DIFFERENCES BETWEEN HIVE AND RDBMS

HIVE VS RDBMS

HIVE IS A DATA WAREHOUSE ON TOP OF HADOOP

HIVE VS RDBMS

DATAWAREHOUSES ARE INHERENTLY DIFFERENT FROM RELATIONAL DATABASES

HIVE VS RDBMS

THEY ARE GEARED TOWARDS
ANALYTICAL PROCESSING

HIVE VS RDBMS

RDBMS ARE GEARED TO TRANSACTION PROCESSING

HIVE VS RDBMS

WHILE THERE ARE MANY DIFFERENCES IN THESE 2 PARADIGMS

LET'S CONCENTRATE ON 2 SPECIFIC DIFFERENCES

HIVE VS RDBMS

2 SPECIFIC DIFFERENCES

BATCH PROCESSING

SCHEMA-ON-READ

HIVE VS RDBMS

2 SPECIFIC DIFFERENCES

BATCH PROCESSING

SCHEMA-ON-READ

HIVE VS RDBMS

BATCH PROCESSING

HIVE IS OPTIMAL FOR PROCESSING REALLY LARGE-SCALE DATASETS

GIGABYTES/PETABYTES

HIVE VS RDBMS

BATCH PROCESSING

THE DATA IN HIVE IS MEANT TO BE USED FOR ANALYTICAL PURPOSES

HIVE VS RDBMS

BATCH PROCESSING

FOR EX: TO PROCESS ORDERS DATA
FOR TRENDS IN LAST 3 YEARS

HIVE VS RDBMS

BATCH PROCESSING

SINCE HIVE USES HADOOP, MAPREDUCE

IT LEVERAGES THE TREMENDOUS PARALLEL COMPUTING POWER OF THESE FRAMEWORKS

HIVE VS RDBMS

BATCH PROCESSING

TRADITIONAL DATABASES IMPROVE PERFORMANCE USING INDEXES

HIVE VS RDBMS

BATCH PROCESSING

BUILDING INDEXES TAKE A LOT OF TIME AND OCCUPIES DISK SPACE

HIVE VS RDBMS

BATCH PROCESSING

INCREASING DISK SPACE ON A SINGLE SYSTEM IS AN EXPENSIVE AFFAIR

HIVE VS RDBMS

BATCH PROCESSING

IN ADDITION PERFORMANCE DOES NOT INCREASE LINEARLY WITH DISK SPACE

HIVE VS RDBMS

BATCH PROCESSING

ON THE OTHER HAND, HIVE/HADOOP USE LARGE NUMBERS OF CHEAP MACHINES IN PARALLEL

HIVE VS RDBMS

BATCH PROCESSING

DOUBLING PERFORMANCE SIMPLY MEANS YOU NEED TO DOUBLE THE NUMBER OF MACHINES

HIVE VS RDBMS

BATCH PROCESSING

HIVE DOES NOT PERFORM WELL FOR TRANSACTIONAL PROCESSING

HIVE VS RDBMS

BATCH PROCESSING

EVEN FETCHING A SINGLE ROW WILL LAUNCH A MAP-REDUCE JOB THAT MIGHT TAKE MINUTES TO RUN

HIVE VS RDBMS

BATCH PROCESSING

A WELL DESIGNED RDBMS CAN ANSWER TRANSACTIONAL QUERIES IN MILLI/MICROSECONDS

HIVE VS RDBMS

BATCH PROCESSING

HIVE ALSO DOES NOT ALLOW ROW LEVEL UPDATES IN ITS TABLES

HIVE VS RDBMS

BATCH PROCESSING

ONCE THE DATA IS WRITTEN TO HIVE,
ITS PURPOSE IS READ-ONLY

HIVE VS RDBMS

BATCH PROCESSING

THIS IS TOTALLY FINE FOR ANALYTICAL PROCESSING

ANALYSTS WOULD ONLY READ THE HISTORICAL DATA, NEVER UPDATE IT

HIVE VS RDBMS

BATCH PROCESSING

WHY ARE UPDATES NOT ALLOWED IN HIVE?

HIVE IS BASED ON SCHEMA-ON-READ

HIVE VS RDBMS

2 SPECIFIC DIFFERENCES

BATCH PROCESSING

SCHEMA-ON-READ

SCHEMA-ON-READ

THE SCHEMA IS THE DESCRIPTION OF A DATABASE TABLE

COLUMN NAMES,
COLUMN TYPES
CONSTRAINTS

SCHEMA-ON-READ

COLUMN NAMES,
COLUMN TYPES
CONSTRAINTS

THESE 3 ARE
DEFINED WHEN YOU
CREATE A TABLE

SCHEMA-ON-READ

COLUMN NAMES,
COLUMN TYPES
CONSTRAINTS

WHEN YOU READ FROM OR WRITE TO A TABLE, YOU EXPECT THAT THIS SCHEMA IS ENFORCED

SCHEMA-ON-READ

HOW IS THE SCHEMA ENFORCED?

TRADITIONAL DATABASES DO IT USING

SCHEMA-ON-WRITE

HIVE USES

SCHEMA-ON-READ

TRADITIONAL DATABASES

SCHEMA-ON-WRITE

TRADITIONAL DATABASES HAVE COMPLETE CONTROL OVER THE DATA STORAGE

TRADITIONAL DATABASES

SCHEMA-ON-WRITE

NO EXTERNAL PROGRAM CAN ACCESS THE DATA WITHOUT GOING THROUGH THE DATABASE

TRADITIONAL DATABASES

SCHEMA-ON-WRITE

THE DATABASE IS THE

SOLE GATEKEEPER

TRADITIONAL DATABASES

SCHEMA-ON-WRITE

THE DATABASE ENFORCES ACID PROPERTIES

ATOMICITY, CONSISTENCY, ISOLATION, DURABILITY

TRADITIONAL DATABASES

SCHEMA-ON-WRITE

THIS MEANS THAT THE DATABASE ENFORCES THE SCHEMA OF A TABLE WHEN DATA IS WRITTEN

TRADITIONAL DATABASES

SCHEMA-ON-WRITE

NO DATA CAN EXIST IN THE TABLE
WITHOUT FOLLOWING THE SCHEMA

SCHEMA-ON-READ

HOW IS THE SCHEMA ENFORCED?

TRADITIONAL DATABASES DO IT USING

SCHEMA-ON-WRITE

HIVE USES

SCHEMA-ON-READ

HIVE

SCHEMA-ON-READ

IN HIVE THE UNDERLYING DATA OF TABLES IS STORED AS FILES IN HDFS

HIVE

SCHEMA-ON-READ

HIVE IS NOT THE SOLE OWNER OF THE FILES

HIVE

SCHEMA-ON-READ

THE FILES CAN BE USED AND MODIFIED BY OTHER CLIENTS

HIVE

SCHEMA-ON-READ

FOR EXAMPLE, YOU CAN HAVE THE SAME FILES SHARED BETWEEN HIVE, HBASE AND CASSANDRA

THESE ARE DIFFERENT DATABASE TECHNOLOGIES THAT WORK USING HADOOP

HIVE

SCHEMA-ON-READ

BECAUSE THE UNDERLYING FILES CAN BE CHANGED AT ANY TIME, HIVE CANNOT ENFORCE

SCHEMA-ON-WRITE

INSTEAD, HIVE ENFORCES

SCHEMA-ON-READ

HIVE

SCHEMA-ON-READ

IN THE HIVE METASTORE, THERE ARE INSTRUCTIONS FOR HIVE ON HOW TO READ AND PARSE THE HDFS FILES

IN THE HIVE METASTORE, THE SCHEMA OF A TABLE IS ALSO STORED

HIVE

SCHEMA-ON-READ

WHEN YOU READ THE DATA IN HIVE, IT WILL PARSE THE FILE AND TRY TO IMPOSE THE SCHEMA

HIVE

SCHEMA-ON-READ

HIVE MAY NOT ALWAYS SUCCEED IN IMPOSING THE SCHEMA

HIVE

SCHEMA-ON-READ

WHAT IF YOU HAD A FILE WITH LINES

VITTHAL, SRINIVASAN
JANANI, RAVI
SWETHA, HANURAJ

HIVE TRIES TO IMPOSE THE SCHEMA

FIRSTNAME, LASTNAME, AGE

HIVE

SCHEMA-ON-READ

FIRSTNAME, LASTNAME, AGE

VITTHAL, SRINIVASAN

JANANI, RAVI

SWETHA, HANURAJ

THE FILE HAS ONLY 2 COLUMNS

HIVE

SCHEMA-ON-READ

FIRSTNAME, LASTNAME, AGE

VITTHAL, SRINIVASAN

JANANI, RAVI

SWETHA, HANURAJ

HIVE EXPECTS THE ROW TO HAVE 3 COLUMNS BUT FINDS ONLY 2 COLUMNS

HIVE

SCHEMA-ON-READ

FIRSTNAME, LASTNAME, AGE
VITTHAL, SRINIVASAN
JANANI, RAVI
SWETHA, HANURAJ

IF YOU TRY TO QUERY THE AGE COLUMN,
IT WILL RETURN NULL VALUES

HIVE

SCHEMA-ON-READ

HIVE DOES IT'S BEST TO IMPOSE THE SCHEMA ON A FILE

HIVE SCHEMA-ON-READ

BECAUSE HIVE IS SCHEMA-ON-READ

IT DOES NOT CHECK THE DATA AT ALL WHEN DOING WRITE OPERATIONS

HIVE

SCHEMA-ON-READ

DURING LOAD/INSERT OPERATIONS, HIVE WILL JUST DUMP THE DATA INTO A FILE WITHOUT CHECKING THE SCHEMA

HIVE

SCHEMA-ON-READ

SINCE IT IS UNAWARE OF SCHEMA DURING WRITE OPERATIONS, YOU CANNOT DO ROW LEVEL UPDATES/DELETES

HIVE

SCHEMA-ON-READ

FOR THE SAME REASON, HIVE CANNOT SUPPORT CONSTRAINTS LIKE PRIMARY KEY, FOREIGN KEY ETC

DIFFERENCES BETWEEN HIVEQL AND SQL

CONSTRAINTS

JOINS

SUBQUERIES

FUNCTIONS

DELETE/UPDATE

ALL OF THESE ARE FEATURES IN SQL

CONSTRAINTS

FUNCTIONS

JOINS

DELETE/UPDATE

SUBQUERIES

HIVEQL ALSO PROVIDES MANY OF THESE FEATURES...

CONSTRAINTS

JOINS

.. WITH SOME LIMITATIONS

FUNCTIONS

SUBQUERIES

DELETE/UPDATE

CONSTRAINTS

IN SQL YOU CAN USE CONSTRAINTS TO MAINTAIN RELATIONSHIPS BETWEEN TABLES

FOREIGN KEY CONSTRAINTS

YOU CAN ALSO IMPOSE CONSTRAINTS ON DATA IN A COLUMN

PRIMARY KEY, NOT NULL, UNIQUE

CONSTRAINTS

FOREIGN KEY CONSTRAINTS

PRIMARY KEY, NOT NULL, UNIQUE

HIVE DOES NOT SUPPORT ANY OF THESE CONSTRAINTS

CONSTRAINTS

CONSTRAINTS ARE ENFORCED BY A DATABASE WHEN DATA IS WRITTEN

CONSTRAINTS

HIVE DOES NOT CARE ABOUT THE SCHEMA OF THE DATA WHEN IT'S WRITTEN

CONSTRAINTS

HIVE IS SCHEMA-ON-READ, WHICH MEANS IT ONLY IMPOSES SCHEMA WHEN IT READS A TABLE

CONSTRAINTS

ONE OTHER PURPOSE OF CONSTRAINTS TRADITIONALLY IS TO SPEED UP QUERY EXECUTION

CONSTRAINTS

HIVE HAS LIMITED SUPPORT FOR INDEXES TO SPEED UP QUERY EXECUTION

CONSTRAINTS

HIVE PRIMARILY USES FEATURES LIKE PARTITIONING AND BUCKETING IN IMPROVE QUERY PERFORMANCE

CONSTRAINTS

JOINS

SUBQUERIES

VIEWS

FUNCTIONS

DELETE/UPDATE

IN SQL YOU CAN DELETE/ UPDATE ROWS THAT SATISFY A CONDITION

DELETE/UPDATE

CONSTRAINTS

FUNCTIONS

SUBQUERIES

VIEWS

```
delete from employees where
firstname="Kiran";

update employees set
lastname="Srinivasan"
where firstname="Vitthal";
```

CONSTRAINTS

JOIN

FUNCTIONS

SUBQUERIES

VIEWS

DELETE/UPDATE

THESE DO NOT WORK IN HIVE

CONSTRAINTS

ONCE AGAIN THIS IS BECAUSE OF SCHEMA-ON-READ

JOINS

FUNCTIONS

SUBQUERIES

DELETE/UPDATE

CONSTRAINTS

DELETE AND UPDATE ARE WRITE OPERATIONS

FUNCTIONS

SUBQUERIES

DELETE/UPDATE

CONSTRAINTS

HIVE DOES NOT IMPOSE ANY SCHEMA DURING WRITE OPERATIONS

SUBQUERIES

DELETE/UPDATE

CONSTRAINTS

JOINS

SUBQUERIES

FUNCTIONS

DELETE/UPDATE

FUNCTIONS

HIVE AND SQL BOTH HAVE SEVERAL FUNCTIONS TO PROCESS DATA

AVG, COUNT, MONTH, YEAR ETC

CONSTRAINTS

JOINS

SUBQUERIES

DELETE/UPDATE

CONSTRAINTS

JOINS

HIVE HAS MANY MORE BUILT-IN FUNCTIONS THAN SQL FOR SOME USE CASES

FUNCTIONS

DELETE/UPDATE

MATHEMATICAL FUNCTIONS LIKE COVARIANCE

HIVE HAS MANY MORE BUILT-IN FUNCTIONS THAN SQL FOR SOME USE CASES

FUNCTIONS

DELETE/UPDATE

FUNCTIONS FOR STATISTICAL PROCESSING AND DATA MINING

CONSTRAINTS

JOINS

SUBQUERIES

FUNCTIONS

DELETE/UPDATE

CONSTRAINTS

JOINS

FUNCTIONS

SUBQUERIES

HIVE HAS EXTENSIVE SUPPORT FOR JOINS

DELETE/UPDATE

CONSTRAINTS

JOINS

FUNCTIONS

SUBQUERIES

IT EVEN ALLOWS THE USER TO CONTROL THE EXECUTION OF THE JOIN TO SOME EXTENT

DELETE/UPDATE

CONSTRAINTS

JOINS

FUNCTIONS

SUBQUERIES

DELETE/UPDATE

THIS ALLOWS USERS TO OPTIMIZE JOINS FOR CERTAIN USE CASES

CONSTRAINTS

JOINS

FUNCTIONS

SUBQUERIES

DELETE/UPDATE

THERE ARE 2 MAIN DIFFERENCES IN JOINS BETWEEN HIVE AND SQL

CONSTRAINTS

JOINS

FUNCTIONS

SUBQUERIES

1. HIVE SUPPORTS EQUIJOINS ONLY

DELETE/UPDATE

2. HIVE DOES NOT SUPPORT NATURAL JOINS

EQUIJOINS

CONSTRAINTS

JOINS

FUNCTIONS

SUBQUERIES

HERE IS A JOIN QUERY

VIEWS

DELETE/UPDATE

```
select a.firstname, b.subordinate
from employees join subordinates on
a.empId=b.empId;
```

CONSTRAINTS

EQUIJOINS

JOINS

FUNCTIONS

SUBQUERIES

```
select a.firstname, b.subordinate
from employees join subordinates on
a.empId=b.empId;
```

VIEWS

DELETE/UPDATE

IN SQL THIS CONDITION CAN BY ANYTHING

EQUIJOINS

CONSTRAINTS

JOINS

FUNCTIONS

SUBQUERIES

VIEWS

DELETE/UPDATE

```
select a.firstname, b.subordinate
from employees join subordinates on
a.empId=b.empId;
```

IN HIVE THIS CONDITION HAS TO BE AN EQUALITY

NATURAL JOINS

CONSTRAINTS

JOINS

FUNCTIONS

SUBQUERIES

```
select a.firstname, b.subordinate
from employees,subordinates;
```

VIEWS

DELETE/UPDATE

NATURAL JOINS ARE JOINS WHERE THE JOIN CLAUSE IS IMPLICIT

NATURAL JOINS

CONSTRAINTS

JOINS

FUNCTIONS

SUBQUERIES

VIEWS

DELETE/UPDATE

```
select a.firstname, b.subordinate
from employees,subordinates;
```

THE TABLES ARE JOINED IMPLICITLY
BASED ON COMMON COLUMNS

CONSTRAINTS

NATURAL JOINS

JOINS

FUNCTIONS

SUBQUERIES

HIVE DOES NOT SUPPORT NATURAL JOINS

VIEWS

UPDATE

CONSTRAINTS

JOINS

FUNCTIONS

SUBQUERIES

DELETE/UPDATE

HIVE SUPPORTS SUBQUERIES

CONSTRAINTS

JOINS

FUNCTIONS

SUBQUERIES

DELETE/UPDATE

SOME SPECIFIC TYPES OF SUBQUERIES ARE NOT ALLOWED

CONSTRAINTS

JOINS

SUBQUERIES

FUNCTIONS

DELETE/UPDATE

UNION IS SUPPORTED BUT INTERSECT AND MINUS CONSTRUCTS DON'T EXIST IN HIVE

CONSTRAINTS

JOINS

FUNCTIONS

SUBQUERIES

THERE ARE SOME RESTRICTIONS ON HOW IN/EXISTS ARE USED

CONSTRAINTS

FUNCTIONS

JOINS

DELETE/UPDATE

SUBQUERIES