HIVE

Data Warehousing & Analytics on Hadoop

Joydeep Sen Sarma, Ashish Thusoo
Facebook Data Team
Why Another Data Warehousing System?

- Problem: Data, data and more data
  - 200GB per day in March 2008 back to 1TB compressed per day today
- The Hadoop Experiment
- Problem: Map/Reduce is great but every one is not a Map/Reduce expert
  - I know SQL and I am a python and php expert
- So what do we do: HIVE
What is HIVE?

- A system for querying and managing structured data built on top of Map/Reduce and Hadoop
- We had:
  - Structured logs with rich data types (structs, lists and maps)
  - A user base wanting to access this data in the language of their choice
  - A lot of traditional SQL workloads on this data (filters, joins and aggregations)
  - Other non SQL workloads
Data Warehousing at Facebook Today

Web Servers → Scribe Servers

Oracle RAC → Hive on Hadoop Cluster

Hive on Hadoop Cluster → Filers

Filers → Federated MySQL
HIVE: Components

- HDFS
- Hive CLI
- DDL
- Queries
- Browsing
- MetaStore
- Thrift API
- Parser
- Planner
- Execution
- SerDe
- Thrift, Jute, JSON...
- Hive QL
- Mgmt. Web UI
- Map Reduce
- HDFS
Data Model

//hive/clicks
//hive/clicks/ds=2008-03-25
//hive/clicks/ds=2008-03-25/0
Dealing with Structured Data

- **Type system**
  - Primitive types
  - Recursively build up using Composition/Maps/Lists

- **Generic (De)Serialization Interface (SerDe)**
  - To recursively list schema
  - To recursively access fields within a row object

- **Serialization families implement interface**
  - Thrift DDL based SerDe
  - Delimited text based SerDe
  - You can write your own SerDe

- **Schema Evolution**
MetaStore

- Stores Table/Partition properties:
  - Table schema and SerDe library
  - Table Location on HDFS
  - Logical Partitioning keys and types
  - Other information

- Thrift API
  - Current clients in Php (Web Interface), Python (old CLI), Java (Query Engine and CLI), Perl (Tests)

- Metadata can be stored as text files or even in a SQL backend
Hive CLI

- **DDL:**
  - create table/drop table/rename table
  - alter table add column

- **Browsing:**
  - show tables
  - describe table
  - cat table

- Loading Data
- Queries
Hive Query Language

- **Philosophy**
  - SQL like constructs + Hadoop Streaming

- **Query Operators in initial version**
  - Projections
  - Equijoins and Cogroups
  - Group by
  - Sampling

- **Output of these operators can be:**
  - passed to Streaming mappers/reducers
  - can be stored in another Hive Table
  - can be output to HDFS files
  - can be output to local files
Hive Query Language

- Package these capabilities into a more formal SQL like query language in next version
- Introduce other important constructs:
  - Ability to stream data thru custom mappers/reducers
  - Multi table inserts
  - Multiple group bys
  - SQL like column expressions and some XPath like expressions
  - Etc..
Joins

FROM page_view pv JOIN user u ON (pv.userid = u.id)
INSERT INTO TABLE pv_users
SELECT pv.*, u.gender, u.age
WHERE pv.date = 2008-03-03;

Outer Joins

FROM page_view pv FULL OUTER JOIN user u ON (pv.userid = u.id)
INSERT INTO TABLE pv_users
SELECT pv.*, u.gender, u.age
WHERE pv.date = 2008-03-03;
FROM pv_users
INSERT INTO TABLE pv_gender_uu
    SELECT pv_users.gender, count(DISTINCT pv_users.userid)
    GROUP BY(pv_users.gender)
INSERT INTO TABLE pv_ip_uu
    SELECT pv_users.ip, count(DISTINCT pv_users.id)
    GROUP BY(pv_users.ip);
FROM (  
  FROM pv_users  
  SELECT TRANSFORM(pv_users.userid, pv_users.date) USING 'map_script'  
  AS (dt, uid)  
  CLUSTER BY (dt) map  
) map  
INSERT INTO TABLE pv_users_reduced  
  SELECT TRANSFORM(map.dt, map.uid) USING 'reduce_script'  
    AS (date, count);
FROM pv_users
INSERT INTO TABLE pv_gender_sum
    SELECT pv_users.gender, count_distinct(pv_users.userid)
    GROUP BY(pv_users.gender)
INSERT INTO DIRECTORY '/user/facebook/tmp/pv_age_sum.dir'
    SELECT pv_users.age, count_distinct(pv_users.userid)
    GROUP BY(pv_users.age)
INSERT INTO LOCAL DIRECTORY '/home/me/pv_age_sum.dir'
    FIELDS TERMINATED BY ‘,’ LINES TERMINATED BY \013
    SELECT pv_users.age, count_distinct(pv_users.userid)
    GROUP BY(pv_users.age);
Hadoop Usage @ Facebook

- **Types of Applications:**
  - **Summarization**
    - Eg: Daily/Weekly aggregations of impression/click counts
  - **Ad hoc Analysis**
    - Eg: how many group admins broken down by state/country
  - **Data Mining (Assembling training data)**
    - Eg: User Engagement as a function of user attributes
Usage statistics:
- Total Users: ~140 (about 50% of engineering) in the last 1 ½ months
- Hive Data (compressed): 80 TB total, ~1TB incoming per day
- Job statistics:
  - ~1000 jobs/day
  - ~100 loader jobs/day
Some problems:
- No Fair Sharing: Big tasks can hog the cluster
- No snapshots: What if a software bug corrupts the NameNode transaction log

Solutions:
- Simple fair sharing (Matie Zaharia)
- Investigating Snapshots (Dhrubha Bortharkur)
Conclusion

- Soon to be checked into hadoop trunk
- Release available in hadoop version 0.19
- People:
  - Suresh Anthony
  - Zheng Shao
  - Prasad Chakka
  - Pete Wyckoff
  - Namit Jain
  - Raghu Murthy
  - Joydeep Sen Sarma
  - Ashish Thusoo