Aggregations, JOINs and Nested Queries

Apache Spark™ and Databricks® allow you to create on-the-fly data lakes.

In this lesson you:

- Use basic aggregations.
- Correlate two data sets with a join
- Use subselects

Basic aggregations

Using built-in Spark functions, you can aggregate data in various ways.

Run the cell below to compute the average of all salaries in the `People10M` table.

 By default, you get a floating point value.

Cmd 8

```
%sql
SELECT avg(salary) AS averageSalary
FROM People10M
```

▸ (1) Spark Jobs

averageSalary
72633.0076033

Command took 2.13 seconds -- by huseyinyilmaz01@gmail.com at 4/2/2020, 1:36:14 PM on test-cluster

Cmd 9

Convert that value to an integer using the SQL `round()` function. See the PySpark documentation for `round()` for more details.

Cmd 10

```
%sql
SELECT round(avg(salary),2) AS averageSalary
FROM People10M
```

▸ (1) Spark Jobs

averageSalary
72633.01

Cmd 11

In addition to the average salary, what are the maximum and minimum salaries?

Cmd 12

```
%sql
SELECT max(salary) AS max, min(salary) AS min, round(avg(salary)) AS average
FROM People10M
```

▸ (1) Spark Jobs

max	min	average
180841	-26884	72633

Joining two tables

Correlate the data in two data sets using a SQL join.

The `People10M` table has 10 million names in it.

> How many of the first names appear in Social Security data files?

To find out, use the `SSANames` table with first name popularity data from the United States Social Security Administration.

For every year from 1880 to 2014, `SSANames` lists the first names of people born in that year, their gender, and the total number of people given that name.

By joining the `People10M` table with `SSANames`, weed out the names that aren't represented in the Social Security data.

(In a real application, you might use a join like this to filter out bad data.)

Start by taking a quick peek at what `SSANames` looks like.

Cmd 16

```
%sql
SELECT * FROM SSANames
```

▸ (1) Spark Jobs

firstName	gender	total	year
Jennifer	F	54336	1983
Jessica	F	45278	1983
Amanda	F	33752	1983
Ashley	F	33292	1983
Sarah	F	27228	1983
Melissa	F	23472	1983
Nicole	F	22392	1983
Stephanie	F	22323	1983
Heather	F	20749	1983

Showing the first 1000 rows.

Next, get an idea of how many distinct names there are in each of our tables, with a quick count of distinct names.

Cmd 18

```
%sql
SELECT count(DISTINCT firstName)
FROM People10M
```

▸ (1) Spark Jobs

count(DISTINCT firstName)
5113

Command took 5.59 seconds -- by huseyinyilmaz01@gmail.com at 4/2/2020, 2:01:36 PM on test-cluster

Cmd 19

```
%sql
SELECT count(DISTINCT firstName)
FROM SSANames
```

▸ (1) Spark Jobs

count(DISTINCT firstName)
93889

Command took 3.98 seconds -- by huseyinyilmaz01@gmail.com at 4/2/2020, 2:01:39 PM on test-cluster

By introducing two more temporary views, each one consisting of distinct names, the join will be easier to read/write.

Cmd 21

```
%sql
CREATE OR REPLACE TEMPORARY VIEW SSADistinctNames AS
  SELECT DISTINCT firstName AS ssaFirstName
  FROM SSANames;

CREATE OR REPLACE TEMPORARY VIEW PeopleDistinctNames AS
  SELECT DISTINCT firstName
  FROM People10M
```

OK

Command took 0.42 seconds -- by huseyinyilmaz01@gmail.com at 4/2/2020, 2:02:25 PM on test-cluster

Cmd 22

Next, join the two tables together to get the answer.

Cmd 23

```
%sql
SELECT firstName
FROM PeopleDistinctNames
INNER JOIN SSADistinctNames ON firstName = ssaFirstName
```

▸ (4) Spark Jobs

firstName
Alayna
Melaine

How many are there?

Cmd 25

```
%sql
SELECT count(*)
FROM PeopleDistinctNames
INNER JOIN SSADistinctNames ON firstName = ssaFirstName
```

▸ (1) Spark Jobs

count(1)
5096

Cmd 26

Nested Queries

Joins are not the only way to solve the problem.

A sub-select works as well.

Cmd 28

```
%sql
SELECT count(firstName)
FROM PeopleDistinctNames
WHERE firstName IN (
  SELECT ssaFirstName FROM SSADistinctNames
)
```

▸ (1) Spark Jobs

count(firstName)
5096



Exercise 1

In the tables above, some of the salaries in the `People10M` table are negative

These salaries represent bad data.

Your job is to convert all the negative salaries to positive ones, and then sort t

 **Hint:** See the Apache Spark documentation, built-in functions.

Cmd 30

Step 1

Create a temporary view called `PeopleWithFixedSalaries`, where all the neg

Cmd 31

```
%sql
-- TODO

create or replace temporary view PeopleWithFixedSalaries as
select firstName, lastName, abs(salary) as salary from People10M
```

OK

Cmd 33

Step 2

Starting with the table `PeopleWithFixedSalaries`, create another view called `PeopleWithFixedSalariesSorted` where:

1. The data set has been reduced to the first 20 records
2. The records are sorted by the column `salary` in ascending order

Cmd 34

```
%sql
-- TODO

create or replace temporary view PeopleWithFixedSalariesSorted as
select * from PeopleWithFixedSalaries
order by salary asc
limit(20)
```

OK

Command took 0.02 seconds -- by huseyinyilmaz01@gmail.com at 4/2/2020, 2:29:26 PM on test-cluster

Cmd 35

Exercise 2

As a refinement, assume that all salaries under $20,000 represent bad rows and filter them out.

Additionally, categorize each person's salary into $10K groups.

Cmd 37

Step 1

Create a temporary view called `PeopleWithFixedSalaries20K` where:

1. Start with the table `PeopleWithFixedSalaries`
2. The data set excludes all records where salaries are below $20K
3. The data set includes a new column called `salary10k`, that should be the salary in groups of 10,000. For example:
   - A salary of 23,000 should report a value of "2"
   - A salary of 57,400 should report a value of "6"
   - A salary of 1,231,375 should report a value of "123"

Cmd 38

```
%sql
-- TODO
create or replace temporary view PeopleWithFixedSalaries20K as
select *, round(salary/10000) as salary10k from PeopleWithFixedSalaries
where salary >= 20000
order by salary10k

```

OK

Exercise 3

Using the `People10M` table, count the number of females named Caren who were born before March 1980.

Cmd 41

Step 1

Starting with the table `People10M`, create a temporary view called `Carens` where:

1. The result set has a single record
2. The data set has a single column named `total`
3. The result counts only
   - Females (`gender`)
   - First Name is "Caren" (`firstName`)
   - Born before March 1980 (`birthDate`)

Cmd 42

```
%sql
-- TODO

create or replace temporary view Carens as
select count(*) as total from People10M
WHERE firstName = 'Caren' AND birthDate < to_timestamp('1980-03-01')
```

OK

Command took 0.43 seconds -- by huseyinyilmaz01@gmail.com at 4/2/2020, 5:32:25 PM on test-cluster

Challenge Exercise 4

Use the `SSANames` table to find the most popular first name for girls in 1885, 1915, 1945, 1975, and 2005.

Cmd 6

Step 1

Create a temporary view called `HistoricNames` where:

1. The table `HistoricNames` is created using a **single** SQL query.
2. The result has three columns:
   - `firstName`
   - `year`
   - `total`

Hint: Explore the data before crafting your solution.

Cmd 7

```
%sql
create or replace temporary view HistoricNames as
SELECT year, firstName, total
FROM (SELECT firstName, year, total,
             rank() OVER (PARTITION BY year ORDER BY total DESC) as rank
      FROM (select firstName, year, max(total) as total from SSANames
            where gender = "F" and year in (1885, 1915, 1945, 1975, 2005)
            group by year, firstName
            order by total) ) tmp
WHERE rank = 1 ORDER BY total asc
```

OK

Cmd 8

```
%sql
select * from HistoricNames
```

▸ (2) Spark Jobs

year	firstName	total
1885	Mary	9128
2005	Emily	23928
1975	Jennifer	58185
1915	Mary	58187
1945	Mary	59284




Command took 13.28 seconds -- by huseyinyilmaz01@gmail.com at 4/2/2020, 9:14:45 PM on test-cluster

Cmd 9