1

Nulls and Pig Latin

There are two ways that you can run into a null value:

- Nulls may occur naturally in the data
- Nulls may be the result of an operation

It might sound kind of boring, but data scientists run into these problems constantly with real-world data!

Comparison operators:

= , !=, <, <= etc.

If either subexpression is null, then the result:

Is also null

Arithmetic operators:

+, -, *, / etc.

If either subexpression is null, then the result:

Is also null

2

Nulls and Pig Latin

There are two ways that you can run into a null value:

- Nulls may occur naturally in the data
- Nulls may be the result of an operation

It might sound kind of boring, but data scientists run into these problems constantly with real-world data!

Functions:

AVG, MIN, MAX etc.

If any elements are null, then the functions:

Ignore nulls

Operators:

GROUP, COGROUP, JOIN

If any elements are null, then the operators:

… It's a bit more complicated

Nulls and Pig Latin

There are two ways that you can run into a null value:

- Nulls may occur naturally in the data
- Nulls may be the result of an operation

It might sound kind of boring, but data scientists run into these problems constantly with real-world data!

```
A = LOAD 'characters.txt' AS (name:chararray,
age:int, episodes:int);
dump A;
(Elliot, 28, 45)
(Darlene, 24,)
(Edward, 45,)

C = GROUP A BY episodes;
dump C;
(45, {(Elliot,28,45)})
(,{(Darlene,24,),(Edward,45,)})
```

Records with a **null** group key are grouped together

Pig Built in Functions

According to Pig documentation there are six types of built in functions

Eval functions

Load/store functions

Math functions

String functions

Datetime functions

Tuple/bag/map functions

University of South Australia

5

Pig Built in Functions

According to Pig documentation there are six types of built in functions

Eval functions

Load/store functions

Math functions

String functions

Datetime functions

Tuple/bag/map functions

University of South Australia

Eval functions

AVG
Compute the average of numeric values in a single-column bag. Requires a preceding GROUP statement.

COUNT
SUM
SUBTRACT
MAX
MIN

6

Pig Built in Functions

According to Pig documentation there are six types of built in functions

Eval functions

Load/store functions

Math functions

String functions

Datetime functions

Tuple/bag/map functions

Math functions

SIN
SINH
ASIN

EXP

RANDOM

University of South Australia

7

Pig Built in Functions

According to Pig documentation there are six types of built in functions

Eval functions

Load/store functions

Math functions

String functions

Datetime functions

Tuple/bag/map functions

String functions

STRSPLIT

TRIM

UPPER

University of South Australia

8

Pig Built in Functions

According to Pig documentation there are six types of built in functions

Eval functions

Load/store functions

Math functions

String functions

Datetime functions

Tuple/bag/map functions

Datetime functions

SECONDSBETWEEN

HOURSBETWEEN

Tuple/bag/map functions

TOP

TOBAG
TOTUPLE
TOMAP

User Defined Functions (UDFs)

UDF development is supported in six programming languages; Java, Jython, JavaScript, Ruby, Groovy and Python.

Pig makes writing UDFs in Python really easy:

MyUDFs.py

```
from pig_util import outputSchema

@outputSchema('word:chararray')
def greetings():
        return "Hello, "
```

We would save that script in a file called something like 'myUDFS.py' and we'd have to **register** it from within a Pig Latin script.

We can then call it as if it was a regular in-built Pig function

MyPigLatinScript.pig

```
REGISTER 'myUDFS.py' using streaming_python as myudfs

characters = LOAD 'my_characters.txt' AS (name:chararray);
say_hello = FOREACH users GENERATE myudfs.greetings(), name
```

User Defined Functions (UDFs)

UDF development is supported in six programming languages; Java, Jython, JavaScript, Ruby, Groovy and Python.

Pig makes writing UDFs in Python really easy:

MyUDFs.py

```
from pig_util import outputSchema

@outputSchema('word:chararray')
def greetings(name):
    return "Hello, " + name
```

We would save that script in a file called something like 'myUDFS.py' and we'd have to **register** it from within a Pig Latin script.

We can then call it as if it was a regular in-built Pig function

MyPigLatinScript.pig

```
REGISTER 'myUDFS.py' using streaming_python as myudfs

characters = LOAD 'my_characters.txt' AS (name:chararray);
say_hello = FOREACH users GENERATE myudfs.greetings(name)
```

Our UDFs can also take input arguments

Piggy Bank

`http://svn.apache.org/repos/asf/pig/trunk/contrib/piggybank`

11

12