

<http://poloclub.gatech.edu/cse6242>

CSE6242 / CX4242: **Data** & **Visual** Analytics

Scaling Up

Hive

Duen Horng (Polo) Chau

Assistant Professor

Associate Director, MS Analytics

Georgia Tech

Partly based on materials by

Professors Guy Lebanon, Jeffrey Heer, John Stasko, Christos Faloutsos, Parishit Ram (GT PhD alum; SkyTree), Alex Gray

Hive



<http://hive.apache.org>

Use SQL to run queries on large datasets

Developed at Facebook

Similar to Pig, Hive runs on client computer that submit jobs (no need to install on Hadoop cluster)

- You write **HiveQL** (Hive's query language), which gets converted into MapReduce jobs

Example: starting Hive

```
% hive
```

```
hive>
```

```
hive> SHOW TABLES;
```

```
OK
```


```
Time taken: 10.425 seconds
```

Example: create table, load data

```
CREATE TABLE records (year STRING, temperature INT,  
quality INT)
```

```
ROW FORMAT DELIMITED  
  FIELDS TERMINATED BY '\t';
```

Specify that data file is
tab-separated



```
LOAD DATA LOCAL INPATH 'input/ncdc/micro-tab/sample.txt'  
OVERWRITE INTO TABLE records;
```

Overwrite old file



This data file will be copied to
Hive's internal **data directory**



Example: Query

```
hive> SELECT year, MAX(temperature)
> FROM records
> WHERE temperature != 9999
> AND (quality = 0 OR quality = 1 OR quality = 4 OR
quality = 5 OR quality = 9)
> GROUP BY year;
1949      111
1950      22
```

Same thing done with Pig

```
records = LOAD 'input/ ncdc/ micro-tab/ sample.txt'  
  AS (year:chararray, temperature:int, quality:int);
```

```
filtered_records =  
  FILTER records BY temperature != 9999  
  AND (quality == 0 OR quality == 1 OR  
    quality == 4 OR quality == 5 OR  
    quality == 9);
```

```
grouped_records = GROUP filtered_records BY year;
```

```
max_temp = FOREACH grouped_records GENERATE  
  group, MAX( filtered_records.temperature);
```

```
DUMP max_temp;
```

Pig (vs SQL)

1. Pig is **procedural** (SQL is declarative)
2. **Checkpointing** data in the pipeline
3. Use **specific** operator implementations
vs. relying on optimizer
4. **Splitting** pipeline
e.g., do multiple things to intermediate data
5. Use developer's **own code**
e.g., different ways of loading data