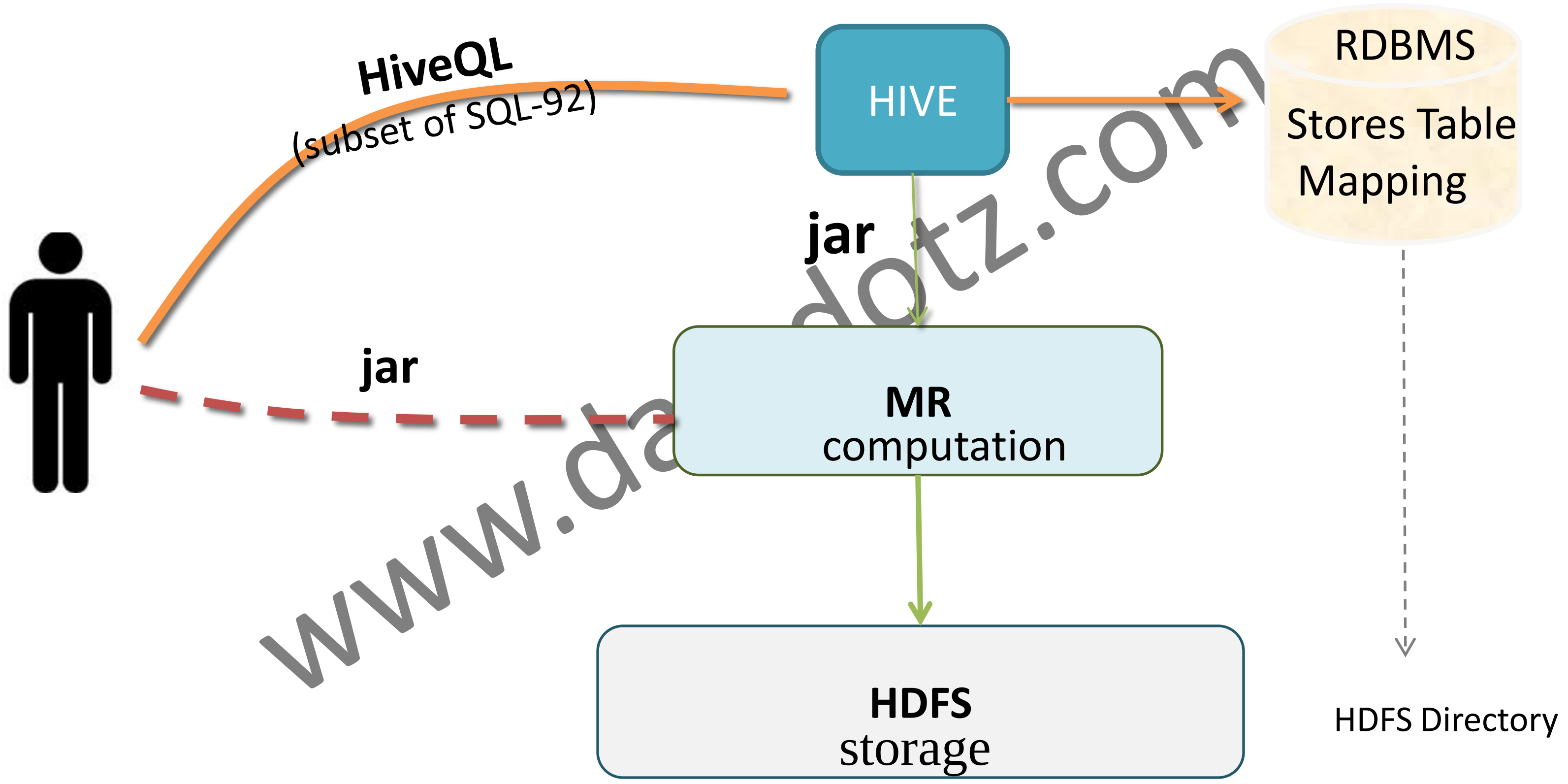
HIVE

Senthil Kumar A

What is Hive??


HiveQL
(subset of SQL-92)
HIVE
RDBMS
Stores Table Mapping
jar
jar
MR computation
HDFS storage
HDFS Directory

Introduction

- An Abstraction on top of MapReduce
- Allows users to query data in the Hadoop cluster without knowing Java or MapReduce
- Structured Data in HDFS logically into Tables
- Uses the HiveQL Language
  - Very similar to SQL
  - Turns HiveQL into MapReduce Jobs

Comparing Hive with Rdbms

Hive	RDBMS
HiveQL Interface. subset of SQL	SQL Interface.
Focus on analytics.	May focus on online or analytics.
No Transactions.	Transactions usually supported.
Partition adds, no random INSERTs. In-Place updates not natively supported (but are possible).	Random INSERT and UPDATE supported.
Distributed processing via map/reduce.	Distributed processing varies by vendor (if available).
Scales to hundreds of nodes.	Seldom scale beyond 20 nodes.
Built for commodity hardware.	Often built on proprietary hardware (especially when scaling out).
Low cost per petabyte.	What’s a petabyte?

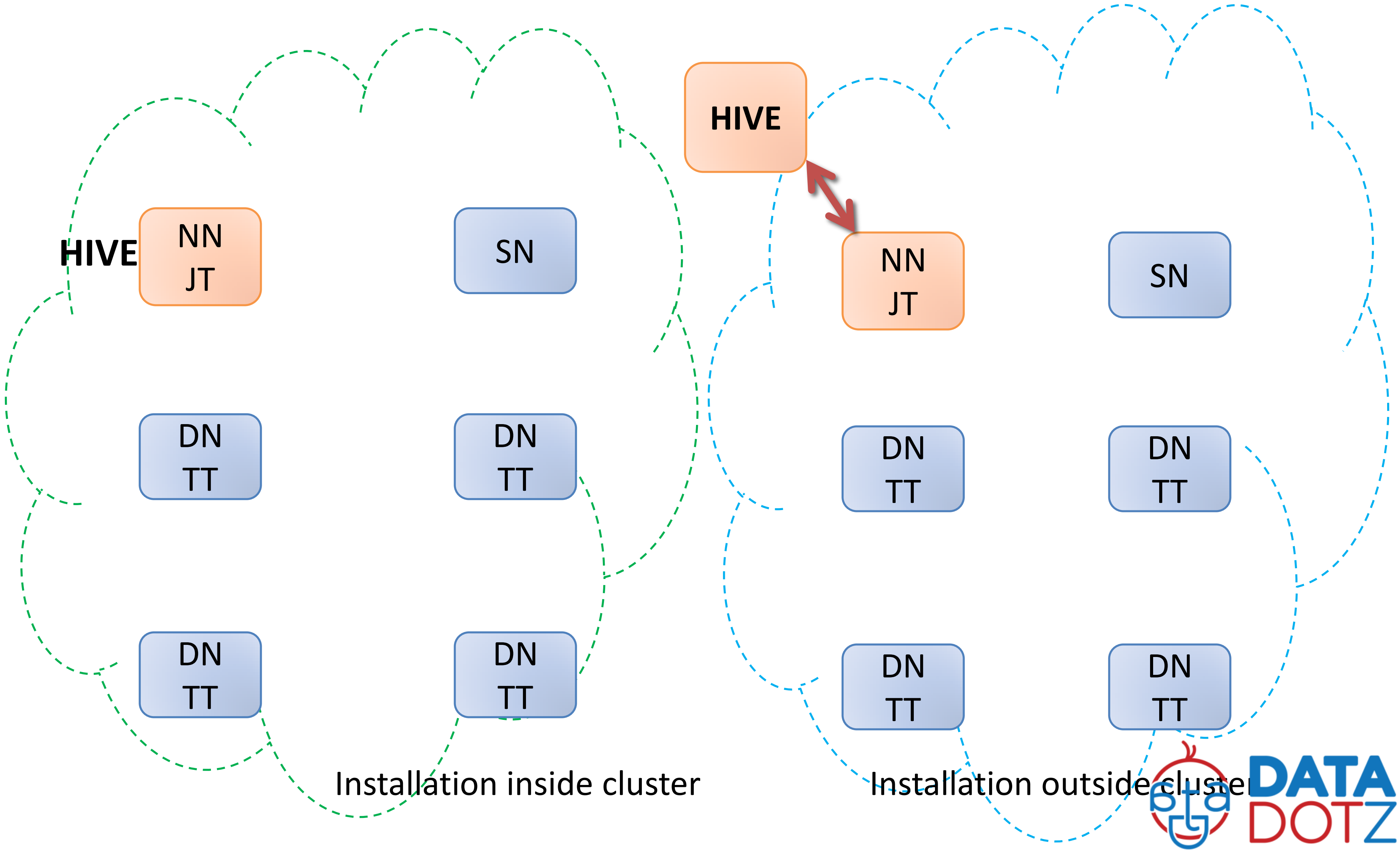
Installation

- Download http://hive.apache.org
- tar -xvf hive-*x.y.z*-dev.tar.gz
- export HADOOP_HOME=/<HADOOP DIR>
- bin/hive

Hive Execution Modes

- CLI
  - bin/hive
  - bin/hive - -service cli
- Batch Mode
  - bin/hive -f queries.hql
  - bin/hive -i script.sql
- Exec
  - bin/hive –e ‘select count(*) from pt’;
  - **Slient mode**
  - bin/hive –S –e ‘select count(*)from pt’;

Installation in Production Cluster



Installation inside cluster

Installation outside cluster

Create table

```
01,senthil,paracetamol,male,12
02,saravanan,avil,male,44
03,rajesh,metacin,male,26
04,usha,paracetamol,female,20
05,alex,paracetamol,male,48
06,nasir,metacin,male,37
07,singh,paracetamol,male,15
```

- create table patient(pid INT, pname STRING, drug STRING,gender STRING, tot_amt INT) row format delimited fields terminated by ',' stored as textfile;
- show tables;

ROW FORMAT DELIMITED - Hive to expect one record per line

FIELDS TERMINATED BY - Columns will be separated by the specified character

Default delimiter - Ctrl+A

Directory - creates a /user/hive/warehouse/patient directory

Loading data into tables

- load data inpath '/datagen_10.txt' into table patient;
  - *Load data inpath – HDFS mv shell command*

- load data local inpath '/home/students/datagen_10.txt' into table patient;
  - *Load data local inpath – HDFS copyFromLocal command*

Some More (Assignments)

- select count(*) from patient;
- select sum(tot_amt) from patient where drug = ‘paracetamol’;
- select max(tot_amt) from patient group by drug;
- show tables;
- desc patient;
- desc extended patient;
- show functions;
- select * from patient where drug in (‘avil’,’metacin’);
- select * from patient;

  does not run as MR. it’s a just a File Read from HDFS.

Store the output

- insert overwrite local directory ‘/home/senthil/results’ select count(*) from patient;
  - *Stores the result of the query in local directory*
- insert overwrite directory ‘/results’ select * from patient;
  - *Stores the result of the query in hdfs directory*

Some more..

```
paracetamol
avil
metacin
```

- create table drug(drugname STRING) row format delimited fields terminated by ‘,’ stored as textfile;
- load data local inpath ‘/home/senthil/drug_file.txt‘ into table drug;
- select * from patient join drug on patient.drug=drug.drugname;
- select patient.* from patient left outer join drug on patient.drug = drug.drugname where drug.drugname is NULL;
- create table drug_ new(drug STRING) row format delimited fields terminated by ‘,’ stored as textfile;
- insert overwrite table drug_new select * from drug;
- select * from drug_new;
- insert into table drug_new select * from drug;
- select * from drug_new;

External Table

- Hive has a relational database on the master node it uses to keep track of state.
- For Example when you CREATE TABLE FOO(foo string) LOCATION 'hdfs://tmp/'; this table schema is stored in the database
- When you drop an internal table, it drops the data, and it also drops the metadata.
- When you drop an external table, it only drops the meta data. That means hive is ignorant of that data now. It does not touch the data itself.

External Table

- create EXTERNAL table patient_external(pid INT, pname STRING, drug STRING,gender STRING,tot_amt INT) row format delimited fields terminated by ',' stored as textfile LOCATION '/patient_new';

- LOAD DATA local INPATH '/datagen_10.txt' INTO table patient_external;
- select * from patient_external ;
- “Please check the table directory in hdfs path"

- drop table patient_external;
- “Please check the table directory in hdfs path"

Partitions

- A way of dividing table into multiple parts based upon a column value such as Date
- defined at table creation time using the PARTITIONED BY clause
- create table patient (pid INT, pname STRING, drug STRING, tot_amt INT) partitioned by (dt STRING, country STRING)row format delimited fields terminated by ‘,’ stored as textfile;
- LOAD DATA LOCAL INPATH ‘/tmp/file_ind.txt‘ INTO TABLE logs PARTITION (dt='2012-11-01', country=‘IND');
- Above code creates a sub partition called country in date
- Please look at the directory structure in the HDFS
- Our data in the files should not contain the columns used for partitioning

Partition querys

- create table patient_partition_1(pid INT, pname STRING, drug STRING,gender STRING,tot_amt INT)partitioned by(country STRING)row format delimited fields terminated by ',' stored as textfile;
- LOAD DATA LOCAL INPATH '/home/username/Desktop/patient_file.txt' INTO TABLE patient_partition_1 PARTITION (country='IND');
- **Check the /user/hive/warehouse/patient_partition_1 directory in HDFS**
- create table patient_partition_2 (pid INT, pname STRING, drug STRING,gender STRING,tot_amt INT) partitioned by (dt STRING, country STRING)row formatdelimited fields terminated by ',' stored as textfile;
- desc patient_partition_2;
- LOAD DATA LOCAL INPATH '/home/username/Desktop/patient_file.txt' INTO TABLE patient_partition_2 PARTITION (dt='2012-11-01', country='IND');
- **Check the /user/hive/warehouse/ patient_partition_2**
- Select * from patient_partition_1;
- Select * from patient_partition_2;

Buckets

- To do sampling
- *CREATE TABLE bucketed_patient (pid INT, pname STRING, drug STRING, gentder STRING,tot_amt INT) CLUSTERED BY (pid) INTO 4 BUCKETS;*
- *set hive.enforce.bucketing = true;*
- *INSERT OVERWRITE TABLE bucketed_patient SELECT * FROM patient;*
- *Can contain multiple fields for bucketing.*
- To enable map side join effectively

Some more queries – Alter,sub query,views

(Assignments)

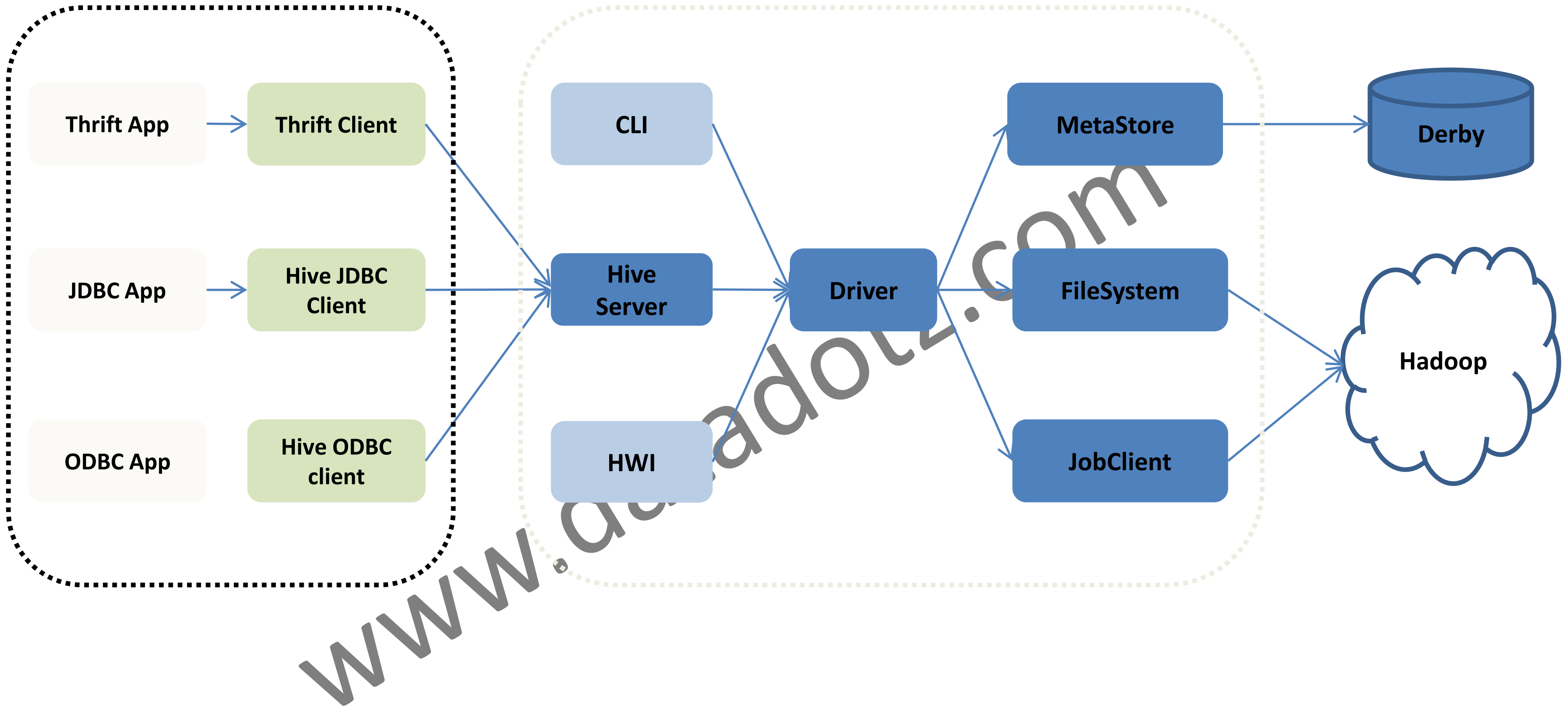
- **Alter**
- Alter Table patient RENAME TO patient_new;
- ALTER TABLE patient_new ADD COLUMNS (extra STRING);
- **Sub Queries**
- **Views**

Data Types (knowledge)

- Primitive types
  - Integers: TINYINT, SMALLINT, INT, BIGINT.
  - Boolean: BOOLEAN.
  - Floating point numbers: FLOAT, DOUBLE .
  - String: STRING.
- Complex types
  - Structs: {a INT; b INT}.
  - Maps: M['group'].
  - Arrays: ['a', 'b', 'c'], A[1] returns 'b'.

Date Format ?

Hive Architecture



JOINS (Assignments)

- SELECT a.* FROM a JOIN b ON (a.id = b.id)
- Join on multiple columns
  - SELECT a.* FROM a JOIN b ON (a.id = b.id AND a.department = b.department)
- Join on Multiple tables
  - SELECT a.val, b.val, c.val FROM a JOIN b ON (a.key = **b.key1**) JOIN c ON (c.key = **b.key1**)
  - SELECT a.val, b.val, c.val FROM a JOIN b ON (a.key = **b.key1**) JOIN c ON (c.key = **b.key2**)
- LEFT, RIGHT, and FULL OUTER joins – available in HiveQL
  - SELECT a.val, b.val FROM a LEFT OUTER JOIN b ON (a.key=b.key)
  - provides more control over ON clauses for which there is no match
- WHERE clause available using JOIN
- SELECT a.val, b.val FROM a LEFT OUTER JOIN b ON (a.key=b.key) WHERE a.ds='2009-07-07' AND b.ds='2009-07-07'

DATA DOTZ

Built in operators (Knowledge)

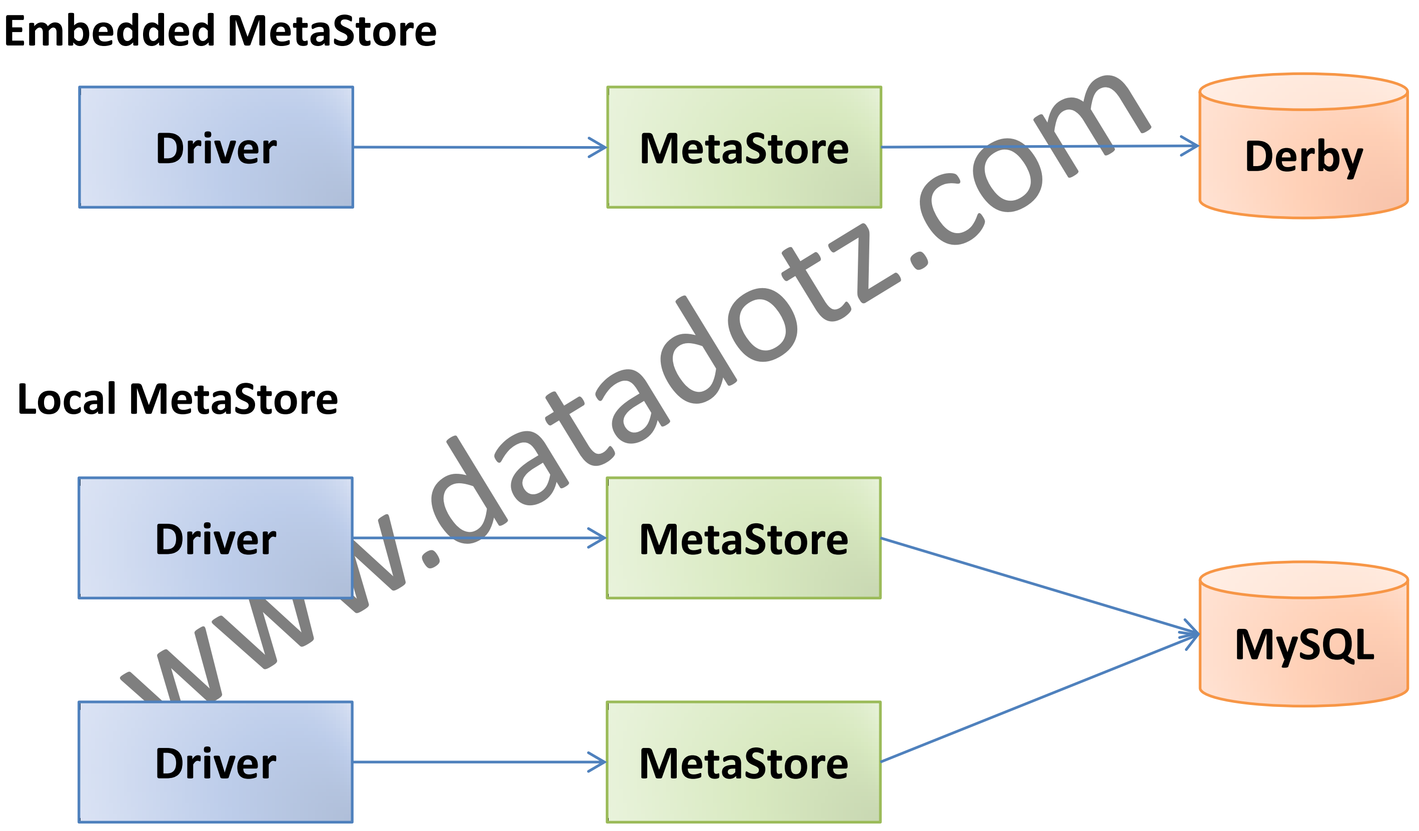
- Arithmetic Operators
  - +, -, *, /,%, |,^,~
- Relational Operators
  - = , <=> , !=, <>, < , >, <=, =>, IS NULL, IS NOT NULL, LIKE, RLIKE, REGEXP, BETWEEN , NOT BETWEEN
- Logical Operators
  - AND, OR, &&, ||, NOT, !

Built-in Functions (Knowledge)

- Mathematical: round, floor, ceil, rand, exp...
- Collection: size, map_keys, map_values, array_contains.
- Type Conversion: cast.
- Date: from_unixtime, to_date, year, datediff...
- Conditional: if, case, coalesce.
- String: length, reverse, upper, trim...

MetaStore

central repository of Hive metadata

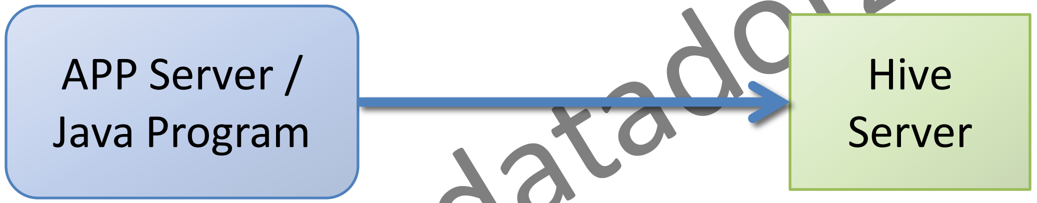


Hive Metastore in Mysql

- **$**sudo apt-get install mysql-server
- **$**hive-0.12.0 → conf → paste hive-site.xml
- **$**hive-0.12.0 →lib →paste mysql-connector.jar
- **$**mysql -u root -p
- **$**Enter password:root
- **mysql>**show databases;
- **$**hive cli stop and start
- **hive>**Show tables;
- **hive>**Create table and load the data.
- **mysql>**show databases;
- **mysql>**use metastore
- **mysql>**Show tables;
- **mysql>**select * from TBLS;

Hive JDBC

- bin/hive --service hiveserver
- uri is just “jdbc:hive://localhost:10000/default”



Other clients

- ODBC
- Thrift Clients
  - A software framework, for scalable cross-language services development, combines a software stack with a code generation engine to build services that work efficiently and seamlessly between C++, Java, Python, PHP, Ruby, Erlang, Perl, Haskell, C#, Cocoa, JavaScript, Node.js, Smalltalk, OCaml and Delphi and other languages
  - Python, PHP, Java, C++

Custom User Defined Functions

```
package com.example.hive.udf;
import org.apache.hadoop.hive.ql.exec.UDF;
 import org.apache.hadoop.io.Text;
public final class Lower extends UDF
 {
        public Text evaluate(final Text s){
                if (s == null)
                        { return null; }
                return new Text(s.toString().toLowerCase());
        }
}
```

UDF operates on a single row and produces a single row as its output.

UDF continued...

- hive> add jar my_jar.jar;
- hive> list jars;
- hive> create temporary function my_upper as 'package name.class name';
- hive> select my_upperr(pname) from patient;

- Simple API - org.apache.hadoop.hive.ql.exec.UDF
- Complex API-org.apache.hadoop.hive.ql.udf.generic.GenericUDF

DATA DOTZ

UDAF

- Works on multiple input rows
- It may either creates a single output row or create a multiple output rows
- Methods to override
  - init, iterate, terminatePartial, merge, terminate

Storage Formats

- Default
  - delimited - Control-A character with a row per line
- Two dimensions – row format & file formats
- Use ROW FORMAT or STORED AS for the above two
- ROW format uses Serde
  - Serde – serializers and deserializers
- Binary FileFormats
  - SequenceFile
  - RCFILE
  - ORC
  - JSON
  - XML
    - Default in industry
    - ***CREATE TABLE … ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.columnar.ColumnarSerDe' STORED AS RCFILE;***

Hive – Hbase integration

- Create a Hbase table “testtable” with column family “data” and column Qualifies “name”
- Replace the hbase-*.jar , zookeeper-*.jar,guava-*.jar,protobuf-*.jar in hive-*/lib from hbase-*/lib
- Copy the all the jars from hbase-*/lib folder to HADOOP_HOME/lib folder
- bin/hive
- *set hbase.zookeeper.quorum=localhost;*
- *create external TABLE hbase_table(key int, value string) STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler' WITH SERDEPROPERTIES ("hbase.columns.mapping" = ":key,data:name") TBLPROPERTIES ("hbase.table.name" = “testtable");*
- *insert overwrite table hbase_table select pid, pname from patient;*
- *select * from hbase_table;*

Hive Web Interface

- Configuration - hive-site.xml
  - hive.hwi.listen.port -- 9999
  - hive.hwi.listen.host -- 0.0.0.0
  - hive.hwi.war.file -- /lib/hive_hwi.war
- bin/hive --service hwi
- Features
  - Schema Browsing
  - Detached query execution
  - No local installation
  - Results are stored locally in hive server machine

THANK YOU

DATA DOTZ