HadoopDB
ein großer Schritt in die falsche Richtung

Seminar 01912
Sommersemester 2011

Thomas Koch

Lehrgebiet Datenbanksysteme für neue Anwendungen
Fernuniversität Hagen

21. Juni 2011

HadoopDB

Hive

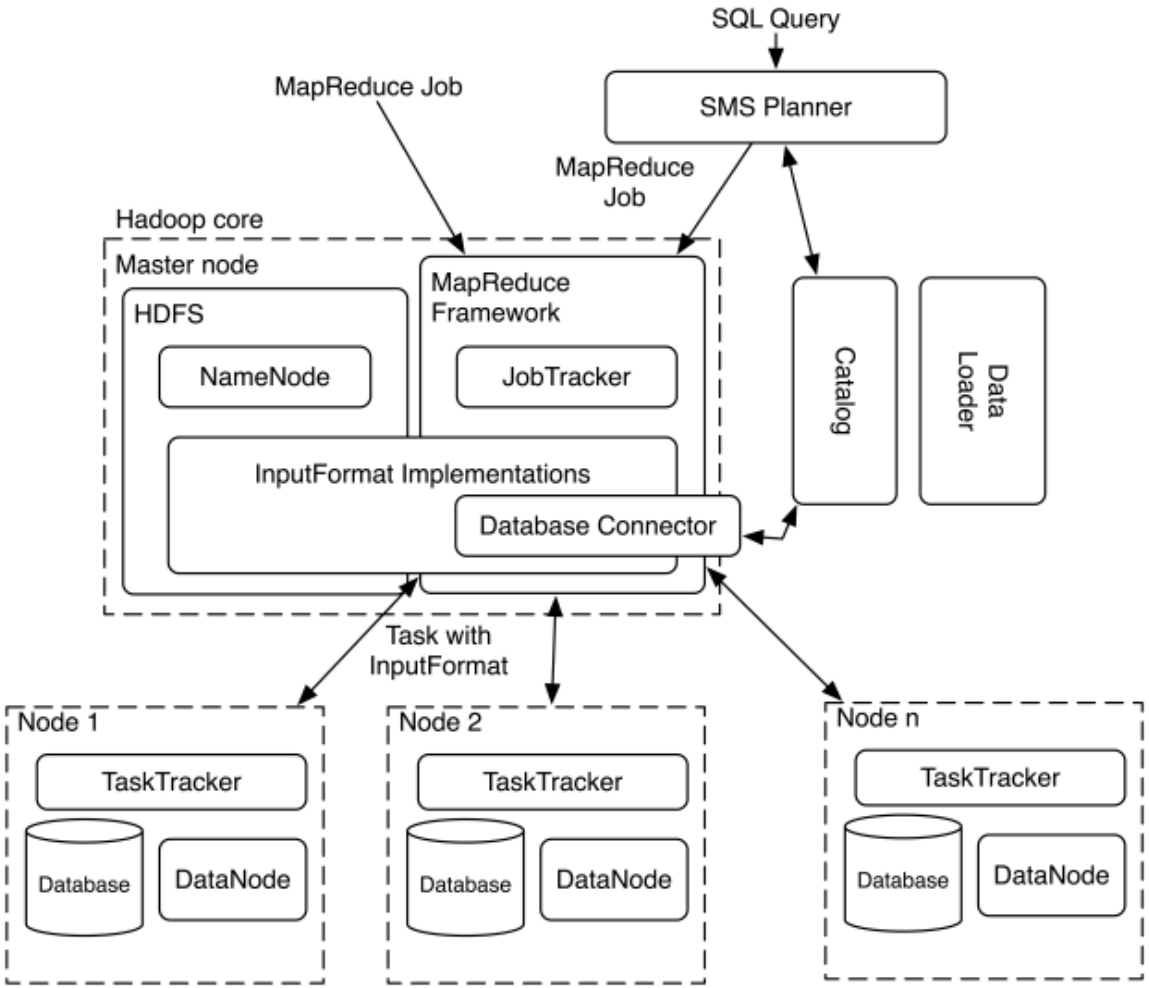
Anforderungen

- Performanz
- Fehlertoleranz
- Heterogene Server
- Flexible / Erweiterbare Abfrageschnittstelle

(Energieeffizienz?)

	Parallele DBs	MapReduce
Performanz	++	?
Fehlertoleranz	–	++
Heterogene Server	–	++
Abfrageschnittstelle	++ (?)	+

Architektur



Datenladephase

1. globaler Hasher (2r, 2w + 1 network copy)
2. Export ins lokale Dateisystem (1r, 1w)
3. lokaler Hasher (1r, 1w)
4. Import in lokale Datenbank (1r, 1w + Indizes)

Datenladephase

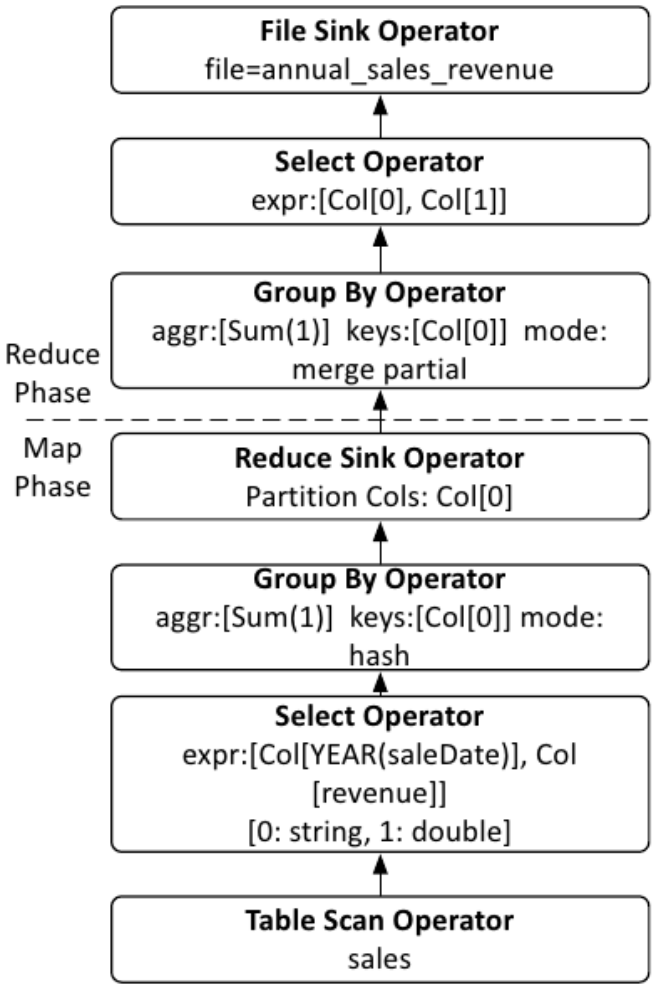
1. globaler Hasher (2r, 2w + 1 network copy)
2. Export ins lokale Dateisystem (1r, 1w)
3. lokaler Hasher (1r, 1w)
4. Import in lokale Datenbank (1r, 1w + Indizes)

5x lesen, 5x schreiben

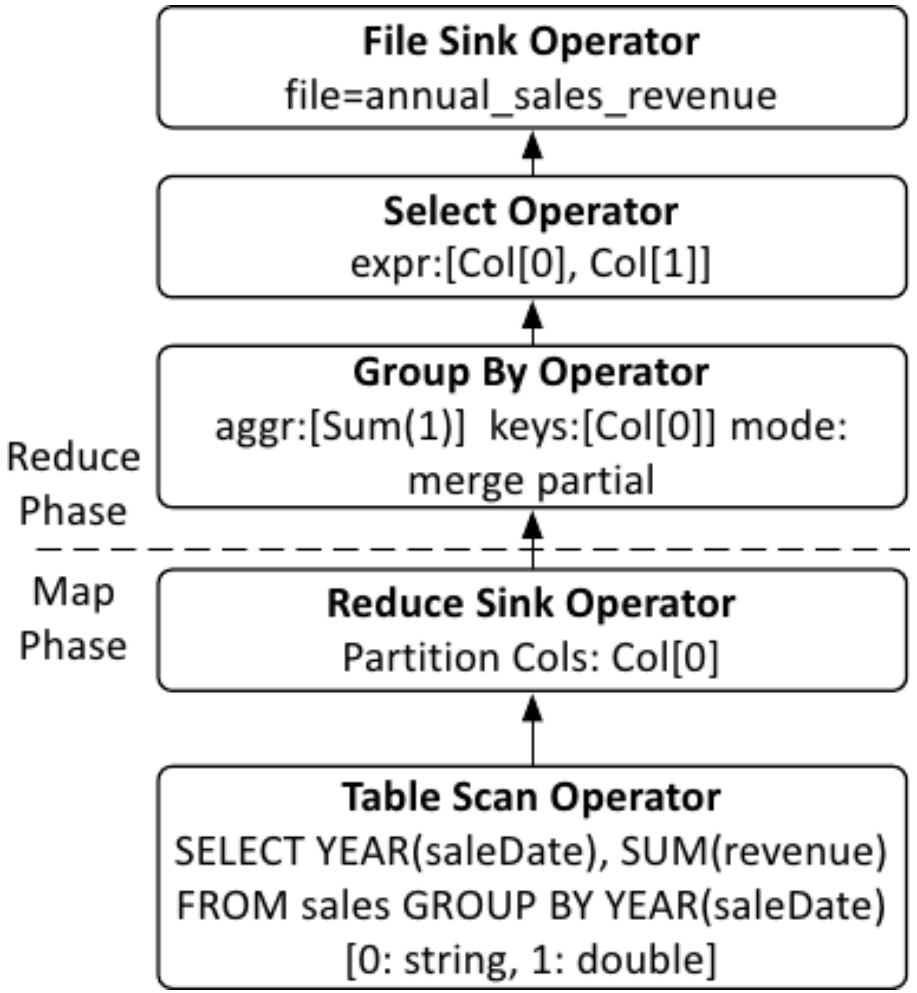
SQL-MapReduce-SQL Planer

```
SELECT YEAR(saleDate), SUM(revenue)
FROM sales GROUP BY YEAR(saleDate)
```

SQL-MapReduce-SQL Planer

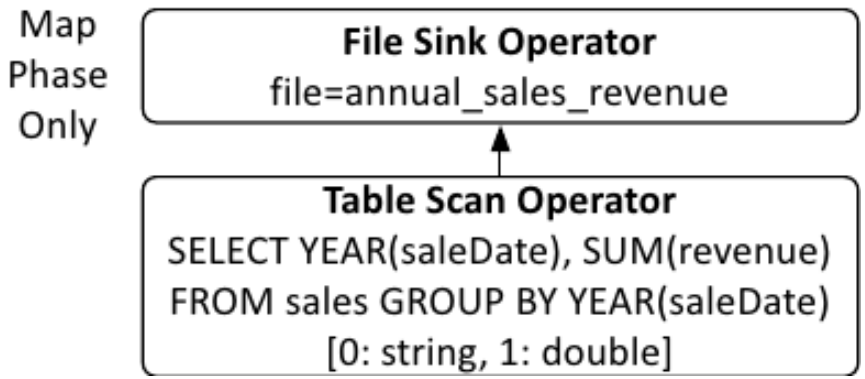


SQL-MapReduce-SQL Planer



SQL-MapReduce-SQL Planer

falls per YEAR(saleDate) partitioniert wurde:



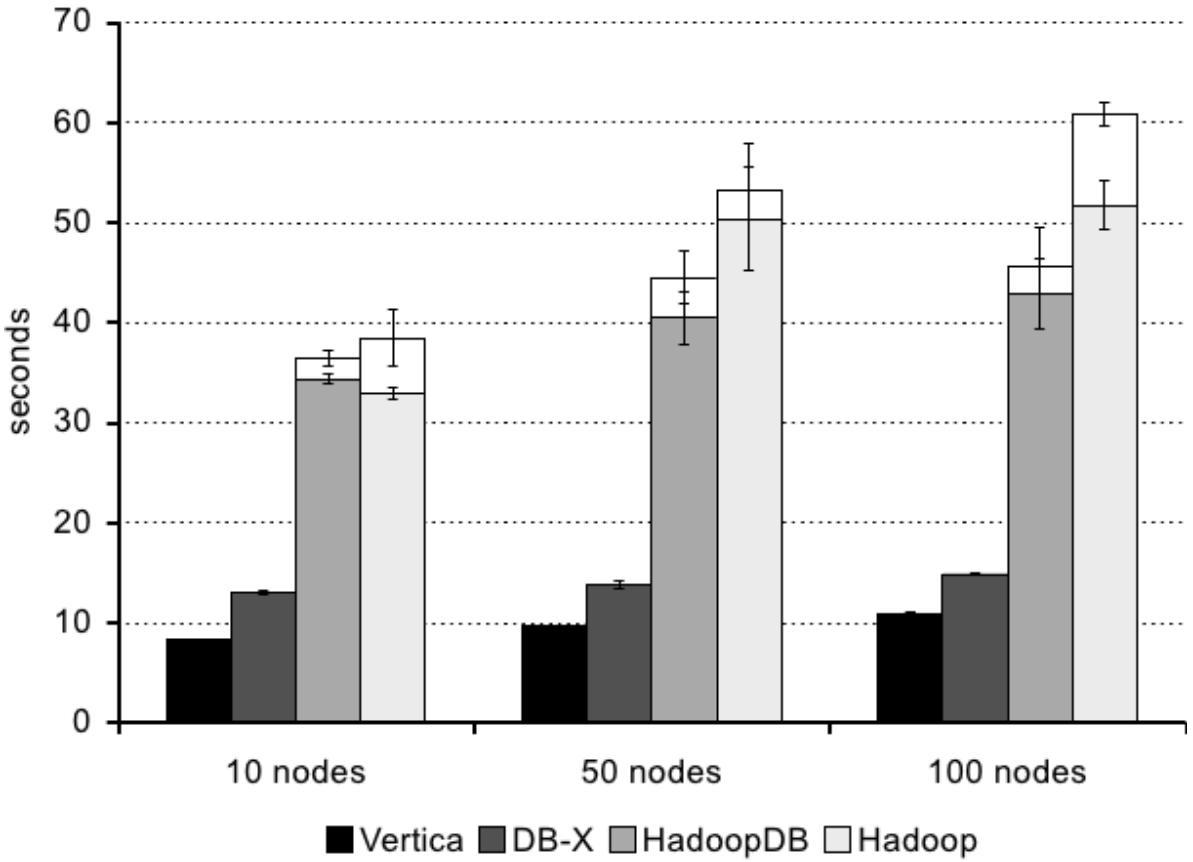
weitere HadoopDB Komponenten

- Datenbank Connectoren
- Catalog:
  - Datenbankverbindungsparameter
  - Metainformationen der Tabellen
  - Speicherorte von Replikationen
  - Partitionseigenschaften

Grep Task

```
SELECT * FROM Data
 WHERE
field LIKE '%XYZ%';
```

Compression?



Schemas der analytischen Tasks

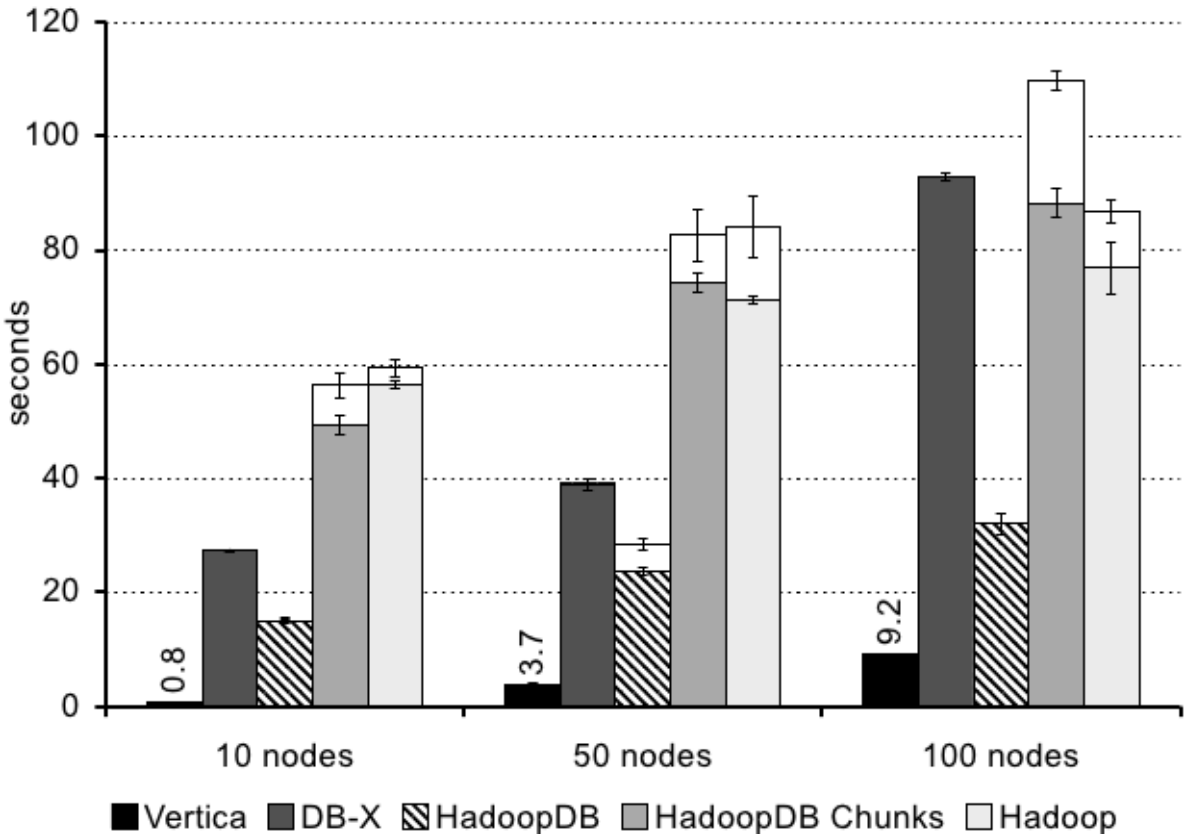
```
CREATE TABLE Documents
( url VARCHAR(100)
      PRIMARY KEY,
  contents TEXT
);

CREATE TABLE Rankings
( pageURL VARCHAR(100)
          PRIMARY KEY,
  pageRank INT,
  avgDuration INT
);
```

```
CREATE TABLE UserVisits
( sourceIP VARCHAR(16),
  destURL VARCHAR(100),
  visitDate DATE,
  adRevenue FLOAT,
  userAgent VARCHAR(64),
  countryCode VARCHAR(3),
  languageCode VARCHAR(6),
  searchWord VARCHAR(32),
  duration INT
);
```

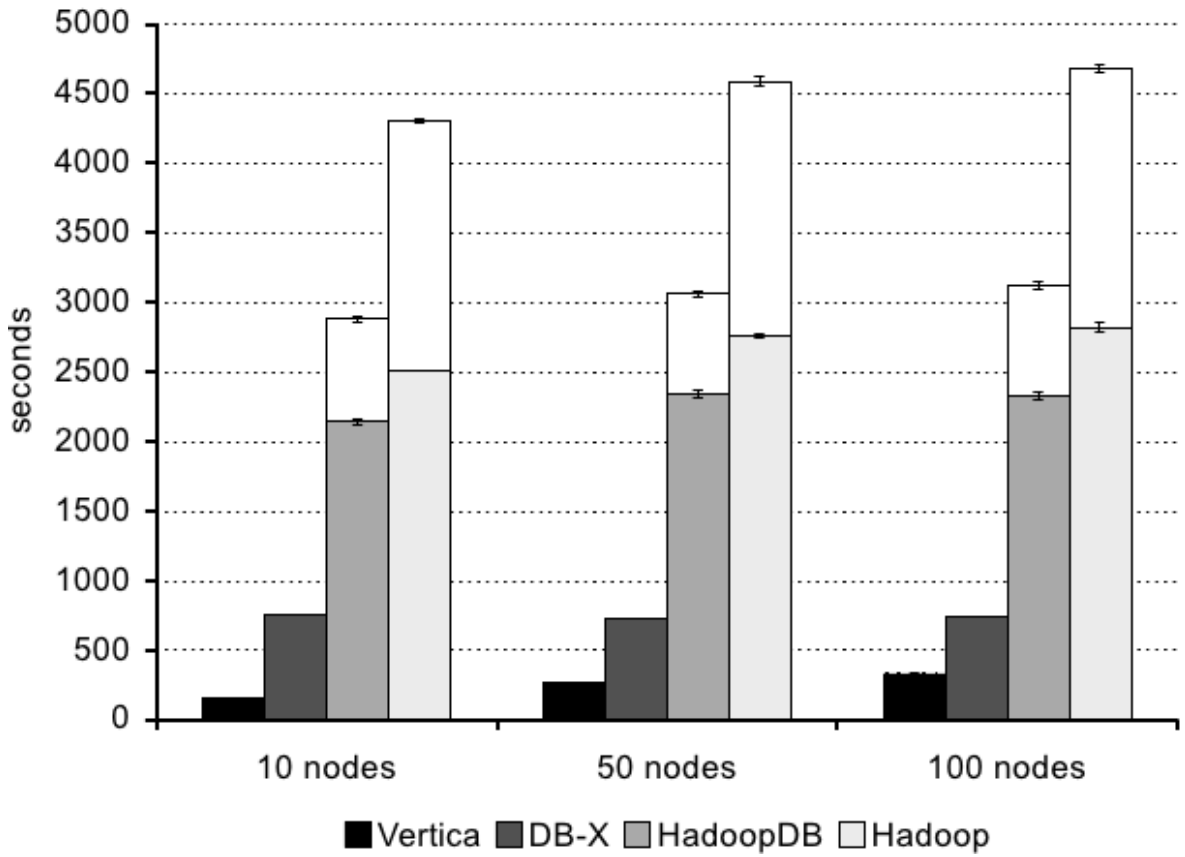
Selection Task

```
SELECT pageUrl,
pageRank
FROM Rankings
WHERE pageRank > 10
```



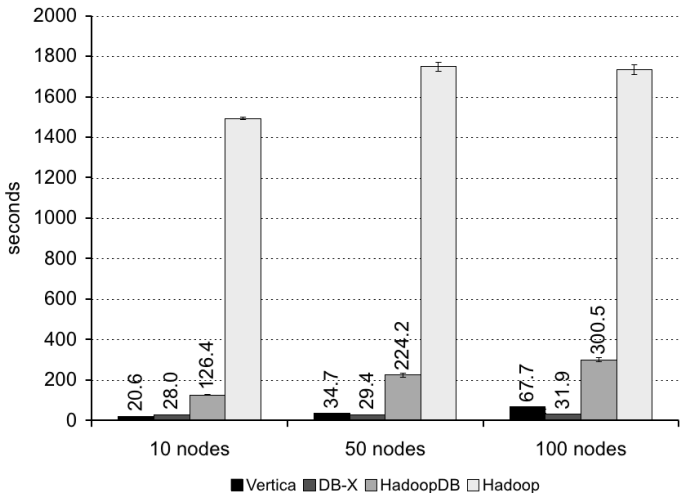
Aggregation Task

```
SELECT sourceIP,
SUM(adRevenue)
FROM UserVisits
GROUP BY sourceIP;
```



Join Task

```
SELECT sourceIP, COUNT(pageRank),
SUM(pageRank), SUM(adRevenue)
FROM Rankings AS R,
     UserVisits AS UV
WHERE
R.pageURL = UV.destURL AND
UV.visitDate BETWEEN
  '2000-01-15' AND '2000-01-22'
GROUP BY UV.sourceIP;
```



Linkgraph-Invertier Task

HTML parsen, Links extrahieren, Inlinks für Seiten zählen.
Musterbeispiel für MapReduce!

Diskussion

- unrealistische Benchmarks
- sehr kleine Anzahl Server
- Fehlbedienung von Hadoop?
- keine richte Fault-Tolerance
- separate Read-Only Datenbank

Praktische Evaluation - Feedback

- <10 unabhängige Google Treffer
- 15 (4 letztes Jahr) Mails auf PostgreSQL Liste
- 22 Revisions in Subversion

Praktische Evaluation - Feedback

- $<$10 unabhängige Google Treffer
- 15 (4 letztes Jahr) Mails auf PostgreSQL Liste
- 22 Revisions in Subversion

. . . obwohl es **Hadoop**DB heißt!

Praktische Evaluation

Kombination von mind. 2 komplizierten Systemen:
Hadoop (+Hive) + Datenbank (PostgreSQL)

Überblick

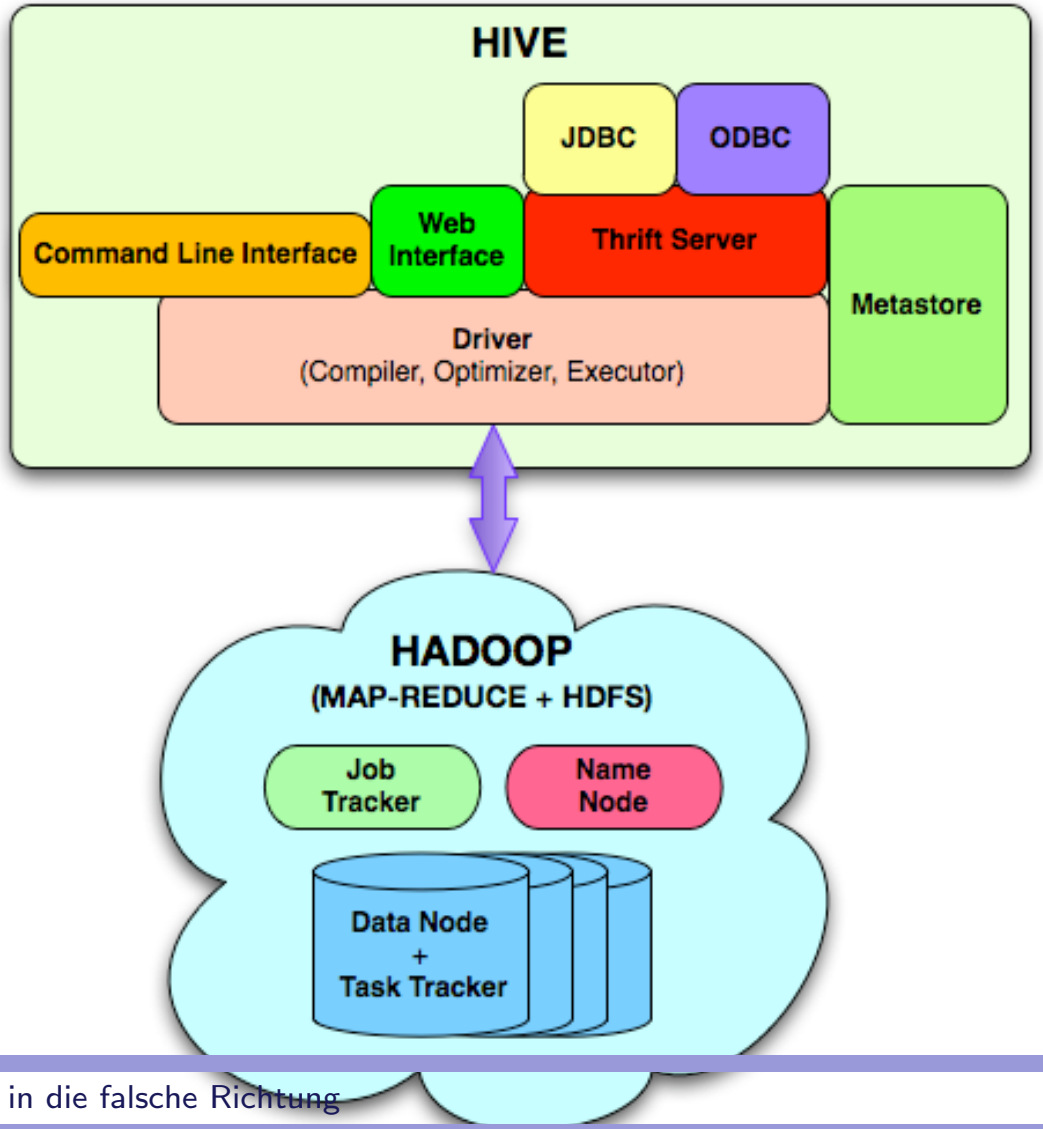
- entwickelt von FaceBook
- SQL ähnliche Abfragesprache
- Ausführung als DAG von MapReduce Jobs

Datenmodell

- Tabellen
- Partitionen
- Buckets

$$/wh/\underbrace{daily\_status}_{\text{table name}}/\underbrace{ds = 20090101/ctry = US}_{\text{nested partitions}}/\underbrace{0001}_{\text{bucket}}$$

Architektur



Hive-Metastore

- Schema: browse, query parsing
- (geplant) Statistiken: Optimierung

Demnächst: HCatalog

Hive-Compiler

1. Query String → Parsebaum
2. → interne Queryrepresentation +
   - Verifikation gegen Metastore
   - SELECT * expandieren
   - Typcheck bzw. conversion
3. → logischer Operatorbaum
4. Optimierer
5. → MapReduce jobs

Vergleich mit SQL

- standard Datentypen + Array, Maps + user programed
- kein update, delete, insert into
- user defined (aggregation) functions
- Mehrere INSERTs aus einem SELECT

Vergleich mit SQL

- standard Datentypen + Array, Maps + user programed
- kein update, delete, insert into
- user defined (aggregation) functions
- Mehrere INSERTs aus einem SELECT

```
FROM (SELECT ...) subq1
INSERT TABLE a SELECT subq1.xy, subq1.xz
INSERT TABLE b SELECT subq1.ab, subq1.xy
```