

Complex types: map <key-type, value-type>
 List <element-type>
 struct <file-name, field-type, ...>

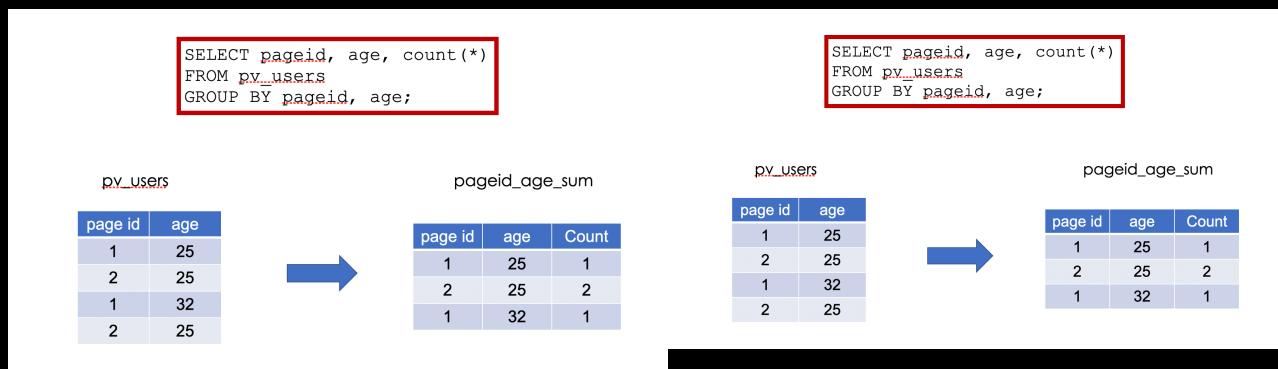
Hive Query Language

HQL \subseteq SQL + some extensions

↳ Support basic SQL statements

select, project, join, group by, aggregation, create table

Hive Join: Only support equality join



Hive Cannot insert data is designed for data analytics.

Data is generated from outside of Hive

Pros: better concurrency control

Cons: hard to optimize, cannot ensure best storage layout

In DB's they figure out the best storage layout for fast query.

Hive Architecture

Metastore: component that stores the system catalog and meta data about tables, cols, parts, . . . stored on RDBMS.

Driver: component that manages the lifecycle of a `hiveSQL` statement as it moves through the hive. Also, maintains a session handle and any session statistics.

Query Compiler: component that compiles `hiveSQL` into a directed acyclic graph of map/reduce tasks.

Optimizer: chain of transformations such that the operator DAG resulting from one trans. is passed as input to the next trans.
(col. pruning, part. pruning, repartitioning of data)

Execution Engine: component that executes the tasks produced by the compiler in proper dependency order. The execution engine interacts w/ the underlying Hadoop instance.

Thrift server: component that provides a thrift interface and a JDBC/ODBC server and provides a way of integrating Hive with other apps.

Client components: Command Line, Interface (CLI)
web UI, JDBC/ODBC driver

software that
let programs talk
to the DB.

Some Techniques We Know and Love Are not Directly Applicable	
■ Indexing ■ Zone-maps ■ Co-located joins ■ Query rewrites ■ Cost-based optimization	■ Databases own their storage SQL-on-Hadoop systems do not ■ Metadata management is tricky ■ Data inserted/loaded without SQL system knowledge ■ No co-location of related tables ■ HDFS is for most practical purposes, read-only