT/JAQL

Henning Kropp
wir02jcr@studserv.uni-leipzig.de

Übersicht

- Parallele Datenverarbeitung
- Map-Reduce
- Pig
- Hive
- SystemT/JAQL
- Zusammenfassung

Parallele Datenverarbeitung

- Meist Verarbeitung von großen Datenmengen
- Voraussetzung ist die Datenunabhängigkeit in den einzelnen Verarbeitungsschritten
- Map-Reduce als geeignetes Modell
- Alle hier vorgestellten System erzielen Parallelität durch das Map-Reduce Konzept
  - Alle basieren auf Hadoop http://hadoop.apache.org

Map-Reduce

- Einfaches Modell zur Parallelisierung
- map() und reduce() der funktionalen Programmiersprachen wie:
  - Haskell, Lisp, R usw.
- Datenfluss-Verarbeitung für ETL Systeme
- Erfreut sich seit seiner Veröffentlichung 2004 großer Beliebtheit, vor allem bei Analytiker in Webunternehmen
- Seit 19. Januar 2010 patentiert durch Google Inc.

Kritik an Map-Reduce

- Einfaches Konzept := geringe Abstraktion
- Starres Konzept
- Map-Reduce beschäftigt sich mit Problemen die vor Jahren schon in parallelen DBMS gelöst wurden
- Entgegen der allgemeinen Auffassung zeigen parallele DBMS fast lineare Skalierbarkeit
- De Witt & Stonebraker: [8],[10],[14]
  „Map-Reduce is a major step backwards“

MR und parallele DBMS

- Hauptkritikpunkte an MR: [8], [10], [14]
  - Starr
    - Wiederholt map() u. reduce()
    - Fehleranfälligkeit durch fehlende Funktionalität
  - Geringe Abstraktion
    - Keine Joins
    - Keine Datentypen
- Hier vorgestellte Konzepte ergänzen MR in dieser Hinsicht

Pig

- Von Yahoo! Inc. entwickelte adhoc Analysesystem und offizielles Apache Foundation Project
- Datenfluss-Umgebung mit eigener Sprache
- Umgebung besteht aus vers. Werkzeugen
- ca. 30% aller Hadoop Aufgaben bei Yahoo! Inc. werden mittels Pig ausgeführt [1]
  - Yahoo! zählt sich zu den größten Hadoop Anwendern

Pig – Philosophie

- Pigs Eat Anything (Schweine sind Allesfresser)
  - unstrukturiert/strukturiert & Bytes, String, ...
- Pigs Live Anywhere (Schweine leben überall)
  - Nicht System gebunden / Unabhängige Sprache
- P. Are Domastic Animals (Schweine sind heimisch)
  - Benutzer freundlich / Eingebettet / UDF
- Pigs Fly (Schweine können fliegen)
  - Pig soll Schnell sein (z.Z. x1.4, Ziel x1.2)

http://hadoop.apache.org/pig/philosophy.html

Pig – Umgebung

- Überblick über die Pig Bestandteile:

pig.jar	Laufzeitumgebung (Parser, Compiler)
PigServer	Für eingebettete Ausführung
PigPen	Eclipse Plugin für die Entwicklung von PigLatin
PigMix	Benchmark
PigLatin	SQL ähnliche Sprache.
Grunt	Benutzerschnittstelle
PiggyBank	UDF Bibliotheken Sammlung.
Test/Debug	Test und Debugging Umgebung.

Pig – PigLatin

- SQL ähnliche Sprache in R Syntax
- Beispiel: mit *A (1,3,4) (1,5,6)* und *B (1,5,6)*

A = LOAD 'data1' as (a1:int, a2:int, a3:int);

B = LOAD 'data2' as (b1:int, b2:int, b3:int);

X = JOIN A BY a:1, B BY b1;

DUMP X;

=> { (1,3,4,5,6) , (1,5,6,5,6) }

Pig - PigLatin

- Datentypen:

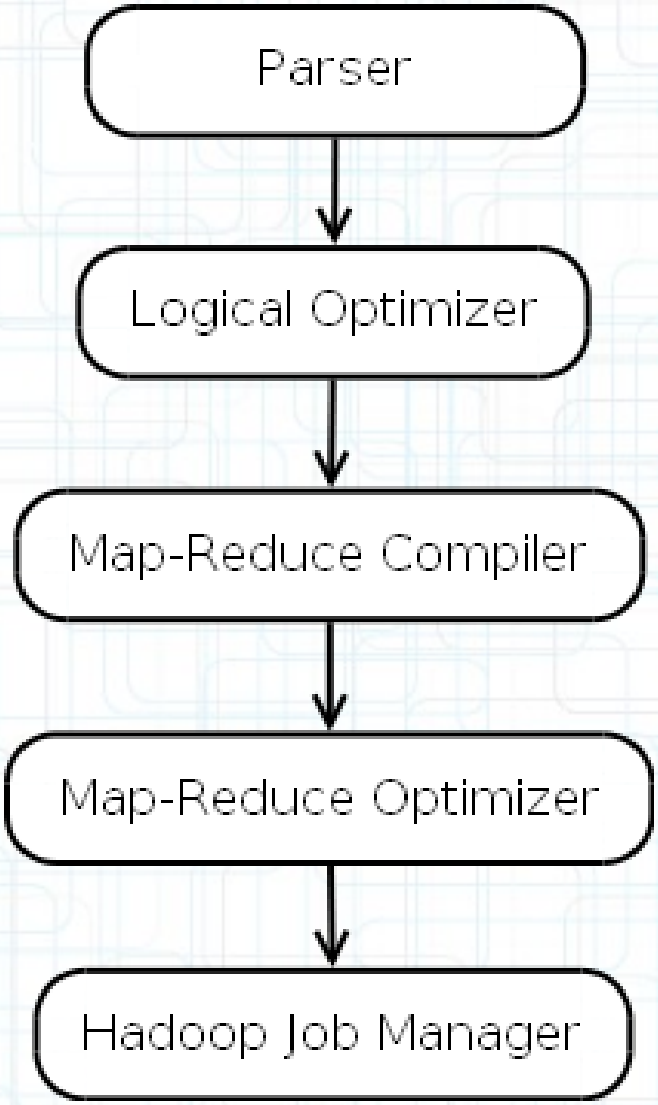
Datentype	Beschreibung	Beispiel
int	Signed 32-bit integer	10
long	Signed 64-bit integer	10L
float	32-bit floating point	10.5F
double	64-bit floating point	10.5
chararray	Char Array in UTF-8	Hello world
bytearray	Byte array (BLOB)	
tuple	Ordered set of fields	(19,2)
bag	Collection of tuples	{(19,2),(18,3)}
map	Key value pairs	[open#apache]

Pig – PigLatin

- Einige relationale Operatoren:
  - COGROUP/GROUP
  - FILTER
  - JOIN; JOIN OUTER
  - DISTINCT
  - UNION
  - ORDER
  - SPLIT

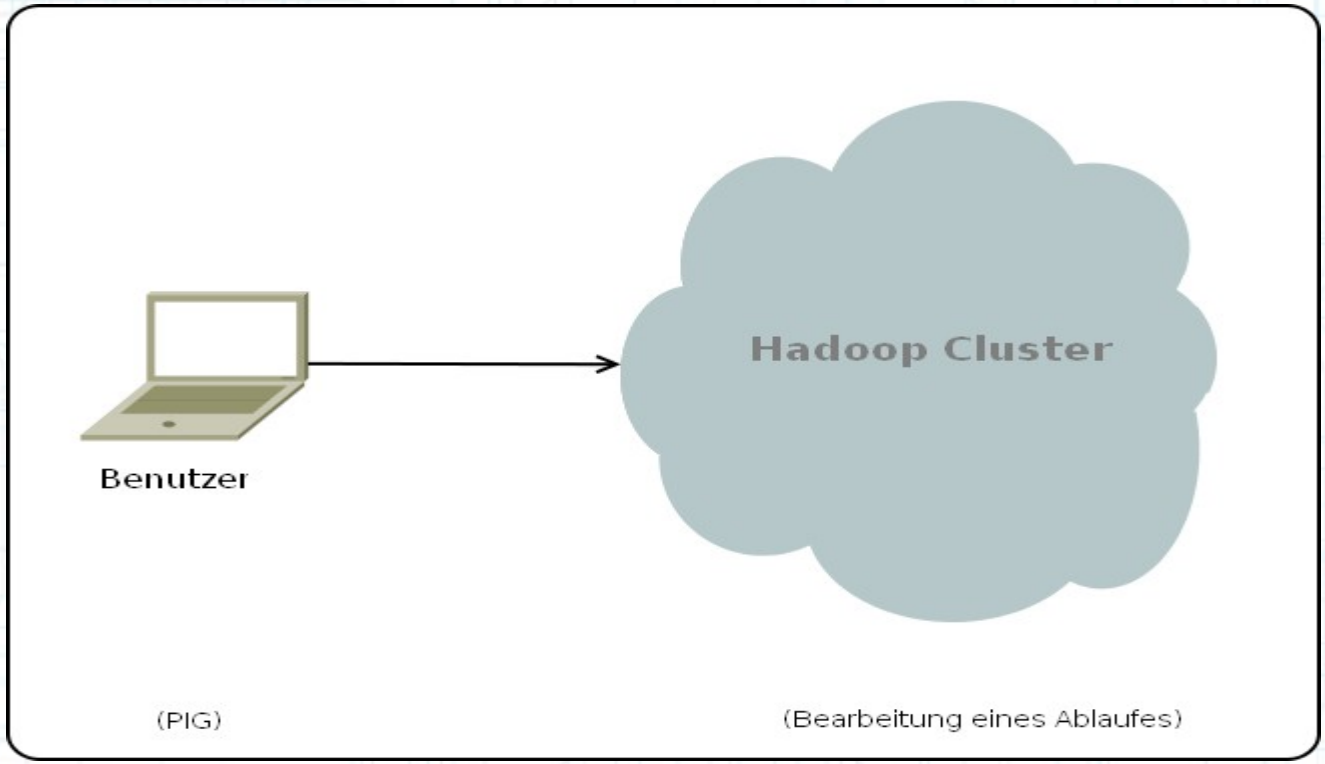
Pig - Compiler

- PigLatin als Input
- Umwandlung in MR Anweisungen
- Späte Bindung von Daten-typen (Lazy Databinding)
- Erst Ausführung, wenn Ausgabe (Lazy Execution)



Pig – Umgebung

- Ausführung von Pig:
  - CLI
  - Eingebettet



Pig – Einsatz bei Yahoo!

- Analyse von Log-Dateien
- Datenverarbeitung für Suchplattform
- Adhoc Anfragen auf große Daten
- Schnelle Prototypen Entwicklung

[1] Alen Gates – Introduction to Pig

Hive

- Von Facebook entwickeltes Data-Warehouse System basierend auf MR
- Offizielles Projekt der Apache Foundation
- Eigene Anfragesprache - HiveQL
- Bietet Datenschema mit DML und DDL
- Schnittstelle über CLI, JDBC/ODBC oder WebGUI

Hive - Prinzip

- Skalierbarkeit vor Schnelligkeit
- Da beruhend auf Stapelverarbeitung (MR) keine geringe Latenz garantiert
  - Selbst kleine Anfragen können lange dauern
- Erweiterbarkeit durch UDF
  - *hive> ADD JAR example.jar;*
  - *Datentypen SerDe Schnittstelle*
- Hive eher Prototype Charakter

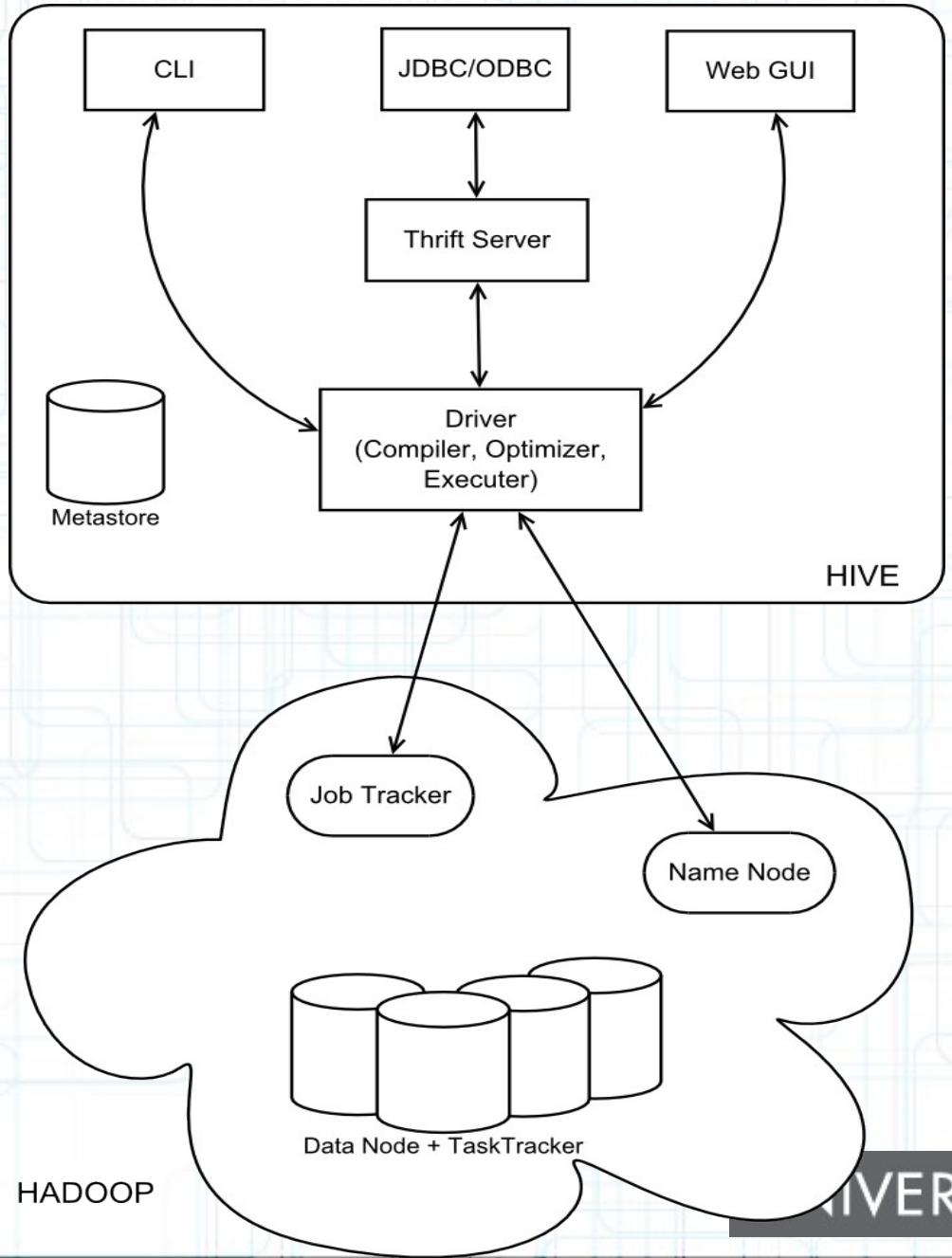
Hive - Funktionsweise

- Speichert Daten in Dateien, typischerweise im HDFS Dateisystem
- Tabellen sind Ordner und Einträge sind serialisierte Textdateien
- Tabellen können noch in Partitionen anhand eines Wertes unterteilt werden
- Daten in jeder Partition können noch in Buckets anhand von Hash-Werten des Schlüssels unterteilt werden
- Anfragen mittels HiveQL

Hive - Datenstruktur

- Metastore
  - Enthält bei jeder Anfrage nachgefragte Schema Informationen
  - Ist selbst DB (derby, MySQL, BD, …)

Hive - Umgebung



Hive - HiveQL

- Einfache Datentypen
  - tinyint, smallint, int, bigint, boolean, float, string
- Verschachtelt Datentypen
  - Struct *col. c: STRUCT{ a INT }* => *a.c*
  - Map *M: 'group'->gid* => *M['group']*
  - Array *A: ['a','b']* => *A[1] = a*
- Eigene Datentypen
  - SerDe (Serailization/Deserialization) Interface

Hive - HiveQL

- DDL:
  - CREATE / DROP TABLE
  - ALTER TABLE
    - add/drop partition
    - rename table / columns (change type, position, comment)
    - add/replace column
    - add proerties table / serde
    - alter table file format and organistaion
  - SHOW / DESCRIBE TABLE(S), PARTITION(S), FUNCTION(S)
  - CREATE / DROP VIEW

Hive - HiveQL

- DML:
  - LOAD DATA (copy/move)
    - LOAD DATA [LOCAL]
      INPATH 'filepath' [OVERWRITE]
      INTO TABLE tablename
      [PARTITION (partcol1=val1, ...)]
  - INSERT OVERWRITE
    - INSERT OVERWRITE
      TABLE tablename1
      [PARTITION (partcol1=val1, ...)]
      select_statement1
      FROM from_statement

Hive - HiveQL

- SELECT:
  - GroupeBy
  - SortBy (key;value)
    DistributeBy
    ClusterBy (sort & distribute)
  - Transform and Map-Reduce Scripts
  - Opterators and UDF
- JOINS {LEFT|RIGHT|FULL}
- UNION

SystemT/JAQL

- SystemT IR System mittels AQL von IBM
- AQL regelbasierte Sprache zur semantischen Extraktion
- SystemT und AQL Bestandteil der InfoSphere Produktreihe bei IBM
  - z. B. Lotus Mail (Extraktion von Telefonnummern, Spam Erkennung)
- JAQL Anfragesprache über MR (Hadoop) mit JSON Datentypen für Ein- u. Ausgabe
  - http://www.json.org/

SystemT/JAQL - SystemT

- SystemT
  - Entwicklungsumgebung
  - Laufzeit Umgebung
- AQL
  - Höhere Anotationssprache
  - Beispiel:

```
create view Url as
extract
    regex /http:\/\/\S+/
    on D.text as url
from Document D;
```

SystemT/JAQL – AQL Bsp

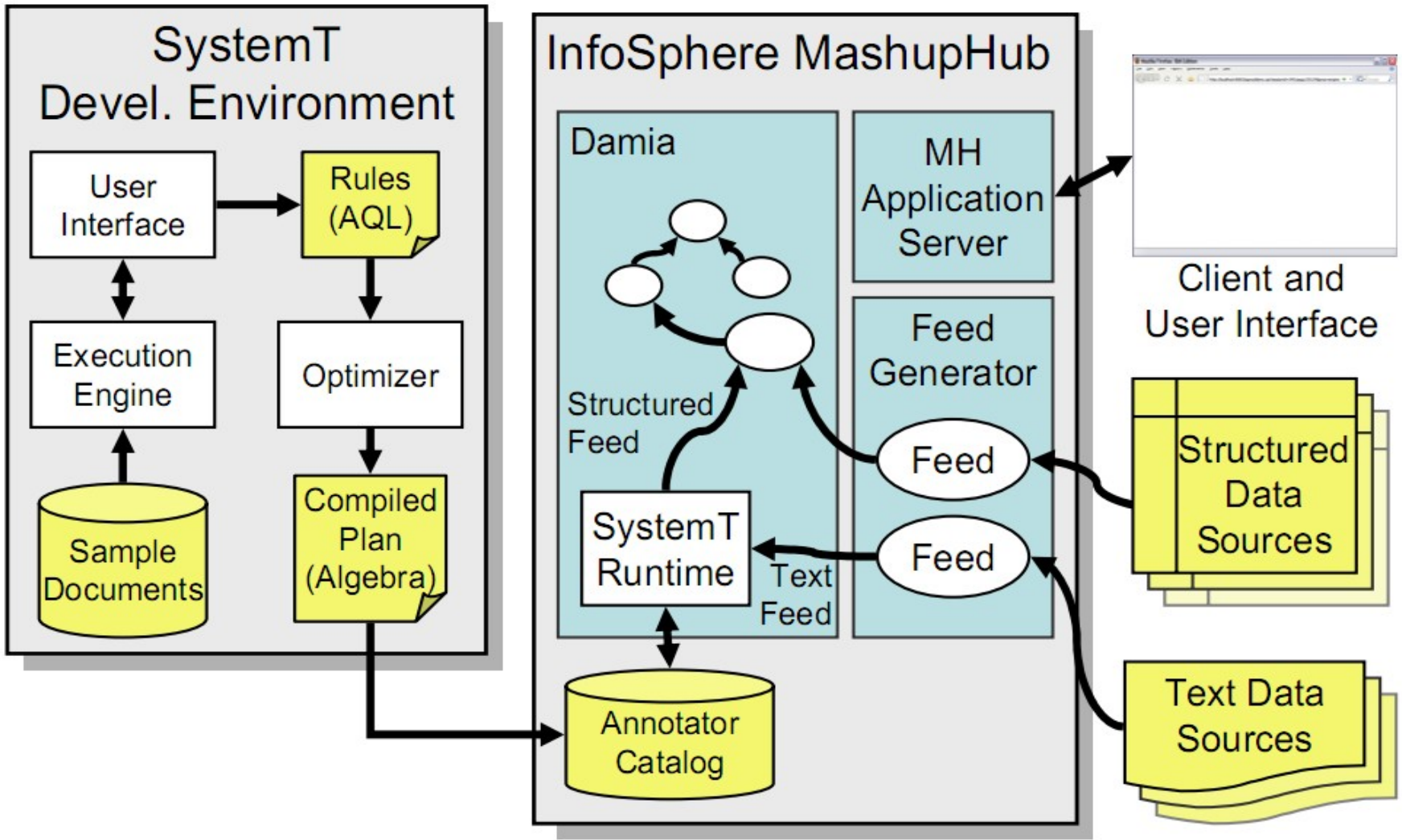
```
create dictionary corp as
   ('IBM','Yahoo','Facebook',..);

create view URL as
   extract
        regex /http:\/\S+/
        on D.text as url
   from Document D;

create view CorpUrl as
   select CombineUrlCorp (B.url, I.inst) as instance
   from URL,
        (extract dictionary 'corp' on D.text as inst
        from Document D) I
   where
        Follows(I.inst, B.url, 0, 30 )
   consolidate on CombineUrlCorp(I.inst, B.url);
```

SystemT/JAQL - Umgebung

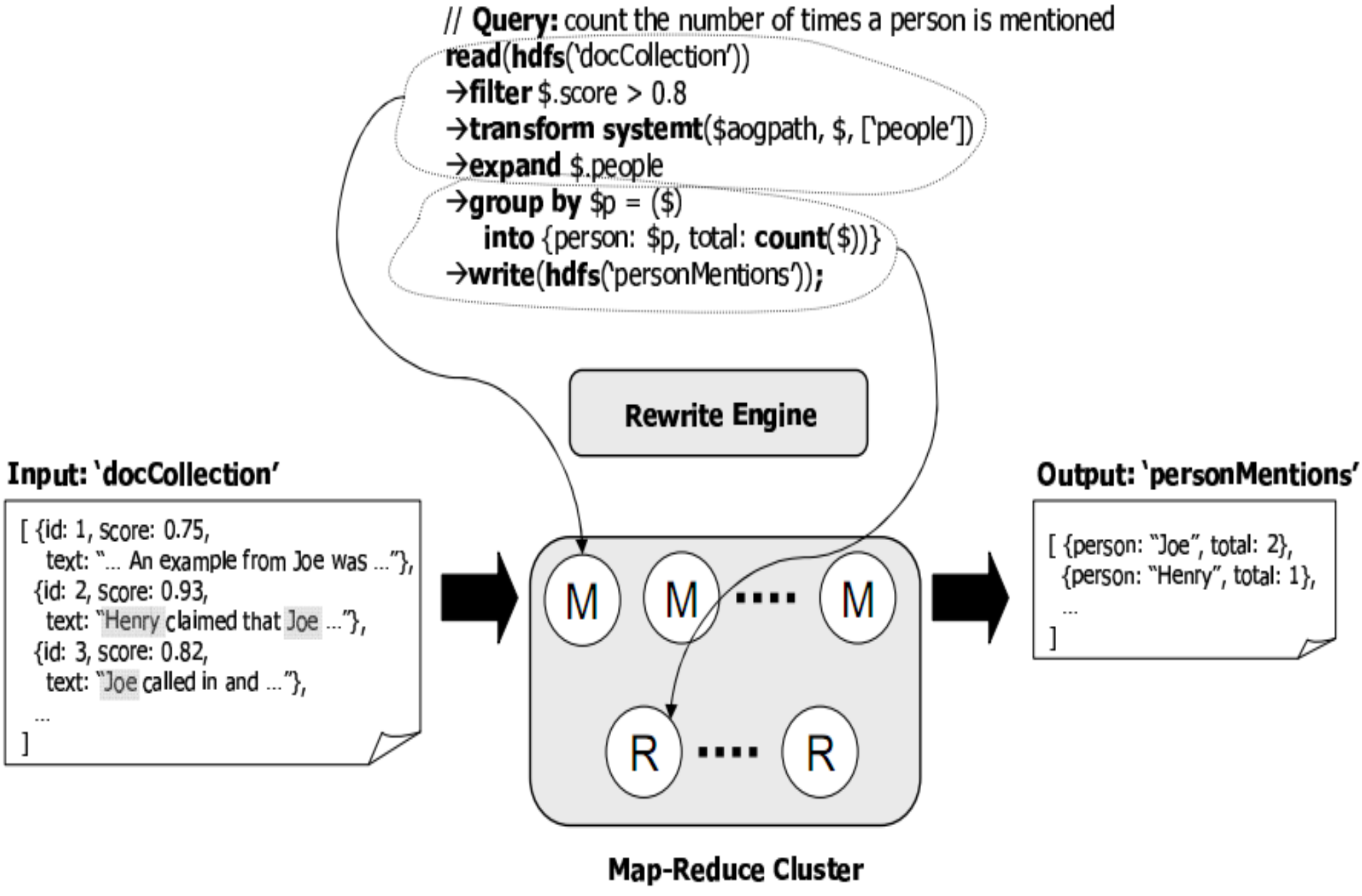
- Überblick über die Entwicklungs- und Laufzeitumgebung [17]:



SystemT/JAQL – JAQL

- Verwendung von JSON als Ein- und Ausgabedaten
- Übersetzung von Anfrage in Map-Reduce
- Speichern und lesen von verschiedenen System z. B. lokales Dateisysten, HDFS und Hbase
- Erweiterung durch UDF
- JAQL Syntax unter ständiger Entwicklung und noch nicht final (Stand: 29.09.2009) [18]

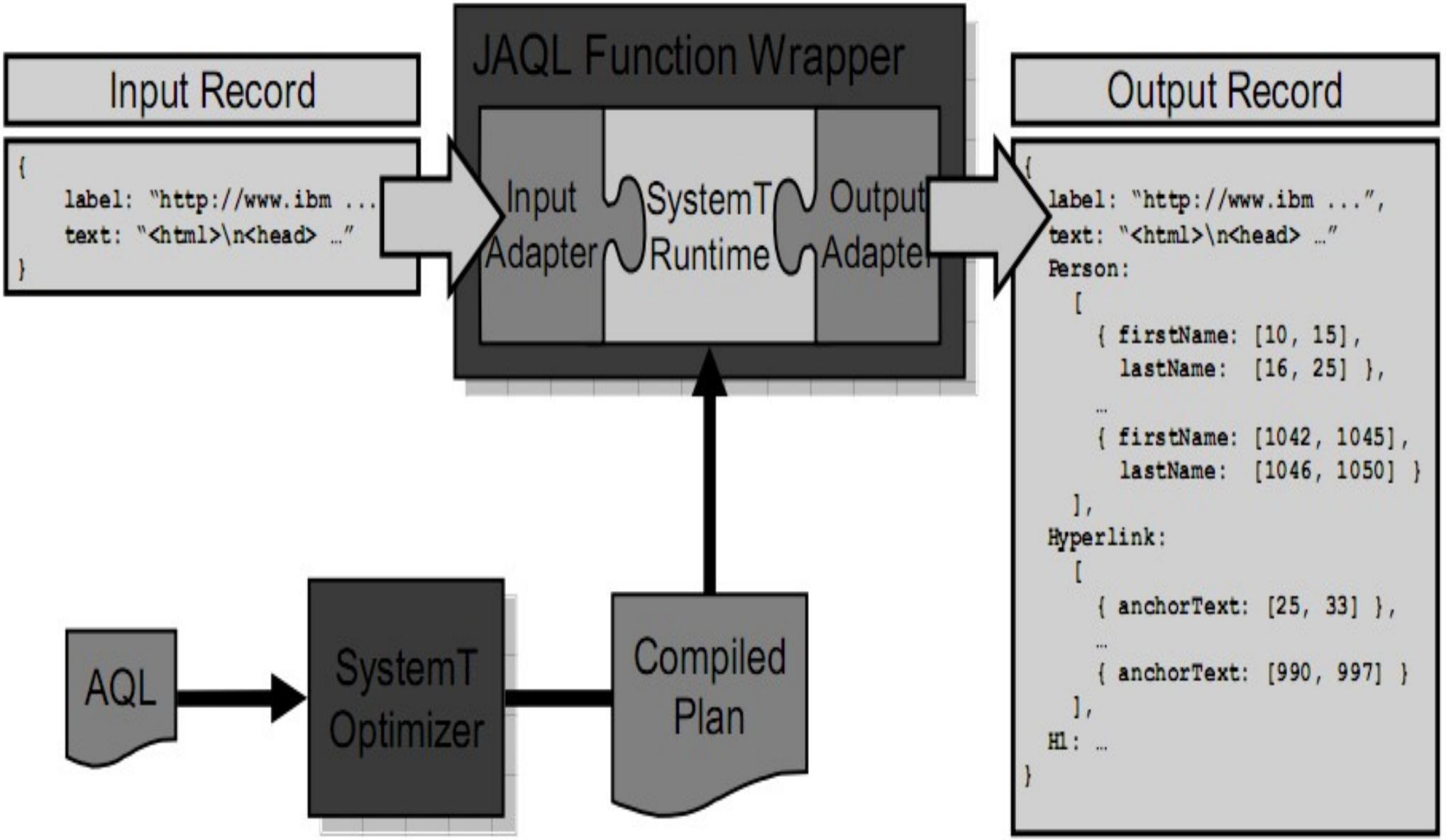
SystemT/JAQL – JAQL [6]



SytemT/JAQL - JAQL

- Kernfunktionen:[20]
  - Filter
  - Transform
  - Group
  - Join
  - Sort
  - Expand
- SystemT/JAQL in der Entwicklungsphase bei IBM

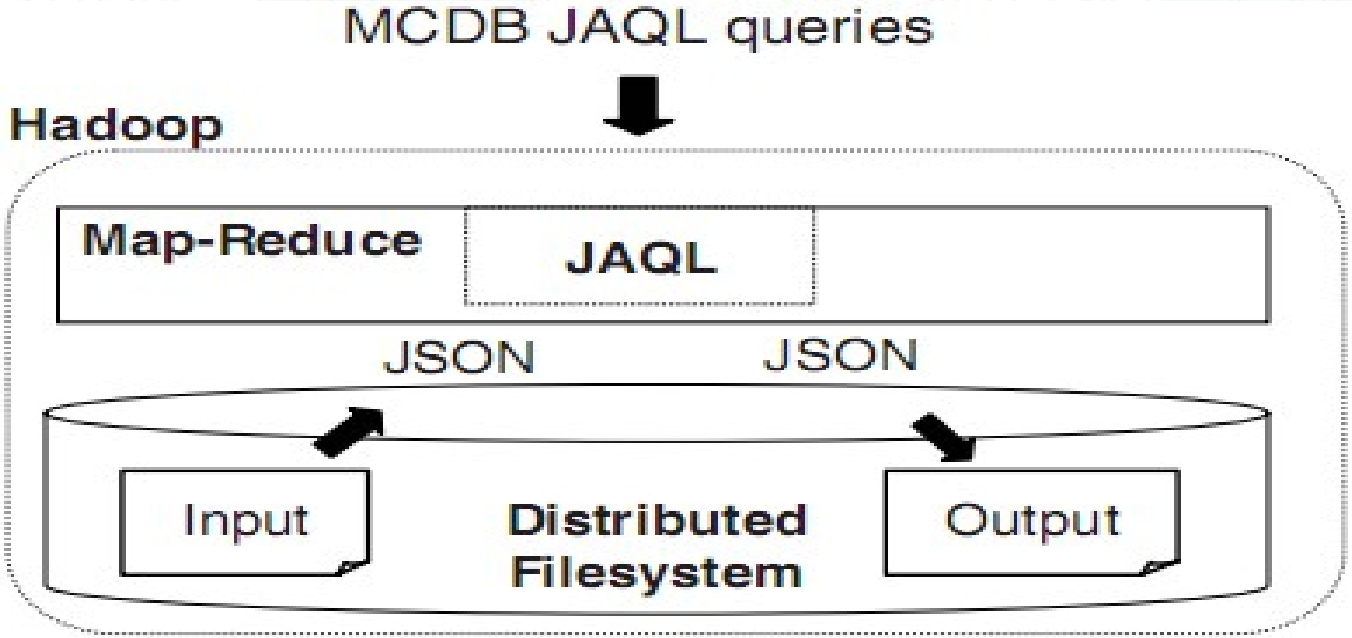
SystemT/JAQL – Umgebung [6]



SystemT/JAQL - JAQL

- Regelbasierte Anfragesprache für JSON Daten
- Ausführung mittels Map-Reduce
- Implementierung in der MCDB

[13]



Zusammenfassung

- Alle hier vorgestellten System implementieren eine höhere Sprache basierend auf Map-Reduce
- Vergleiche könnten über Compiler und Sprachumfang geführt werden
- Aber jedes hier vorgestellte System verfolgt einen anderen Ansatz
  - Pig *adhoc Datenanalyse*
  - Hive *Datawarehouse*
  - JAQL *Semantische Extraktion*

Ende

Vielen Dank!

Fragen?

Literatur

1: Alan Gates
Cloudera Inc.: Introduction to Pig, 2009
http://www.cloudera.com/sites/default/files/IntroToPig.pdf

2: Christopher Olston, Benjamin Reed, Utkarsh Srivastava, Ravi Kumar, Andrew Tomkins
SIGMOD'08: Pig Latin: A Not-So-Foreign Language for Data Processing, 2008
http://research.yahoo.com/files/sigmod08.pdf

3: Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao,Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff and Raghotham Murthy
VLDB 09: Hive - A Warehousing Solution Over a Map-Reduce Framework, 2009
http://wiki.apache.org/hadoop-data/attachments/Hive(2f)Design/attachments/hive.pdf

4: Azza Abouzeid, Kamil Bajda-Pawlikowski, Daniel Abadi, Avi Silberschatz, Alexander Rasin
VLDB 09: HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads, 2009

5: Alan F. Gates, Olga Natkovich, Shubham Chopra, Pradeep Kamath, Shravan M. Narayanamurthy, Christopher Olston, Benjamin Reed
VLDB 09: Building a High-Level Dataflow System on top of Map-Reduce: The Pig Experience, 2009

Literatur

6: Kevin Beyer, Vuk Ercegovac, Rajasekar Krishnamurthy, Sriram Raghavan, Jun Rao, Frederick Reiss, Eugene J. Shekita, David Simmen, Sandeep Tata, Shivakumar Vaithyanathan, Huaiyu Zhu
IEEE Data Engineering 2009: Towards a scalable enterprise content analytics platform, 2009 http://sites.computer.org/debull/A09mar/sandeep.pdf

7: Christopher Olston, Benjamin Reed, Utkarsh Srivastava, Ravi Kumar, Andrew Tomkins
SIGMOD'08: Pig Latin: A Not-So-Foreign Language for Data Processing, 2008 http://research.yahoo.com/files/sigmod08.pdf

8: Michael Stonebraker, Daniel abadi, David J. DeWitt, Sam Madden, Erik Paulson, Andrew Pavlo Alexander Rasin
ACM: MapReduce and Parallel DBMSs: Friends or Foes?, 2010

9: David A. Patterson
ACM: Technical Perspective: The Data Center Is The Computer, 2008

10: Andrew Pavlo, Erik Paulson, Alexander Rasin, Daniel J. Abadi, David J. DeWitt, Samuel Madden, Michael Stonebraker
SIGMOD'09: A Comparison of Approaches to Large-Scale Data Analysis, 2009

Literatur

11: Christopher Olston, Benjamin Reed, Adam Silberstein, Utkarsh Srivastava
USENIX08: Automatic Optimization of Parallel Dataflow Programs, 2008

12: Jeffrey Dean and Sanjay Ghemawat
OSDI'04: MapReduce: Simplified Data Processing on Large Clusters, 2004
http://labs.google.com/papers/mapreduce.html

13: Fei Xu, Kevin Beyer, Vuk Ercegovac, Peter J. Haas, Eugene J. Shekita
SIGMOD'09: E = MC3: Managing Uncertain Enterprise Data in a Cluster - Computing Environment, 2009

14: David J. DeWitt, Michael Stonebraker
Databasecolumn: MapReduce: A major step backwards, 2008
http://databasecolumn.vertica.com/database-innovation/mapreduce-a-major-step-backwards/

15: Rajasekar Krishnamurthy, Yunyao Li, Sriram Raghavan, Frederick Reiss, Shivakumar Vaithyanathan, Huaiyu Zhu
SIGMOD'09: SystemT: A System for Declarative Information Extraction, 2009

Literatur

17: David E. Simmen, Frederick Reiss, Yunyao Li, Suresh Thalamati
SIGMOD'09: Enabling Enterprise Mashups over Unstructured Text Feeds with InfoSphere MashupHub and SystemT, 2009

18: Thilo Goetz
Hadoop users group meeting, Berlin: Text Analysis with JAQL, 2009
http://isabel-drost.de/hadoop/slides/hadoop-ug-jaql.pdf

19: Hive Wiki
http://wiki.apache.org/hadoop/Hive/ (2010-01-20)

20: JAQL
http://jaql.org/ (2010-01-20)

21: Pig Latin Reference Manual (Version 0.5.0)
http://hadoop.apache.org/pig/docs/r0.5.0/piglatin_reference.html (2010-01-20)