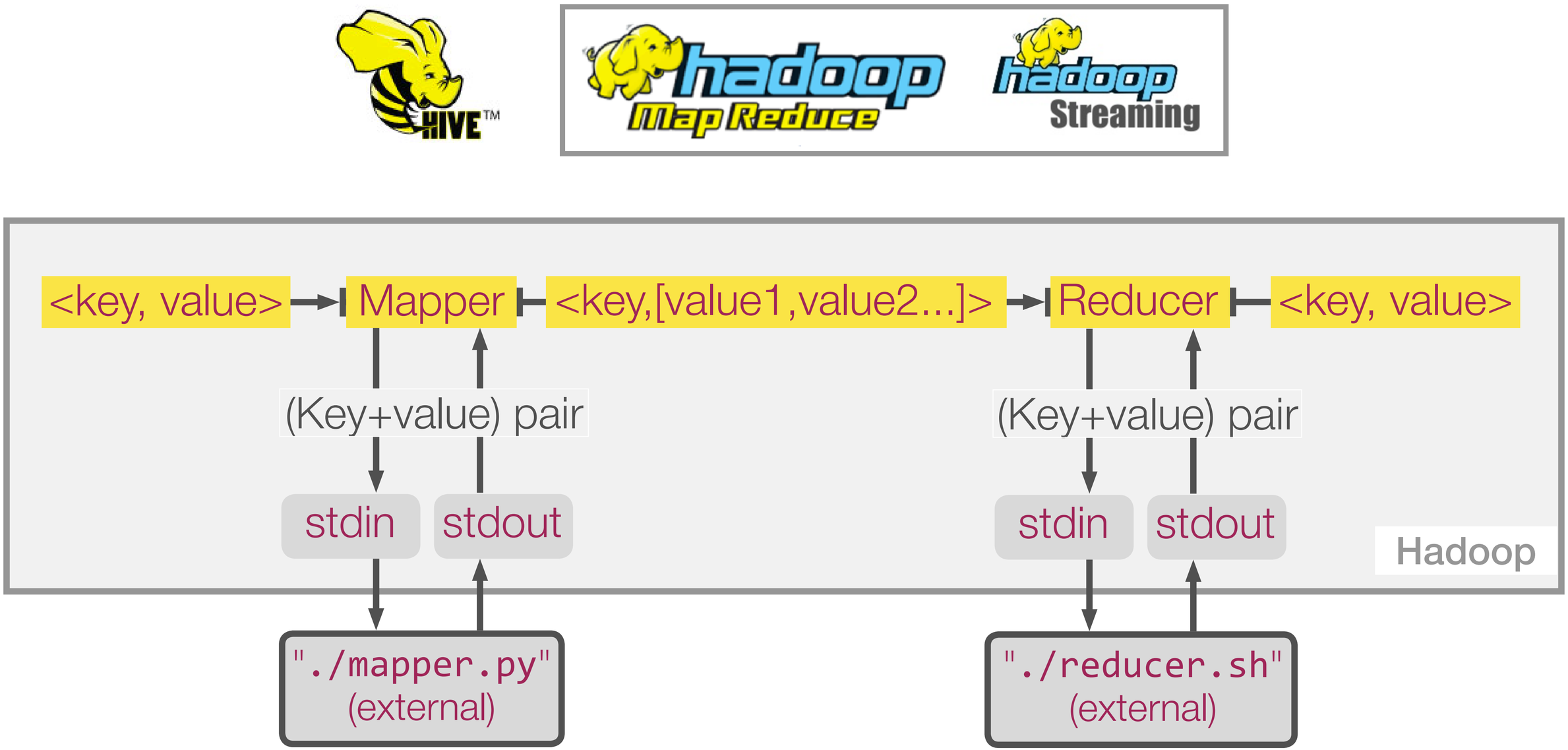
Hive Streaming

HIVE™

hadoop
Map Reduce


<key, value>
Mapper
<key,[value1,value2...]>
Reducer
<key, value>
(Key+value) pair
(Key+value) pair
stdin
stdout
stdin
stdout
Hadoop
"./mapper.py"
(external)
"./reducer.sh"
(external)

```
FROM my_table
SELECT TRANSFORM ...
```

```
FROM my_table
SELECT TRANSFORM (column_A, column_B)
...
```

```
FROM my_table
SELECT TRANSFORM (column_A, column_B)
USING "/bin/cat"
```


```
FROM my_table
SELECT TRANSFORM (column_A, column_B)
USING "/bin/cat"
AS new_A, new_B
```

```
"a"     "1"
"b"     "2"
...
```


```
FROM my_table
SELECT TRANSFORM (column_A, column_B)
USING "/bin/cat"
AS (new_A STRING, new_B DOUBLE)
```

↓

```
"a"     1.0
"b"     2.0
...
```

```
FROM my_table
SELECT TRANSFORM (column_A, column_B)
USING "/bin/cat"
AS (new_A STRING, new_B DOUBLE)
```

↓

```
"a"     1.0
"b"     2.0
...
```

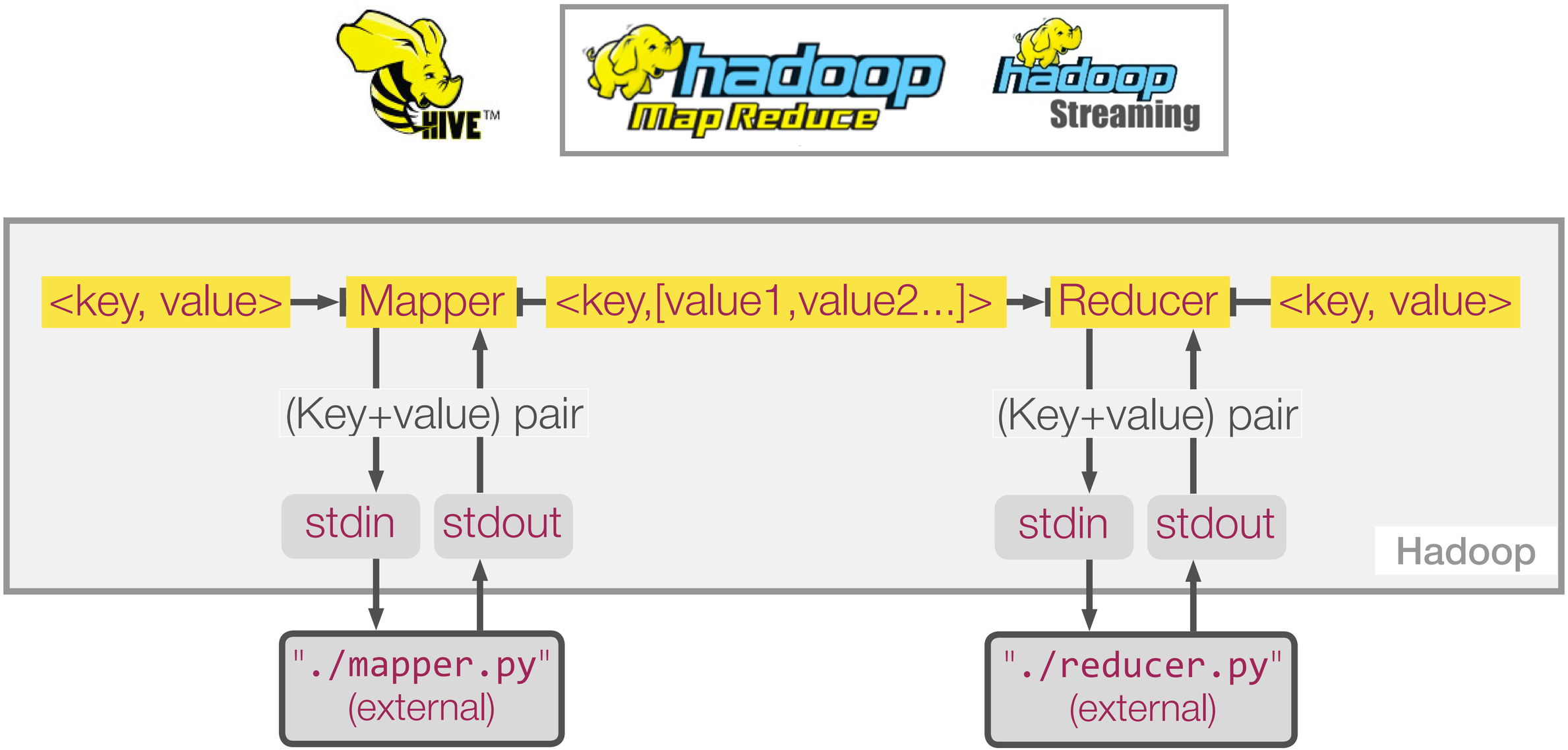
See: https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Transform

```
FROM my_table
SELECT TRANSFORM (column_A, column_B)
USING "/bin/cat"
AS new_A, new_B
```

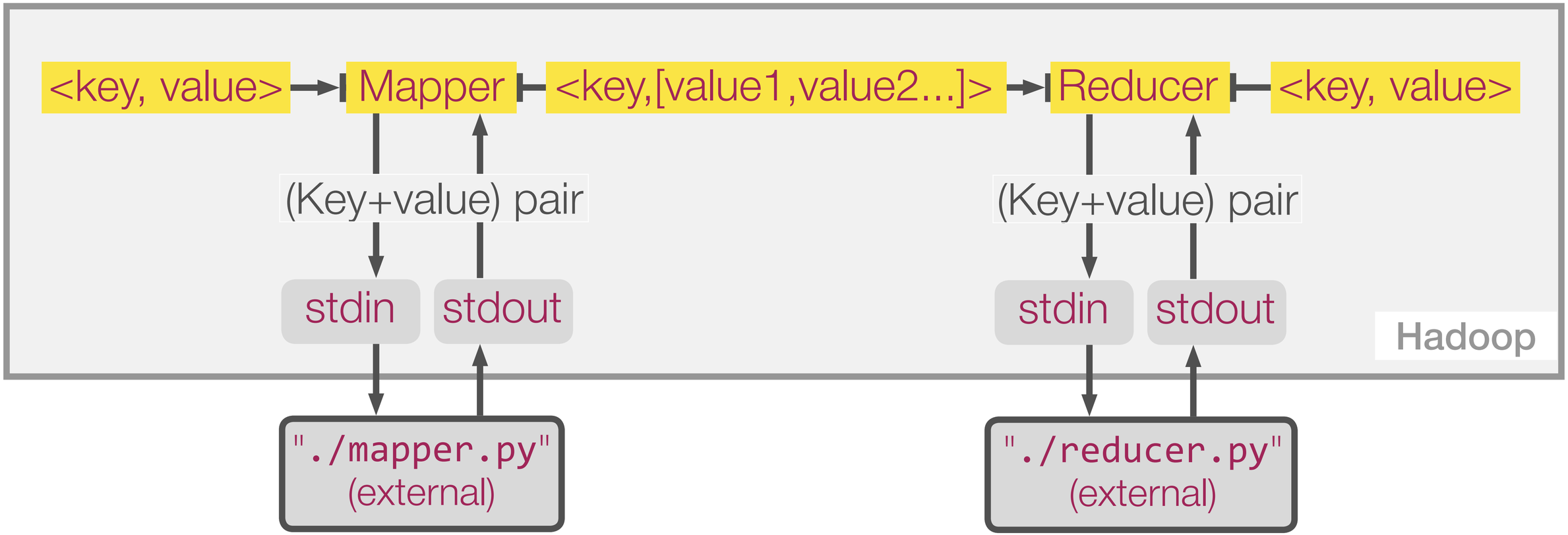
```
"a"     "1"
"b"     "2"
...
```

```
FROM my_table
SELECT TRANSFORM (column_A, column_B)
USING "/bin/cat -f1"
AS new_A, new_B
```

```
"a"    NULL
"b"    NULL
...
```


HIVE™
hadoop
Map Reduce
hadoop
Streaming
<key, value>
Mapper
<key,[value1,value2...]>
Reducer
<key, value>
(Key+value) pair
(Key+value) pair
stdin
stdout
stdin
stdout
Hadoop
"./mapper.py"
(external)
"./reducer.py"
(external)

WordCount


<key, value>
Mapper
<key,[value1,value2...]>
Reducer
<key, value>
(Key+value) pair
(Key+value) pair
stdin
stdout
stdin
stdout
Hadoop
"./mapper.py"
(external)
"./reducer.py"
(external)

stdin

Mapper (Python): **mapper.py**

```python
from __future__ import print_function
import sys

for line in sys.stdin:
    article_id, content = line.split("\t", 1)
    words = content.split()
    for word in words:
        print(word, 1, sep="\t")
```

stdout

stdin

Mapper (Python): **reducer.py**

```
# …
for line in sys.stdin:
    word, counts = line.split("\t", 1)
    counts = int(counts)
    if word == current_word:
        word_count += counts
    else:
        if current_word:
            print(current_word, word_count, sep="\t")
        current_word = word
        word_count = counts
# …
```

stdout

WIKIPEDIA
The Free Encyclopedia

HIVE™

WordCount

```
FROM (
   FROM wikipedia_sample
   SELECT TRANSFORM (line)
   USING "./mapper.py" AS word, counts
   DISTRIBUTE BY word SORT BY word
) word_pairs
SELECT TRANSFORM (word_pairs.word, word_pairs.counts)
USING "./reducer.py"
AS word, counts
```

WIKIPEDIA
The Free Encyclopedia

HIVE™

WordCount

```
FROM (
   FROM wikipedia_sample
   SELECT TRANSFORM (line)
   USING "./mapper.py" AS word, counts
   DISTRIBUTE BY word SORT BY word
) word_pairs
SELECT TRANSFORM (word_pairs.word, word_pairs.counts)
USING "./reducer.py"
AS word, counts
```

WIKIPEDIA
The Free Encyclopedia

HIVE™

WordCount

```
FROM (
   FROM wikipedia_sample
   SELECT TRANSFORM (line)
   USING "./mapper.py" AS word, counts
   DISTRIBUTE BY word SORT BY word
) word_pairs
SELECT TRANSFORM (word_pairs.word, word_pairs.counts)
USING "./reducer.py"
AS word, counts
```

CLUSTER BY

WIKIPEDIA
The Free Encyclopedia

HIVE™

WordCount

```
FROM (
   FROM wikipedia_sample
   MAP (line)
   USING "./mapper.py" AS word, counts
   DISTRIBUTE BY word SORT BY word
) word_pairs
REDUCE (word_pairs.word, word_pairs.counts)
USING "./reducer.py"
AS word, counts
```

INCORRECT

```
hive> ADD FILE /path/to/file.py;
```

WordCount

```
ADD FILE /path/to/mapper.py;
ADD FILE /path/to/reducer.py;
FROM (
   FROM wikipedia_sample
   SELECT TRANSFORM (line)
   USING "./mapper.py" AS word, counts
   DISTRIBUTE BY word SORT BY word
) word_pairs
SELECT TRANSFORM (word_pairs.word, word_pairs.counts)
USING "./reducer.py"
AS word, counts
```

Summary

Summary

- You can **extend** Hive functionality with the help of streaming scripts (bash, Python, ...)

Summary

- You can **extend** Hive functionality with the help of streaming scripts (bash, Python, ...)
- You can **execute** streaming scripts on Map and Reduce phases (see: TRANSFORM; **danger** of: MAP / REDUCE)