FUJITSU

FUJITSU Software NetCOBOL V12.2

Hadoop Integration Function User's Guide

Linux(64)

J2UL-2325-02ENZ0(00)
May 2020

Preface

The Hadoop integration function works in concert with "Interstage Big Data Parallel Processing Server" and "Apache Hadoop". To use the Hadoop integration function, knowledge of "Interstage Big Data Parallel Processing Server" and "Apache Hadoop" is required.

- For information about "Interstage Big Data Parallel Processing Server", refer to "Interstage Big Data Parallel Processing Server User's Guide."
- The replacement for the Interstage Big Data Parallel Processing Server is the Big Data Integration Server. For information about combining NetCOBOL with the Big Data Integration Server, contact your Big Data Integration Server representative.
- For information about "Apache Hadoop", refer to https://hadoop.apache.org/.

Purpose of This Manual

This manual explains the operating procedures for the COBOL program using the Hadoop integration function.

Intended Readers

This manual is for users who operate the COBOL programs using the Hadoop integration function.

Abbreviations

Names of products described in this manual are abbreviated as:

Product Name	Abbreviation
Red Hat(R) Enterprise Linux(R) 8 (for Intel64)	Linux
Red Hat(R) Enterprise Linux(R) 7 (for Intel64)	or
Red Hat(R) Enterprise Linux(R) 6 (for Intel64)	Linux(64)

"Linux(64)" is a 64-bit mode of Linux.

Trademarks

- Linux(R) is the registered trademark of Linus Torvalds in the U.S. and other countries.
- Red Hat and Red Hat Enterprise Linux are trademarks or registered trademarks of Red Hat, Inc. in the United States and/or other countries.
- Intel is a trademark of Intel Corporation or its subsidiaries in the U.S. and/or other countries.
- Apache Hadoop, Hadoop and HDFS are trademarks of the Apache Software Foundation in the United States and other countries.
- All other trademarks are the property of their respective owners.

Export Controls

Exportation/release of this document may require necessary procedures in accordance with the regulations of your resident country and/or US export control laws.

The contents of this manual may be revised without prior notice. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Fujitsu Limited.

May 2020

Copyright 2013-2020 FUJITSU LIMITED

Contents

Chapter 1 Overview

In recent years, problems have occurred in mission critical batch processing due to an increase in the amount of data. Because of this, approaches have been taken to enhance the server's performance to reduce the batch processing time, and to perform multi-processing on the data after splitting it. With such approaches expensive hardware becomes necessary, and new processing becomes necessary for multi-processing and problems related to high-cost occur.

By using the Hadoop integration function, sequential processing performed by traditional batch processing can be accelerated by splitting data and performing parallel processing on multiple servers. Since existing applications can be used, compatibility and upgradability is ensured.

As a result, there are the following advantages besides accelerating processing.

- Low Cost

  By performing parallel processing on relatively inexpensive servers, an economical system can be configured.

- High Availability

  By splitting the duplex data, reliability is improved. Even if the server (Slave Server) performing parallel processing goes down, another other Slave Server takes over and continues the processing.

- Scale out

  Scale out can be performed easily by adding Slave Servers.

In Hadoop integration function, distributed processing is performed by using the MapReduce of Hadoop.

1.1 MapReduce

MapReduce is a programming model and framework for distributed processing of large amounts of segmented data in a cluster. In the MapReduce framework, distributed processing is implemented in the following three steps:

- Map Task
- Shuffle&sort
- Reduce Task

Map Task and Reduce Task can run any application created by the user.

The flow of batch processing using MapReduce is shown below. The example shows the output processing of the total of every product by entering data from voucher numbers, product names and sales counts.

1. Input File available on DFS (*) is split and passed on to each Map Task. ([1])
2. During Map Task, filtering of input data is performed. ([2])

   In the example, product name and sales count are extracted from each record, and are required for totalization.
3. Extracted data is passed to Shuffle&sort. ([3])
4. In Shuffle&sort, Sorting and Aggregation of records where data is taken as key, is performed. ([4])

   Aggregation of record uses product name as key.
5. Records are sorted with sales as a key and passed to each Reduce Task. ([5])
6. During Reduce Task, totalization processing of record is performed. ([6])

   Aggregated sales of the identical product are calculated.
7. Output data is stored on DFS. ([7])

*: Depending on the integration software, DFS refers to the following file systems:

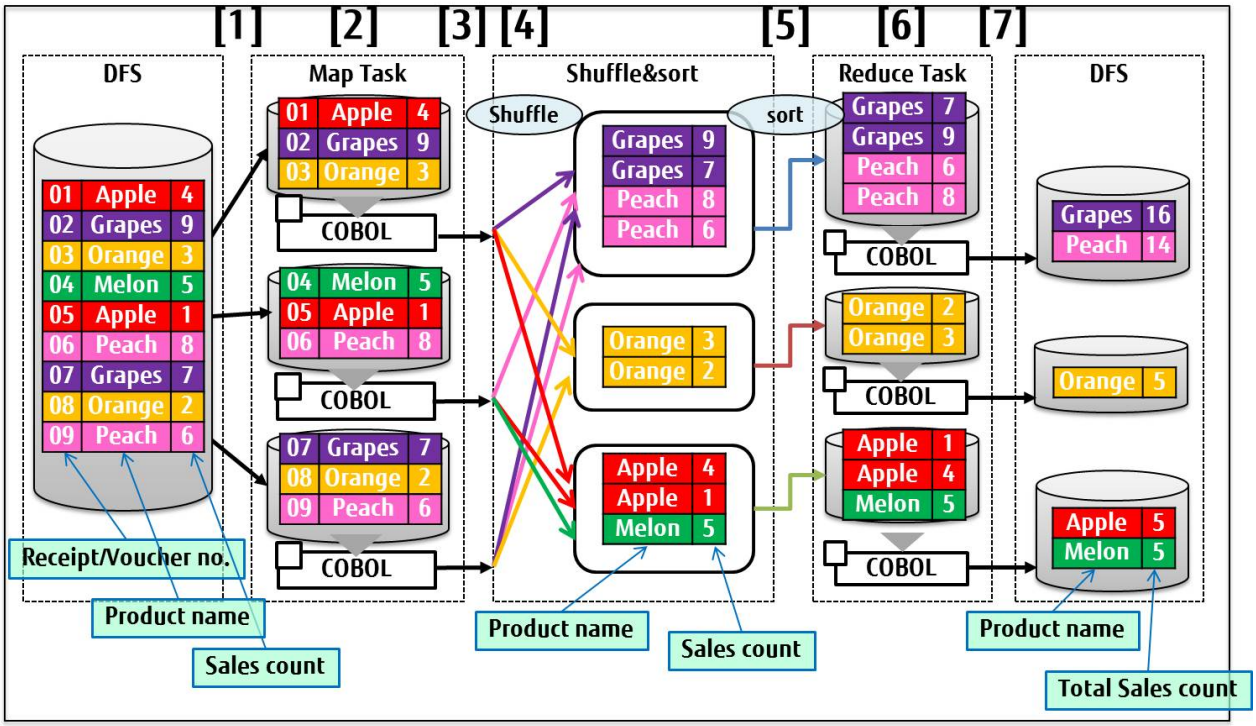
- In the case of integrating with Interstage Big Data Parallel Processing Server:

  Any distributed file system of your choice usable with Interstage Big Data Parallel Processing Server.

- In the case of integrating with Apache Hadoop:

  Hadoop distributed file system (HDFS: Hadoop Distributed File System).

Figure 1.1 MapReduce Data Flow



1.2 Behavior of MapReduce Framework

In Map Reduce framework, the task unit requested by the user is called Hadoop Job. Hadoop job consists of the following three elements:

- Input Data
- MapReduce application
- Configuration Information

In this Hadoop job, the segmented unit, which can be parallel processed in distributed processing, is called as Task.

Hadoop job is executed by integrating with the following 3 functionalities available in Map Reduce framework:

- JobClient

  JobClient receives Hadoop job request from the user and sends it to JobTracker.

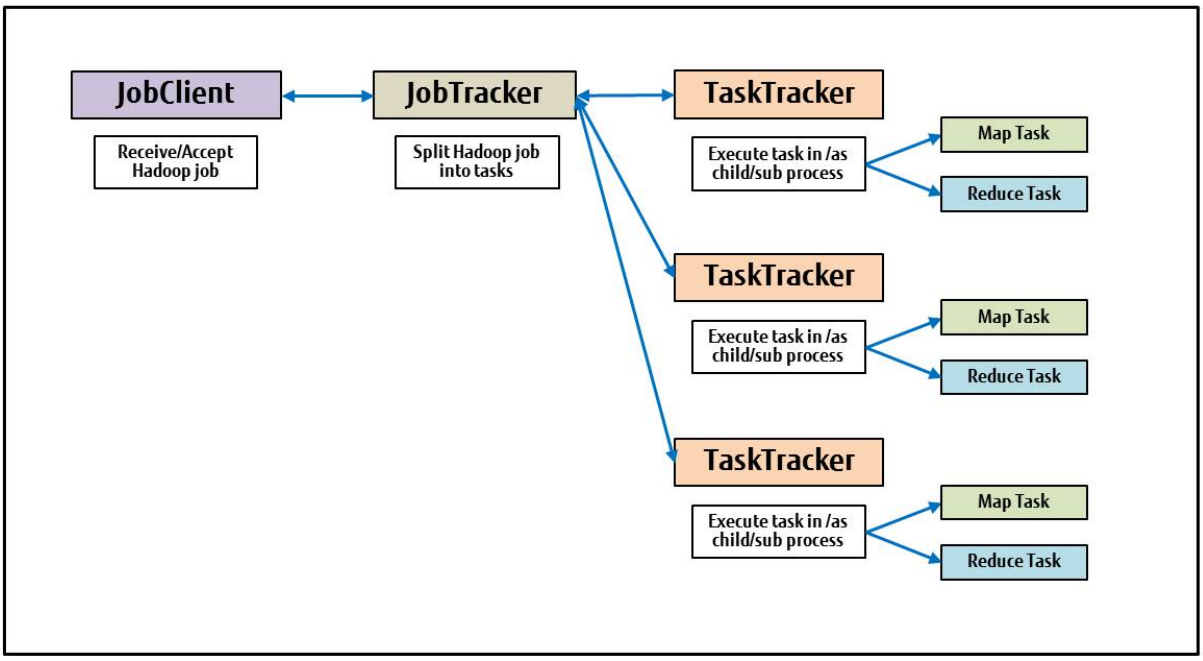
- JobTracker

  Hadoop job request sent by the JobClient is divided into task optimum for distributed processing and assigned to each TaskTracker.

- TaskTracker

  Function to execute Map Task and Reduce Task assigned by the JobTracker. Each task is executed on child process.

Figure 1.2 Role of MapReduce Framework Function



1.3 Hadoop Integration Features and Overview

The Hadoop integration function has the following features:

- Each Map Task and Reduce Task can be executed as a COBOL application.
  (Hereafter this COBOL application will be called MapReduce)
- It is possible to access the COBOL file through a READ/WRITE statement with the MapReduce application.
- Multiple files can be given as input.

Input and output files of MapReduce applications are represented by the following nicknames:

- Hadoop input data file
- Map input data file
- Map output data file
- Reduce input data file
- Reduce output data file
- Hadoop output data file

It is possible to use the following file organization with these files:

- Line sequential Files
- Record Sequential Files
- Physical sequential Files

To access these files with the MapReduce application, a unique name or file structure must be specified in advance in the configuration file. This configuration file is called "MapReduce Configuration File". In addition, the COBOL Runtime system automatically performs the allocation of the file that will be the target of input and output processing. For this reason, the user does not need to specify the file name in regard to the file identifier.

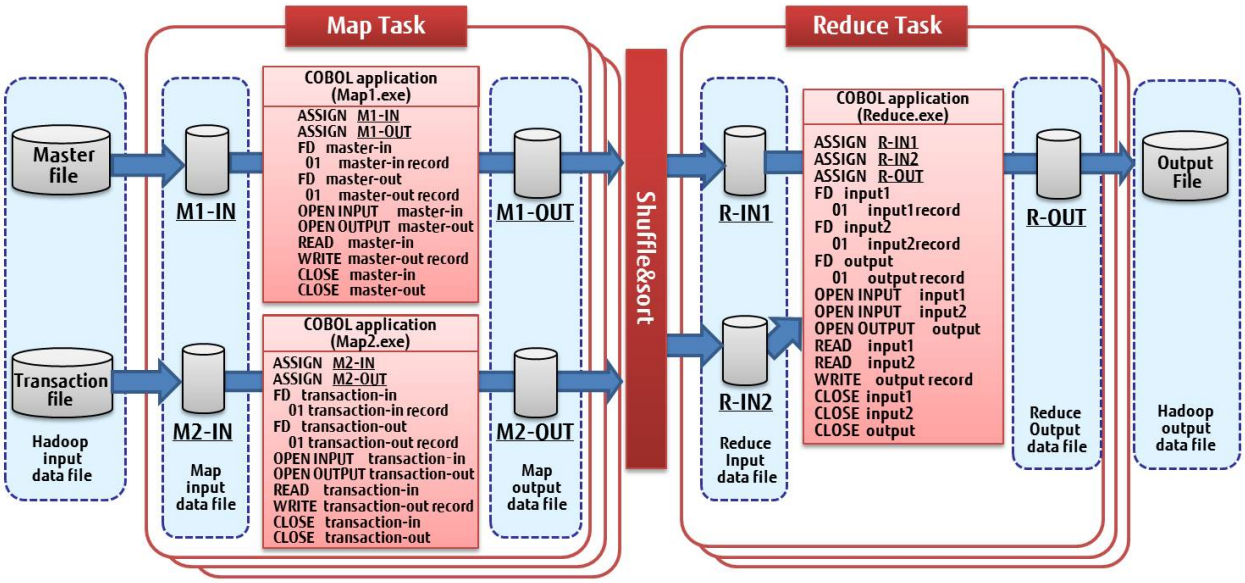
Note

Hadoop can use the following physical sequential file record types

- Physical sequence file of variable-length record form (format V record) (*) and
- File without BDW (Block Descriptor Word) and
- A file with RDW (Record Descriptor Word).

* In regards to a fixed-length record format (F-type record), it can be used as an order of a record fixed-length file.

Concept of MapReduce application and input output file, MapReduce Configuration File is shown in the following diagram.



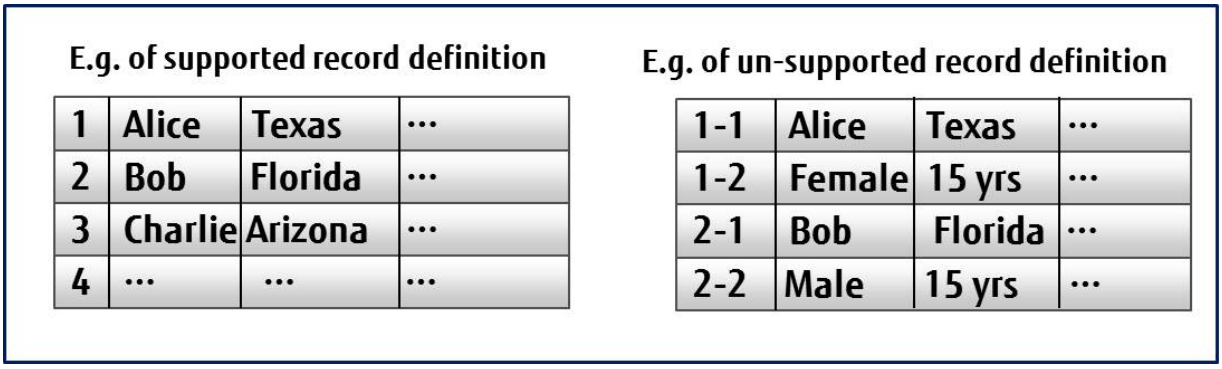
Hereafter, explanations regarding each file and Map Task, Shuffle&sort, and Reduce Task are given.

1.3.1 Hadoop Input data file

The divided file passed on to each Map Task is called "Hadoop Input Data File". A Hadoop input data file must be a data file that can be split, a transaction file, or a master file used as Hadoop Input data file.

Since the unit for splitting the Hadoop Input file is record, dependency of records cannot be used.

Figure 1.3 Possible record definition to use with Hadoop Input data file



Information

The file passed to the Map Task is split automatically, but it is not split in the middle of a record.

1.3.2 Map Task

In the Map Task, the Hadoop input file is simply divided into a fixed size that can be read as a COBOL file. The file assigned to this Map Task is called "Map Input data file". After processing each record as needed, it is subsequently passed as a COBOL file to Shuffle&sort. The file passed is called "Map Output data file".

In the Map Task, files simply divided by the MapReduce framework are taken as input and all records are processed in the same way. For this reason, processing the input file after splitting is implemented in the Map Task in such a way that there should be no problem.

Information

The Map Task can be omitted. If it is omitted, the Hadoop input data file is passed on to Shuffle&sort as a Map output data file.

1.3.3 Shuffle&sort

Based on the specified key, Shuffle & sort are grouped by the primary key record that was passed from the Map Task. It is also possible to sort the records based on the primary key and the secondary key. The sorted record is subsequently passed to the Reduce Task. It is possible to specify multiple primary keys and multiple sub-keys to the key information.

The COBOL data type or CSV format data can be specified to the key.

Information

In the key information, different types of sort (ascending/descending) order can be specified for each key

1.3.4 Reduce Task

In the Reduce Task, records that are grouped based on the specified key can be read as a COBOL file. This file is called "Reduce Input data file".

After processing the record as needed, it is output as a COBOL file. This outputted file is called "Reduce Output Data File".

In the Reduce Task, records that are grouped based on the primary key are passed from Shuffle & sort. Moreover, the record is already aligned with the primary key and the secondary key. For this reason, processing such as totalization of data, and comparison is implemented in the Reduce Task.

Information

The Reduce Task can be omitted. If it is omitted, the Reduce input data file is outputted as the Reduce output data file.

If you omit a Reduce Task, then please specify "0" as the property "mapred.reduce.tasks" of Hadoop.

1.3.5 Hadoop Output data file

The file outputted by each Reduce Task is called "Hadoop Output Data File". The Hadoop output data file is the COBOL file outputted by Reduce Task. A Hadoop output data file is generated for each Reduce Task that was executed.

1.4 Operations applicable to Hadoop integration function

Hadoop integration function possesses operations possible to perform from the characteristics of parallel distributed processing framework.

Following is an example showing the actions possible with Hadoop.

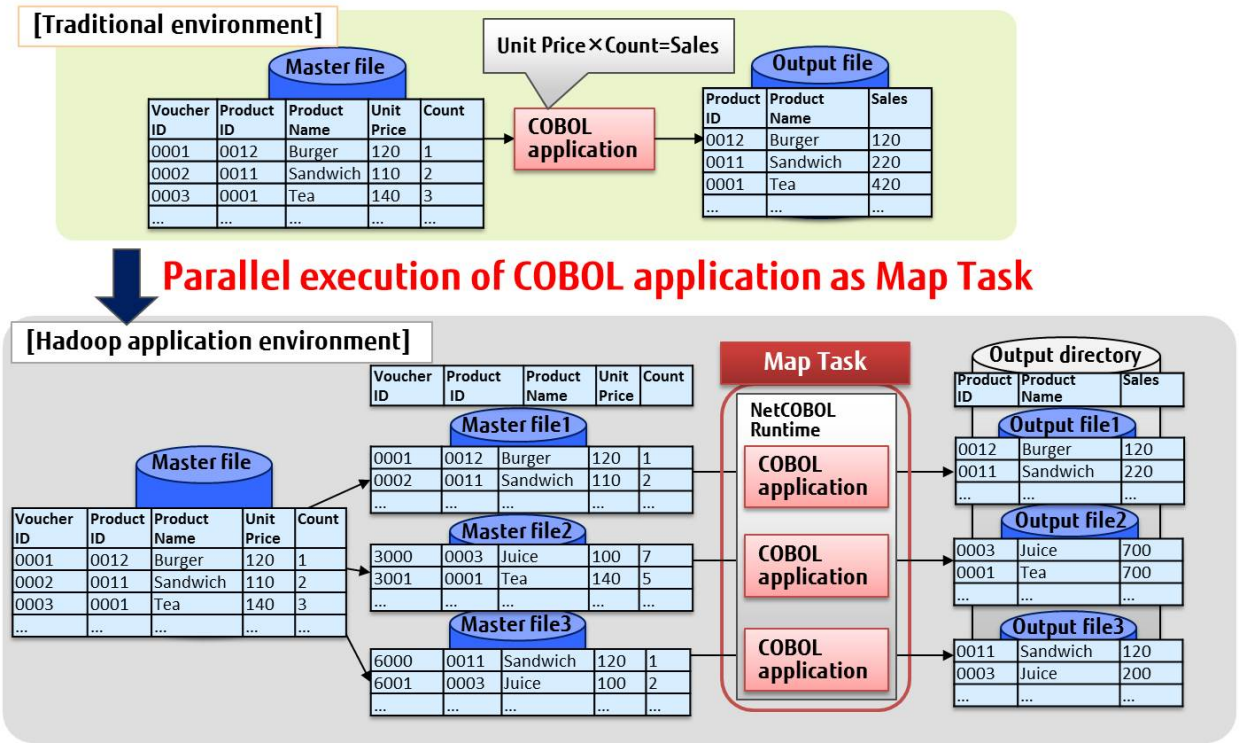
- Simple processing of the record
- Aggregation of records with a specific key
- Comparison processing between files

An example of these is explained below.

1.4.1 Simple processing of the record

Simple processing of records can be performed through Map Task. In the following example, the master file is read and used to calculate the unit price and quantity, and output a sales file for each product.

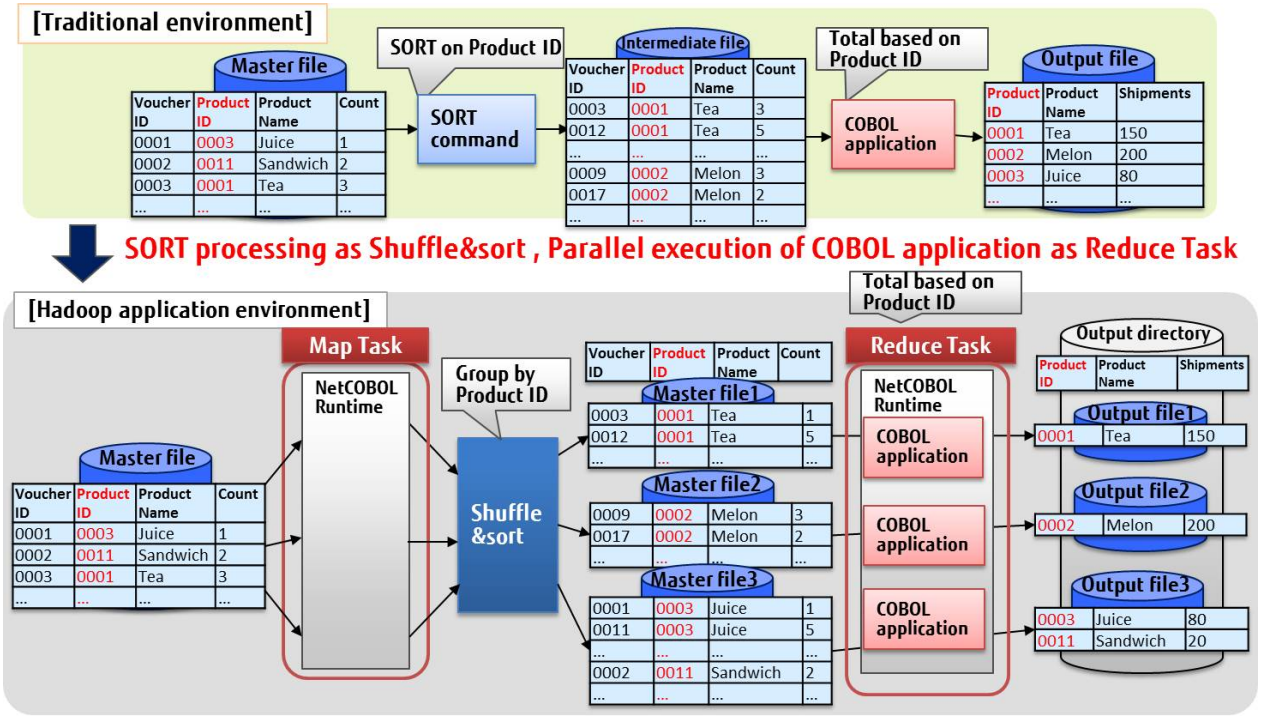
Figure 1.4 Example for processing to perform simple processing of the record



1.4.2 Aggregation of records

The process of aggregation of records after aligning can be performed by Reduce Task. In the following example, after arranging the master file based on the product ID, and then totaling the quantity for each product ID, the shipment count file for each product ID is generated.

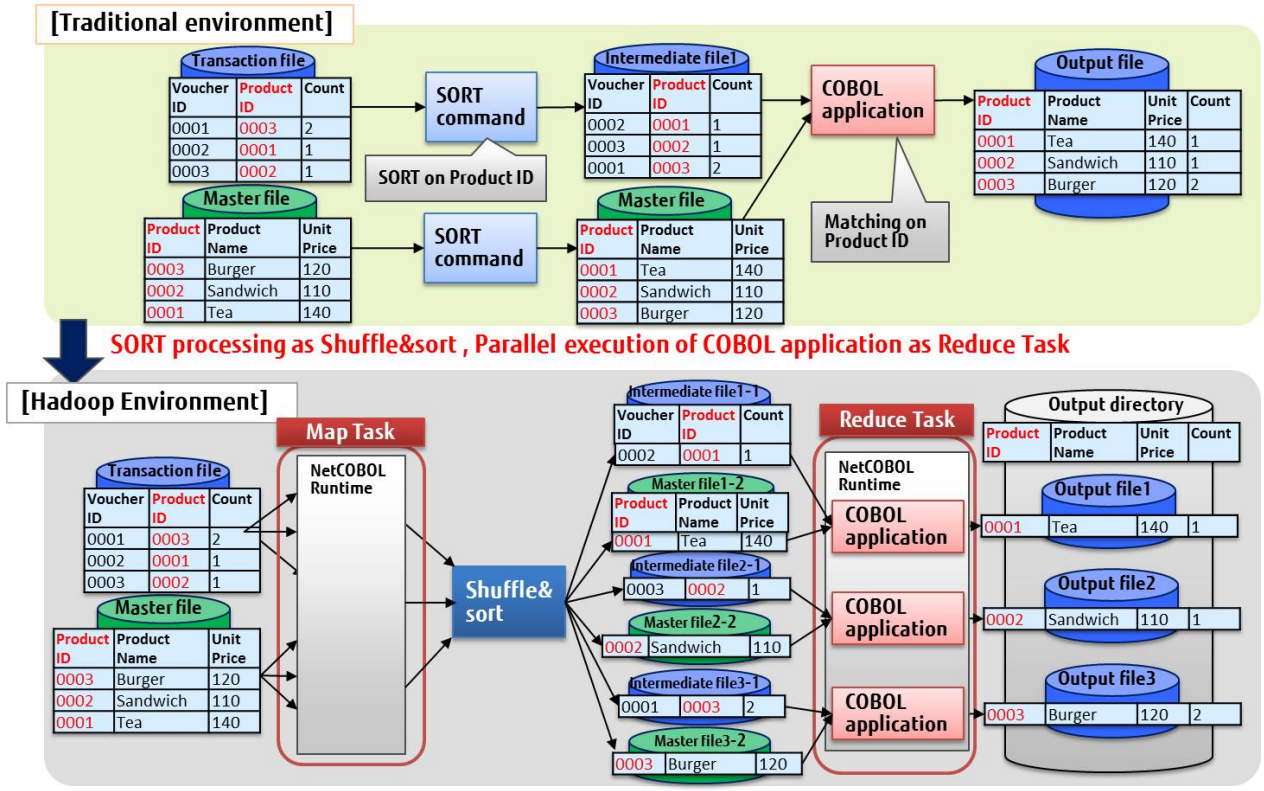
Figure 1.5 Example for Aggregation of records processing



1.4.3 Comparison processing between files

To align the records of multiple files, the Reduce Task can perform processing to match the keys. In the following example, the master file and transcript file are sorted after aligning them based on product ID. A file having product name, unit price, and quantity for each product ID is generated.

Figure 1.6 Example for Comparison processing between files



Chapter 2 Using the Hadoop integration function

This chapter explains the Hadoop integration function.

The steps may differ depending on the Integration software; refer to the following in accordance with the software used.

- Integration with Interstage Big Data Parallel Processing Server:

  System configuration for integration with Interstage Big Data Parallel Processing Server

  Processing procedure for Interstage Big Data Parallel Processing Server

- Integration with Apache Hadoop:

  System configuration for integration with Apache Hadoop

  Processing procedure for Apache Hadoop

2.1 System configuration for integration with Interstage Big Data Parallel Processing Server

The system is constructed with the following hardware configurations with the Interstage Big Data Parallel Processing Server.

- Master Server

  A file with big data is split into blocks and turned into a distributed system file. Master Server is the centralized server that manages the file name and storage location. Moreover, it can accept requests for execution on Hadoop and can run parallel distributed processing for the Slave Server.

- Slave Server

  By splitting data files into blocks by Master Server, and performing parallel distributed processing on multiple Slave Servers, analysis processing can be done in a short span of time.

- Server for Development Execution

  Server where Pig or Hive is installed to facilitate the development of parallel distributed applications (MapReduce).

- Integration Server

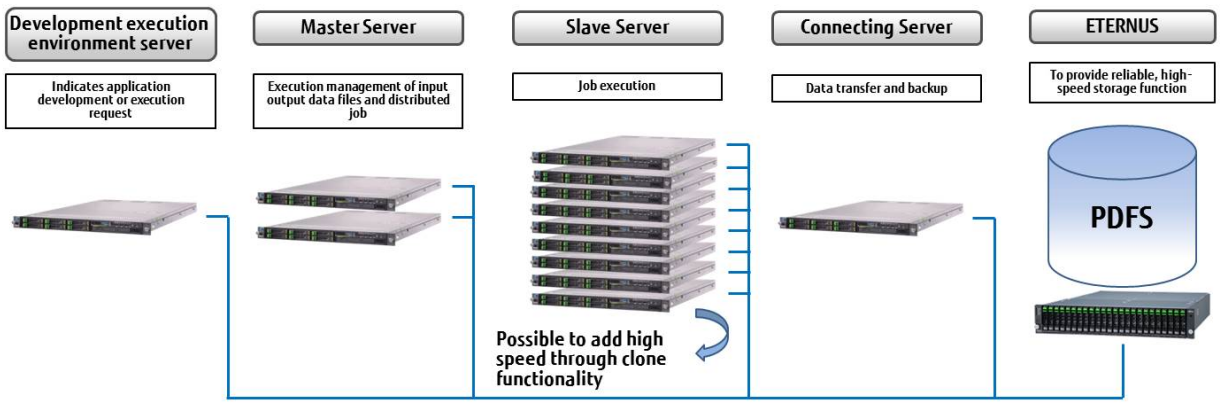
  Server to directly transmit the big data available on Production server and perform the analysis.

- ETERNUS

  Storage to store the input and output data used in the Hadoop.

Moreover, with the Interstage Big Data Parallel Processing Server in addition to the Hadoop distributed file system (HDFS) any other distributed file system (DFS) can be used. Hadoop processing will be performed after transferring data of a business application (HDFS). If a file system of one's own choice is used, then the data transfer will be unnecessary and processing time can be significantly reduced.

DFS is the common storage area accessible from all servers that comprise the Interstage Big Data Parallel Processing Server.



MapReduce is executed on the Slave Server. For this reason, it is necessary to have NetCOBOL Runtime on the Slave Server.

With the Hadoop integration function, Hadoop job is executed on the development execution environment server. For this reason, it is necessary to have a NetCOBOL development execution environment server. Moreover, if the COBOL application will be built on a development execution environment server, then on the development execution environment server, NetCOBOL's development and Runtime environment is necessary. If the COBOL application built on a different server is used, NetCOBOL's Runtime will only be installed on the development execution environment server.

Information

Even if there are multiple Slave Servers, with the clone functionality of "Smart Setup", installation of NetCOBOL can be performed in a short period of time. For information on "Smart Setup", refer to "Interstage Big Data Parallel Processing Server User's Guide."

2.2 System configuration for integration with Apache Hadoop

The system is constructed with the following hardware configurations with the Apache Hadoop.

- Master Server

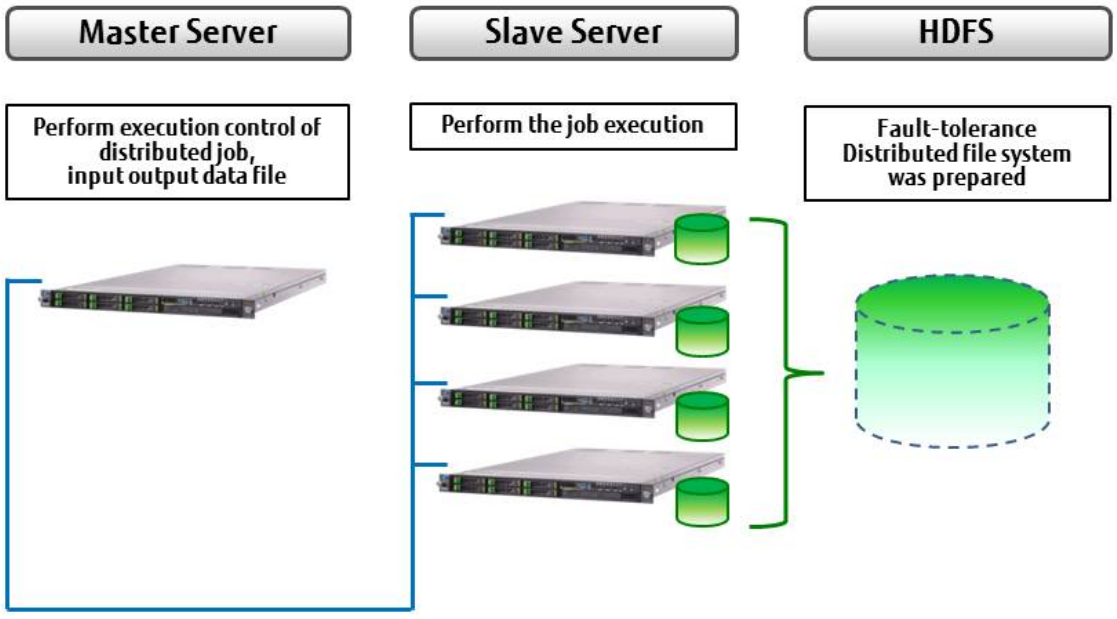
  File with big data is split into blocks and turned into a distributed system file. Master Server is the centralized server that manages the file name and storage location. Moreover, it can accept a request for execution on Hadoop and can run parallel distributed processing for the Slave Server.

- Slave Server

  By splitting data files into blocks by Master Server, and performing parallel distributed processing on multiple Slave Servers, analysis processing can be done in a short span of time.

- HDFS

  Distributed file system to store data files divided into blocks. HDFS is configured by mounting a local disk onto the Slave Server.



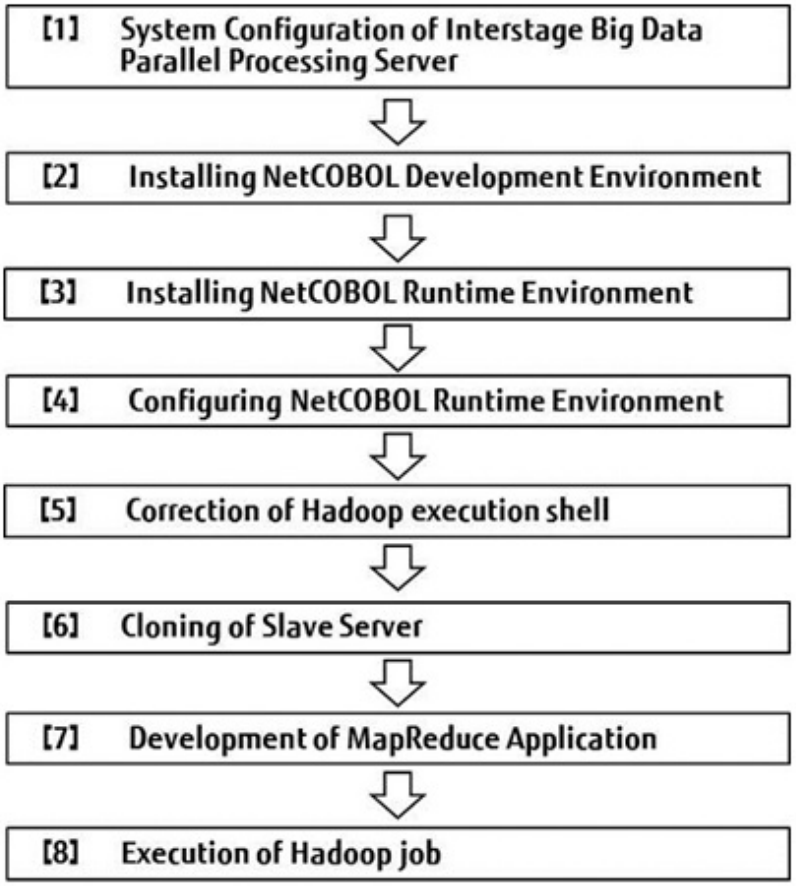
The MapReduce application is executed on the Slave Server. For this reason, it is necessary to have the NetCOBOL Runtime environment on a Slave Server.

With the Hadoop integration function, the Hadoop job is executed on the Master Server. For this reason, it is necessary to have NetCOBOL Runtime environment on the Master Server. Moreover, if the COBOL application will be built on the Master Server then NetCOBOL's

development and Runtime environment is necessary on the Master Server. If the COBOL application is built on a different server, NetCOBOL's Runtime will only be installed on the Master Server.

2.3 Processing procedure for Interstage Big Data Parallel Processing Server

The steps for using the Hadoop integration function are shown below.



[1] System Configuration of Interstage Big Data Parallel Processing Server

Interstage Big Data Parallel Processing Server is installed. Proceed to step [2] after completing the acquisition of first Slave Server.

[2] Installing NetCOBOL Development Environment

NetCOBOL's development and Runtime package is installed to the server where the COBOL application will be built.

[3] Installing NetCOBOL Runtime Environment

NetCOBOL Runtime package is installed on the development execution environment server and the first Slave Server. If the development execution environment server is selected for building the COBOL application then it is not necessary to install the NetCOBOL Runtime package on the development execution environment server.

[4] Configuring NetCOBOL Runtime Environment

MapReduce application is started by executing remote commands (default is SSH) from a Master Server onto a Slave Server. In the MapReduce job execution user's login shell, set the necessary information (such as environment variables and permissions) for execution.

[5] Correction of Hadoop execution shell

Hadoop execution shell is corrected according to the version of Hadoop which is used. Please refer to "Executing a Hadoop job" for details.

[6] Cloning of Slave Server

Using the cloning feature of Interstage Big Data Parallel Processing Server Slave Server is added.

If cloning feature is not used, install NetCOBOL Runtime environment's on each Slave Server and configure NetCOBOL Runtime environment.

[7] Development of MapReduce Application

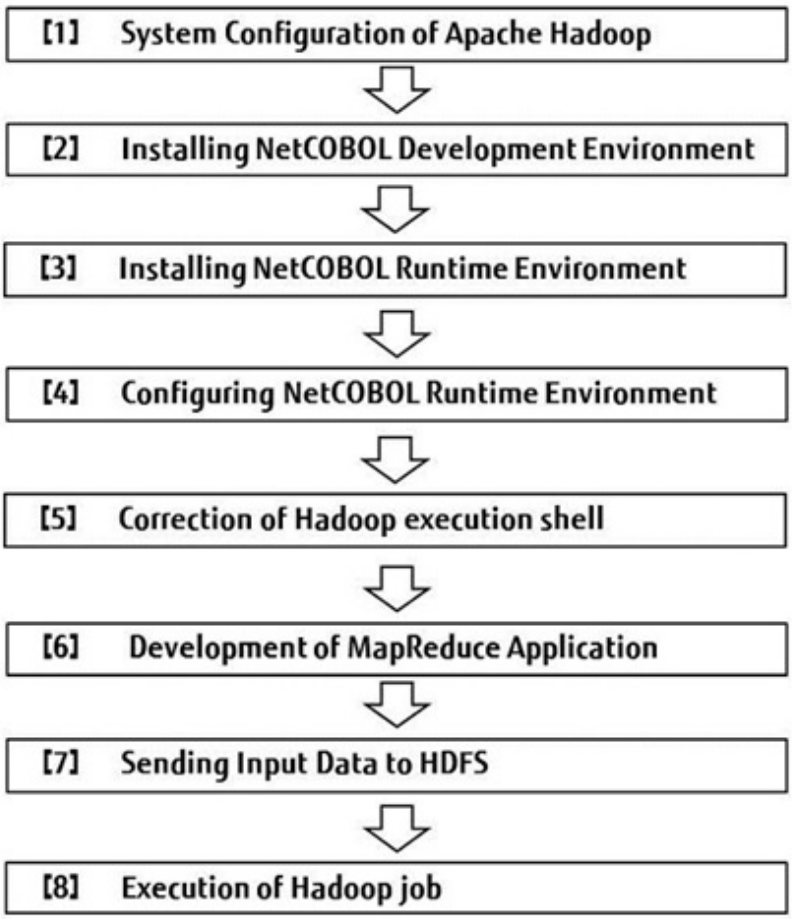
MapReduce applications are developed using the NetCOBOL development environment. For details refer to "Development of MapReduce Application".

[8] Execution of Hadoop Job

MapReduce applications are executed by using Hadoop execution shell. For Details refer to "Executing a Hadoop job".

2.4 Processing procedure for Apache Hadoop

Steps for using the Hadoop integration function are shown below.



[1] System Configuration of Apache Hadoop

Apache Hadoop is installed. Proceed to step [2] after completing the acquisition of Master Server, Slave Server and HDFS.

[2] Installing NetCOBOL Development Environment

NetCOBOL's development and Runtime package is installed to the server where the COBOL application will be built.

[3] Installing NetCOBOL Runtime Environment

The NetCOBOL Runtime package is installed on the Master Server and Slave Server. If the Master Server is selected for building the COBOL application then it is not necessary to install the NetCOBOL Runtime package on the Master Server.

[4] Configuring NetCOBOL Runtime Environment

The MapReduce application is started by executing remote commands (default is SSH) from a Master Server onto a Slave Server. In the MapReduce job execution user's login shell, set the necessary information (such as environment variables and permissions) for execution.

[5] Correction of Hadoop execution shell

Hadoop execution shell is corrected according to the version of Hadoop which is used. Please refer to "Executing a Hadoop job" for details.

[6] Development of MapReduce Application

MapReduce applications are developed using the NetCOBOL development environment. For details refer to "Development of MapReduce Application".

[7] Sending Input Data to HDFS

By using the Apache Hadoop command, input data is sent to HDFS.

[8] Execution of Hadoop Job

MapReduce applications are executed by using the Hadoop execution shell. For more information, refer to "Executing a Hadoop Job".

2.5 Development of MapReduce Application

The MapReduce application should be an executable application. For this reason, multi-threaded programs cannot be executed. While developing a MapReduce application there is no need to specify any special link or compiler option.

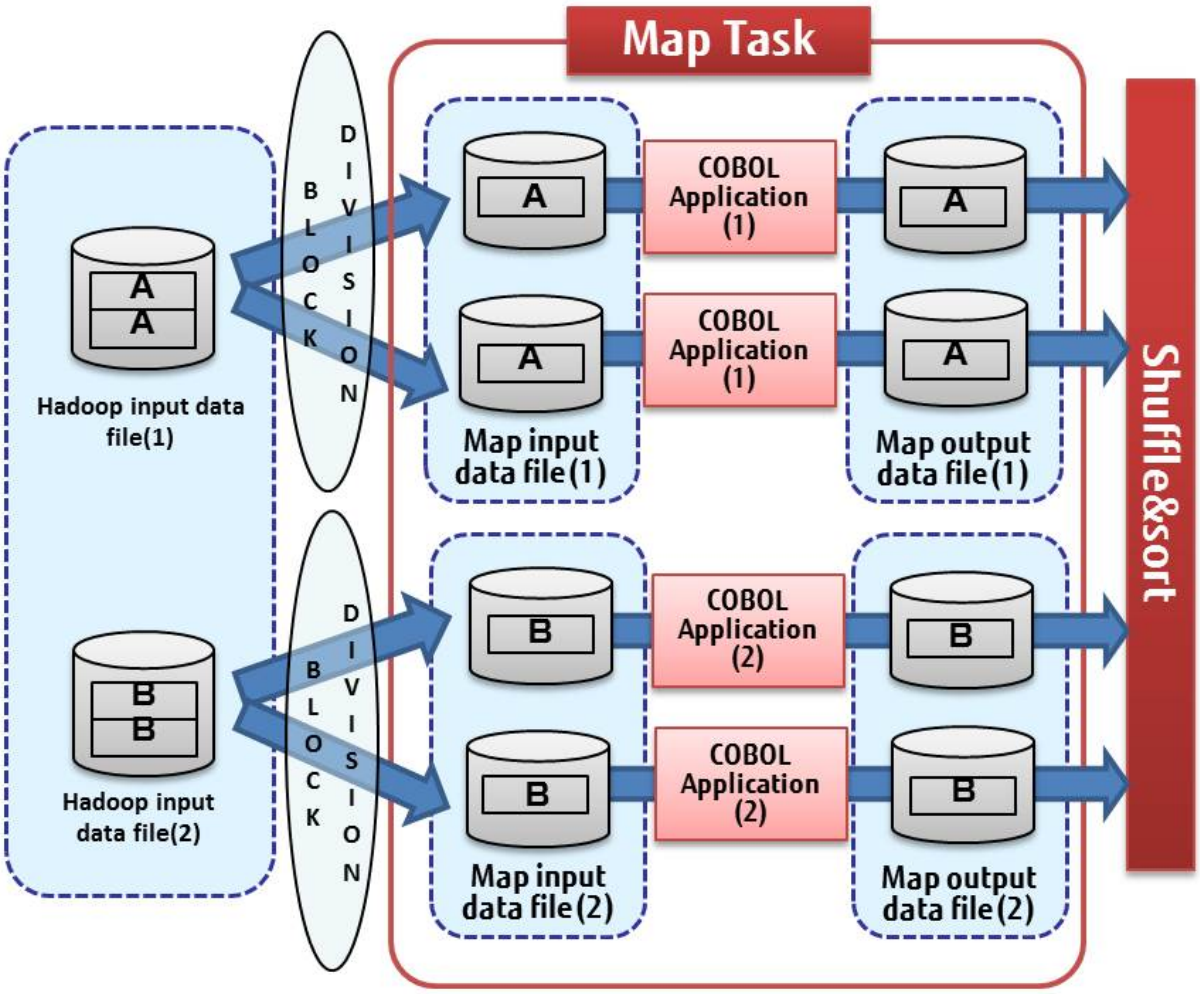
2.5.1 File used with Map Task

The COBOL Runtime system automatically performs the allocation of the Map input and output data file used by the Map Task for COBOL applications. Therefore, it is necessary to specify the unique name of the "MapReduce Configuration File" in advance.

The COBOL application of Map Task can open one file each of Map input data file and Map output data file.

In the following example, Map Task is executed by assigning 2 different Hadoop input date files to 2 COBOL applications.

```
Hadoop Input Data file(1) is assigned to COBOL application(1)
Hadoop Input Data file(2) is assigned to COBOL application(2)
```



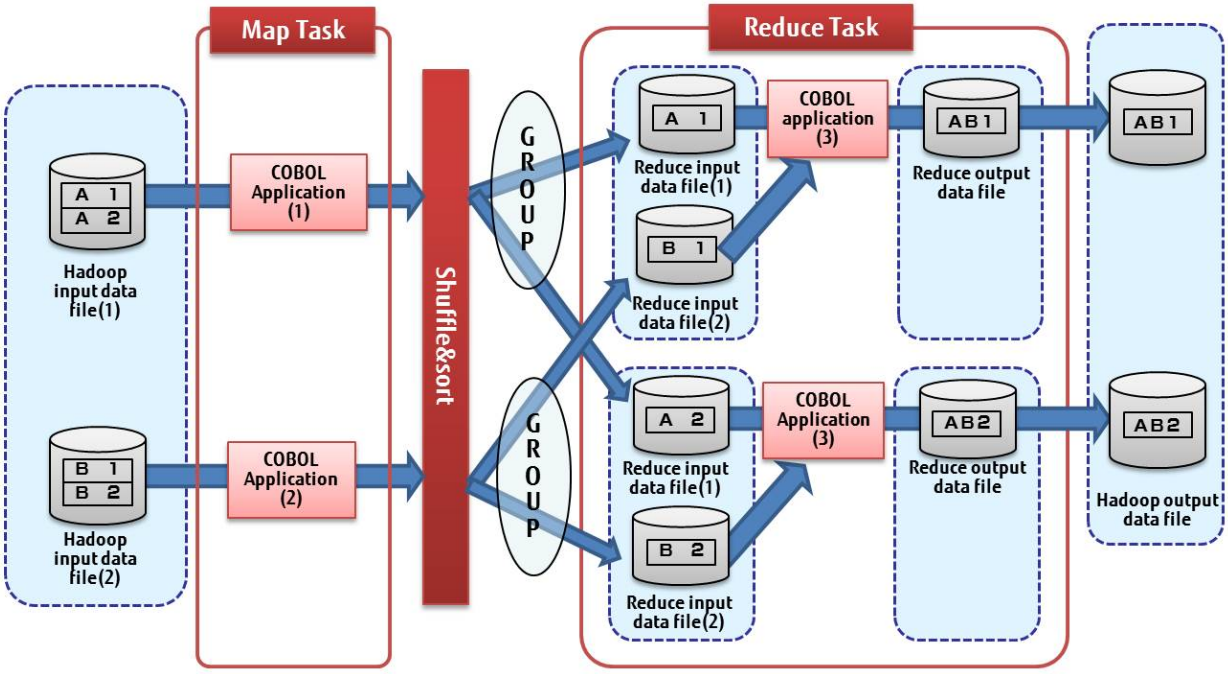
2.5.2 File used with Reduce Task

The COBOL Runtime system automatically performs the allocation for COBOL applications of Reduce input and output data files used by the Reduce Task. Therefore, it is necessary to specify the unique name of the "MapReduce Configuration File" in advance.

The COBOL application of Reduce Task can open all the files specified in the Hadoop Input data file as Reduce Input data files. Moreover, it can open multiple Reduce output data files.

In the following example, Reduce Task is executed by assigning 2 different Hadoop Input data files.

Numeric item is specified as the key to Shuffle&sort and grouped. With the Reduce Task COBOL application, grouped records are read, and records of the same key are joined.



2.5.3 Using a variable-length record sequential file

If "Variable-length record sequential file" is specified in the Hadoop Input data file, it is necessary to create the record length information file in advance and specify the path on DFS in the "MapReduce Configuration File".

The record length information file is used in order to access the variable length records file divided by the MapReduce framework.

The record length information file is created by running the following shell.

Command Format

```
cobgenrecinf.sh [-v|-p] -i record sequential file name -o output directory name of record length file
-b blocksize [-t maptasks] [-n]
```

[-v|-p]

File format of a Hadoop input data file is specified.

- When a Hadoop input data file is a variable length record sequential file, specify "-v."
- When a Hadoop input data file, is a physical sequential order file, specify "-p."

The default value, when file format specification omitted is "-v."

-i record sequential file name

The variable length record sequential file or Physical file name is specified. It is also possible to specify the directory where multiple variable length record sequential files or Physical sequence file are stored.

-o output directory name of record length file

Outputted to record length information file. This directory name is described in the specification of "MapReduce Configuration File" 's record length information file.

-b blocksize

The data size (block size) to be divided in Map Task is specified in bytes.

- When Linking with Interstage Big Data Parallel Processing Server

  Specify the same value as the below given property.

pdfs.fs.local.block.size

- When Linking with Apache Hadoop:

  Specify the same value as the below given property.

  dfs.block.size

[-t maptasks]

At the time of Hadoop execution, if multiple Map Tasks are specified then also specify the count of Map Tasks to this parameter. The count of Map Tasks is specified to the following property of Interstage Big Data Parallel Processing Server or Apache Hadoop:

mapred.map.tasks

[-n]

After the record length information file is created, the number of records of each input file is displayed.

Note

Because the processing of the record length information file is simplified when the size of the input file is smaller than the blocksize, the number of records is not displayed. Include the "-C" option when you want to display the number of records without simplified processing.

Information

MapReduce framework divides the Hadoop Input data file into block size and assigns it to Map Task.

If the count of Map Tasks is specified, the Hadoop Input data file is divided by the number of Map Tasks and assigned.

However, even if you specify the number of Map Tasks, if the size divided by the number of Map Tasks exceeds the block size, then the Map Task is assigned for each block size and the Map Task count specification is ignored.

2.5.4 Using a user definition counter

User definition counter of Hadoop can be used from MapReduce application. When using the user definition counter, any output value produced from each MapReduce application can be totaled. When there is a success in a job, then the totaled value is output in the console of the Hadoop execution shell (an aggregate is not output, when a job fails) .User definition counter is created by the COB_HADOOP_COUNTER subroutine.

Call format

```
CALL "COB_HADOOP_COUNTER"  USING  DataName-1  DataName-2  DataName-3.
```

Explanation of parameter

DataName-1

```
01 DataName-1    PIC X(1024).
```

Counter group name, is specified.

DataName-2

```
01 DataName-2    PIC X(1024).
```

Counter name, is specified.

DataName-3

```
01 DataName-3    PIC S9(18) COMP-5.
```

Counter value, is specified.

Interface

In the group name and counter name, a comma (,) cannot be used. Moreover, leading and trailing blanks are deleted. The record length of Group name and counter name is to be set to 1024 bytes (when data exceeding 1024 characters is passed, then it is rounded to 1024 characters). Moreover, delete leading and trailing blanks cannot be specified when a data name has a character string length of 0.

Return value

Return value from a subroutine is received using special register PROGRAM-STATUS.

0 is returned when it succeeds. Return value at the time of failure and the cause of an error are given below.

Value	Meaning
1	Since it was not the application started by Hadoop task, so the counter information writing was not performed.
2	Improper character string is specified as Group name.
3	The null character was specified as the character string of Group name.
4	The inaccurate character string is specified as counter name.
5	The null character was specified as the character string of counter name.
6	There was a failure in the opening of a counter information file.
7	There was a failure in the writing of the counter information.

Note

When you create the program which uses subroutine COB_HADOOP_COUNTER, then please link librcobee.so, when linking with the shared library.

librcobee.so is a shared library stored in the installation directory of a NetCOBOL Runtime system.

Calling from applications other than COBOL

When setting up counter information from languages other than COBOL, please perform the additional writing of counter information to a counter information configuration file (environment variable MAPRED_COUNTER_FILE) with the following format.

Group name ,counter name, counter value

Example) When counter is used from shell script

echo "MYGROUPE, MYCOUNTER,100" >> $MAPRED_COUNTER_FILE

(Group name =MYGROUPE counter name=MYCOUNTER counter value=100 is set)

Warning when there is a format error

Even when writing counter information in a counter information configuration file directly even when the write format is not protected, the job ends normally. In this case, the message which notifies about the total failure of counter information is displayed, and number of each error is output as the following counter information. Refer to the task log for the details of an error.

counter Group name	counter name	Meaning
Extjoiner.userCounter.FormatError	Map.CommaNumError	There is an error in the counter value format of Map Task.
	Map.NumberFormatException	The counter value of Map Task could not be converted to a numerical value.
	Reduce.CommaNumError	There is an error in the counter value format of Reduce Task.

	Reduce.NumberFormatException	The counter value of Reduce Task could not be converted to a numerical value.

Note

When Adding (appending) counter information to a counter information configuration file, please ensure that it is in the Add write mode.

When a counter information configuration file is edited without being in Add write mode, the existing counter information is overwritten.

Incorrect Example)

echo "MYGROUPE, MYCOUNTER,100" > $ MAPRED_COUNTER_FILE

In this case, since it is not additional writing, all previous task settings for the counter information are overwritten.

2.6 Operation of MapReduce Application

The operation of the MapReduce application is explained below.

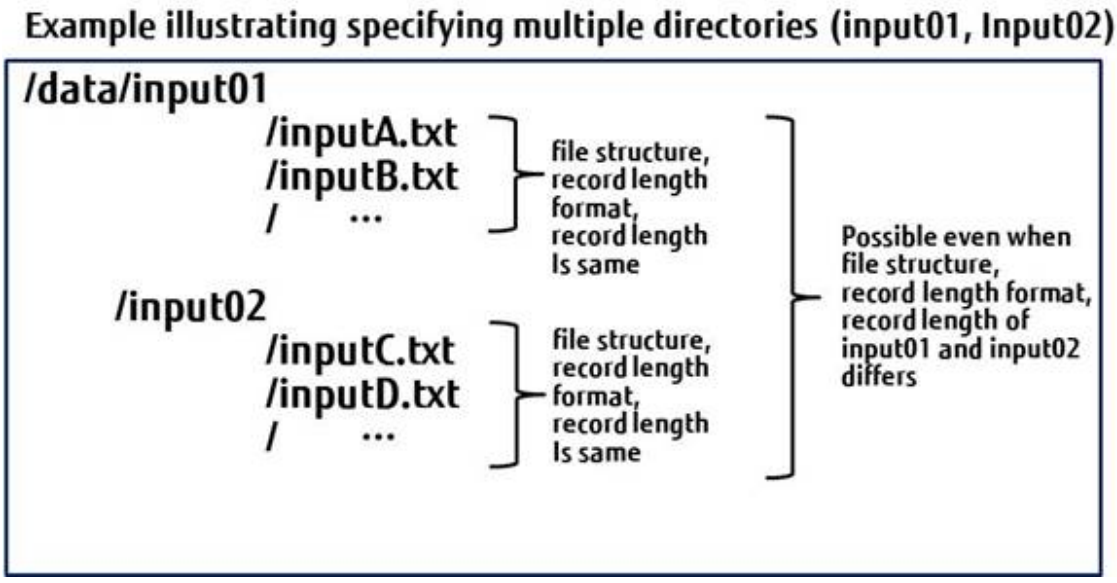
2.6.1 Preparing Hadoop Input Data File

Before executing the Hadoop job it is necessary to save the Hadoop input data file to the DFS.

The path of the Hadoop Input data file on the DFS is specified in the "MapReduce Configuration File". If the directory is specified as the Hadoop Input Data file path, then all files in that directory are read.

It is possible to input multiple files with different structures or with different record layouts as a Hadoop Input file.

Figure 2.1 Example of specifying multiple (input01, input02) directories



Note

If a directory is specified to the Hadoop input data file, then the file structure of files in a directory, record length format (fixed-length or variable-length format) and the record length must be the same.

2.6.2 Executing MapReduce Application

Task Tracker executes the MapReduce application. TaskTracker executes the MapReduce application by the following procedures:

- Local working directory for the task is created
- Local working directory is set as current directory and MapReduce application is started
- After completion of MapReduce application, local working directory for task is deleted

2.6.3 Return value and error of MapReduce Applications

In this function, the maximum return code value is returned from the MapReduce application to the Hadoop execution shell.

In the default setting, a job failure message is produced when the return value from MapReduce exceeds the default setting of "128". Otherwise, the status of Job Tracker is successful (SUCCEEDED) and a Hadoop output data file is saved.

The threshold value for a return value, which handles a job as an error, can be changed by a "MapReduce Configuration File".

Further, MapReduce application can be rerun for every task by setting up the threshold value for the return value which reruns application. (When not set, then as the default value is 255, so it cannot be re-executed)

When the return value exceeds the threshold value and is returned from MapReduce application, then this function notifies an abnormal end to TaskTracker.

Although, though the TaskTracker repeats the re-execution of maximum test count applications which are set in advance, but when the error of the maximum test count is repeated then it ends the Hadoop job with an error.

Here, the task which is under execution, is forced to terminate and a Hadoop output data file is totally deleted including what was outputted from the completed task.

It is possible to change the threshold return value handled by the application performing the re-run, from the MapReduce Configuration File.

2.7 Important Points of MapReduce Application

With the Hadoop integration function, the same program runs parallel on multiple machines. This should be kept in mind while developing a program. Be careful while migrating existing applications. Hereafter points to consider are mentioned for each function.

2.7.1 Database Access Function

As the application is executed in parallel, concurrent access to the database server occurs. Do performance estimation according to the multiplicity of MapReduce applications. In addition, due to speculative execution in Hadoop, there is a possibility that the same task as the task under execution is run. For this reason, please do not update the table.

2.7.2 Print Function

As the application is executed in parallel concurrent access to the print or reporting server occurs. Do performance estimation according to the multiplicity of MapReduce applications. Based on the requirements, please consider increasing the print or reporting servers.

2.7.3 Small Input Output (I/O) Function

Following are small I/O functions for use.

Function	I/O Destination	Availability	Remarks and Precautions
ACCEPT	Standard Input	No	Data from the standard input cannot be read because the console does not exist. Replace it with file input function.
	File Input	Yes	For location of the Input file refer to "Allocating resources required for execution"
	Systemwalker's Admin console	No	Systemwalker Centric Manager admin console gets input requests from applications started from multiple machines. In this case, it cannot be used as requesting application cannot be identified.
DISPLAY	Standard Output	Yes	Saved as a file in directory created for each task.

Function	I/O Destination	Availability	Remarks and Precautions
	Standard Error Output	Yes	
	Syslog	Yes	Output to the OS syslog.
	File Output	Yes	The output file should output to the current directory to avoid conflicts. A file outputted to the current directory is saved to the directory created for each task.
	Systemwalker's Admin console	No	Systemwalker Centric Manager admin console gets output from applications started from multiple machines (simultaneously). In this case, it cannot be used as application generating output cannot be identified.
	General Log	No	As Hadoop integration function cannot be used under the Interstage Business Application Server, so cannot output to general log.
Command line arguments	ARGC,ARGV	Yes	Can be retrieved by using an ACCEPT statement. The arguments are specified in the MapReduce Configuration File.
Environment Variable Operation	Environment Variable	Yes	Specified in COBOL.CBR or "MapReduce Configuration File"

2.7.4 How to debug COBOL application using the debugger

The MapReduce application is launched from MapReduce framework. For this reason, debugging of the application is performed by using the remote debugging (attach function) of NetCOBOL Studio.

When multiple MapReduce applications are started, the recently started Map application will be attached to debugger. Other applications will work normally without being connected to the debugger.

Note

A program connected to the debugger is forcibly terminated as soon as the debugger exits. For this reason, in order to continue with a program connected to the debugger, please do not exit the debugger until the break points are released and the program completes.

2.7.5 Caution related to the timeout of the task

MapReduce application is started from MapReduce framework. When there is no response for a pre-defined period of time from MapReduce application (data is not output) processing time ends as a job failure. To extend processing time for records that take longer to process, please increase the timeout setting (specified in the mapred.task.timeout of Apache Hadoop property) in advance.

2.7.6 Caution related to input output file attribute

The data transfer between the MapReduce application and the MapReduce framework uses named pipes. Therefore, when the MapReduce application refers to the file attribute of the input output file, it is recognized that they are not usual files but instead are named pipes.

2.8 Allocation Process of Shuffle&sort

In Shuffle&sort, dividing conditions are applied to the main key specified in the "MapReduce Configuration File" and a record is distributed to the Reduce Task number, which can be executed. The method of distributing which can be used by a Hadoop integration function is explained below.

2.8.1 Distribution with the hash value

This is the default distribution methodology. The hash value is calculated from a primary key and a record is distributed to the number of Reduce Tasks (mapred.reduce.tasks) which can be executed.

Note

By duplication of a hash value, two or more primary keys can be distributed to the same Reduce Task.

Please create Reduce application, in the case where the record in which primary keys differ is passed to the same Reduce Task.

2.8.2 Automatic optimum partitioning based on key allocation

It is a function which shortens the Hadoop job's execution duration, by not allocating key with numerous counts to the same task, by scheduling the execution of appropriate task, in accordance to the examination of the distribution of primary key of the Hadoop input data file and its distribution,

The count of Reduce Tasks possible to execute (mapred.reduce.tasks) can also be automatically specified to an appropriate value for the execution environment.

Hence, when a change due to failure or addition of a Slave Server happens, then the time and effort for tuning the Reduce Task count can be saved.

Hence for using this method of distribution, it is required to create "Primary Key List file" in advance.

Hadoop execution shell is executed after this primary key list file is specified through "Specifying Primary key list file" of "MapReduce Configuration File".

Information

In the distribution which considers the key distribution, if there is a deviation the primary key, then it is possible to reduce the execution time of the Hadoop job. On executing the "Executing the Primary Key List file creation shell", the deviation of distribution of a primary key can be checked.

Note

Since priority is given to count of Reduce Tasks by optimization, so in case when Reduce Tasks (mapred.reduce.tasks) is specified in the MapReduce Configuration File, then that value is ignored.

During the execution of Hadoop execution shell, "_mainkeyList" directory is generated in the temporary directory (Directory of system property java.io.tmpdir in Java, by default it is /tmp directory).

The file size used by the "_mainkeyList" directory is same as the primary key list file, which is used.

The work file is deleted simultaneously with the end of a job.

2.8.3 Distributing different primary keys to different tasks

This is a method for distributing the different primary keys of records that have more than one primary key to different Reduce tasks.

Because each Reduce task only handles one type of primary key, this method is useful for applications that cannot handle primary keys in groups.

Following three procedures are necessary, for using this distribution method.

- The primary keys which are included in the Hadoop input data file are already defined in the "Primary Key List file".
- The primary key list file is specified in the "Specifying Primary Key List file" of MapReduce Configuration File.
- Perform "Specifying unique partition" in the MapReduce Configuration File.

Note

The records assigned to a spare Reduce Task does not become a unique key, as all records with the primary key which is not contained in a primary key list file, are assigned to the "Spare Reduce Task".

The value where 1 is added as a spare Reduce Task to the primary key which is included in the primary key list file, becomes an executable Reduce Task count (mapred.reduce.tasks).Hence that value is ignored, when the number of Reduce Tasks (mapred.reduce.tasks) is specified in the MapReduce Configuration File.

To prevent the unexpected number of tasks from being executed, the maximum count of the distributable key is being specified.

Perform "Specifying maximum number of key for unique partition" of "MapReduce Configuration File" for improving the maximum count.

2.8.4 Primary Key List file

Primary key list file is a file that contains the definition of the primary key in Hadoop input file.

Creating this file and considering the key distribution, it is possible to automatically distribute with the appropriate condition, and distribute to the different tasks to each key.

The following are two methods for the creation of a primary key list file.

- Executing the Primary key list file creation shell
- Creating the Primary Key List file using Text Editor

Information

A primary key list file is automatically distributed to each Slave Server as side data during the job instruction execution.

For this reason, the user does not need to specify it as side data or copy to each slave server.

Primary key which is not defined in the Primary key list file, will be automatically distributed to a spare Reduce Task.

2.8.4.1 Correcting the Primary Key List file creation shell

Primary key list file creation shell is the shell which executes the Hadoop job which creates a primary key list file.

The primary key list file creation shell is stored at the following path.

```
/opt/FJSVcbl64/bin/cobmkmlist.sh
```

It is necessary to define the commons-logging jar file and the hadoop-core jar file which are provided by Hadoop to be used as primary key list file creation shell.

On initial execution and when updating the Hadoop version, open the Hadoop execution shell with a text editor, and correct the path of the commons-logging file and the Hadoop-core file as in the following lines.

```
class_path_log="/usr/share/hadoop-1.2.1/lib/commons-logging-1.1.1.jar"
```

```
class_path_core="/usr/share/hadoop-1.2.1/hadoop-core-1.2.1.jar"
```

Note

Root authority is required for updating the Primary Key List File creation shell.

By default the commons-logging and the hadoop-core are defined as jar files "commons-logging-1.1.1.jar" and "hadoop-core-1.2.1.jar".

2.8.4.2 Executing the Primary Key List file creation shell

In Primary Key List file creation shell, a MapReduce Configuration File is specified as a parameter and executed.

```
$ cobmkmlist.sh -conf  MapReduce Configuration File
```

Primary Key List file creation shell returns"0" if the job is a success and "except zero" if it fails.

When the Primary key list file creation shell job is successful, then it will be created in the directory specified in the "Specifying Primary Key List file".

Possibility of improvement that can be done for performance with the distribution function are displayed, while executing the Primary key list file creation shell.

Example when the distribution function is valid

```
Result of analysis of the key distribution of the input file, possible to improve performance by "80%"
based on automatically distributing on the optimum conditions.
```

Example when the distribution function is invalid

```
As a result of analysis of the key distribution of the input file, the effect automatically
distributing on the optimum conditions cannot be anticipated.
```

In addition, it is a logical value that takes into account only the distribution of key and even when the possibilities of improvements are displayed as "100%", depending on the overhead of Hadoop job etc., there are cases when it is not possible to shorten the job time.

The information defined by the MapReduce Configuration File is given below. Refer to "MapReduce Configuration File" for the details of each setup.

	Setting name	Meaning	Specification
Specifying input output file	extjoiner.input.*nn*.filename	Hadoop input data file name (*1) (*2)	Required
	mapred.output.dir	Hadoop output data file storage standard directory name (*1) (*3)	Required
	extjoiner.map.input.*nn*.filename	Map input data file Distinguished Name (*4)	Optional
	extjoiner.map.output.*nn*.filename	Map output data file Distinguished Name (*4)	Optional
Specifying Key information	com.fujitsu.netcobol.hadoop.sortkey.*nn*.main	Used in the Shuffle&sort Primary Key information	Required
Specifying Primary Key List file	extjoiner.mainkeylist	File name of Primary Key List file (*1)	Required
Specifying MapReduce Application	extjoiner.map.streamprocessor.*nn*	Map application name (*5)	Optional
Threshold return value for re- execution of Application	extjoiner.command.retryexitstatus	From the Return value which are returned from the Map application, specify the threshold value whether the re-execution is done or not for the Map application.	Optional
Specifying environment variables	extjoiner.map.environment	Map application environment variables are specified.	Optional
Specifying file format	extjoiner.input.*nn*.format	File sequence of Hadoop input data file	Optional
	extjoiner.map.output.*nn*.format	File sequence of Map output data file	Optional
Specifying record length (Only fixed-length record sequential file)	extjoiner.input.*nn*.recordlength	Record length of Hadoop input data file	Optional
	extjoiner.map.output.*nn*.recordlength	Record length of Map output data file	Optional
Specifying record length information file(For variable-length record sequential file or physical sequence file only)	extjoiner.input.*nn*.recinfdir	Storage directory name of the record length information file for the Hadoop input data file (*1)	Optional

	Setting name	Meaning	Specification
Specifying Character Encoding (Only line sequential file)	extjoiner.input.*nn*.codeset	Hadoop input data file's character code	Optional
	extjoiner.map.output.*nn*.codeset	Map output data file's character code	Optional
CSV FORMAT data usage specifications	extjoiner.csv.separator	CSV data separator is specified	Optional
	extjoiner.partitioner.csv.padding	CSV data distribution processing related, primary Key's blank handling	Optional
	extjoiner.csv.floatfield	Float field specification	Optional

(*1) Path name on the DFS is specified.

(*2) It is possible to specify the directory name. When a directory is specified, all the files in the directory are treated as an input file.

(*3) The Hadoop job performed from primary key list file creation shell uses it as a temporary work directory. It is automatically deleted after the end of primary key list file creation shell.

(*4) File Distinguished Name, is specified with the Map application.

(*5) Application used in the Map application, is specified.

Information

When performing the primary key list file creation shell, a task can be interrupted by using "Ctrl+C" key.

General-purpose Hadoop command line option, can be specified as the argument of primary key list file creation shell.

Please refer to "Executing a Hadoop job" for details.

2.8.4.3 Creating the Primary Key List file using Text Editor

Primary key list file is a CSV text file. By using a text editor etc., a file can be edited or a new file can be created.

Specify the ratio/percentage of the primary key and the value of the primary key in the primary key list file.

Example of Primary key list file, with 50 specified as a rate of primary key, and primary key as ABCDE:

```
"ABCDE",50
```

Primary Key's value

Specify the Primary Key value. When multiple primary keys are specified, they are separated by a comma.

The value of primary key specified as a primary key list file, must match with the key value including the digit number.

There are two methods of specifying the value of a primary key.

1. Enclosed in double quotations

   When the attribute of a primary key is the following, it should be enclosed in double quotation marks.

   - When the Map output data file is a sequential file

     Alphanumeric items, Zoned decimal item, Packed decimal item, CSV,CSVN

   - When the Map output data file is other than sequential, or when a Map output data file is in multiple, then in cases when there are file sequence included other than even one sequential order file.

     Zoned decimal item, Packed decimal item.

2. Specify with a binary value

   When it cannot be enclosed with double quotation marks, binary value can be specified as a value of the primary key.

   "0x" is added to a prefix and it is specified as a binary value (0~F).

Ratio of the Primary Key

Ratio or number of a primary key or item count, is specified numerically.

The ratio is used for scheduling the execution of optimal task.

Example

A primary key list file, when the Map output data file is a sequential file and there is an alphanumeric item (character) and signed zoned decimal item in the Primary key.

```
; This line is a comment
"CCCCC","+1234",30000
"AAAAA","+0000",7000
"BBBBB","-1234",50000
```

A primary key list file, when the Map output data file is a sequential file and there is an alphanumeric item (binary value) and unsigned packed decimal item in the Primary key.

```
; This line is a comment
0xFFFDF3,"01234",30000
0xFEFDF1,"00000",7000
0xFFFDF2,"01234",50000
```

Note

- The file must use the Unicode (UTF-8) character code.
- A 1 byte new line character must be created (0x0A).
- Either Unicode (UTF-8) with BOM (Byte Order Mark) or BOM-less Unicode (UTF-8) can be used.
- When a line begins with a semicolon (;) in the first byte, the entire line until the new-line character is regarded as a comment.
- In the packed decimal item, two digits are stored per byte. When a packed decimal item is enclosed by double quotation marks and specified as Primary key list file, please compensate a high-order digit with 0 and specify an integer digit number (except for a sign) by odd number of digits.
- In the primary key list file, only the value of a primary key and the ratio of a primary key can be specified. Except for the comments, the blank character etc. cannot be included.
- When in the primary key list file, the value of the primary key is specified as enclosed in double quotation marks, then when double quotation marks are included in the key values, there is a need to further enclose the key value in double quotation marks.

2.8.5 Spare Reduce Task

The Spare Reduce Task is a task which is distributed by keys not included in the Primary key list file.

The spare Reduce Task is started automatically in case of "Automatic optimum partitioning based on key allocation ", or "varying tasks are distributed to each key".

When all primary keys are defined in the Primary key list file, a key cannot be distributed to a spare Reduce Task.

Note

Entire keys that are not contained in a primary key list file, can be distributed to a spare Reduce Task.

Hence, when there are many such keys then a spare Reduce Task may become a bottleneck.

In this case, it is recommended improving the primary key list file.

2.9 Executing a Hadoop Job

With the Hadoop integration function, a Hadoop job is executed by using the Hadoop execution shell provided by the following NetCOBOL servers:

- When Integrating with Interstage Big Data Parallel Processing Server:

  Development Execution Environment Server

- When Integrating with Apache Hadoop:

  Master Server

Correction of Hadoop execution shell

It is necessary to define the jar file of Hadoop Streaming offered by Hadoop to be used as Hadoop execution shell.

Hadoop execution shell is stored at the following path.

```
STREAMING_JAR=/usr/share/hadoop/contrib/streaming/hadoop-streaming-1.2.1.jar
```

Note

Correction of Hadoop execution shell requires root authority.

"hadoop-streaming-1.2.1.jar" is defined as a jar file of Hadoop Streaming by the default.

How to execute Hadoop execution shell

A Hadoop execution shell is run by specifying the MapReduce Configuration File as a parameter.

```
$ cobhadoop.sh -conf  MapReduce Configuration File
```

It is necessary to specify the MapReduce Configuration File.

A Hadoop execution shell returns "0" for successful execution and "non 0" for failed execution.

However, if the MapReduce application returns the return value then in that case the maximum value is returned.

The task is interrupted if "Ctrl+C" is entered during the execution of a Hadoop execution shell.

Information

A general-purpose Hadoop command line option can be specified as the argument of Hadoop execution shell.

For example, the value of the Hadoop property can be specified by using "-D" option.

This can be used to specify the timeout setting, number of Reduce Tasks, etc. for every job.

Example: When "300 seconds" is specified as task timeout time and "0" is set to the number of Reduce Tasks.

```
$ cobhadoop.sh -conf  MapReduce Configuration File -D mapred.task.timeout=300000 -D
mapred.reduce.tasks=0
```

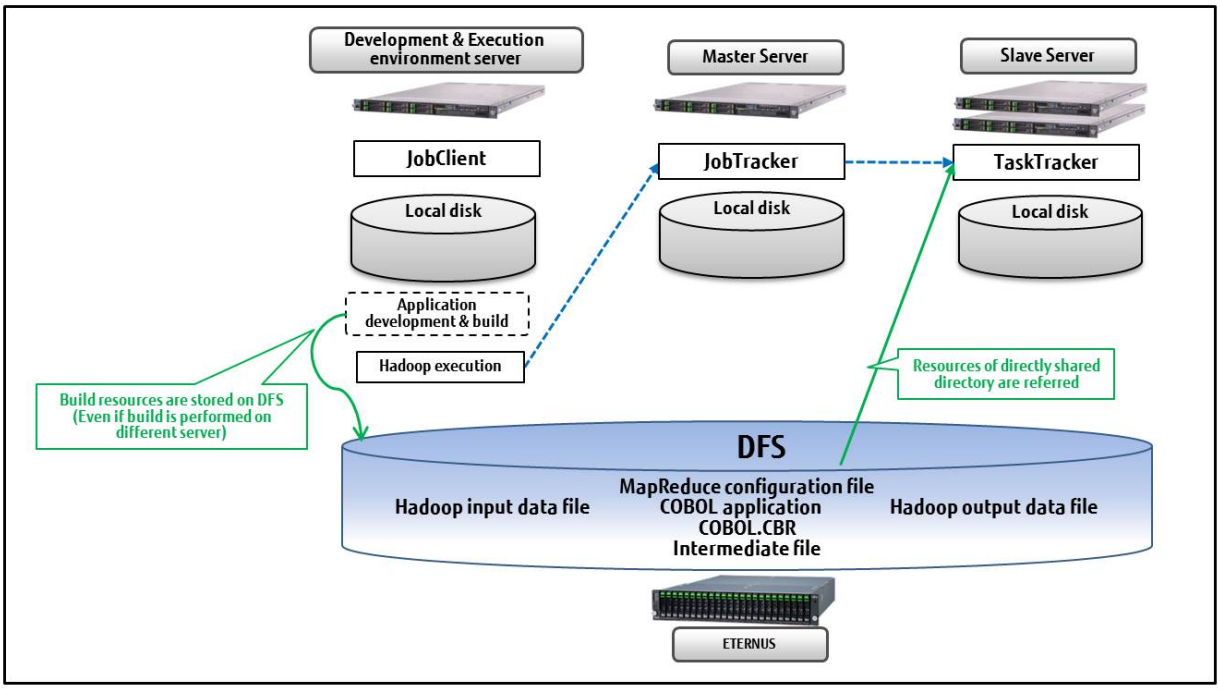
2.9.1 Allocating resources required for execution

When executing a Hadoop job the following resources are required:

- MapReduce Configuration File
- Map application and Reduce application
- File used by COBOL Application
- Libraries, etc. required for COBOL.CBR or calling Subroutine

2.9.1.1 Integration with Interstage Big Data Parallel Processing Server

Resources other than the MapReduce Configuration File are required by each Slave Server, but users need not copy the resources to the local disk of each Slave Server. These resources are saved in the DFS directory share mounted between the Development execution environment and each Slave Server.



2.9.1.2 Integration with the Apache Hadoop

Resources other than a MapReduce Configuration File are resources required of each Slave Server.

While it is not necessary for a user to copy to the local disk of each Slave Server, it is necessary to make access available from each Slave Server using one of the following methods.

- Use a shared directory

  The resources required for execution can be referred to from each Slave Server, by storing resources in the directory which carried out share mount (NFS etc.) between a Master Server and each Slave Server,

  When the resources are saved on the directory currently shared between each server, then a shared directory is used

- Use side data

  The side data, is a function of Apache Hadoop which distributes read-only data, in each Slave Server using the mechanism of a distributed file. During the execution of MapReduce application reference can be made from the current directory by specifying side data as a real line command.

  Side data is used when resources are saved on the local disk of the Master Server.

  Example:When Map application "map1.exe" and Reduce application "reduce1.exe" are used as side data.

```
$ cobhadoop.sh -conf MapReduce Configuration File -files ./map1.exe,./reduce1.exe
```

2.10 Output file of MapReduce Application

The file that the MapReduce application output to the current directory can be saved. If a definition is given to a MapReduce configuration file, it is saved after job execution to the following directories.

```
{Base directory storing Hadoop output data file}/currentdir/part-{m|r}-{nnnnn}/
```

```
  { Base directory storing Hadoop output data file}:Path specified in the MapReduce Configuration
Information file
  {m|r}:When outputting Map application"m", When outputting Reduce application"r"
  {nnnnn}:Task trial run ID generated by JobTracker.
```

Also, a message is outputted to the following files in the case of standard output or standard error output.

Regardless of the settings for saving a file, these two files are always saved in the above directory.

stdout	Contents outputted to standard output are shown
stderr	Contents outputted to standard error are shown

Information

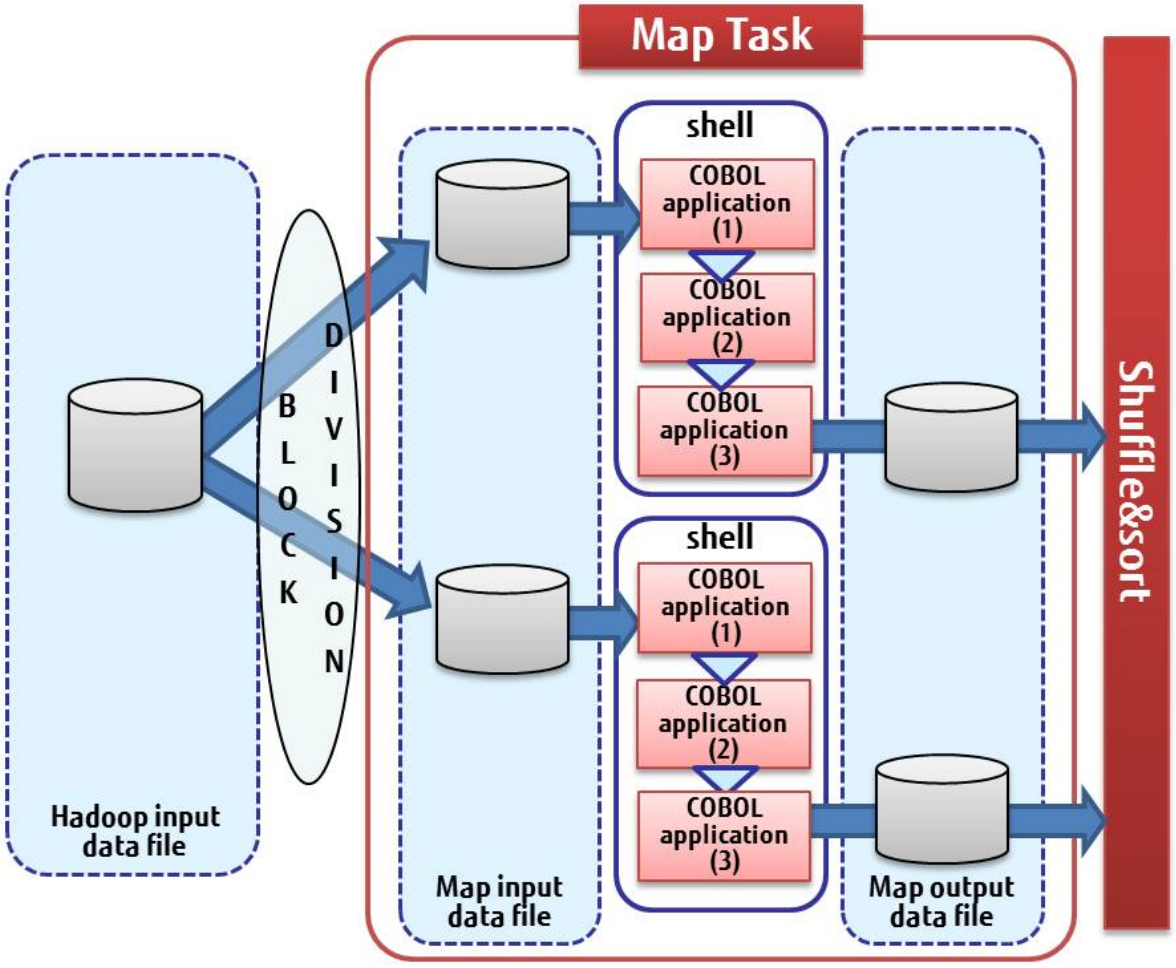
In addition to the files output from the MapReduce application, the files required to run the Map Tasks and Reduce Tasks are stored in the current directory. Therefore, if the save setting is selected, the output file is saved along with these files.

2.11 Executing shell through MapReduce

MapReduce applications can be executed by specifying the COBOL application; it is also possible to specify a shell. By using the shell, execution of multiple COBOL applications, and executing COBOL applications in conjunction with the command, can be done.

In Hadoop, each time the job is run the overhead of distributed processing occurs. By using the shell and by executing multiple COBOL applications or multiple commands in one job, it is possible to reduce the impact of the overhead.

However, the data input and output files can be read or written only once in any of the applications those are executed in the shell.



Map application executed as shell, COBOL application (1), (2), (3) is executed inside the shell.

Map Input data file is read by COBOL application (1), Map Output data file is written by COBOL application (3).

Passing data between COBOL applications (1), (2), (3), intermediate file outputted to the current directory is used.

2.12 MapReduce Configuration File

With the Hadoop integration function, the following information necessary for execution is defined in the MapReduce Configuration File.

- MapReduce application name
- The unique name and the file (path) name of file
- Organization and record length, line sequential file's character encoding
- Key Information for use in Shuffle&sort

The MapReduce Configuration File is an XML file (text file). Create it by using a text editor.

Information

Hadoop property values such as task timeout time, number of Reduce Tasks, etc., can also be defined in a MapReduce Configuration File.

2.12.1 Format of MapReduce Configuration File

In the MapReduce Configuration File, create a <property> element for one item to be set. In that <property> element, configure each element by specifying <name> element and <value> element.

The format of the MapReduce Configuration File is shown below:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<configuration> <!---root element. Necessary to specify. -->
        <property> <!--- property element is created and name, value is specified inside it.  -->
                <name>set name 1</name>
                <value>set value 1</value>
        </property>
        <property>
                <name>set name 2</name>
                <value>set value 2</value>
        </property>
        <!---setting is repeated for necessary times only -->
</configuration>
```

2.12.2 List of information specified in the MapReduce Configuration File

List of parameters specified in the MapReduce Configuration File.

Table 2.1 Parameter List

	Parameter Name	Meaning
Hadoop job name specification	extjoiner.jobname	Hadoop job name
Specifying MapReduce Application	extjoiner.map.streamprocessor.*nn*	Map application name
	extjoiner.reduce.streamprocessor	Reduce application name
Threshold return value for re-execution of application	extjoiner.command.retryexitstatus	Based on the return value returned by the MapReduce application, the threshold value for

	Parameter Name	Meaning
		whether or not to re-execute the MapReduce application
Threshold return value for treating job as error	extjoiner.command.jobfailurestatus	Based on the return value returned by the MapReduce application, the threshold value for whether or not to make job as error
Specifying environment variables	extjoiner.map.environment	Environment variables of Map application
	extjoiner.reduce.environment	Environment variables of Reduce application
Specifying Input Output file	extjoiner.input.*nn*.filename	Hadoop Input data file name (*1) (*2)
	mapred.output.dir	Hadoop output data file base directory name (*1)
	extjoiner.output.*nn*.filename	Hadoop output data file directory name
	extjoiner.map.input.*nn*.filename	Map input data file unique name (*3)
	extjoiner.map.output.*nn*.filename	Map output data file unique name (*3)
	extjoiner.reduce.input.*nn*.filename	Reduce input data file unique name (*3)
	extjoiner.reduce.output.*nn*.filename	Reduce output data file unique name (*3)
Specifying the standatrd ditectory to output data file to writing	extjoiner.output.dir.removeifexist	Specify whether to overwrite the Hadoop output data file standard Directory or not.
Specifying file format	extjoiner.input.*nn*.format	Hadoop input data file's file organization
	extjoiner.output.*nn*.format	Hadoop output data file's file organization
	extjoiner.map.output.*nn*.format	Map output data file's file organization
	extjoiner.reduce.input.*nn*.format	Reduce input data file's file organization
Specify the record length (Only fixed-length record sequential file)	extjoiner.input.*nn*.recordlength	Hadoop input data file's record length
	extjoiner.output.*nn*.recordlength	Hadoop output data file's record length
	extjoiner.map.output.*nn*.recordlength	Map output data file's record length
	extjoiner.reduce.input.*nn*.recordlength	Reduce input data file's record length
Specify record length information file (For variable-length record sequential file or physical sequence file only)	extjoiner.input.*nn*.recinfdir	Directory name of the file containing the record length information of the Hadoop input data file (*1)
Specify character encoding (Only line sequential file)	extjoiner.input.*nn*.codeset	Hadoop input data file's character encoding
	extjoiner.output.*nn*.codeset	Hadoop output data file's character encoding
	extjoiner.map.output.*nn*.codeset	Map output data file's character encoding
	extjoiner.reduce.input.*nn*.codeset	Reduce input data file's character encoding
Specifying key information	com.fujitsu.netcobol.hadoop.sortkey.*nn*.main	Primary key information to be used in Shuffle & sort
	com.fujitsu.netcobol.hadoop.sortkey.*nn*.sub	Secondary key information to be used in Shuffle & sort
	com.fujitsu.netcobol.hadoop.sortkey.*nn*.colseq	Order of the ASCII code
	com.fujitsu.netcobol.hadoop.sortkey.*nn*.decimal	Order of the decimal items with no SEPARATE phrase
	com.fujitsu.netcobol.hadoop.sortkey.*nn*.float	Order of the internal floating-point item

	Parameter Name	Meaning
Specifying Primary Key List file	extjoiner.mainkeylist	Primary Key List file name
Specifying unique partition	extjoiner.partitioner.unique	The record where the primary key is different is distributed to a different Reduce Task.
Specifying maximum number of key for unique partition	extjoiner.partitioner.unique.max.keys	maximum number of keys for unique partition
CSV FORMAT data usage specifications	extjoiner.csv.separator	CSV data's separator is specified
	extjoiner.partitioner.csv.padding	Primary Key blank handling, related to the CSV data distribution processing
	extjoiner.comparator.csv.padding	Blank handling of the key related to the CSV data sort processing
	extjoiner.csv.floatfield	Float field specification
Auto sort specification for the Map output data file	extjoiner.input.sort	After skipping the Reduce application specification, Shuffle&sort is operated
Get the log of the processing record count	extjoiner.map.input.getRecordCount	The number of Map input records is displayed.
	extjoiner.map.output.getRecordCount	The number of Map output records is displayed.
	extjoiner.reduce.input.getRecordCount	The number of Reduce input records is displayed.
	extjoiner.reduce.output.getRecordCount	The number of Reduce output records is displayed.
Specifying saving of the current directory	extjoiner.copyworkingdir	MapReduce application's current directory is saved after completion of task
Specifying buffer size	extjoiner.maxbufferrecords	The maximum number of records to be buffered
Map Task multiple file output mode	extjoiner.map.multioutput	Use the multiple file output in the application assigned to the Map Task
Start of Map application when Hadoop input data file is 0 bytes in size	extjoiner.map.alwaysRun	Map application starts even when Hadoop input data file is 0 bytes in size.
Returning the return value of Reduce application when Hadoop input data file is 0 bytes in size	extjoiner.reduce.alwaysReturnExitcode	Reduce application returning return value even when Hadoop input data file is 0 bytes in size.
Creating a Hadoop output data file of 0 bytes	extjoiner.reduce.alwaysOutput	Even when data is not written in Hadoop output data file, a Hadoop output data file of 0 byte size is created.

*1: Specify the path name of DFS

*2: It is possible to specify directory name also. If directory is specified, all files in that directory are treated as input files.

*3: Specify the unique name of file used by MapReduce application

Inside the table "*nn*" represents the count of Hadoop input data file. 01-64 can be specified in "*nn*".

The following explains the value for each setting name.

2.12.3 Hadoop job name specification

Setting	Setting Name (NAME element)	Setting Value (VALUE element)	Remarks
Hadoop job name	extjoiner.jobname	Specify job name by alphanumeric character	Optional default name is "External Joiner utility"

2.12.4 Specifying MapReduce Application

Map application, specify for each file to be used.

In the Reduce application, only one is specified regardless of the number of files used.

Setting	Setting Name (NAME element)	Setting Value (VALUE element)	Remarks
Map application	extjoiner.map.streamprocessor.*nn*	(*1) (*2)	Optional to omit either Map or Reduce application
Reduce application	extjoiner.reduce.streamprocessor		

*1: When Linking with Interstage Big Data Parallel Processing Server

Specify MapReduce application's path to absolute path of the mount directory of DFS.

*2: When Linking with Apache Hadoop

If the shared directory method is used for distributing MapReduce applications, the full path of the shared directory is given. If side data is used, then the path specified as side data is given.'

When you omit Reduce application, please specify 0 as the property "mapred.reduce.tasks" of Apache Hadoop.

Information

The parameters for the Map and Reduce applications can be specified at the same time. If multiple parameters are specified, they should be space separated.

2.12.5 Threshold return value for re-execution of application

Regarding the return value passed by the Map application or Reduce application, the threshold return value for re-execution of the application is specified with the following setting name.

Setting	Setting Name (NAME element)	Setting Value (VALUE element)	Remarks
Threshold return value for re-execution of application	extjoiner.command.retryexitstatus	Regarding return value passed by MapReduce application, threshold value is specified whether re-execution will be performed or not. If the return value exceeds the value specified, the task has failed and re-execution is performed by MapReduce framework	Optional default value is 255

2.12.6 Threshold return value for treating job as error

Regarding the return value passed by the Map application or Reduce application, the threshold return value for treating job as error is specified with the following setting name.

Setting	Setting Name (NAME element)	Setting Value (VALUE element)	Remarks
Threshold return value for treating job as error	extjoiner.command.jobfailurestatus	Regarding the return value returned from multiple MapReduce applications running the job, the maximum value is considered as the set value If the maximum value specified is exceeded, job failure message is generated	Optional default value is 128

2.12.7 Specifying environment variables

Environment variables that enable the Map application or Reduce application are specified with the following settings.

(The path necessary for COBOL execution is already set and there is no need to specify.)

Setting	Setting Name (NAME element)	Setting Value (VALUE element)	Remarks
Environment variables of Map application	extjoiner.map.environment	Environment variable name = Environment variable value If multiple are specified then they should be space separated	Optional
Environment variables of Reduce application	extjoiner.reduce.environment		
Example	<name>extjoiner.reduce.environment</name> <value>CBR_MESSOUTFILE=message.txt</value>		

If any path is specified to the "LD_LIBRARY_PATH" then add "${com.fujitsu.netcobol.ld_library_path}:" before the value.

For example:

```
<name>extjoiner.reduce.environment</name>
<value>LD_LIBRARY_PATH=${com.fujitsu.netcobol.ld_library_path}:/mnt/nfs/user/lib</value>
```

Information

It is possible to use the execution initialization file "COBOL.CBR" also. "COBOL.CBR" saved at the same path as MapReduce application on DFS will be used.
If the environment variable regarding Map application or Reduce application is specified and the contents of the initialization file for execution "COBOL.CBR" are duplicate, contents of initialization file for execution "COBOL.CBR" are given precedence.

Information

The following environment variables are set to Map application and Reduce application, by the Hadoop linking function

environment variables name	overview
MAPRED_TASK_ID	Task ID which allocated to the various tasks

MAPRED_COUNTER_FILE	File storage path for writing the User definition counter

2.12.8 Specifying Input Output file

File name is specified for each file used.

Setting	Setting Name (NAME element)	Setting Value (VALUE element)	Remarks
Hadoop Input data file name	extjoiner.input.*nn*.filename	Specify file name on DFS or directory name (*1)	Required If directory is specified then all files inside that directory are considered as input file
Base directory name to save Hadoop output data file	mapred.output.dir	Specify directory name on DFS (*1)	Required
Directory name of Hadoop output data file	extjoiner.output.*nn*.filename	Specify directory name on DFS (*1)	Required Under the name of the Hadoop output data file base directory, directory with specified name is created, the output files will be stored under each task.
Map input data file unique name	extjoiner.map.input.*nn*.filename	Specify unique name of file used with Map application	Optional if Map application is not used
Map output data file unique name	extjoiner.map.output.*nn*.filename		
Reduce input data file unique name	extjoiner.reduce.input.*nn*.filename	Specify unique name of file used with Reduce application	Optional if Reduce application is not used
Reduce output data file unique name	extjoiner.reduce.output.*nn*.filename		

*1: The directory name can be specified with a full path or relative path on the DFS. Relative path is a path relative to the home directory on the DFS.

Information

The file specified in the Hadoop input data file name passes the records to the MapReduce application in their order of definition (nn).

If multiple Hadoop input data files are used in the file comparison processing then by specifying the master file first, performance is improved as wait time for read is reduced and at the same time memory is used efficiently.

Note

Hadoop input data file names and the Hadoop output data file directory name should not contain characters ',' or '='.

2.12.9 Specifying the standard directory to output data file to writing

Specify whether a Hadoop output data file standard directory is overwritten or not. When specification is omitted, if the Hadoop output data file standard directory already exists, an error occurs.

The contents of specification are as follows.

Setting	Setting name (NAME ELEMENT)	Setting value (*1) (VALUE COMPONENT)	Remarks
Specify writing to the output data file standard Directory	extjoiner.output.dir.removeifexist	- true Overwrite if Directory already exists. - false An error occurs if the Directory already exists.	Optional Default value is false

2.12.10 Specifying file format

File composition is specified for each file used. If specifying file composition is optional then it is considered as if line sequential file is specified. Settings for file composition are as follows:

Setting	Setting Name (NAME element)	Setting Value (*1) (VALUE element)	Remarks
Hadoop input data file's file composition	extjoiner.input.*nn*.format	- CobolLineSeqInputFormat Line sequential file - CobolSeqFixInputFormat Fixed length record sequential file - CobolSeqVarInputFormat Variable-length record sequential file - CobolPhysicalInputFormat physical sequential file	Optional Default value is line sequential file
Hadoop output data file's file composition	extjoiner.output.*nn*.format	- CobolLineSeqOutputFormat Line sequential file - CobolSeqFixOutputFormat Fixed length record sequential file - CobolSeqVarOutputFormat Variable-length record sequential file - CobolPhysicalOutputFormat physical sequential file	Optional Default value is line sequential file
Map output data file's file composition	extjoiner.map.output.*nn*.format	- CobolLineSeqOutputFormat Line sequential file - CobolSeqFixOutputFormat Fixed length record sequential file - CobolSeqVarOutputFormat Variable-length record sequential file	Optional Default value is line sequential file

Setting	Setting Name (NAME element)	Setting Value (*1) (VALUE element)	Remarks
		- CobolPhysicalOutputFormat physical sequential file	
Reduce input data file's file composition	extjoiner.reduce.input.*nn*.format	- CobolLineSeqInputFormat Line sequential file - CobolSeqFixInputFormat Fixed length record sequential file - CobolSeqVarInputFormat Variable-length record sequential file - CobolPhysicalInputFormat physical sequential file	Optional Default value is line sequential file

(*1) The Setting Value is modified by "com.fujitsu.netcobol.hadoop.mapred."

2.12.11 Specifying record length (Only fixed-length record sequential file)

If the file composition is fixed-length record sequential then it is necessary to specify the length of the record file. Contents that can be specified in the record length are as follows:

Setting	Setting Name (NAME element)	Setting Value (VALUE element)	Remarks
Hadoop input data file's record length	extjoiner.input.*nn*.recordlength	Integer	Required Record length is specified in bytes
Hadoop output data file's record length	extjoiner.output.*nn*.recordlength		
Map output data file's record length	extjoiner.map.output.*nn*.recordlength		
Reduce input data file's record length	extjoiner.reduce.input.*nn*.recordlength		

Note

If file composition is other than fixed-length record sequential file, do not specify the record length.

If the file composition is fixed-length record sequential file and the record length is not specified, a Runtime error will occur.

2.12.12 Specifying record length information file (For variable-length record sequential file or physical sequence file only)

If the file composition is variable-length record sequential file or physical sequence file, it is necessary to specify the record information file. For the record information file, refer to "Using a variable-length record sequential file".
Contents that can be specified in the record length information file are as follows:

Setting	Setting Name (NAME element)	Setting Value (VALUE element)	Remarks
Hadoop input data file's record information file	extjoiner.input.*nn*.recinfdir	Specify the absolute path of the directory containing the record	Required

Setting	Setting Name (NAME element)	Setting Value (VALUE element)	Remarks
		length information file on DFS	

Note

The Record length information file cannot be saved in the directory specified in the Hadoop input data file.

2.12.13 Specifying Character Encoding (Only line sequential file)

It is necessary to specify the character encoding if the file composition is line sequential file. Following are the character encodings that can be specified.

Setting	Setting Name (NAME element)	Setting Value (VALUE element)	Remarks
Hadoop input data file's character encoding	extjoiner.input.*nn*.codeset	UTF-8 UTF-16LE UTF-16BE UTF-32LE UTF-32BE SJIS	Optional Default is UTF-8 - UTF-8 Character encoding is UTF-8 - UTF-16LE Character encoding is UTF-16 little endian - UTF-16BE Character encoding is UTF-16 big endian - UTF-32LE character code is, UTF-32 Little-Endian - UTF-32BE character code is, UTF-32 Big Endian - SJIS Character encoding is SHIFT-JIS
Hadoop output data file's character encoding	extjoiner.output.*nn*.codeset		
Map output data file's character encoding	extjoiner.map.output.*nn*.codeset		
Reduce input data file's character encoding	extjoiner.reduce.input.*nn*.codeset		

Information

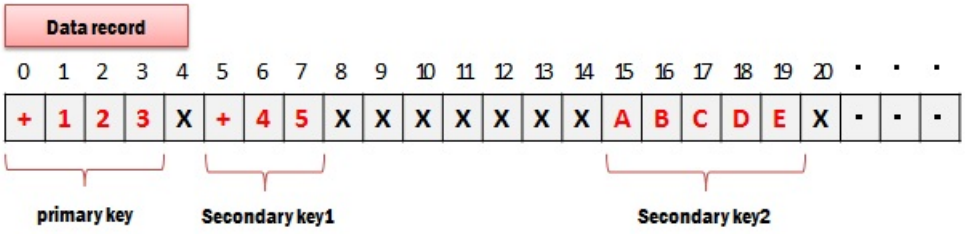
If character encoding other than line sequential file is specified for file composition, then it is ignored.

2.12.14 Specifying key information

In the key information used in Shuffle&sort, primary key used to group the data and secondary key used to arrange the data exists. It is possible to specify multiple primary and secondary keys. It is also possible to specify attributes and sorting order (ascending or descending) for each key.

It is necessary to specify the key information regarding each Hadoop input data file.

<table>
<tr><th>Setting</th><th>Setting Name
(NAME element)</th><th>Setting Value
(VALUE element)</th><th>Remarks</th></tr>
<tr><td>Primary key</td><td>sortkey.nn.main</td><td rowspan="2">- When the key is the COBOL data type
Specify as comma separated.
<code>Key attribute, offset,
length, Sort order</code>
- Key attribute
Refer "Value to specify with key attribute"
- Offset
Specify the starting offset of key in bytes
- Length
Specify key length in bytes
- Sort order
If specifying A (ascending) or D (descending) is omitted, ascending is considered.
- When the key is a CSV FORMAT data
comma separated values
<code>Key attributes,Column
item,order of a row</code>
or
<code>Key attributes,Column
item:Start offset
length,order of a row</code>
, is specified.
Key attributes
CSV or CSVN
CSV: key is evaluated as a character
CSVN: key is evaluated as a character
Column item:Start offset length
The column item where a key exists and the top offset of a key in a column and its length are specified in the byte length.</td><td>Required
Specify primary key.
If multiple are specified, then they should be / (slash) separated.
Data grouped by this key is passed to each Reduce Task.</td></tr>
<tr><td>Secondary key</td><td>sortkey.nn.sub</td><td>Optional
Specify secondary key.
If multiple are specified, then they should be / (slash) separated.
For CSV data, the column starts from 0.
When a key includes a separator character in the data it must enclosed in double quotation marks.
However, when true is specified to float field, the double quotation mark surrounding the field is not considered.</td></tr>
</table>

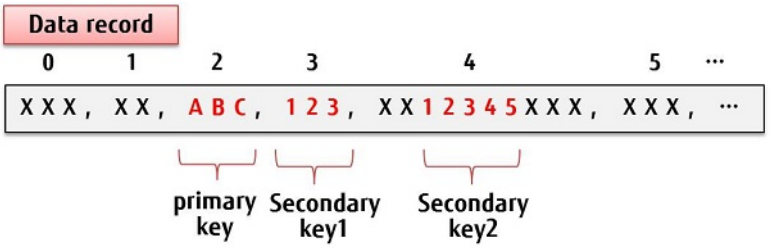
Setting	Setting Name (NAME element)	Setting Value (VALUE element)	Remarks
		When only start offset is specified, then it is considered that even the end of a column is length (when length is not specified). order of a row A (ascending order) or D (descending order) is specified and - when not specified, default order is ascending	
Sort order of ASCII code	sortkey.*nn*.colseq	- ASCII Sort by (JIS8) ASCII code - EBCDIC-ASCII Sort by EBCDIC ASCII code - EBCDIC-KANA Sort by EBCDIC KANA code - EBCDIC-LOWER Sort by EBCDIC lower case character	Optional This attribute is enabled in the case of following key attributes: - Alphabetic item - Alphanumeric item - Alphanumeric edited item
Sort order of decimal items with no SEPARATE phrase	sortkey.*nn*.decimal	- FJ Sort by Fujitsu format - MF Sort by Micro Focus-compatible format - 88 Sort by 88 Consortium format	Optional This attribute is enabled in the internal format of zoned decimal item with no SEPARATE phrase.
Order of the internal floating-point items	sortkey.*nn*.float	- IEEE Sort as IEEE format data - M Sort as M format data	Optional This attribute is enabled in the internal floating-point item.
Example	<name>com.fujitsu.netcobol.hadoop.sortkey.01.main</name> <value>S9LS,0,4,D</value> <name>com.fujitsu.netcobol.hadoop.sortkey.01.sub</name> <value>S9P,5,3/ASC,15,5</value> - Zoned decimal LEADING SEPARATE in primary key, offset from 0 byte with 4 bytes length, descending order - Signed packed decimal in first secondary key, offset from 5th byte with 3 bytes length, ascending - Alphanumeric data item in the second secondary key, offset from 15th byte with 5 bytes length, ascending - When the key is a COBOL data  	

Setting	Setting Name (NAME element)	Setting Value (VALUE element)	Remarks

```
<name>com.fujitsu.netcobol.hadoop.sortkey.01.main</name>
<value>S9LS,0,4,D</value>
```

```
<name>com.fujitsu.netcobol.hadoop.sortkey.01.sub</name>
<value>S9P,5,3/ASC,15,5</value>
```

- Primary Key, zoned decimal using LEADING SEPARATE with offset 0 byte for length of 4 bytes, descending order
- In the 1st key of the sub key, signed packed decimal, offset 5th byte for length of 3 bytes, ascending order
- In the 2nd item sub key, alphanumeric items, offset 15th byte for length of 5 bytes, ascending order.
- When the key is CSV data




```
<name>com.fujitsu.netcobol.hadoop.sortkey.01.main</name>
<value>CSV,2,A</value>
```

```
<name>com.fujitsu.netcobol.hadoop.sortkey.01.sub</name>
<value>CSVN,3,D/CSVN,4:2-5,D </value>
```

- 3rd column of CSV data in the primary key, is considered as character in ascending order.
- In the 1st sub key, 4th column of CSV data is considered as numeric in descending order.
- In the 2nd sub key, in the 5th column of CSV data 5 bytes length from 2nd byte of the offset is considered as numeric in descending order.

*: Setting name is qualified as "com.fujitsu.netcobol.hadoop".

Table 2.2 Values specified in key attribute

COBOL Data Types						Specified Value
Class	**Section**	**USAGE Clause**	**SIGN Clause**	**PICTURE Clause**	**Nickname**	
Numeric	Numeric	DISPLAY	No	9(4)	Zoned decimal	9
				S9(4)		S9
			LEADING	S9(4)		S9L
			TRAILING			S9T
			LEADING SEPARATE			S9LS
			TRAILING SEPARATE			S9TS
		PACKED-DECIMAL	-	9(4)	Packed decimal	9P
		PACKED-DECIMAL	-	S9(4)		S9P

COBOL Data Types						Specified Value
Class	Section	USAGE Clause	SIGN Clause	PICTURE Clause	Nickname	
		COMP-6	-	9(4)	Packed decimal data item without sign half-byte	9PC6
		COMP-6	-	S9(4)	Packed decimal data item with sign half-byte	S9PC6
		BINARY,COMP	-	9(4)	Normal binary	9B
		BINARY,COMP	-	S9(4)		S9B
		COMP-5	-	9(4)	System binary item	9C
		COMP-5	-	S9(4)		S9C
		BINARY-CHAR	-	9(4),S9(4)	Int type Binary Integer item	BC
		BINARY-SHORT	-			BS
		BINARY-LONG	-			BL
		BINARY-DOUBLE	-			BD
				+99.99E+99	External floating point item	EXFL
		COMP-1	-	9(4),S9(4)	Single-precision Internal floating-point items	INFL
		COMP-2	-	9(4),S9(4)	Double-precision Internal floating-point items	
	Numeric edited	DISPLAY	-	B / P V Z 0 9 , . * + - CR DB \		ASCE
Alphabetic character	Alphabetic character (*1)	DISPLAY	-	A		ASC
Alphanumeric	Alphanumeric (*1)	DISPLAY	-	A X 9		
	Alphanumeric edited (*1)	DISPLAY	-	A X 9 B 0 /		
National	National	-	-	N		NLE16 (*2) NBE16 (*2) NLE32 (*3) NBE32 (*3)
	National edited	-	-	NB		
Boolean	Boolean	DISPLAY	-	1(8)	External boolean item	BOOL

(*1) Data of ASCII (JIS8) code can be arranged as EBCDIC code. The following EBCDIC codes are supported:

- EBCDIC ASCII
- EBCDIC lower case characters
- EBCDIC kana

(*2) In UTF-16 Little-Endian "NLE16" is specified. In UTF-16 Big Endian, "NBE16", is specified.

(*3) In UTF-32 Little-Endian "NLE32" is specified. UTF-32 Big Endian, "NBE32", is specified.

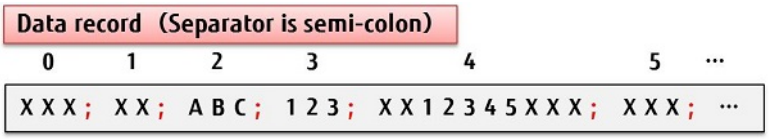
Information

When "NLE" is specified as Key attributes, it means that "NLE16" was specified. When "NBE" is specified as Key attributes, it means that "NBE16" was specified. This designation is for compatibility.

2.12.15 CSV FORMAT data usage specifications

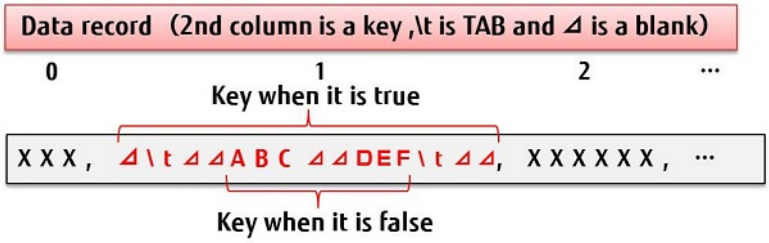
When using CSV FORMAT data, it is possible to change the separator, handling of blank spaces leading or trailing the column.

Setting	Setting name (NAME ELEMENT)	Setting value (VALUE COMPONENT)	Remarks
CSV data separator	extjoiner.csv.separator	, (comma) \t (TAB) Alphabet (a-z, A-Z) symbol (The separator value can be of 1 character only)	Optional A default is comma. However, when true is specified in the Float field specification, a blank and a tab become the default separator.

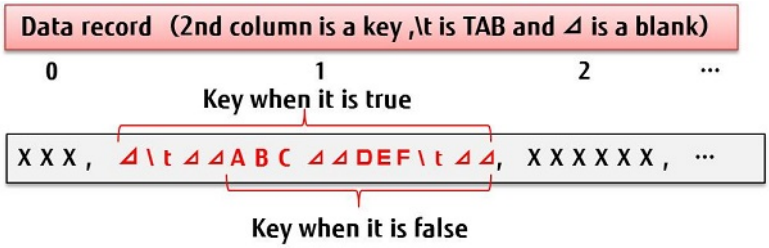
Specification example	
	`<name> extjoiner.csv.separator </name>`
	`<value>;</value>`
	- When the value is an Alphabet, a capital letter is distinguished from a lower case letter.
	- When the key is other than CSV FORMAT, the specified separator is ignored.		

Handling blanks in Primary Key for CSV data distribution processing	extjoiner.partitioner.csv.padding	true false	Optional Default is true Enabled only when the key attributes are "CSV"

Setting	Setting name (NAME ELEMENT)	Setting value (VALUE COMPONENT)	Remarks

Specification example	
	`<name> extjoiner.partitioner.csv.padding </name>`
	`<value>true</value>`
	- When true, leading and trailing blank, leading and trailing TAB are considered as part of Primary Key.
	- When false, value inside the column not including leading blank spaces, trailing blank spaces, leading TAB, trailing TAB is considered as Primary Key for partition.		

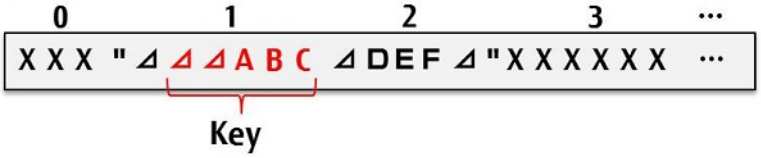
Handling blank of the key related to the CSV data sort processing	extjoiner.comparator.csv.padding	true <u>false</u>	Optional Default is false Enabled only when the key attributes are "CSV"

Specification example	
	`<name> extjoiner.comparator.csv.padding </name>`
	`<value>true</value>`
	- When true, value in the column including leading blank spaces, trailing blank spaces, leading TAB, trailing TAB is considered as key for comparison.
	- When False, value in the column including trailing blank spaces, trailing TAB is considered as key for comparison (leading blank spaces, leading TAB are not included).		

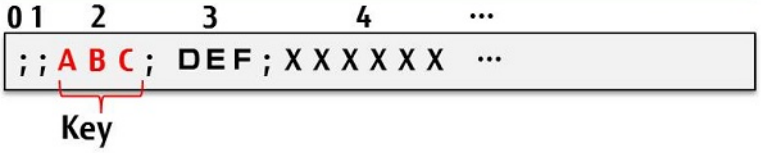
Float field specification	extjoiner.csv.floatfield	true <u>false</u>	Optional Default is false

Setting	Setting name (NAME ELEMENT)	Setting value (VALUE COMPONENT)	Remarks
Specification example			

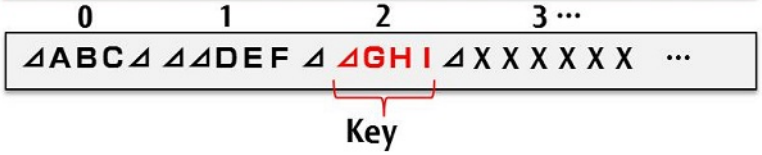
Data record (2nd column is a key , ⊿ is a blank)
Example of case where True is skipped when there is separator specification:



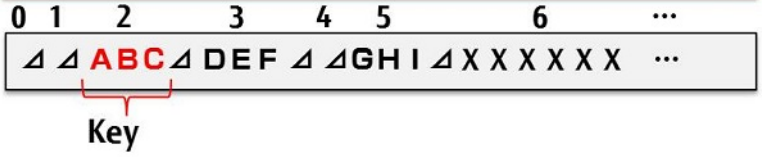
Data record (3rd column is a key)
Example when True and separator specify [;]:



Data record (3rd column is a key,⊿ is a blank)
Example when True and separator specification is skipped and at the start of the record, there is a blank :



Data record (3rd column is a key,⊿ is a blank)
Example when in True and separator specification Blank is specified and when at the start of record, there is a blank.




```
<name> extjoiner.csv.floatfield </name>
<value>false</value>
```

- When true and if a double quotation mark exists at the start of the column, it will not be treated as a double quote mark.
- When true and a CSV data separator is omitted, a blank and a tab become a separator. In such case if the blank spaces are continuing, then the first blank serves as separator and the remaining are considered as part of CSV data.
- When true and a CSV data separator is specified, in such case if separator continues, processing is done assuming that an empty column exists.it will be considered that an empty column exists and will be processed.
- When true and a CSV data separator is omitted and there is a blank or tab is in the start of a record, then those blank or tab are not taken as field separate characters.
- When true and a CSV data separator is specified and in the record header, the separator is used as a field separator character.

Note

In the CSV data separator, different values can be specified from the Float field specification. The value which can be specified is as follows.

- When Float field specification is used

  ,(comma), \t (TAB), Alphabet (a-z, A-Z),symbol

- When Float field specification is not used

  ,(comma), \t (TAB)

Using an invalid separator will generate an error in the Hadoop job instruction execution.

2.12.16 Auto sort specification for the Map output data file

When the specification of Reduce application is omitted, the Shuffle&sort does not start and the Map output file is output as un-sorted.

Specify "true" if you want to omit the Reduce application specification and start the Shuffle&sort.

Setting	Setting name (NAME ELEMENT)	Setting value (VALUE COMPONENT)	Remarks
Auto sort specification for the Map output data file	extjoiner.input.sort	true false	Optional Default is false

Note

Shuffle&sort uses Reduce Task for execution. If 0 is specified to mapred.reduce.tasks, Shuffle&sort will not start.

2.12.17 Get the log of the processing record count

When true is specified, MapReduce writes an input-output record count for each task it processes. When the specification is omitted it is considered as false is specified.

Setting	Setting name (NAME ELEMENT)	Setting value (VALUE COMPONENT)	Remarks
Specification for record count processing log	extjoiner.map.input.getRecordCount extjoiner.map.output.getRecordCount extjoiner.reduce.input.getRecordCount extjoiner.reduce.output.getRecordCount	true false	Optional Default is false

Note

The number of processing record count logs depends on the number of tasks. The error might occur when there are a high number of tasks because the counter information number has an upper limit. To start a large number of tasks, increase the counters limit setting (specified in the mapreduce.job.counters.limit property) in advance.

2.12.18 Specifying saving of the current directory

Specify whether or not the current directory of the MapReduce application will be saved after completion of the task. If omitted, it is considered as if false is specified.

If true is specified, the current directory is saved. Specify true if you want to verify the output file generated in the current directory after completion of the task.

If false is specified, the current directory will not be saved and it will be deleted at the completion of the task.

Setting	Setting Name (NAME element)	Setting Value (VALUE element)	Remarks
Specify saving of current directory	extjoiner.copyworkingdir	true false	Optional Default is false

Information

Even if false is specified and saving is omitted, files containing standard output, and standard error output (stdout, stderr) are saved.

2.12.19 Specifying buffer size

Input and output files of MapReduce applications are designed in a way that they perform buffering on the memory to increase the speed. Depending on the processing of the MapReduce application and buffering, Java heap might get used up. In that case, specify maximum value for the buffering size.

Setting	Setting Name (NAME element)	Setting Value (VALUE element)	Remarks
The number of records to be buffered	extjoiner.output.maxbufferrecords	The maximum number of records to be buffered	Optional Default is 25000

2.12.20 Specifying Primary Key List file

Specify the primary key list file.

Setting	Setting name (NAME ELEMENT)	Setting value (VALUE COMPONENT)	Remarks
Primary Key List file	extjoiner.mainkeylist	The primary Key List file name is specified with a full path or relative path.	Required

2.12.21 Specifying unique partition

It is specified when records with different primary keys are distributed to different Reduce Tasks.

To specify this setting, it is necessary to create Primary Key List file.

Setting	Setting name (NAME ELEMENT)	Setting value (VALUE COMPONENT)	Remarks
Specification of unique partition	extjoiner.partitioner.unique	true false	Optional Default is false

2.12.22 Specifying maximum number of key for unique partition

It is specified when maximum count of key is increased for a unique partition.

The upper limit of "512" is specified by default to prevent unexpected numbers of tasks from getting executed.

Setting	Setting name (NAME ELEMENT)	Setting value (VALUE COMPONENT)	Remarks
Specification of the max number of the key in unique distribution	extjoiner.partitioner.unique.max.keys	Numeric	Optional Default is 512

2.12.23 Map Task multiple file output mode

Normally, in Map application, only a Map input data file and one Map output data file can be opened, respectively. Please enable this property to output two or more files with Map application (job etc. which divide an input file according to data).

When the Map Task multiple file output is enabled, the Map output file setting is all used as setting specified in the "extjoiner.map.streamprocessor.01" of an application output file setting.

Setting	Setting name (NAME ELEMENT)	Setting value (VALUE COMPONENT)	Remarks
Map application supports multiple file output.	extjoiner.map.multioutput	true false	Optional Default is false

Note

Enabling the Map Task multiple file has the following restrictions.

- Cannot use more than 2 of MapApplication.

  When Map application is specified other than the extjoiner.map.streamprocessor.0, an error occurs in job.

- ReduceApplication cannot be specified.

  To enable this property, specify 0 to the mapred.reduce.tasks property at the same time and disable the ReduceApplication.

2.12.24 Start of Map application when Hadoop input data file is 0 byte

When Hadoop input data file is 0 byte, the Map application is not usually started. In this case, when you enable this property, you can start the application.

Setting	Setting name (NAME element)	Setting Value (VALUE element)	Remarks
Specifies to start the Map application when Hadoop input data file is 0 bytes in size	extjoiner.map.alwaysRun	true Map application starts false Map application has not started	Optional Default is false

Note

When this property is enabled and Hadoop input data file is 0 bytes in size, the following actions occur.

- A temporary file is created in the Hadoop file system while Hadoop job is executing. This temporary file is not deleted after job completion.

- This temporary file is created in the following directories.
  "extjoiner_dummy" directory under Hadoop file system directory of job execution user.

  If you want to delete temporary files after the job ends, delete this directory manually. (There is no impact on job execution, even if temporary file exists)

- The size of the temporary file is a maximum of 64 bytes.
- Hadoop log is displayed as if the one record has been passed.
  For example, when the extjoiner.map.input.getRecordCount property is true, the number of input records of Map task that are displayed in the log appears as if one record has been passed in Map application, but the records are not passed.

2.12.25 Returning the return value of Reduce application when Hadoop input data file is 0 bytes

When Hadoop input data file size is 0 bytes, the Reduce application does not return a return value. In this case, when you enable this property, the Reduce application returns a return value.

Setting	Setting name (NAME element)	Setting Value (VALUE element)	Remarks
Specifies to returning the return value of Reduce application when Hadoop input data file is 0 bytes	extjoiner.reduce.alwaysReturnExitcode	true false	Optional Default is false

Note

When this property is enabled and Hadoop input data file is 0 bytes in size, the following actions occur.

- A temporary file is created in the Hadoop file system while Hadoop job is executing. This temporary file is not deleted after job completion.
- This temporary file is created in the following directories.
  "extjoiner_dummy" directory under Hadoop file system directory of job execution user.

  If you want to delete temporary files after the job ends, delete this directory manually. (There is no impact on job execution, even if temporary file exists)
- The size of the temporary file is a maximum of 64 bytes.
- Hadoop log is displayed as if the record has been passed.

  For example, when the extjoiner.reduce.input.getRecordCount property is true, the number of input records of Reduce task that are displayed in the log appears as if number of input data files record has been passed in Reduce application, but the records are not passed.

2.12.26 Creating a Hadoop output data file of 0 byte

When data is not written in a Hadoop output data file, a Hadoop output data file is not usually created. In this case, when you enable this property, you can create a 0 byte size file.

When data is written in a Hadoop output data file regardless of the specification of this property, the output data is written as usual.

Setting	Setting name (NAME element)	Setting Value (VALUE element)	Remarks
Specifies to create a Hadoop output data file of 0 byte size	extjoiner.reduce.alwaysOutput	true 0 byte file is created false 0 byte file is not created	Optional Default is false

Note

If this property is enabled, even when file operation is not done in MapReduce application, output data file is created to the directory which is specified with extjoiner.output.nn.filename property.

Chapter 3 Trouble shooting

This chapter explains the corresponding method for troubleshooting, with Hadoop integration function.

3.1 Error during Hadoop integration function

This chapter explains the errors that occur using Hadoop integration function.

When an error occurs in the Hadoop integration function, the error message is outputted as follows.

- Executing Hadoop job in the console

  When Hadoop execution shell "cobhadoop.sh"is executed, a log is outputted in the console. Hadoop or Hadoop linking function error message is outputted in this log. Check the error message and take appropriate action.

  When the task has failed (output is FAILED), check the MapReduce task log for the cause of the failure.

- MapReduce task log

  Log file produced by the slave server when a task is executed.

  The log file destination is defined in the Hadoop "/etc/hadoop/hadoop-env.sh" configuration and is created for each job ID and task ID.

  There are three file(s) in a task log.

  - stdout

    Stores the data output by standard output from the task.

  - stderr

    Stores the data output by standard error output from the task.

  - syslog

    Messages generated by Hadoop or Hadoop linking function.

    When the task fails, please check the task log (syslog) of task ID which failed in order to correct the error. Furthermore when the execution status in the log is 134, please check the system log on the Slave Server.

- Slave Server syslog

  Log file outputted according to the syslog daemon of the operating system.

  The log file destination is indicated in the syslog configuration file (default /etc/syslog.conf). When errors occur during COBOL execution an error message is outputted. Check the error message and take appropriate action.

Check the following for errors from the console and the task log.

ID	Error Details	Action
LB0001:	There is an error in the specification of primary key information in MapReduce Configuration File. Primary key information is not specified.	Primary key information specification is required. Specify the primary key information in the MapReduce Configuration File.
LB0006:	There is an error in the CSV key information specification of the MapReduce Configuration File regarding handling of blanks in the sorting application. Setting value is ='$1'	Please verify the value specified as extjoiner.comparator.csv.padding for the MapReduce Configuration File.
LB0007:	Specification of code set of the map output file of a MapReduce Configuration File has an error. Specified value =$1	Please verify the value specified to the extjoiner.map.output.nn.codeset of MapReduce Configuration File.
LB0101:	There is an error in the specification of key information (key attribute) in MapReduce Configuration File. Specified value='$1'	Value of primary key or secondary key is not specified correctly. Incorrect value is outputted in the error message, verify the MapReduce Configuration File.

ID	Error Details	Action
LB0102:	There is an error in the specification of key information (ASCII code sequence) in MapReduce Configuration File. An empty string was specified.	Order of ASCII code is not specified. Verify the MapReduce Configuration File.
LB0103:	There is an error in the specification of key information (ASCII code sequence) in MapReduce Configuration File. Specified value='$1'	Value for the order of ASCII code is not specified correctly. Incorrect value is outputted in the error message, verify the MapReduce Configuration File.
LB0104:	There is an error in the specification of primary key information in MapReduce Configuration File.Value of the primary key information is not specified.	It is necessary to specify Primary key information. In the MapReduce Configuration File, specify primary key information.
LB0105:	There is an error in the specification of key information in MapReduce Configuration File.	Format for specifying is value to be specified as key information, key attribute, offset, length, order (optional). Verify the key information of MapReduce Configuration File.
LB0106:	There is an error in the specification of key information (key attribute) in MapReduce Configuration File. An empty string was specified.	Key attribute is not specified. Verify the MapReduce Configuration File.
LB0107:	There is an error in the specification of key information (offset) in MapReduce Configuration File. An empty string was specified.	Offset is not specified. Verify the MapReduce Configuration File.
LB0108:	There is an error in the specification of key information (offset) in MapReduce Configuration File. Specified value='$1'	Value specified for offset is not correct. Offset value can be specified in the range of 0-32759. Incorrect value is outputted in the error message, verify the MapReduce Configuration File.
LB0109:	There is an error in the specification of key information (length) in MapReduce Configuration File. An empty string was specified.	Length is not specified. Verify the MapReduce Configuration File.
LB0110:	There is an error in the specification of key information (length) in MapReduce Configuration File. Specified value='$1'	Length is not specified correctly. Length can be specified in the range of 1-32760. If an incorrect value is outputted in the error message, verify the MapReduce Configuration File.
LB0111:	There is an error in the specification of key information (length) in MapReduce Configuration File. Length from the offset exceeds the maximum record length.	Length is not specified correctly. Length specified should not exceed maximum record length by adding the offset. Verify the MapReduce Configuration File.
LB0112:	There is an error in the specification of key information (sequence) in MapReduce Configuration File. An empty string was specified.	Order is not specified. Verify the MapReduce Configuration File.
LB0113:	There is an error in the specification of key information (sequence) in MapReduce Configuration File. Specified value='$1'	Order is not specified correctly. Incorrect value is outputted in the error message. Verify the MapReduce Configuration File.
LB0117:	There is an error in the (primary) key information of the MapReduce Configuration File. The preset value of the primary key of each Hadoop input data file must match (except for offset).	When multiple primary keys are specified, consistency should be maintained in length and key attribute of each of them
LB0118:	There is an error in the (primary) key information of the MapReduce Configuration File. The CSV key and a non-CSV key are joined.	The CSV key and a non-CSV key cannot be mixed. Please verify MapReduce Configuration File.
LB0119:	There is an error in the CSV key information of the MapReduce Configuration File (column item [:start offset length]). The null character was specified.	Please check the MapReduce Configuration File for the CSV key specification.

ID	Error Details	Action
LB0120:	There is an error in the specification of the CSV key information of a MapReduce Configuration File (column item [:start offset length]). Specified value = $1	Please check the MapReduce Configuration File for the CSV key specification.
LB0121:	There is an error in the CSV key information specification of the MapReduce Configuration File (column item).The null character was specified.	Please check the MapReduce Configuration File for the CSV key specification.
LB0122:	There is an error in the CSV key information specification of the MapReduce Configuration File (column item) . Setting value is ='$1'	Please check the MapReduce Configuration File for the CSV key specification.
LB0123:	There is an error in the specification of the CSV key information of a MapReduce Configuration File (start offset-length). The null character was specified.	Please check the MapReduce Configuration File for the CSV key specification.
LB0124:	There is an error in the specification of CSV key information of a MapReduce Configuration File (start offset-length) has an error. Specified value =$1	Please check the MapReduce Configuration File for the CSV key specification.
LB0125:	There is an error in the CSV key information specification of the MapReduce Configuration File (start offset). The null character was specified.	Please check the MapReduce Configuration File for the CSV key specification.
LB0126:	There is an error in the CSV key information specification of the MapReduce Configuration File (start offset) has an error. Setting value is ='$1'	Please check the MapReduce Configuration File for the CSV key specification.
LB0127:	There is an error in the CSV key information specification of the MapReduce Configuration File (length). The blank character is specified.	Please check the MapReduce Configuration File for the CSV key specification.
LB0128:	There is an error in the CSV key information specification of the MapReduce Configuration File (length). Setting value is ='$1'	Please check the MapReduce Configuration File for the CSV key specification.
LB0129:	There is an error in the specification of key information (DECIMAL sequence) in MapReduce Configuration File. An empty string was specified.	Please check the MapReduce Configuration File for the sortkey.nn.decimal specification.
LB0130:	There is an error in the specification of key information (DECIMAL sequence) in MapReduce Configuration File. Specified value='$1'	Please check the MapReduce Configuration File for the sortkey.nn.decimal specification.
LB0131:	There is an error in the specification of key information (FLOAT sequence) in MapReduce Configuration File. An empty string was specified.	Please check the MapReduce Configuration File for the sortkey.nn.float specification.
LB0132:	There is an error in the specification of key information (FLOAT sequence) in MapReduce Configuration File. Specified value='$1'	Please check the MapReduce Configuration File for the sortkey.nn.float specification.
LB0201:	There is an error in the specification of primary key information in MapReduce Configuration File. Primary key information is not specified.	It is necessary to specify Primary key information. In the MapReduce Configuration File, specify primary key information.
LB0202:	Primary key list file specified in the MapReduce Configuration File cannot be found. FILE=$1	Please verify the primary key file name specified in the MapReduce Configuration File.

ID	Error Details	Action
LB0207:	The total of length of primary key of primary key list file is not matching with the total of length of the primary key that is specified in the MapReduce Configuration File.	Please match the length of primary keys of primary key list file with the length of sum of primary keys specified in the MapReduce Configuration File.
LB0211:	Record length information file is not valid. Offset information could not be read.	The Record length information file is not valid. Verify the record length information file created from the Hadoop input data file.
LB0212:	There is an error in the CSV key information specification of the MapReduce Configuration File regarding handling blanks in the primary key in distribution processing. Setting value is ='$1'	Please check the value specified as extjoiner.partitioner.csv.padding of the MapReduce Configuration File.
LB0217:	Specification of code set of the map output file(%s) of a MapReduce Configuration File has an error. Specified value =$1	Please verify the value specified to the extjoiner.map.output.nn.codeset of MapReduce Configuration File.
LB0218:	Primary list file could not be found nn order to perform distribution of key distribution, the FILE=$1	Please check whether extjoiner.mainkeylist is set up correctly.
LB0219:	In order to perform distribution in consideration of key distribution, it failed in reading the primary list file. FILE =$1	Please verify that primary key list file is in readable format
LB0221:	Multiple keys exist in a primary key list file. key=$1	Please review whether the same key is defined as the primary key list file.
LB0222:	Conversion of the primary key went wrong.	Please contact Support.
IO0001:	There is an error in the CSV key information specification of the MapReduce Configuration File regarding the float field. Setting value is ='$1'	Please check the value specified as extjoiner.csv.floatfield for the MapReduce Configuration File.
IO0002:	Specification of separator of the CSV data of a MapReduce Configuration File has an error. Specified value =$1	Please verify the value specified to the extjoiner.csv.separator of MapReduce Configuration File.
EX0001:	Internal error: Uncaught exception has occurred.	Please contact Support.
EX0002:	Incorrect property name. Input file number can be up to '$1': extjoiner.input.'$1'.filename	Count of Hadoop input data file name is incorrect. Verify the MapReduce Configuration File.
EX0003:	Job execution failed.	Either verify the other message generated at the same time or the task log.
EX0004:	The execution status of the command is greater than retryexitstatus('$1':Part ID of task='$2'Execution status='$3'	The return value of MapReduce application has exceeded the specified threshold value. Task is re-executed. Check the MapReduce application or MapReduce Configuration File.
EX0007:	Temporary directory for the task did not get created. Review the Hadoop settings.Temporary directory:'$1'	Task is not operating correctly. Review the Hadoop settings. Check samples provided with Hadoop and verify that the simple MapReduce application can run. If the problem persists, please contact Support.
EX0009:	Named pipe cannot be created. Review Hadoop's settings. Named Pipe:'$1'	
EX0010:	Named pipe cannot be created. Review Hadoop's settings. Named Pipe:'$1'	
EX0011:	Failed to execute the named pipe creation command mkfifo:'$1'	

ID	Error Details	Action
EX0013:	There was a failure in the start of the command	MapReduce application was not executed. Check whether it is possible to execute MapReduce application.
EX0014:	Job has been killed by the user. Attempted to forcefully terminate the input to the command	An Interrupt has been detected during the execution of the COBOL application. If a user interrupted the job, this message is generated and interruption is not affected.
EX0015:	Job has been killed by the user. Attempted to close the input to the command	
EX0016:	Input to the command was closed in the middle. Perhaps command did not read to the end of the file:	COBOL application did not open the input file or file was not read to the end. Check whether it is possible to execute the COBOL application. Verify the COBOL application and if it is possible to execute it.
EX0017:	Input to the command was closed in the middle. Perhaps command did not read to the end of the file:	
EX0024:	Value of the property is not an appropriate directory name: mapred.output.dir	There is a problem with the directory specified to mapred.output.dir. Verify the MapReduce Configuration File.
EX0025:	The environment variable name or value of environment variable is inappropriate	There is an error in the specified environment variable name or environment variable value. System dependent error occurred due to changing or setting not permitted variable name. Verify the MapReduce Configuration File.
EX0030:	Job has been killed by the user. Was waiting for the task to start.	An Interrupt has been detected during the execution of the COBOL application. If a user interrupted the job, this message is generated and interruption is not affected.
EX0033:	Job has been killed by the user. Named pipe was under creation	
EX0035:	Hadoop output data file name contains ',' or '=' : nn='$1' filename='$2'	Hadoop data file name cannot contain ',' or '='. Verify the MapReduce Configuration File.
EX0038:	1 or more is specified in mapred.reduce.tasks, but the command to be executed is not specified.	Specify Reduce application in extjoiner.reduce.streamprocessor. If Reduce application is omitted then specify 0 to mapred.reduce.tasks property.
EX0039:	Output directory exist:'$1'	Output directory already exists. Re-execute the job after deleting the output directory.
EX0045:	Failed in passing the command output to Hadoop	Review Hadoop's configuration. Check whether it is possible to run samples provided with Hadoop and a simple MapReduce application. If the problem persists, please contact Support.
EX0046:	Job has been killed by the user. Was waiting for the command execution to complete	An Interrupt has been detected during the execution of the COBOL application. If a user interrupted the job, this message is generated and interruption is not affected.
EX0047:	Job has been killed by the user. Attempted to end the reading of named pipe	
EX0048:	Job has been killed by the user. Attempted to end the writing of named pipe	
EX0049:	Job has been killed by the user. Was waiting for the completion of command execution	
EX0050:	Job has been killed by the user. Attempted to end the reading of named pipe	
EX0052:	Value of property is not a number: property='$1' value='$2'	Other than number is specified to the property to which number should be specified. By referring to the generated information verify the MapReduce Configuration File.
EX0054:	Value of the property is not a correct class name. property='$1' value='$2'	There is an error in the specified file structure. By referring to the generated information verify the MapReduce Configuration File.

ID	Error Details	Action
EX0056:	Failed in job deletion. For more information please refer the below exception job id='$1'	Manually delete the Hadoop job after verifying Hadoop's configuration.
EX0058:	Property value is out of range. Set in the range '$1'-'$2': property='$3' value='$4'	Property has crossed the specified range. By referring to the generated information verify the MapReduce Configuration File.
EX0065:	Job has been killed by the user. Attempted to close the input to the command	An Interrupt has been detected during the execution of the COBOL application. If a user interrupted the job, this message is generated and interruption is not affected.
EX0066:	Job has been killed by the user. Attempted to forcefully terminate the input to the command	
EX0070:	Line sequential file's character encoding is invalid: property='$1' value='$2'	There is an error in specifying character encoding. By referring to the generated information verify the MapReduce Configuration File.
EX0071:	Record length of the record sequential fixed-length file is not specified: property='$1'	It is necessary to specify the record length information in case fixed-length record sequential file is used. Verify the MapReduce Configuration File.
EX0073:	Record length information file of variable-length record sequential file is not specified: property='$1'	It is necessary to specify the record length information in case variable-length record sequential file is used. Verify the MapReduce Configuration File.
EX0074:	Record length information file of variable-length record sequential file is invalid: property='$1' value='$2'	Record length information file could not be accessed. Verify the path of the record length information file and the MapReduce Configuration File.
EX0075:	Failed in starting the execution of command. Refer to the exception details for more information.	While starting with command execution RuntimeException is detected. Please contact Support.
EX0076:	Failed in recovering the files of current working directory. Refer to the exception details for more information.	IOException occurred during the recovery of files. Please contact Support.
EX0077:	Failed in recovering the name of current working directory. Re-validate Hadoop's setting (mapred.working.dir).	Confirm Hadoop's setting (mapred.working.dir) and check whether or not a correct path is specified.
EX0083:	Failed in writing to the input file of command.	While writing the input file an IOException occurred. At times this message is generated when the reading of COBOL application's input file is completed without reading till the end. If the problem persists, please contact Support.
EX0084:	Job has been killed by the user. Attempted to export the input to the command	An Interrupt has been detected during the execution of the COBOL application. If a user interrupted the job, this message is generated and interruption is not affected.
EX0086:	There was no data to be processed in the reduce	0 records are entered in the Reduce Task.
EX0087:	There was no data to be processed in the map	0 records are entered in the Map Task.
EX0088:	Value of the required property is not specified. Specify the output directory name: mapred.output.dir	Property mapred.output.dir is required. Verify the MapReduce Configuration File.
EX0089:	Value of the required property is not specified. Property name: '$1'	Required property is omitted. Verify the MapReduce Configuration File by referring to the generated information.
EX0090:	Not even a single Hadoop input data file is specified. Property name: extjoiner.input.NN.filename	Specification of Hadoop input data file property is required. Verify the MapReduce Configuration File.
EX0091:	There was a problem in specifying the Hadoop input data file. '$1'	There is an error in the path specified in the Hadoop input data file. Verify the path and the MapReduce Configuration File.
EX0095:	Inappropriate value is assigned to a required property. Could not create the output directory. '$1' '$2'='$3'	There is an error in the path specified in the Hadoop output data file. Verify the path and the MapReduce Configuration File.

ID	Error Details	Action
EX0097:	Temporary file for buffer could not be created.	Failed in creating a temporary file required for creation of input of COBOL application. Please contact Support.
EX0098:	Invalid environment variable is set. Property name: '$1' Value: '$2'	There is an error in the specified environment variable name or value. It is necessary to separate environment variable name by '=' from environment variable's value. Verify the MapReduce Configuration File.
EX0100:	Hadoop input data file name contains ',' or ';' : nn=%s filename=%s	Please check the value specified as extjoiner.input.nn.filename of MapReduce Configuration File.
EX0102:	Failed to read the Counter file	IOException occurred while updating the user definition counter. Please contact support.
EX0103:	Failed to update the user definition counter. The value of counter cannot be converted. $1	The format of the User definition counter is invalid. In the counter value, please specify the numeric value.
EX0104:	Failed to update the user definition counter. The format is incorrect. $1	The format of the User definition counter is invalid. Please describe group name, counter name, counter value in their respective format.
EX0105:	When Map multioutput is specified, extjoiner.input. 01 must be specified	When using Map multiple output function, extjoiner.input.01 must be specified. Please specify an input file.
EX0106:	When Map multioutput is specified, Hadoop input data file cannot be specified excluding extjoiner.input.01	When using Map multiple output function, no input files can be specified except extjoiner.input.01. Please confirm the MapReduce Configuration File.
EX0107:	When Map multioutput is specified, please specify 0 for the number of Reduce Tasks	When using the Map multiple output function, It is required to specify Reduce Task count as 0.
EX0108:	When Map multioutput is specified, map.streamprocessor.01 must be specified	When using the Map multiple output function, it is required to specify map.streamprocessor.01.Please specify the Map application.
EX0109:	When Map multioutput is specified, Map application cannot be specified excluding map.streamprocessor. 01	When using the Map multiple output function map.streamprocessor. Map application only 01 can be specified. Please confirm, the MapReduce Configuration File.
EX0111:	Specification of extjoiner.input.sort is disregarded because that mapred.reduce.tasks is 0	When the map output data file uses the auto sort function, it is required to specify more than 1 in the Reduce Task count.
EX0112:	Specification of extjoiner.reduce.streamprocessor is disregarded because that mapred.reduce.tasks is 0	There is an IOException while updating the user definition counter. Please contact the support.
EX0113:	The record length information file of physical sequential file is not specified: property=$1	When using the physical sequential file, record length information file must be specified. Please confirm the MapReduce Configuration File.
EX0114:	The record length information file of physical sequential file is incorrect: property=$1 value=$2	The record length information file cannot be accessed. Please confirm the MapReduce Configuration File and the record length information file.
EX0115:	There is no write authority in the directory. The output directory cannot be made	Please confirm the access rights of the path specified in the mapred.output.dir of the MapReduce Configuration File.
EX0116:	The parent directory of the output directory cannot be made. The output directory cannot be made	
EX0117:	There is no write authority in the parent directory of the output directory. The output directory cannot be made	
EX0118:	The name of the parent directory of the output directory cannot be acquired. The output directory cannot be made	

ID	Error Details	Action
EX0119:	The map and reduce application is not specified	When the auto sorting of Map output data file is not specified it is not possible to skip both map and reduce application.
EX0120:	The format of user definition counter has an error. It was not correctly totaled	Please check the format of the counter information written in the counter information configuration file.
EX0121:	There is contradiction in the value of the property. $1=$2 $3=$4	By referring to the displayed message, please check if there is any inconsistency in setup name and the installation value.
EX0122:	The wrong parameter was primary key list file creation shell.	Please check the argument for the primary key list file creation shell.
EX0124:	There is a failure in the close processing of the input file of reducer.	When job execution is successful, no problem exists even if this message is produced. When job execution fails, please contact Support.
EX0126:	There is a failure in the close processing of the output file of reducer.	It is possible that there is insufficient free space on the disk used by Reduce application. Please check the disk storage capacity of the Slave Server.
EX0128:	Primary key conversion failed.	Please contact Support.
EX0129:	There is an error in the specification of the key information of a MapReduce Configuration File.	Please check the specification of key information of the MapReduce Configuration File.
EX0130:	Key information (key attribute) of Mapreduce information file is not specified correctly. Blank space is specified.	Please check the key information (key attribute) of the MapReduce Configuration File.
EX0131:	The primary key list file contains inaccurate lines. Line:$1 $2	Please correct the format or character code referring indicated in the message
EX0132:	The primary key list file could not be found.	Please check whether extjoiner.mainkeylist of the MapReduce Configuration File is set up correctly.
EX0133:	Unable to read the primary key list file.	Please check whether access authority for the file name specified as extjoiner.mainkeylist of the MapReduce Configuration File is set up correctly.
EX0134:	Failed to create optimization information on the distribution of a primary key.	Please check whether a current directory has authority to create the file.
EX0135:	Failure in the optimization of the number of Reduce tasks.	Please contact Support.
EX0137:	The number of Reduce tasks could not be optimized. Could not acquire environment execution information.	Please contact Support.
EX0140:	Job could not complete due to exceeding the maximum count for unique key distribution.	Please correct the maximum count of the unique key distribution if needed.
EX0142:	Specification of blank handling of the key in the shuffle&sort of the CSV data of a MapReduce Configuration File has an error. Specified value =$1	Please check the value specified in the extjoiner.partitioner.csv.padding of the MapReduce Configuration File.
EX0143:	The primary key is not defined in the primary key list file.	When using Shuffle&sort, primary key must be specified. Please confirm the MapReduce Configuration File.
	Internal inconsistency has occurred. (CODE=NNNN)	Please contact Support.
	Message other than the above	Please contact Support.

3.2 Specification of the Slave Server by which the task was performed

The Hadoop execution environment consists of multiple Slave Servers.

There is no issue when the task log with Job error etc. is configured to write to the ETERNUS shared directory but when configured to write to the local disk of the Slave Server, when an error occurs, the Slave Server executing the task must be specified.

The role of JobTracker is to divide the job into multiple tasks and assign them to a Slave Server.

The JobTracker log file records how tasks are assigned.

The log location is defined in the Hadoop configuration file "/etc/hadoop/hadoop-env.sh".

Check the Master Server task log.

```
$ more /var/log/hadoop/mapred/hadoop-mapred-jobtracker-hadoop1.log
```

(Log excerpt)

```
INFO org.apache.hadoop.mapred.JobTracker: Adding task (MAP) 'attempt_201403281518_0347_m_000001_0'
 to tip task_201403281518_0347_m_000001, for tracker 'tracker_hadoop2:localhost.localdomain/
127.0.0.1:57728'
INFO org.apache.hadoop.mapred.JobTracker: Adding task (MAP) 'attempt_201403281518_0347_m_000002_0'
 to tip task_201403281518_0347_m_000002, for tracker 'tracker_hadoop2:localhost.localdomain/
127.0.0.1:57728'
INFO org.apache.hadoop.mapred.JobTracker: Adding task (REDUCE) 'attempt_201403281518_0347_r_000000_0'
 to tip task_201403281518_0347_r_000000, for tracker 'tracker_hadoop2:localhost.localdomain/
127.0.0.1:57728'
```

To determine that the Map Task 0 number, 1 number and Redcuce task 0 number was allocated to Hadoop 2, please check the task log of the Hadoop2.

3.3 Example of outputting log and confirm method of the content

When the Hadoop linkage function is used a log is generated with an explanation and example for each successful and failed job.

When execute a job using a correct input data, program or MapReduce Configuration File, job information is generated after the Reduce Task is completed and the Hadoop execution shell ends normally. Hadoop generates Job information such as input-output record count of the Map/Reduce Task and return value of the task.

When execute a job using an incorrect input data, program or MapReduce Configuration File, the job ends with an error. However, Hadoop retry to execute the job (default 3 times) when an error occurs in a task. If the error persists, the job ends in an error.

3.3.1 Example of job success

The following is an example in which Hadoop Job execution succeeds. The Map Task and a Reduce Task both complete without error messages at the end of the Hadoop execution shell.

```
[hadoop@hadoop1 hadoop01]$ cobhadoop.sh -conf  conf/configuration.xml -files reduce.exe
14/04/24 10:20:41 INFO util.NativeCodeLoader: Loaded the native-hadoop library
14/04/24 10:20:41 WARN snappy.LoadSnappy: Snappy native library not loaded
14/04/24 10:20:41 INFO mapred.FileInputFormat: Total input paths to process : 1
14/04/24 10:20:41 INFO mapred.FileInputFormat: Total input paths to process : 1
14/04/24 10:20:41 INFO mapred.JobClient: Running job: job_201403281518_0347
14/04/24 10:20:42 INFO mapred.JobClient:  map 0% reduce 0%
14/04/24 10:21:00 INFO mapred.JobClient:  map 25% reduce 0%
14/04/24 10:21:01 INFO mapred.JobClient:  map 50% reduce 0%
14/04/24 10:21:05 INFO mapred.JobClient:  map 75% reduce 0%
14/04/24 10:21:07 INFO mapred.JobClient:  map 100% reduce 0%
14/04/24 10:21:10 INFO mapred.JobClient:  map 100% reduce 33%
14/04/24 10:21:13 INFO mapred.JobClient:  map 100% reduce 100%
14/04/24 10:21:16 INFO mapred.JobClient: Job complete: job_201403281518_0347
14/04/24 10:21:16 INFO mapred.JobClient: Counters: 31
```

```
14/04/24 10:21:16 INFO mapred.JobClient:   Job Counters
14/04/24 10:21:16 INFO mapred.JobClient:     Launched reduce tasks=1
14/04/24 10:21:16 INFO mapred.JobClient:     SLOTS_MILLIS_MAPS=28884
14/04/24 10:21:16 INFO mapred.JobClient:     Total time spent by all reduces waiting after reserving
slots (ms)=0
14/04/24 10:21:16 INFO mapred.JobClient:     Total time spent by all maps waiting after reserving
slots (ms)=0
14/04/24 10:21:16 INFO mapred.JobClient:     Launched map tasks=4
14/04/24 10:21:16 INFO mapred.JobClient:     Data-local map tasks=4
14/04/24 10:21:16 INFO mapred.JobClient:     SLOTS_MILLIS_REDUCES=12510
14/04/24 10:21:16 INFO mapred.JobClient:   File Input Format Counters
14/04/24 10:21:16 INFO mapred.JobClient:     Bytes Read=0
14/04/24 10:21:16 INFO mapred.JobClient:   File Output Format Counters
14/04/24 10:21:16 INFO mapred.JobClient:     Bytes Written=1260
14/04/24 10:21:16 INFO mapred.JobClient:   FileSystemCounters
14/04/24 10:21:16 INFO mapred.JobClient:     FILE_BYTES_READ=320586
14/04/24 10:21:16 INFO mapred.JobClient:     HDFS_BYTES_READ=194818
14/04/24 10:21:16 INFO mapred.JobClient:     FILE_BYTES_WRITTEN=809649
14/04/24 10:21:16 INFO mapred.JobClient:     HDFS_BYTES_WRITTEN=1260
14/04/24 10:21:16 INFO mapred.JobClient:   Extjoiner.ExitStatus
14/04/24 10:21:16 INFO mapred.JobClient:     reduce.00=0
14/04/24 10:21:16 INFO mapred.JobClient:   Map-Reduce Framework
14/04/24 10:21:16 INFO mapred.JobClient:     Map output materialized bytes=320604
14/04/24 10:21:16 INFO mapred.JobClient:     Map input records=10020
14/04/24 10:21:16 INFO mapred.JobClient:     Reduce shuffle bytes=320604
14/04/24 10:21:16 INFO mapred.JobClient:     Spilled Records=20040
14/04/24 10:21:16 INFO mapred.JobClient:     Map output bytes=300540
14/04/24 10:21:16 INFO mapred.JobClient:     Total committed heap usage (bytes)=826540032
14/04/24 10:21:16 INFO mapred.JobClient:     CPU time spent (ms)=5870
14/04/24 10:21:16 INFO mapred.JobClient:     Map input bytes=190320
14/04/24 10:21:16 INFO mapred.JobClient:     SPLIT_RAW_BYTES=1032
14/04/24 10:21:16 INFO mapred.JobClient:     Combine input records=0
14/04/24 10:21:16 INFO mapred.JobClient:     Reduce input records=10020
14/04/24 10:21:16 INFO mapred.JobClient:     Reduce input groups=40
14/04/24 10:21:16 INFO mapred.JobClient:     Combine output records=0
14/04/24 10:21:16 INFO mapred.JobClient:     Physical memory (bytes) snapshot=877379584
14/04/24 10:21:16 INFO mapred.JobClient:     Reduce output records=20
14/04/24 10:21:16 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=4984156160
14/04/24 10:21:16 INFO mapred.JobClient:     Map output records=10020
[hadoop@hadoop1 hadoop01]$
```

3.3.2 Example of a job failure (Example when execution program is not found)

Following is an example where the program not existing is taken as Reduce application, and is specified in the MapReduce Configuration File and there is a failure of the Hadoop Job execution.

Reduce Task progresses halfway (this example completes 66%) until a timeout occurs and a failure in the Reduce Task. Upon retry job still ends in an error condition.

```
[hadoop@hadoop1 hadoop01]$ cobhadoop.sh -conf  conf/configuration.xml
14/04/24 16:04:02 INFO util.NativeCodeLoader: Loaded the native-hadoop library
14/04/24 16:04:02 WARN snappy.LoadSnappy: Snappy native library not loaded
14/04/24 16:04:03 INFO mapred.FileInputFormat: Total input paths to process : 1
14/04/24 16:04:03 INFO mapred.FileInputFormat: Total input paths to process : 1
14/04/24 16:04:03 INFO mapred.JobClient: Running job: job_201403281518_0350
14/04/24 16:04:04 INFO mapred.JobClient:  map 0% reduce 0%
14/04/24 16:04:17 INFO mapred.JobClient:  map 50% reduce 0%
14/04/24 16:04:23 INFO mapred.JobClient:  map 75% reduce 0%
14/04/24 16:04:24 INFO mapred.JobClient:  map 100% reduce 0%
14/04/24 16:04:26 INFO mapred.JobClient:  map 100% reduce 25%
14/04/24 16:04:34 INFO mapred.JobClient: Task Id : attempt_201403281518_0350_r_000000_0, Status :
```

```
FAILED
14/04/24 16:04:35 INFO mapred.JobClient:  map 100% reduce 0%
14/04/24 16:04:42 INFO mapred.JobClient:  map 100% reduce 33%
14/04/24 16:04:45 INFO mapred.JobClient: Task Id : attempt_201403281518_0350_r_000000_1, Status :
FAILED
14/04/24 16:04:46 INFO mapred.JobClient:  map 100% reduce 0%
14/04/24 16:04:53 INFO mapred.JobClient:  map 100% reduce 25%
14/04/24 16:04:56 INFO mapred.JobClient: Task Id : attempt_201403281518_0350_r_000000_2, Status :
FAILED
14/04/24 16:04:57 INFO mapred.JobClient:  map 100% reduce 0%
14/04/24 16:05:04 INFO mapred.JobClient:  map 100% reduce 33%
14/04/24 16:05:07 INFO mapred.JobClient:  map 100% reduce 0%
14/04/24 16:05:10 INFO mapred.JobClient: Job complete: job_201403281518_0350
14/04/24 16:05:10 INFO mapred.JobClient: Counters: 24
14/04/24 16:05:10 INFO mapred.JobClient:   Job Counters
14/04/24 16:05:10 INFO mapred.JobClient:     Launched reduce tasks=4
14/04/24 16:05:10 INFO mapred.JobClient:     SLOTS_MILLIS_MAPS=30402
14/04/24 16:05:10 INFO mapred.JobClient:     Total time spent by all reduces waiting after reserving
slots (ms)=0
14/04/24 16:05:10 INFO mapred.JobClient:     Total time spent by all maps waiting after reserving
slots (ms)=0
14/04/24 16:05:10 INFO mapred.JobClient:     Launched map tasks=4
14/04/24 16:05:10 INFO mapred.JobClient:     Data-local map tasks=4
14/04/24 16:05:10 INFO mapred.JobClient:     Failed reduce tasks=1
14/04/24 16:05:10 INFO mapred.JobClient:     SLOTS_MILLIS_REDUCES=50585
14/04/24 16:05:10 INFO mapred.JobClient:   File Input Format Counters
14/04/24 16:05:10 INFO mapred.JobClient:     Bytes Read=0
14/04/24 16:05:10 INFO mapred.JobClient:   FileSystemCounters
14/04/24 16:05:10 INFO mapred.JobClient:     HDFS_BYTES_READ=194818
14/04/24 16:05:10 INFO mapred.JobClient:     FILE_BYTES_WRITTEN=452088
14/04/24 16:05:10 INFO mapred.JobClient:   Map-Reduce Framework
14/04/24 16:05:10 INFO mapred.JobClient:     Map output materialized bytes=320604
14/04/24 16:05:10 INFO mapred.JobClient:     Map input records=10020
14/04/24 16:05:10 INFO mapred.JobClient:     Spilled Records=10020
14/04/24 16:05:10 INFO mapred.JobClient:     Map output bytes=300540
14/04/24 16:05:10 INFO mapred.JobClient:     Total committed heap usage (bytes)=764739584
14/04/24 16:05:10 INFO mapred.JobClient:     CPU time spent (ms)=3130
14/04/24 16:05:10 INFO mapred.JobClient:     Map input bytes=190320
14/04/24 16:05:10 INFO mapred.JobClient:     SPLIT_RAW_BYTES=1032
14/04/24 16:05:10 INFO mapred.JobClient:     Combine input records=0
14/04/24 16:05:10 INFO mapred.JobClient:     Combine output records=0
14/04/24 16:05:10 INFO mapred.JobClient:     Physical memory (bytes) snapshot=777326592
14/04/24 16:05:10 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=4139409408
14/04/24 16:05:10 INFO mapred.JobClient:     Map output records=10020
EX0003:Job execution failed.
[hadoop@hadoop1 hadoop01]$
```

In the example, the task is understood to error due to "attempt_201403281518_0350_r_000000_0".

Hence, check the details of the error in the task log.

Checking the Slave Server task log.

```
$ more /var/log/hadoop/mapred/userlogs/job_201403281518_0350/attempt_201403281518_0350_r_000000_0/
syslog
```

The task execution information for the Hadoop output is included in the log. Search within the log for the error (ERROR) and exception (Exception).

```
2014-04-24 16:04:52,279 INFO com.fujitsu.labs.extjoiner.MidFileCommandBuilder: EX0013:There was a
failure in the start of the command
java.io.IOException: Cannot run program "./reduce.exe": java.io.IOException: error=2, No such file or
directory
        at java.lang.ProcessBuilder.start(ProcessBuilder.java:460)
```

```
        at com.fujitsu.labs.extjoiner.MidFileCommandBuilder.startCommand(MidFileCommandBuilder.java:
841)
        at com.fujitsu.labs.extjoiner.MidFileCommandBuilder.onConfigure(MidFileCommandBuilder.java:
475)
        at com.fujitsu.labs.extjoiner.ExtJoinerReduce.configure(ExtJoinerReduce.java:122)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
        at java.lang.reflect.Method.invoke(Method.java:597)
        at org.apache.hadoop.util.ReflectionUtils.setJobConf(ReflectionUtils.java:88)
        at org.apache.hadoop.util.ReflectionUtils.setConf(ReflectionUtils.java:64)
        at org.apache.hadoop.util.ReflectionUtils.newInstance(ReflectionUtils.java:117)
        at org.apache.hadoop.mapred.ReduceTask.runOldReducer(ReduceTask.java:485)
        at org.apache.hadoop.mapred.ReduceTask.run(ReduceTask.java:420)
        at org.apache.hadoop.mapred.Child$4.run(Child.java:255)
        at java.security.AccessController.doPrivileged(Native Method)
        at javax.security.auth.Subject.doAs(Subject.java:396)
        at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1121)
        at org.apache.hadoop.mapred.Child.main(Child.java:249)
Caused by: java.io.IOException: java.io.IOException: error=2, No such file or directory
        at java.lang.UNIXProcess.<init>(UNIXProcess.java:148)
        at java.lang.ProcessImpl.start(ProcessImpl.java:65)
        at java.lang.ProcessBuilder.start(ProcessBuilder.java:453)
        ... 17 more
2014-04-24 16:04:52,280 ERROR com.fujitsu.labs.extjoiner.MidFileCommandBuilder: EX0075:There was a
failure in the command execution. For details please refer to the Exception information.
java.lang.RuntimeException: There was a failure in the start of the command
```

The missing "./reduce.exe" is understood to be the source of the error.

3.3.3 Job failure example (Error occurs during execution of the COBOL program)

Following are examples of job failure errors during execution of the COBOL program when the COBOL program runs through the Reduce application. The job ends in failure although the Reduce Task completes.

```
[hadoop@hadoop1 hadoop01]$ cobhadoop.sh -conf  conf/configuration.xml -files reduce.exe
14/04/24 10:44:44 INFO util.NativeCodeLoader: Loaded the native-hadoop library
14/04/24 10:44:44 WARN snappy.LoadSnappy: Snappy native library not loaded
14/04/24 10:44:44 INFO mapred.FileInputFormat: Total input paths to process : 1
14/04/24 10:44:44 INFO mapred.FileInputFormat: Total input paths to process : 1
14/04/24 10:44:44 INFO mapred.JobClient: Running job: job_201403281518_0349
14/04/24 10:44:45 INFO mapred.JobClient:  map 0% reduce 0%
14/04/24 10:45:01 INFO mapred.JobClient:  map 25% reduce 0%
14/04/24 10:45:02 INFO mapred.JobClient:  map 50% reduce 0%
14/04/24 10:45:07 INFO mapred.JobClient:  map 75% reduce 0%
14/04/24 10:45:08 INFO mapred.JobClient:  map 100% reduce 0%
14/04/24 10:45:11 INFO mapred.JobClient:  map 100% reduce 33%
14/04/24 10:45:14 INFO mapred.JobClient:  map 100% reduce 100%
14/04/24 10:45:17 INFO mapred.JobClient: Job complete: job_201403281518_0349
14/04/24 10:45:17 INFO mapred.JobClient: Counters: 31
14/04/24 10:45:17 INFO mapred.JobClient:   Job Counters
14/04/24 10:45:17 INFO mapred.JobClient:     Launched reduce tasks=1
14/04/24 10:45:17 INFO mapred.JobClient:     SLOTS_MILLIS_MAPS=29916
14/04/24 10:45:17 INFO mapred.JobClient:     Total time spent by all reduces waiting after reserving
slots (ms)=0
14/04/24 10:45:17 INFO mapred.JobClient:     Total time spent by all maps waiting after reserving
slots (ms)=0
14/04/24 10:45:17 INFO mapred.JobClient:     Launched map tasks=4
14/04/24 10:45:17 INFO mapred.JobClient:     Data-local map tasks=4
14/04/24 10:45:17 INFO mapred.JobClient:     SLOTS_MILLIS_REDUCES=12460
14/04/24 10:45:17 INFO mapred.JobClient:   File Input Format Counters
```

```
14/04/24 10:45:17 INFO mapred.JobClient:     Bytes Read=0
14/04/24 10:45:17 INFO mapred.JobClient:   File Output Format Counters
14/04/24 10:45:17 INFO mapred.JobClient:     Bytes Written=0
14/04/24 10:45:17 INFO mapred.JobClient:   FileSystemCounters
14/04/24 10:45:17 INFO mapred.JobClient:     FILE_BYTES_READ=320873
14/04/24 10:45:17 INFO mapred.JobClient:     HDFS_BYTES_READ=194818
14/04/24 10:45:17 INFO mapred.JobClient:     FILE_BYTES_WRITTEN=809644
14/04/24 10:45:17 INFO mapred.JobClient:     HDFS_BYTES_WRITTEN=287
14/04/24 10:45:17 INFO mapred.JobClient:   Extjoiner.ExitStatus
14/04/24 10:45:17 INFO mapred.JobClient:     reduce.00=134
14/04/24 10:45:17 INFO mapred.JobClient:   Map-Reduce Framework
14/04/24 10:45:17 INFO mapred.JobClient:     Map output materialized bytes=320604
14/04/24 10:45:17 INFO mapred.JobClient:     Map input records=10020
14/04/24 10:45:17 INFO mapred.JobClient:     Reduce shuffle bytes=320604
14/04/24 10:45:17 INFO mapred.JobClient:     Spilled Records=20040
14/04/24 10:45:17 INFO mapred.JobClient:     Map output bytes=300540
14/04/24 10:45:17 INFO mapred.JobClient:     Total committed heap usage (bytes)=826212352
14/04/24 10:45:17 INFO mapred.JobClient:     CPU time spent (ms)=6180
14/04/24 10:45:17 INFO mapred.JobClient:     Map input bytes=190320
14/04/24 10:45:17 INFO mapred.JobClient:     SPLIT_RAW_BYTES=1032
14/04/24 10:45:17 INFO mapred.JobClient:     Combine input records=0
14/04/24 10:45:17 INFO mapred.JobClient:     Reduce input records=10020
14/04/24 10:45:17 INFO mapred.JobClient:     Reduce input groups=40
14/04/24 10:45:17 INFO mapred.JobClient:     Combine output records=0
14/04/24 10:45:17 INFO mapred.JobClient:     Physical memory (bytes) snapshot=876621824
14/04/24 10:45:17 INFO mapred.JobClient:     Reduce output records=0
14/04/24 10:45:17 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=5048295424
14/04/24 10:45:17 INFO mapred.JobClient:     Map output records=10020
EX0003:Job execution failed.
```

A log shows that Reduce Task no. 00 has a return value of 134.

There is a high possibility that a Runtime error has occurred since 134 is a Runtime error return value (U error).

Check the Slave Server syslog.

```
# tail /var/log/messages
```

(Log excerpt)

```
Apr 24 10:45:17 hadoop2 : COBOL:rts64: HALT: JMP0015I-U [PID:000053FA TID:6BEC76E0] CANNOT CALL
PROGRAM 'ABC'. "dlopen-so=libABC.so: cannot open shared object file: No such file or directory dlsym-
out=./reduce.exe: undefined symbol: ABC" PGM=REDUCE. LINE=54
```

An error occurs at the execution time shown above.

Index