Apache Spark SQL Partitioning & Bucketing

Sandhiya M · Follow

5 min read · May 13, 2022

Today Let's get to know Spark Partitioning Concepts..!!

> ***Hello Everyone...! Are You all excited to know what it is....? Come on let's get into it...!!!***

I'll Start with a Short Example:

➡We all might have seen an encyclopedia in our school or college library. It is a set of books that will give you information about almost anything.

➡So, Do you know what is the best thing about the encyclopedia?

➡Yes, as you guessed ,the words are arranged alphabetically. For example, if you have a word in mind say “Data Science”. You will directly go and pick up the book with the title “D”.

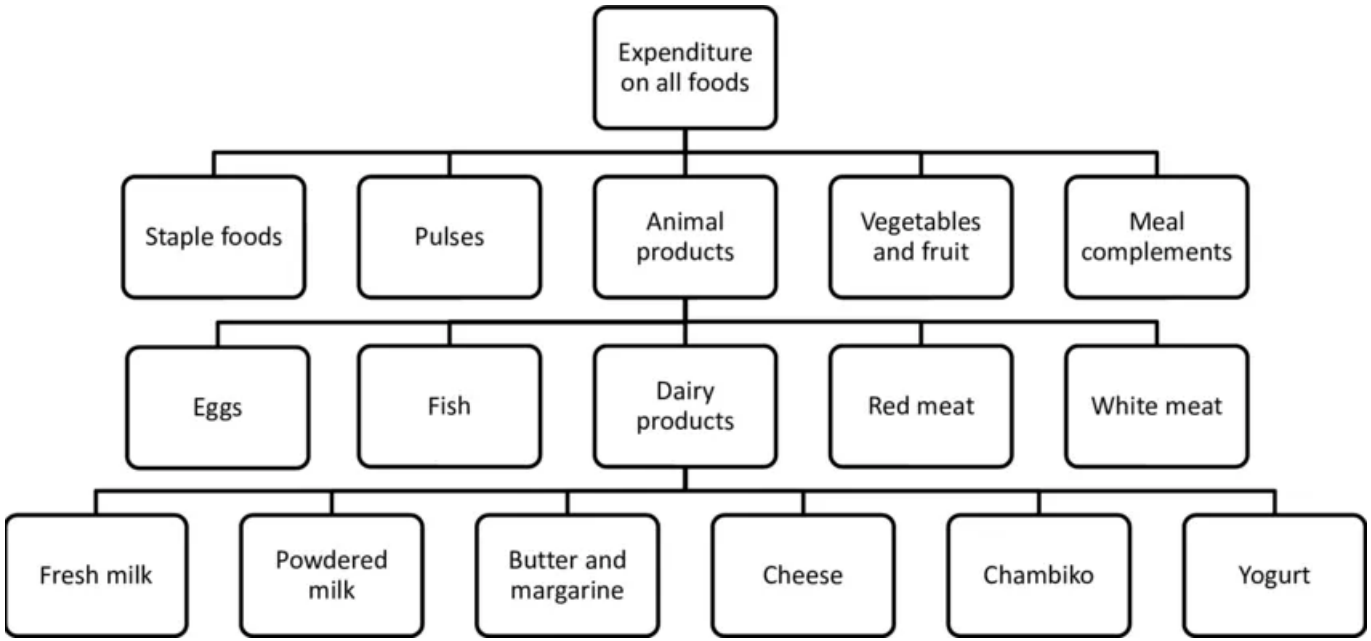
➡Can you imagine how tough would the task be to search for a single book if they were stored without any order?

➡Here storing the words alphabetically represents indexing, but using a different location for the words that start from the same character is known as bucketing.

Similar kinds of storage techniques like partitioning and bucketing are there in Apache spark so that we can get faster results for the search queries. In this article, we will see what is partitioning and bucketing, and when to use which one?

◉SO what is partitioning? and what it does.?

❶ It allows us to organize the table into multiple partitions where we can group the same kind of data together. It is used for distributing the load horizontally.



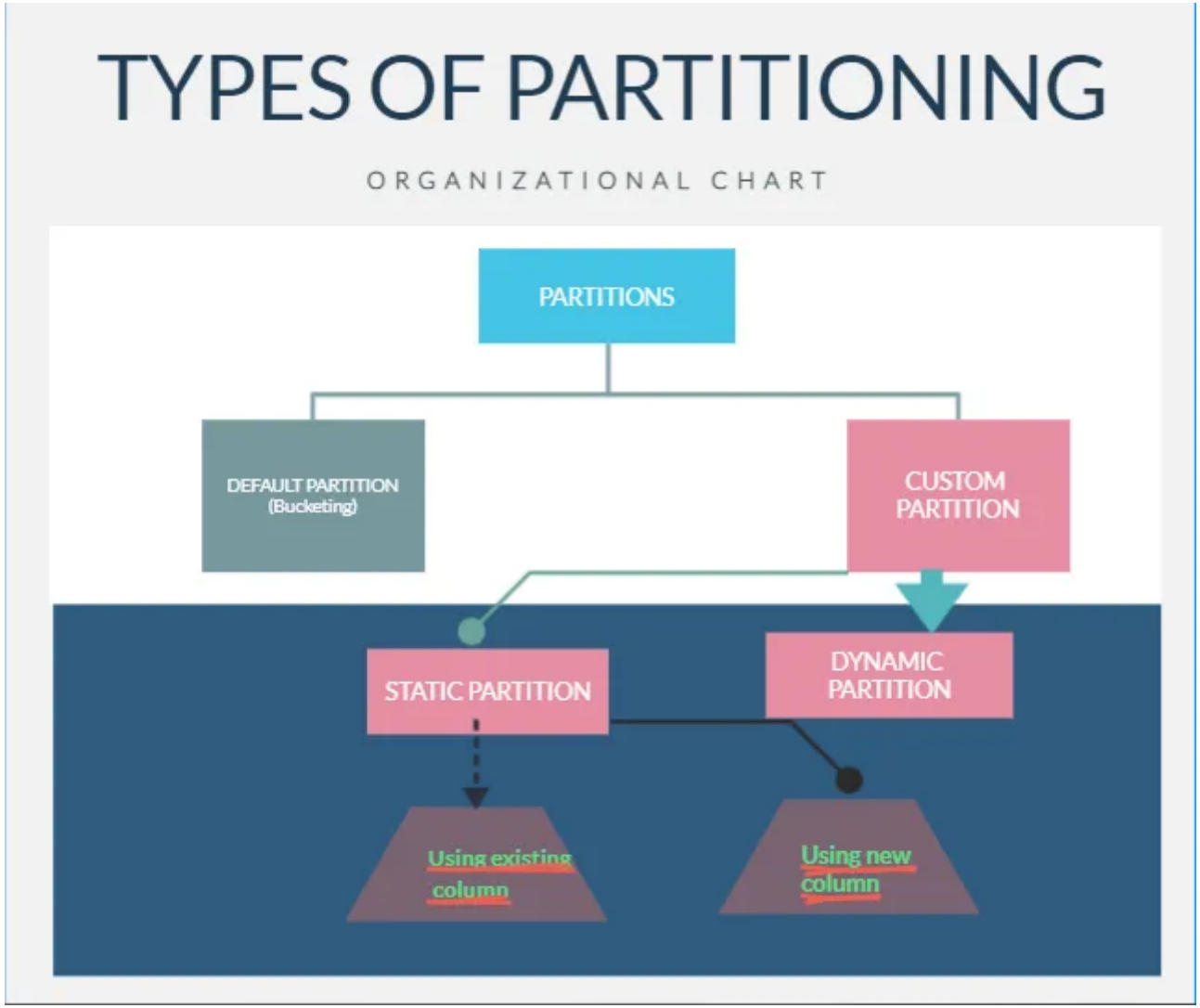
Example

Partitioning like this, the data gives us performance benefits and also helps us in organizing the data. Now, let's see when to use the partitioning in the spark.

Note:

> ⩥The difference between HIVE and SPARK is, Hive makes usage of MapReduce for storage purpose whereas spark will have its own engine. But both results the same output ⩤

Types of Partitioning



Let's get in-depth and see how it works:

➡STATIC PARTITIONING:

❶ Using existing column:

In this method, we are creating the partition based on a particular column that already exists in the table.

Step 1: Check the available databases and tables.

Step 2: Now Create a table

> *create table p_patient2(pid int,pname string,gender string,amt int)*
> ***partitioned by*** *(drug string);*

```
spark-sql> create table p_patient2(pid int,pname string,gender string,amt int)
         > partitioned by (drug string)
         > ;
22/05/10 11:18:19 WARN ResolveSessionCatalog: A Hive serde table will be created
 as there is no table provider specified. You can set spark.sql.legacy.createHiv
eTableByDefault to false so that native data source table will be created instea
d.
```

➡To describe the patient1 table give the below command

```
spark-sql> desc p_patient1;
pid                     int
pname                   string
drug                    string
gender                  string
amt                     int
d                       string
# Partition Information
# col_name              data_type
d                       string
```

Step 3: Insert the data and to select drug partition that contains only ***'Para'***

```
insert overwrite table p_patient2 partition(drug='Para')
select pid,pname,gender,tot_amt from patient where drug='Para';

insert overwrite table p_patient2 partition(drug='hcq')
select pid,pname,gender,tot_amt from patient where drug='hcq';
```

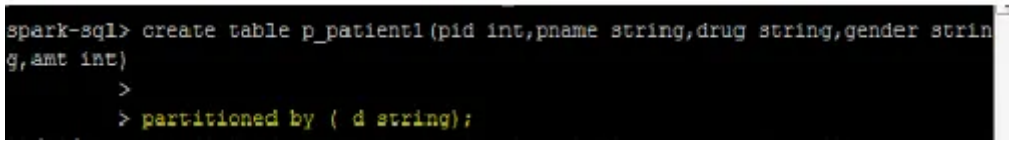
```
spark-sql> insert overwrite table p_patient2 partition(drug='Para')
         > select pid,pname,gender,tot_amt from patient where drug='Para';
Time taken: 0.8 seconds
spark-sql> insert overwrite table p_patient2 partition(drug='hcq')
         > select pid,pname,gender,tot_amt from patient where drug='hcq';
Time taken: 0.521 seconds
spark-sql> select * from p_patient2;
111     aaa     m       900     Para
222     bbb     f       999     Para
333     ccc     m       444     Para
111     aaa     m       500     Para
222     bbb     f       999     Para
333     ccc     m       444     Para
111     aaa     m       500     Para
222     bbb     f       999     Para
333     ccc     m       444     Para
111     aaa     m       900     Para
222     bbb     f       999     Para
333     ccc     m       444     Para
111     aaa     m       500     Para
222     bbb     f       999     Para
333     ccc     m       444     Para
111     aaa     m       500     Para
222     bbb     f       999     Para
333     ccc     m       444     Para
444     ddd     m       500     hcq
111     aaa     m       600     hcq
```

❷Using new column:

In the previous method we used a existing column to create a partition, But in this method we are going to create a new column by which the partition is going to take place.

Step 1: Creating a table with new column.

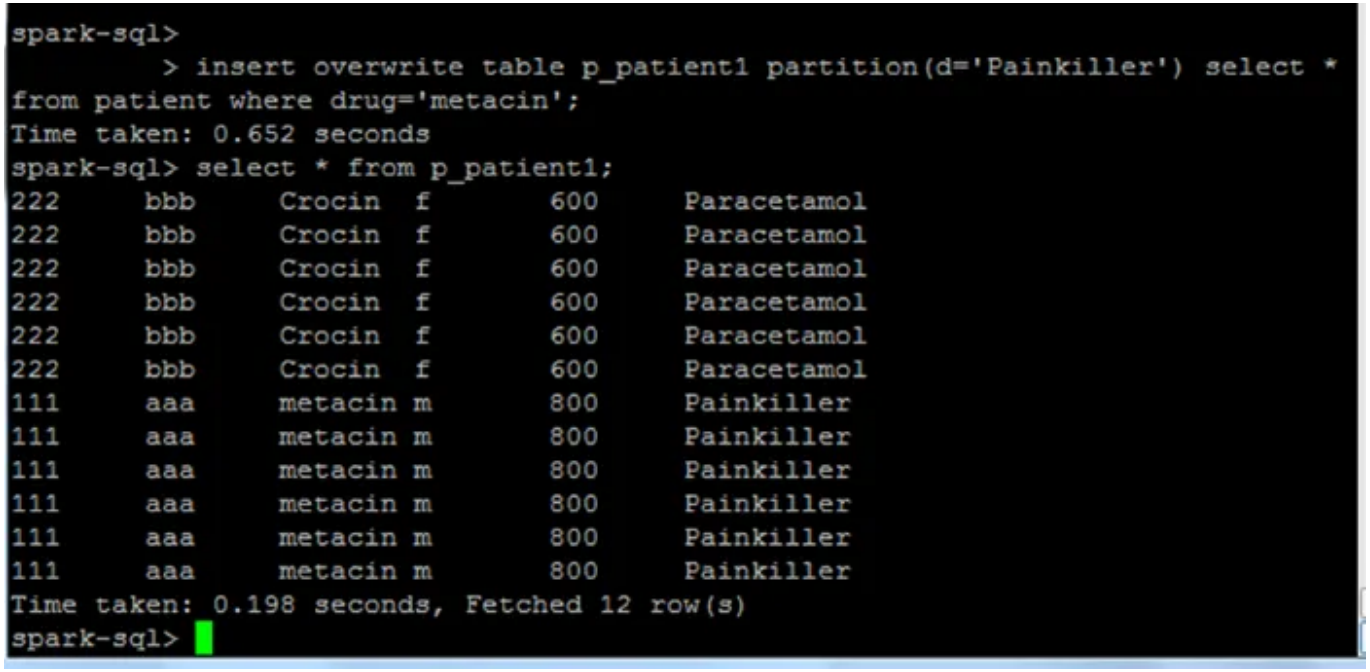
```
create table p_patient1(pid int,pname string,gender string,amt int)
partitioned by (d string);
```



Step-2: Insert the value into the table where the ‘ drug=’Para’ or drug =’Crocin’;

```
>>insert overwrite table p_patient1 partition(d='Paracetamol') select *
from patient Where drug='Para' or drug ='Crocin';
```

Data in our table p_patient1:



```
spark-sql>
         > insert overwrite table p_patient1 partition(d='Painkiller') select *
from patient where drug='metacin';
Time taken: 0.652 seconds
spark-sql> select * from p_patient1;
222     bbb     Crocin  f       600     Paracetamol
222     bbb     Crocin  f       600     Paracetamol
222     bbb     Crocin  f       600     Paracetamol
222     bbb     Crocin  f       600     Paracetamol
222     bbb     Crocin  f       600     Paracetamol
222     bbb     Crocin  f       600     Paracetamol
111     aaa     metacin m       800     Painkiller
111     aaa     metacin m       800     Painkiller
111     aaa     metacin m       800     Painkiller
111     aaa     metacin m       800     Painkiller
111     aaa     metacin m       800     Painkiller
111     aaa     metacin m       800     Painkiller
Time taken: 0.198 seconds, Fetched 12 row(s)
spark-sql>
```

➡Dynamic Partitioning :

- Dynamic partitioning can be performed on the hive external table and managed table.
- In Dynamic partitioning, there is no requirement of the where clause.
- If you want to perform partition on the tables without knowing the number of columns in that case you can use Dynamic partitioning.
- Let’s just ensure it by using the following code.

```
>>set hive.exec.dynamic.partition.mode=nonstrict
```

> ***Step 1:*** *Create a table*

```
spark-sql> create table p_patientdy(pid int,pname string,amt int)
         > partitioned by (drug string,gender string );
22/05/10 12:08:40 WARN ResolveSessionCatalog: A Hive serde table will be created
Default to false so that native data source table will be created instead.
22/05/10 12:08:40 WARN HiveMetaStore: Location: hdfs://localhost:9000/user/hive/
Time taken: 0.089 seconds
```

> ***Step 2:*** *Insert the data from patient without where clause*

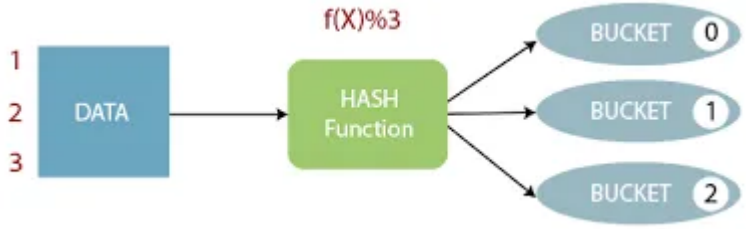
```
>>insert overwrite table p_patientdy partition(drug,gender)  select
pid,pname,tot_amt,drug,gender from patient ;
```

Inside each partition we will be having another sub partition based on the gender.

Now let's look into our Default (Bucketing) Partition.

Default Partition:(Bucketing)

➡ It is similar to partitioning in Hive with an added functionality that it divides large datasets into more manageable parts known as buckets.



- The concept of bucketing is based on the hashing technique.
- Here, modules of current column value and the number of required buckets is calculated (let say, F(x) % 3).
- Now, based on the resulted value, the data is stored into the corresponding bucket.

> ***Step 1:*** *Create Table.*

```
>> create table bucket_patient(pid int,pname string,drug string,gender
string,amt int)  clustered by (drug) into 4 buckets;
```

```
spark-sql> create table bucket_patient(pid int,pname string,drug string,gender string,amt int)
         > clustered by (drug) into 4 buckets;
22/05/10 12:29:27 WARN ResolveSessionCatalog: A Hive serde table will be created as there is no table provider specified. You can set spark.sql.le
gacy.createHiveTableByDefault to false so that native data source table will be created instead.
22/05/10 12:29:28 WARN HiveMetaStore: Location: hdfs://localhost:9000/user/hive/warehouse/sal.db/bucket_patient specified for non-external table:b
ucket_patient
Time taken: 0.912 seconds
spark-sql>
```

> ***Step 2:*** *Before passing on the data into our table we have set some properties.*

```
>>SET hive.enforce.bucketing=false ;

>>SET hive.enforce.sorting=false ;
```

> ***Step 3:*** *Insert data from patient.*


```
>>insert overwrite table bucket_patient select * from patient;
```

> ***Step 4-*** *Select the data of buckets.*

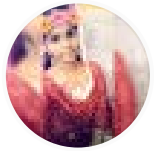
We will be able to view contents of a particular Bucket using the below code.

```
spark-sql> select * from bucket_patient1 TABLESAMPLE(BUCKET 1 OUT OF 4);
333     ccc     calpol  f       500
444     ddd     hcq     m       500
222     bbb     Para    f       999
333     ccc     calpol  f       500
222     bbb     Crocin  f       600
333     ccc     Para    m       444
111     aaa     metacin m       800
222     bbb     Para    f       999
333     ccc     calpol  f       500
444     ddd     hcq     m       500
333     ccc     Para    m       444
444     ddd     hcq     m       500
111     aaa     hcq     m       600
555     eee     cetzine m       700
Time taken: 0.135 seconds, Fetched 14 row(s)
spark-sql>
```

> *Hope this blog will help you a lot to understand what exactly is partition , what is Static partitioning , What is Dynamic partitioning . We have also covered various advantages and disadvantages of spark partitioning.*
>
> *If you have any query related to spark Partitions, so please leave a comment. We will be glad to solve them....!!*

Will catch you all in the next blog .....!

Big Data Spark Data Science Data Analysis Python

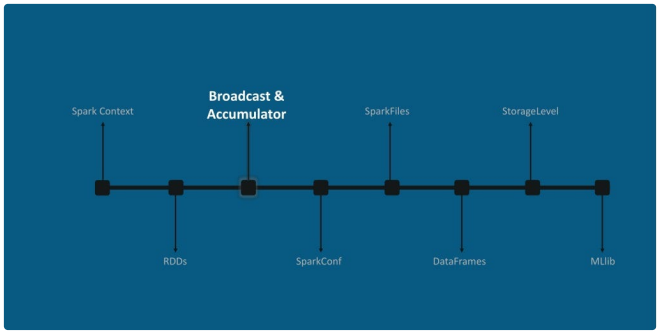


Written by Sandhiya M

47 Followers

Follow

More from Sandhiya M



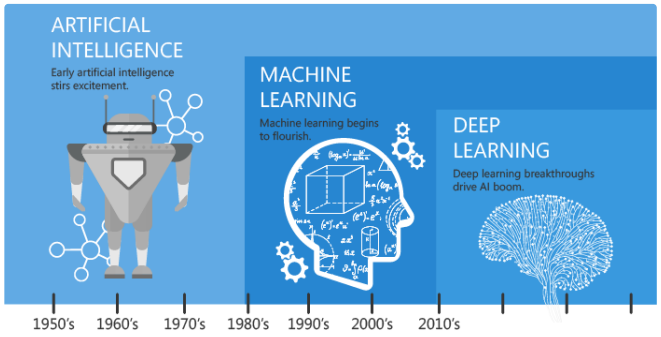
Sandhiya M in Towards Dev

PySpark Broadcast variables and Accumulator

Hello Everyone! Today let's look into broadcast and accumulator variables in...

4 min read · May 9, 2022

87



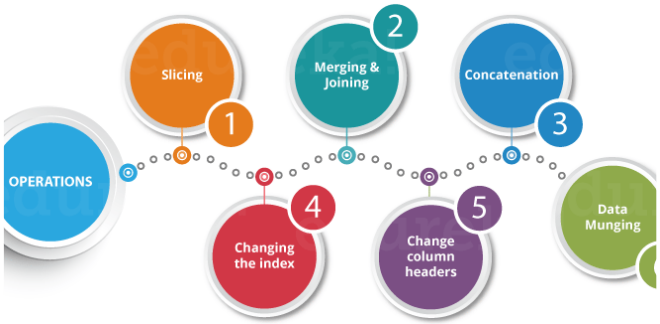
Sandhiya M

Explanation of Principal Component Analysis (PCA)

A Step-by-Step Explanation of Principal Component Analysis (PCA)

7 min read · May 31, 2022

21



Sandhiya M

A sneak peek on PANDAS..!

“Never stop learning because life never stops teaching.”

5 min read · Apr 13, 2022

110

```
c.count('tas')
```

```
1
```

```
c.append('tas')
```

Sandhiya M

DATA STRUCTURES IN PYTHON

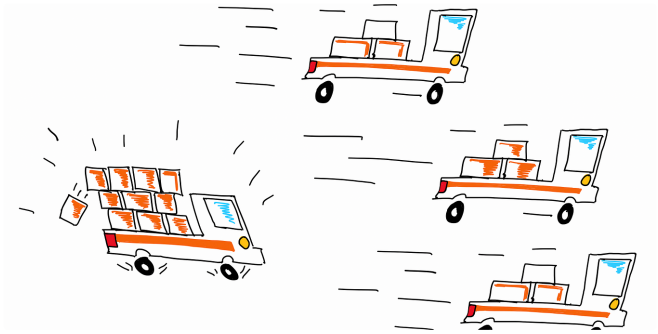
In this article, we will discuss the Data Structures in the Python Programming...

4 min read · Apr 6, 2022

53 1

See all from Sandhiya M

Recommended from Medium



Suffyan Asad

Handling Data Skew in Apache Spark: Techniques, Tips and Trick...

Discover how to detect and mitigate data-skew in Spark. Learn about the impact of...

12 min read · Jan 31

22

Zaid Erikat

Handling Skewed Data in Apache Spark

What is Spark?

8 min read · Apr 30

12

Lists

What is ChatGPT?

9 stories · 13 saves

Stories to Help You Level-Up at Work

19 stories · 11 saves

Staff Picks

294 stories · 55 saves

Gaurav Patil in Globant

How to solve the “large number of small files” problem in Spark

A solution for the “large number of small files” problem

8 min read · Nov 25, 2022

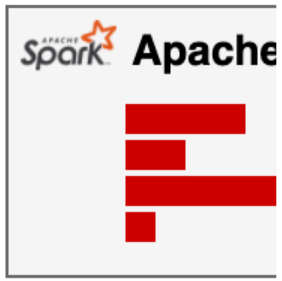
57 1

Chenglong Wu

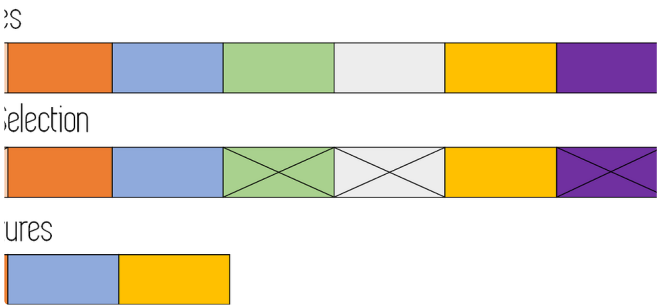
The Spark 5S Optimization Series, Part 1: Unleashing the Power of...

Optimizing Shuffle in Apache Spark: Strategies to Improve Performance Using th...

5 min read · Apr 23



Manish Shrivastava



shorya sharma

Data Skew Problem and different ways to resolve it in Pyspark

Introduction:

4 min read · Mar 6

1

Variable Selection In Pyspark: Part 1

Variable Selection , also known as feature selection, is a process that can be used to...

5 min read · Feb 14

19 1

See more recommendations