GRUTER
Enjoy Connecting

SQL-on-Hadoop 그리고 Tajo

Tech Planet 2013
11월 14일

소개

- **최현식 (Hyunsik Choi)**
  - Director of Research, Gruter Corp (2013. 6 ~)
  - PhD (Computer Science & Engineering, 2013. 8)
  - Apache Tajo (incubating) PPMC member & committer (2013.3 ~)
  - Apache Giraph PMC member and committer (2011. 8 ~)

- **Contacts**
  - Email: hyunsik AT apache.org
  - Linkedin: http://linkedin.com/in/hyunsikchoi/
  - Twitter: @hyunsik_choi

SQL-on-Hadoop 시스템

- SQL-on-Hadoop 시스템이란?
  - HDFS에 저장된 데이터에 대한 SQL 처리를 제공하는 시스템
  - 탈 MapReduce 모델 & 프레임워크
    - 새로운 분산 처리 모델 & 프레임워크를 기반
- **다양한 설계 목표 지향 (외형적으로 비슷해 보이지만 다름)**

왜 SQL-on-Hadoop 시스템?

- **MapReduce의 한계**
  - 데이터 처리 모델상의 한계 (관계형 처리를 위해 고안된 것이 아님)
    - Map -> Shuffle -> Reduce
  - Pig, Hive는 MapReduce가 제공하는 기능 이상의 최적화 불가능
  - 느린 속도 (초기화 및 스케쥴링 지연)
    - MapReduce가 느린 것이 아님, 하둡 구현이 느린 것

- **높은 learning curve 및 Legacy 시스템들과 호환성 문제**
  - MapReduce는 어렵고, 개발 노력이 많이 들고, 성능 보장이 어려움
    - MapReduce의 직접 사용은 점차 줄고 있음
  - HiveQL != SQL

- **Ad-hoc 질의에 대한 속도 문제로 DBMS 병행 사용 불가피**
  - 부담이 큰 ETL의 문제 (HDFS <-> DBMS)
  - 추가적인 스토리지 공간 필요, 비싼 DBMS 라이센스 비용의 벽

SQL-on-Hadoop 시스템 분류

- **주요 기준**
  - Long time 질의를 지원하는가? 아닌가?
    - '특정 시스템이 이 워크로드에 적합한가'를 판단하게 하는 기준

- **Long time 질의 지원를 위한 설계 시 주요 고려사항**
  - Fault tolerance
  - Dynamic scheduling

> *"Machine failures are common in a large cluster"*
> *- Jeff Dean*

Fault Tolerance

- 질의 처리 중 발생하는 오류를 핸들링하여 질의를 완료하는 기능
- Long time 질의 (수 십분 이상) 에 필수적
- 하나의 질의를 작은 단위의 테스크로 나누어 처리하고 오류의 범위를 테스크로 한정 및 테스크 재시작
- 단, 중간 데이터 materialization 요구
  - 디스크 부하 유발

Throughput  Fault Tolerance

Trade-off 관계

Dynamic Scheduling

- Fixed scheduling
  - 작업 시작 시 균등하게 클러스터 노드들에 분할된 작업을 한번에 할당
- Dynamic scheduling
  1. 각 노드에 노드가 한번에 실행할 수 있는 테스크를 우선적으로 분배
  2. 노드가 할당 받은 테스크가 끝나면 다시 할당
- **실제 클러스터의 환경적 특성**
  - 같은 사양의 노드라도 알수 없는 이유로 성능 차이 발생
  - 점진적으로 확장되는 경우 이종의 클러스터 노드로 구성

일부 느린 노드가 분산 작업의 성능을 결정

SQL-on-Hadoop 시스템 분류

Name	Fault Tolerance	Dynamic Scheduling	Long time 질의 적합	클러스터 자원 대비 큰 용량 처리
Tajo	O	O	O	O
Impala	X	X	X	X
Stinger (Hive)	O	O	O	O
Drill	?	?	?	?
Presto	X	X	X	X

표 1. 설계에 따른 비교표

SQL-on-Hadoop 시스템 분류

Data Warehouse System

Query Engine

SQL-on-Hadoop 시스템 분류

- **Data Warehouse System**
  - 수 시간이상 걸리는 질의 수행 가능
  - ETL 작업
    - 데이터 변환 및 데이터 노이즈 제거
    - 데이터 파티셔닝
  - Data integration
    - 다수 데이터 소스에 대한 통합
  - Ad-hoc query

- **Query Engine**
  - 수 초에서 수 분까지의 질의 수행에 최적화
  - 빠른 응답을 가지는 ad-hoc query
  - 중간 데이터 크기와 자원에 따라 질의가 다소 제약됨
    - 인메모리 처리 구조와 파이프라인 연산 방식에 따라

Presto 소개

- Facebook에서 개발하여 오픈소스로 공개
- 수 초에서 수 분이 걸리는 질의 유형을 타겟으로 설계
- No Fault tolerance
- 빠른 응답과 online query processing
  - 우선적으로 처리된 결과가 반환됨
- 일부 질의 유형에 대해 approximate query 지원
- Hive의 보완 시스템으로 개발
- 모든 연산자는 파이프라이닝 (pipelining)과 데이터 전송은 스트리밍
- Facebook 사례
  - 누적된 300 PB Data Warehouse, 일일 30k 질의, 하루에 1PB 처리

"90% of jobs on a Facebook cluster have input sizes under 100 GB"
- Raja Appuswamy et al. "Nobody ever got fired for buying a cluster",
Microsoft Research

Tajo란?

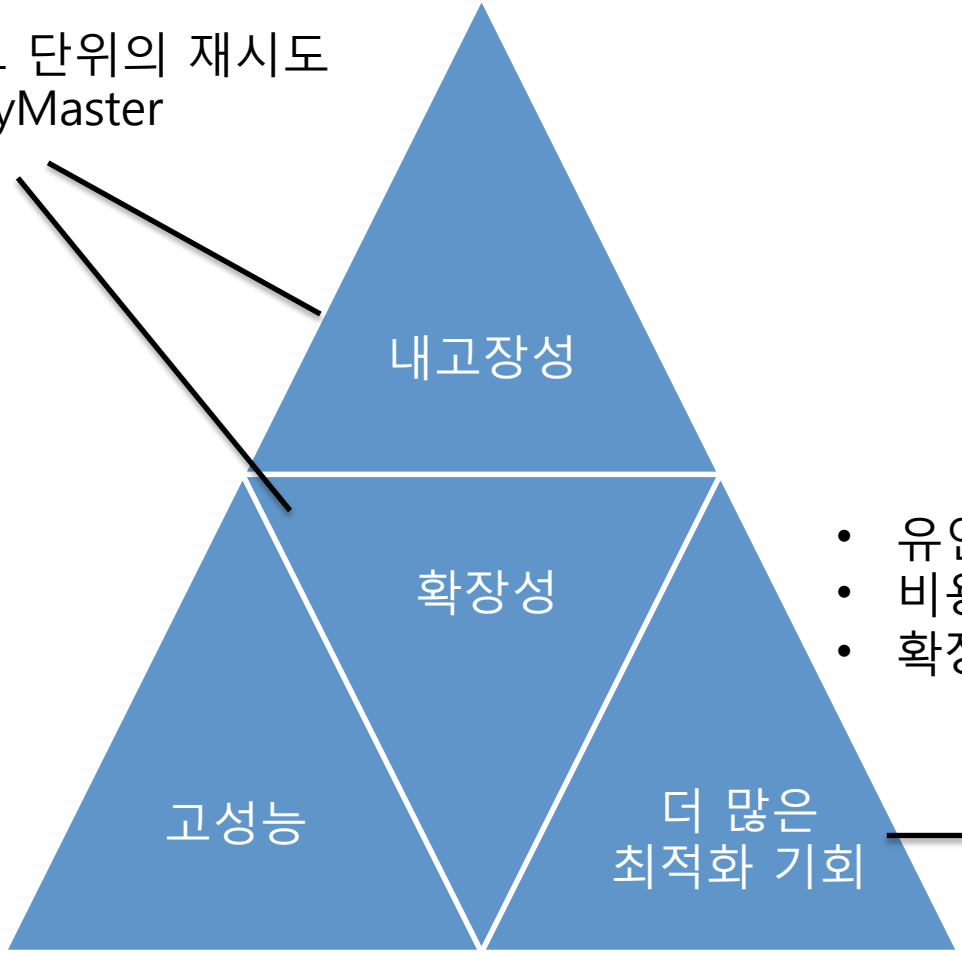
- Tajo
  - 하둡 기반의 대용량 데이터웨어 하우스 시스템
  - 2010년 부터 리서치 프로토타입으로 개발 시작
  - 아파치 인큐베이션 프로젝트 (올해 3월 ASF 인큐베이션 채택)

- Features
  - SQL 표준 호환
  - 질의 전체를 분산 처리
  - HDFS가 기본 스토리지
  - 관계형 모델 (Nested model로 확장 계획 중)
  - ETL 뿐만 아니라 low-latency 질의(100 ms ~ hours)
  - UDF 지원

- News
  - 0.2-incubating: 11월 중 릴리즈
  - 0.8-incubating: 12월 릴리즈 계획

Tajo의 주요 설계 원칙

- 실패한 테스크 단위의 재시도
- 질의 별 QueryMaster

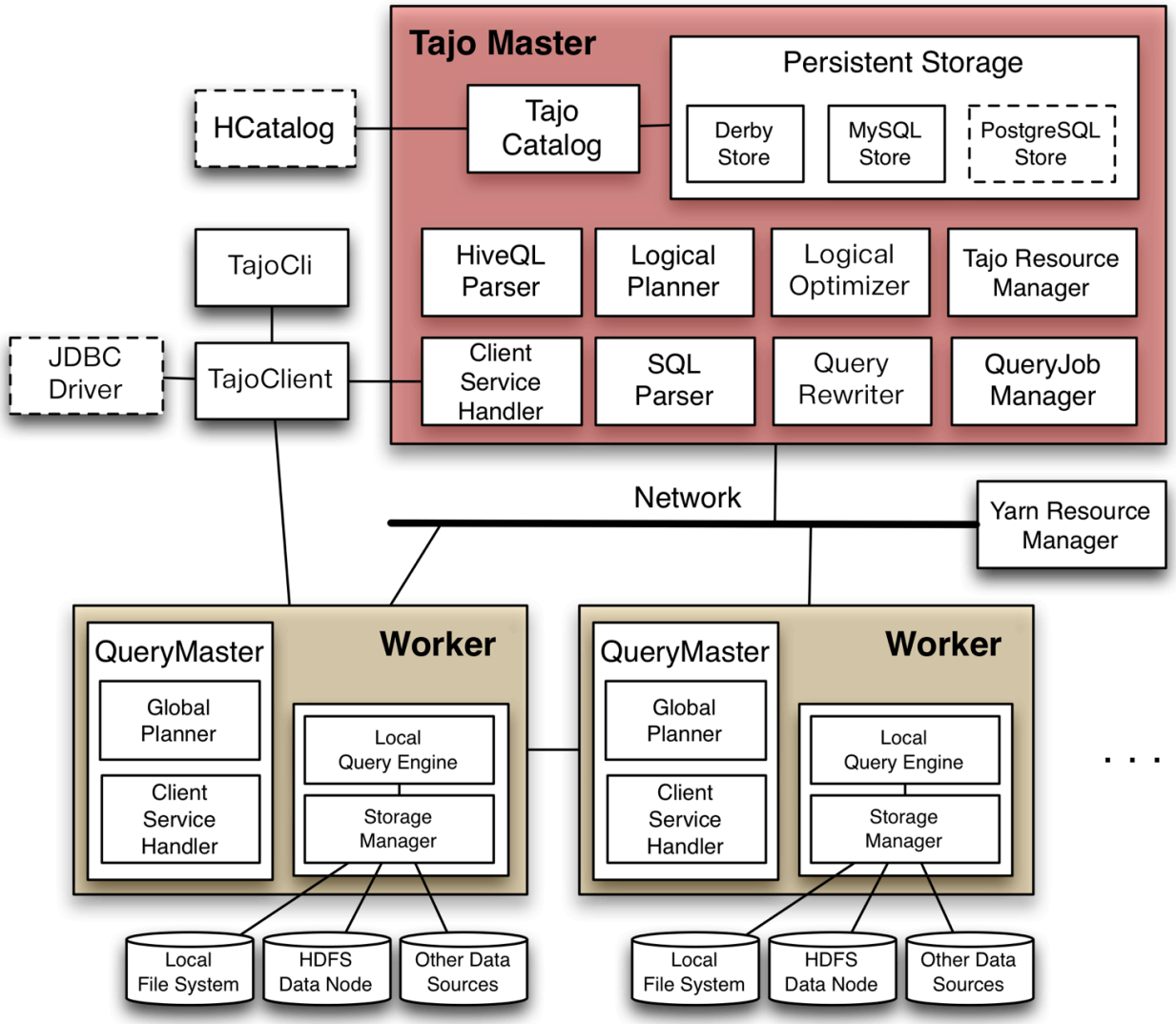


- 유연한 분산 처리 모델
- 비용 기반 최적화
- 확장 가능한 rewrite rule

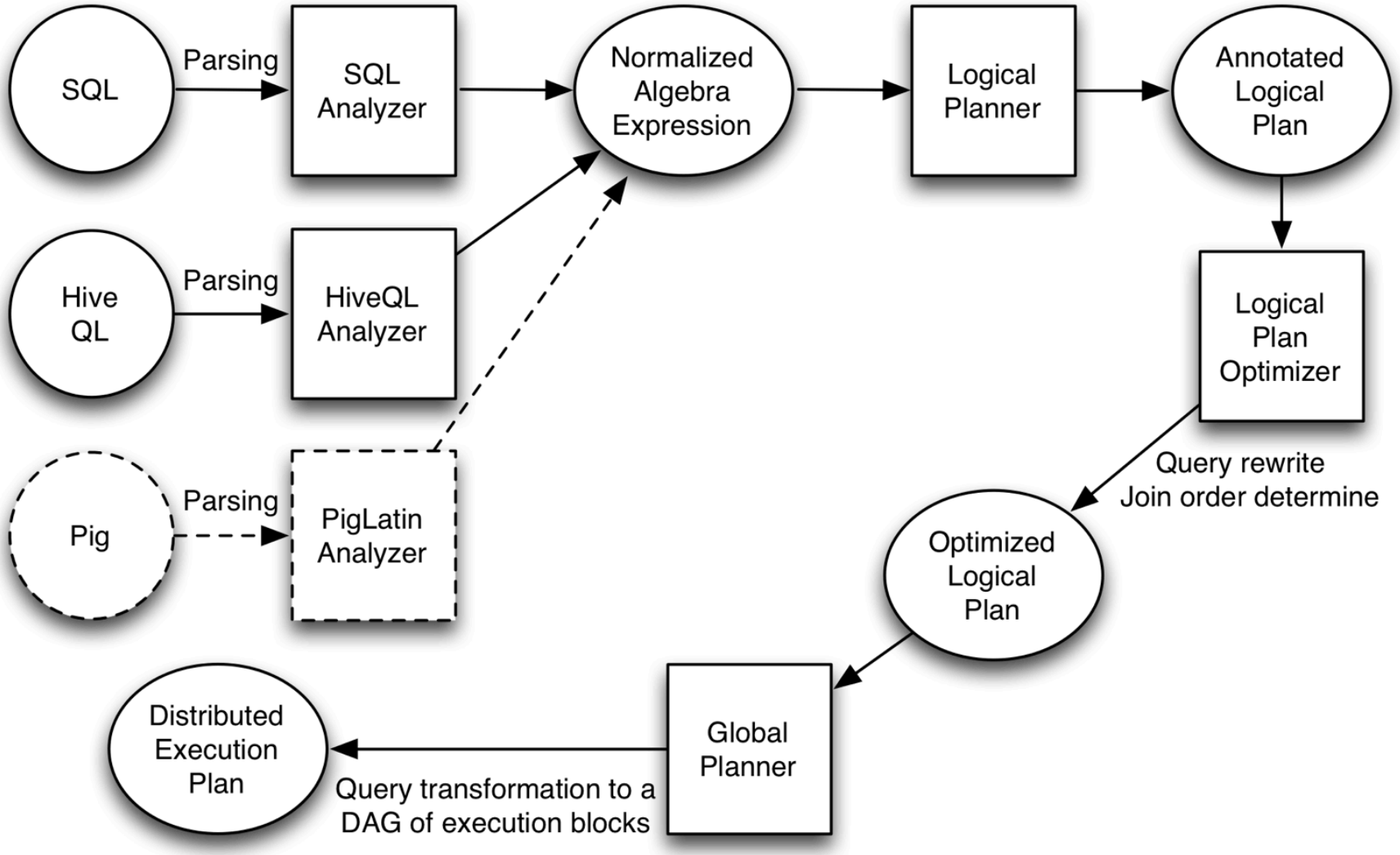
아키텍쳐

- 기본 스토리지는 HDFS, 다양한 데이터 소스를 가질 수 있음
- 마스터 워커 클러스터 모델 + 각 질의 별 Query Master
  - Single point of failure를 제거하고 병목 지점 제거 고려
- 타조 마스터 (Tajo Master)
  - 항시 대기하며 DDL과 같은 분산 처리 없이 처리 가능한 질의 수행
  - Client API를 RPC로 제공
  - 쿼리 파서 및 Query Master들을 관리
  - 카탈로그 서버 내장 (독립 실행 가능)
  - 클러스터 자원 관리
- Query Master (각 질의 별 동작)
  - 논리 실행 계획을 분산 실행 계획으로 변환
  - 분산 실행 계획을 제어
  - 테스크 스케쥴링
- Tajo Worker
  - Storage Manager
  - Local Query Engine

Tajo 아키텍쳐


Tajo Master
Persistent Storage
HCatalog
Tajo Catalog
Derby Store
MySQL Store
PostgreSQL Store
TajoCli
HiveQL Parser
Logical Planner
Logical Optimizer
Tajo Resource Manager
JDBC Driver
TajoClient
Client Service Handler
SQL Parser
Query Rewriter
QueryJob Manager
Network
Yarn Resource Manager
QueryMaster
Worker
Global Planner
Client Service Handler
Local Query Engine
Storage Manager
QueryMaster
Worker
Global Planner
Client Service Handler
Local Query Engine
Storage Manager
Local File System
HDFS Data Node
Other Data Sources
Local File System
HDFS Data Node
Other Data Sources

질의 플래닝 과정


SQL
Parsing
SQL Analyzer
Normalized Algebra Expression
Logical Planner
Annotated Logical Plan
Hive QL
Parsing
HiveQL Analyzer
Logical Plan Optimizer
Pig
Parsing
PigLatin Analyzer
Query rewrite
Join order determine
Optimized Logical Plan
Distributed Execution Plan
Global Planner
Query transformation to a DAG of execution blocks

Tajo 분산 처리 모델

- **질의 = 비순환 방향 그래프**
  - 정점, 방향을 가진 간선으로 구성
  - 사이클 없음
- **정점은 데이터 처리 단계를 표현**
  - 논리 연산자 그래프 - A DAG of Logical Operators
  - Enforcer (properties to force physical planning)
- **간선 (edge) = 정점간 데이터 흐름 표현**
  - 전송 방법 (Pull, Push, File)
  - 셔플 방법 (range, hash, and ..)
  - 파티션 개수

데이터 셔플

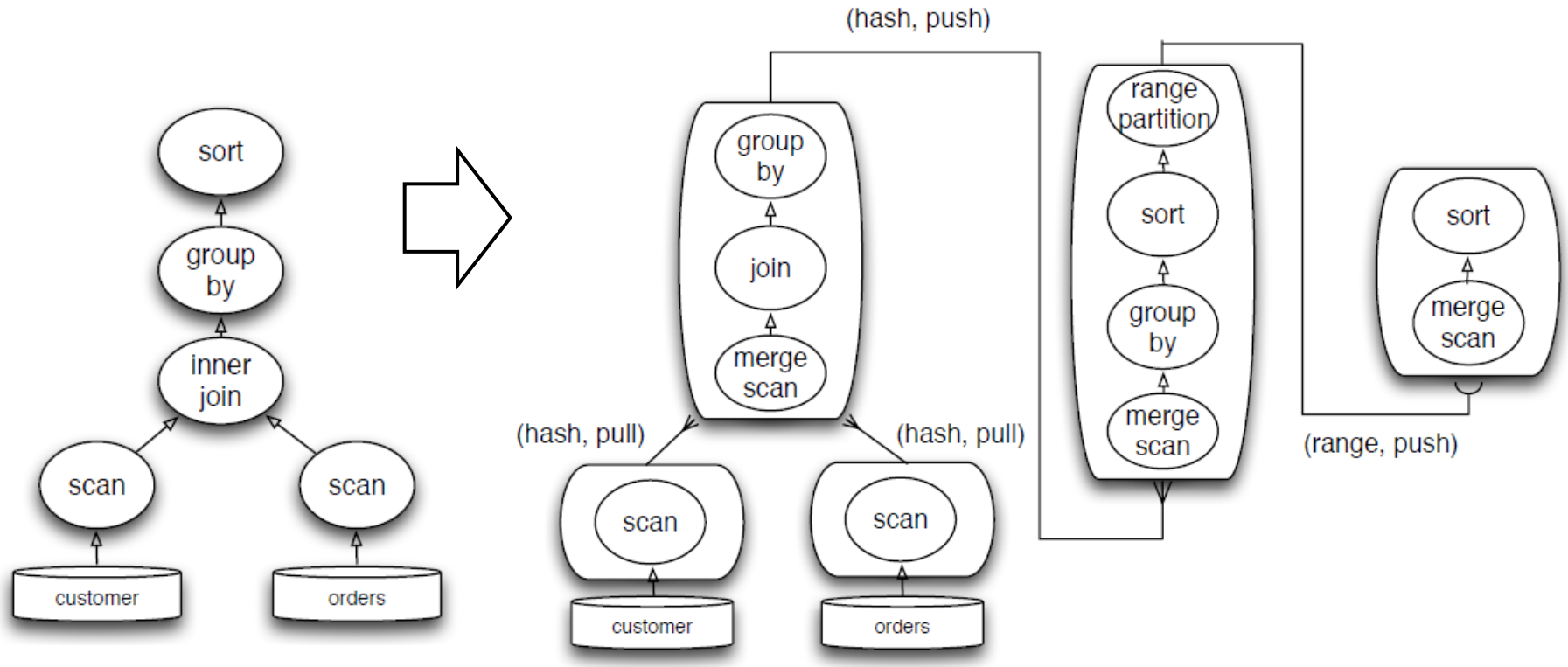
- **셔플 방법**
  - Hash
    - 해쉬 파티셔닝 (해쉬 키를 이용한 중간 데이터 파티셔닝)
  - Range
    - 범위 파티셔닝 (각 노드에 범위를 할당, 범위에 만족하도록 데이터 파티셔닝)
    - 주로 분산 정렬에 사용

- **전송 방법**
  - Pull
    - Step 1: 로컬 디스크에 저장
    - Step 2: 다른 노드들이 HTTP로 끌어감
  - Push (12월 릴리즈 0.8에 추가됨)
    - 중간 데이터를 다음 노드에게 직접 스트리밍 (no materialization)
    - Push되는 양쪽 실행 단계는 파이프라이닝 가능

Tajo 분산 처리 모델에 기반한 분산 실행 계획의 예제

논리 실행 계획
(logical plan)

분산 실행 계획
(distributed execution plan)



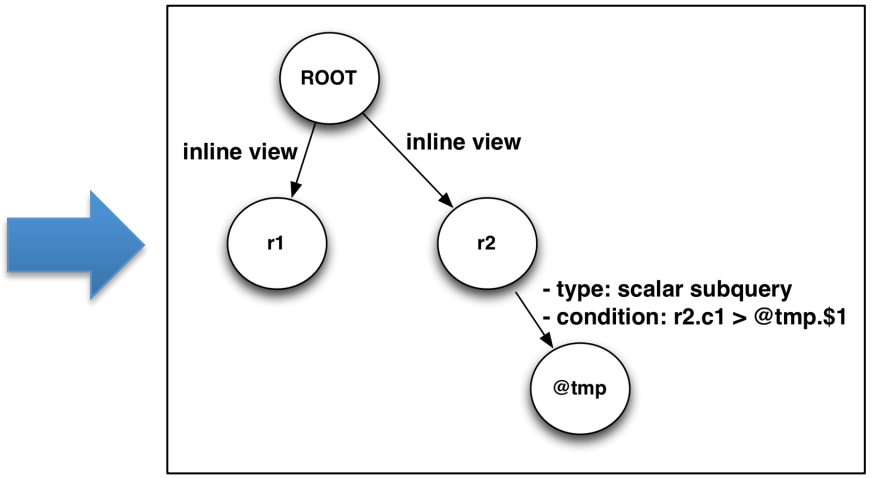

```
select col1, sum(col2) as total, avg(col3) as average from r1, r2
where r1.col1 = r2.col2 group by col1 order by average;
```

질의 최적화

- Cost-based Join Optimization (Greedy Heuristic)
  - 사용자가 최선의 조인 순서를 추측하는 수고 제거

- 확장 가능한 rewrite rule 엔진
  - rewrite rule 인터페이스 제공과 다양한 유틸리티 제공
    - Query block 그래프 자료구조 (블럭간의 관계 표현)
    - 조인 그래프 자료구조 (조인 조건들을 기준으로 그래프 표현)
    - 그 외의 대수관련 유틸리티

```
SELECT … from
 ( SELECT … from A) as r1
 join
 ( SELECT … from B
   WHERE c1 > (SELECT
 ) as r2
```



Query block 그래프 변환

질의 최적화

- **점진적 질의 최적화 (Progressive Query Optimization)**
  - 실행 시간 통계 수집
    - 실행 시간에 Min, Max 값, 중간 데이터 크기 수집
  - 분산 정렬을 위한 범위 분할 (range partitioning)의 적절한 파티션 범위, 개수 등을 런타임에 조정
  - 분산 조인, 그룹바이를 위한 파티션 개수를 런타임에 조정
  - 논리 연산자 단위의 재최적화 (향후 계획)
    - 실행 중 수집한 통계 정보를 바탕으로 조인 순서 다시 최적화
    - 전송 방법 (transmission)도 실행 중 조정

현재 프로젝트 상태

- SQL 지원
  - 표준 부분: ANSI SQL 2003 compliance
  - 비표준 부분: PostgreSQL 호환
    - 예) 함수 regexp_replace, split_part,…
  - 미지원 및 향후 지원 계획
    - Outer Join (v0.8), Exists (v1.0), In Subquery (v1.0)
- 클러스터 노드들에 대한 분산 조인, 그룹바이, 정렬 제공
- Blocking/Asynchronous Java Client API
- JDBC 드라이버 (개발 브렌치)

현재 프로젝트 상태

- Tajo Catalog
  - 테이블의 통계 정보 (용량, row 수)
  - Derby와 MySQL을 Catalog 스토어로 가능
  - HCatalog를 통해 Hive Meta 엑세스 (개발 브렌치)
    - Hive 테이블을 Tajo에서 엑세스 가능
  - 편리한 백업 및 복구 유틸리티 제공 (tajo_dump)

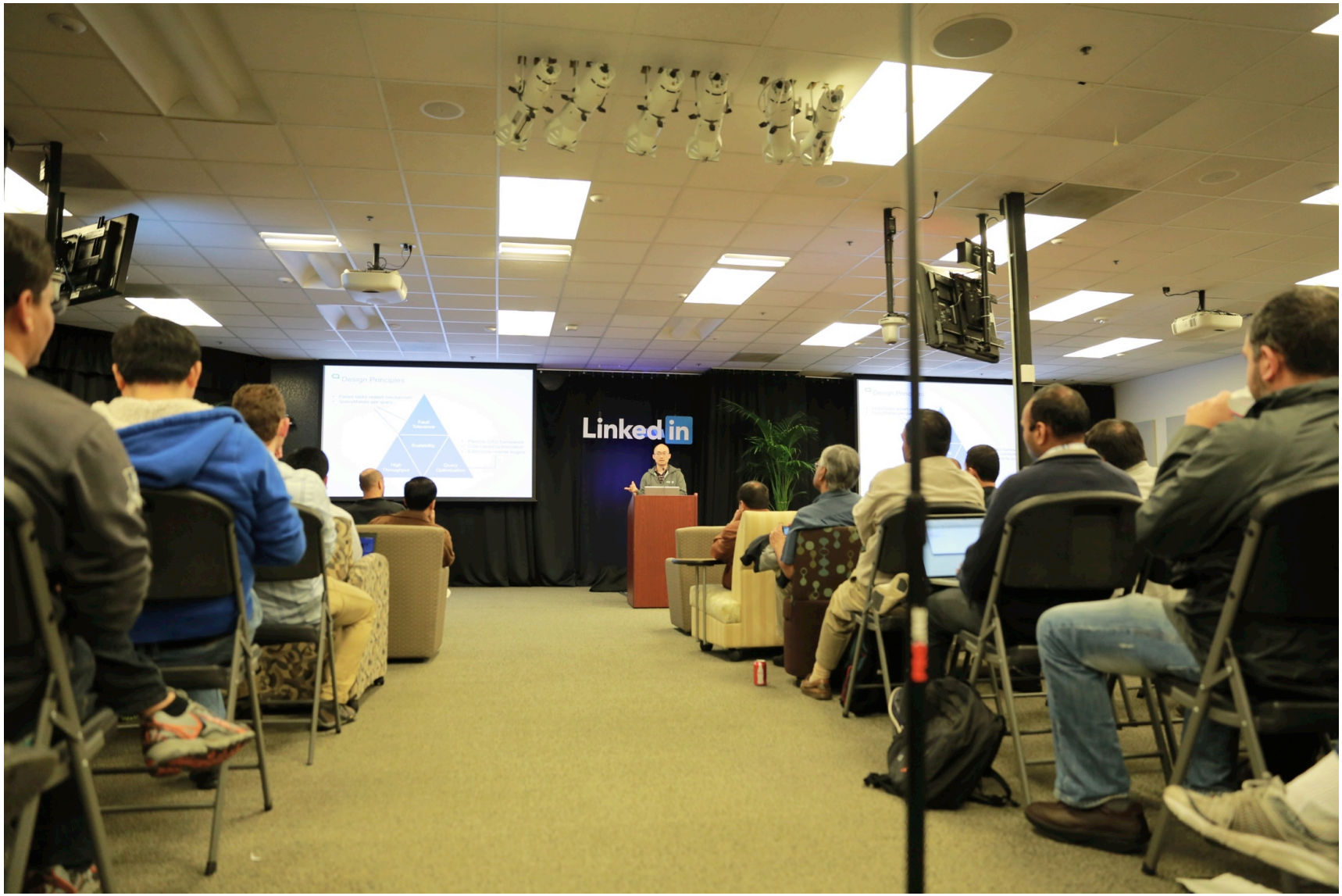
```
-- 카탈로그 백업
$ bin/tajo_dump [tbname] > backup.sql
or
$ bin/tajo_dump –a > backup_all.sql

-- 카탈로그 복구
$ bin/tsql –f backup.sql
```

현재 프로젝트 상태

- **파일 지원**
  - CSV format
  - RowFile (row store 파일 구조)
  - RawFile (for local disk/network materialization)
  - RCFile (Text/Binary (de)serializer 제공 및 Hive 호환) (개발 브렌치)
  - Parquet (다음 릴리즈 v0.8에서 제공)
- **사용자 파일 포맷을 위한 scanner/appender 인터페이스 제공**
- **대폭 향상된 스캔 성능**
  - 클러스터 노드와 각 노드의 디스크 부하까지 고려하여 테스크를 동적으로 할당 (dynamic scheduling)
  - 디스크 바운드 (disk-bound) 질의의 경우 평균 60-110 MB/s per disk (SATA2 and SAS)

현재 프로젝트 상태

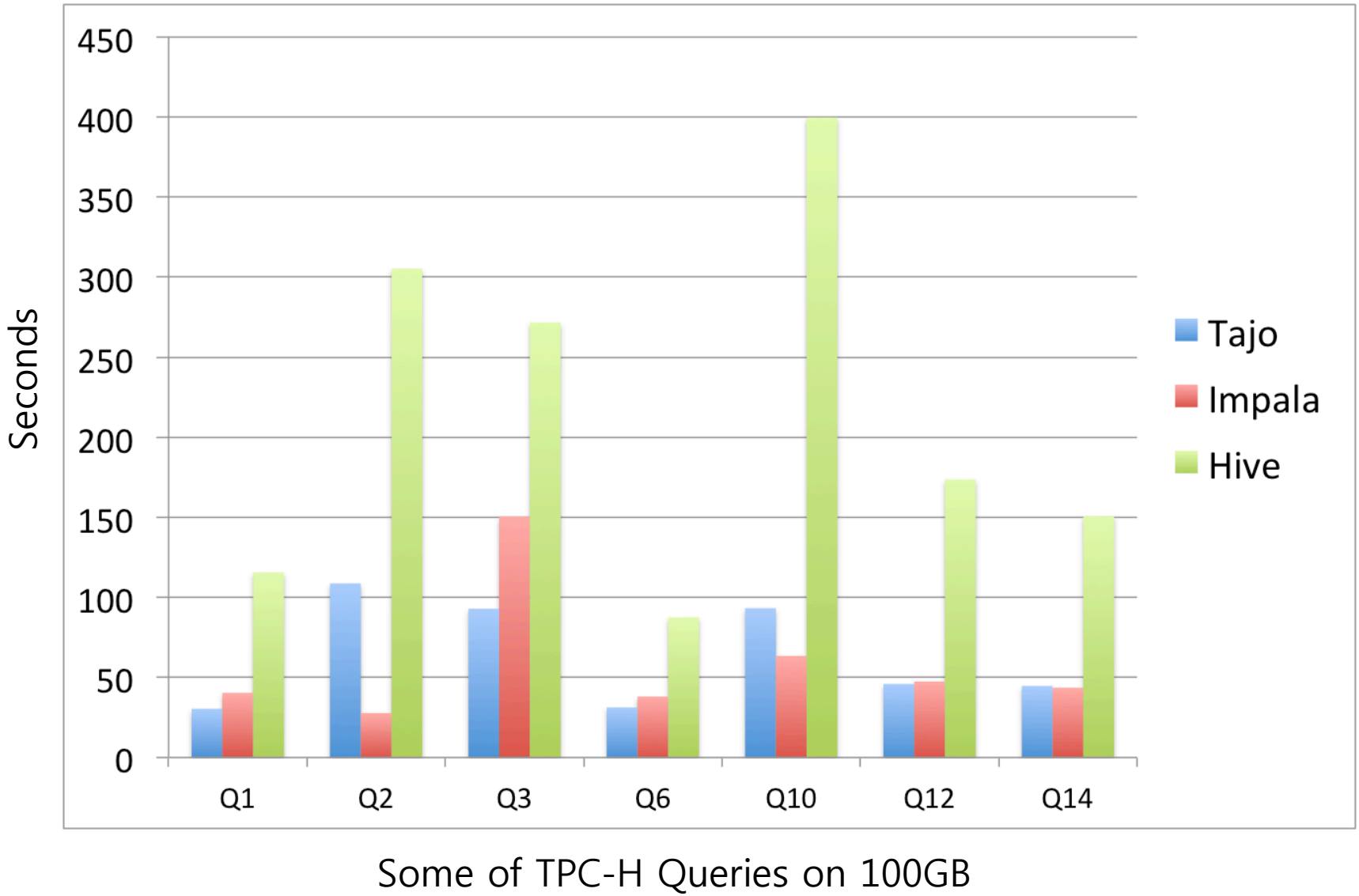
현재 프로젝트 상태



비교 평가

- Tajo (0.2-incubating) vs. Impalad_version 1.1.1 vs. Hive 0.10-cdh4
- TPC-H data set 100GB
- 모든 테스트는 캐쉬 drop 후 수행
- 10G 이더넷
- 6대 클러스터 노드
  - Intel Xeon CPU E5 2640 2.50GHz x 4
  - 64 GB memory
  - 6 SATA2 disks

비교 평가



Some of TPC-H Queries on 100GB

Roadmap

- **현재 0.2 릴리즈 투표중**

- **2013.12 : Apache Tajo 0.8 Release**
  - SQL 지원 향상, 안정성 향상
  - Outer join 지원
  - Hadoop 2.2.0 GA 포팅
  - Table Partitioning: Hash, Range, List, Column (Hive style)
  - Hcatalog 엑세스 지원

- **2014 Q1: Apache Tajo 1.0-alpha release**
  - 강력하고 다양한 rewrite rule 추가
  - 윈도우 함수 지원 (window function)
  - JIT query compilation & vectorized 엔진 프로토타입

참여 방법!

- 시작하기
  - http://wiki.apache.org/tajo/GettingStarted
- 개발 브렌치를 이용한 빌드
  - http://wiki.apache.org/tajo/BuildInstruction
- Jira - 이슈 트래커
  - https://issues.apache.org/jira/browse/TAJO
- 메일링 리스트 가입
  - tajo-dev-subscribe@incubator.apache.org