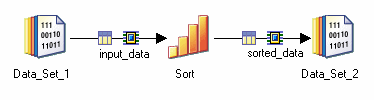
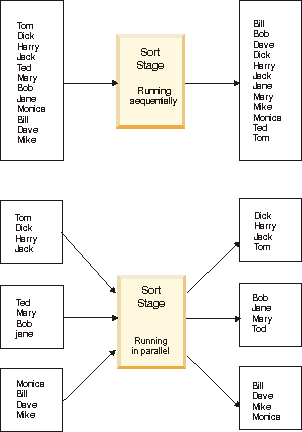
The Sort stage is a processing stage that is used to perform more complex sort operations than can be provided for on the Input page Partitioning tab of parallel job stage editors.

You can also use the Sort stage to insert a more explicit simple sort operation where you want to make your job easier to understand. The Sort stage has a single input link which carries the data to be sorted, and a single output link carrying the sorted data.



You specify sorting keys as the criteria on which to perform the sort. A key is a column on which to sort the data, for example, if you had a name column you might specify that as the sort key to produce an alphabetical list of names. The first column you specify as a key to the stage is the primary key, but you can specify additional secondary keys. If multiple rows have the same value for the primary key column, then InfoSphere® DataStage® uses the secondary columns to sort these rows.

You can sort in sequential mode to sort an entire data set or in parallel mode to sort data within partitions, as shown below:



The stage uses temporary disk space when performing a sort. It looks in the following locations, in the following order, for this temporary space.

1. Scratch disks in the disk pool sort (you can create these pools in the configuration file).
2. Scratch disks in the default disk pool (scratch disks are included here by default).
3. The directory specified by the TMPDIR environment variable.
4. The directory /*tmp*.

You might perform a sort for several reasons. For example, you might want to sort a data set by a zip code column, then by last name within the zip code. Once you have sorted the data set, you can filter the data set by comparing adjacent records and removing any duplicates.

However, you must be careful when processing a sorted data set: many types of processing, such as repartitioning, can destroy the sort order of the data. For example, assume you sort a data set on a system with four processing nodes and store the results to a data set stage. The data set will therefore have four partitions. You then use that data set as input to a stage executing on a different number of nodes, possibly due to node constraints. InfoSphere DataStage automatically repartitions a data set to spread out the data set to all nodes in the system, unless you tell it not to, possibly destroying the sort order of the data. You could avoid this by specifying the Same partitioning method. The stage does not perform any repartitioning as it reads the input data set; the original partitions are preserved.

You must also be careful when using a stage operating sequentially to process a sorted data set. A sequential stage executes on a single processing node to perform its action. Sequential stages will collect the data where the data set has more than one partition, which might also destroy the sorting order of its input data set. You can overcome this if you specify the collection method as follows:

* If the data was range partitioned before being sorted, you should use the ordered collection method to preserve the sort order of the data set. Using this collection method causes all the records from the first partition of a data set to be read first, then all records from the second partition, and so on.
* If the data was hash partitioned before being sorted, you should use the sort merge collection method specifying the same collection keys as the data was partitioned on.

**Note**If you write a sorted data set to an RDBMS there is no guarantee that it will be read back in the same order unless you specifically structure the SQL query to ensure this.

By default the stage will sort with the native InfoSphere DataStage sorter, but you can also specify that it uses the UNIX *sort*command.

The stage editor has three pages:

* [**Stage Page**](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/c_deeref_Stage_Page_sort_stage.html?view=kc). This is always present and is used to specify general information about the stage.
* [**Input Page**](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/c_deeref_Inputs_Page_sort_stage.html?view=kc). This is where you specify details about the data sets being sorted.
* [**Output Page**](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/c_deeref_Outputs_Page_sort_stage.html?view=kc). This is where you specify details about the sorted data being output from the stage.

This example shows the sequential sort of a list of GlobalCo customer, which is sorted by customer number and the sorted data is written into a single partition.

This job sorts the contents of a sequential file, and writes it to a data set. The data is a list of GlobalCo customers sorted by customer number. We are going to sort it by customer name instead.

Here is a sample of the input data:

"JON SMITH","789 LEDBURY ROAD","GC13849","GlobalCoUS"

"MARY GARDENER","127 BORDER ST","GC13933","GlobalCoUS"

"CHRIS TRAIN","1400 NEW ST","GC14036","GlobalCoUS"

"HUW WILLIAMS","579 DIGBETH AVENUE","GC14127","GlobalCoUS"

"SARA PEARS","45 ALCESTER WAY","GC14263","GlobalCoUS"

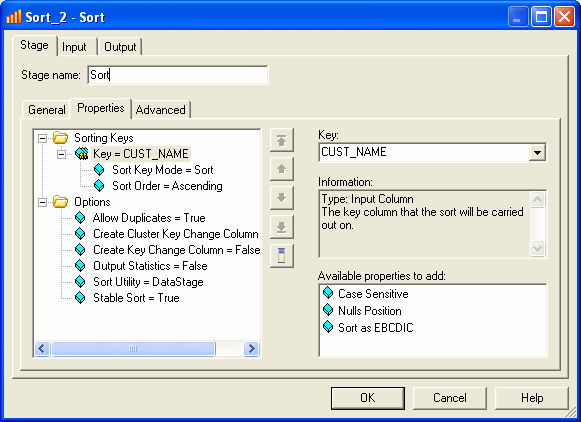
"LUC TEACHER","3 BIRMINGHAM ROAD","GC14346","GlobalCoUS"[Copy](javascript:void(0);)

The meta data for the file is as follows:

| **Column name** | **Key** | **SQL Type** | **Length** | **Nullable** |
| --- | --- | --- | --- | --- |
| CUST\_NAME |  | varchar | 30 | no |
| ADDR\_1 |  | varchar | 30 | no |
| CUSTOMER\_NUMBER | yes | char | 7 | no |
| SOURCE |  | char | 10 | no |
| *Table 1. Example metadata for writing to a sequential file* | | | | |

The Sequential File stage runs sequentially because it only has one source file to read. The Sort stage is set to run sequentially on the Stage page **Advanced** tab. The sort stage properties are used to specify the column CUST\_NAME as the primary sort key:

*Figure 1. Properties tab*



When the job is run the data is sorted into a single partition. The Data Set stage, GlobalCo\_sorted, is set to run sequentially to write the data to a single partition. Here is a sample of the sorted data:

"CHRIS TRAIN","1400 NEW ST","GC14036","GlobalCoUS"

"HUW WILLIAMS","579 DIGBETH AVENUE","GC14127","GlobalCoUS"

"JON SMITH","789 LEDBURY ROAD","GC13849","GlobalCoUS"

"LUC TEACHER","3 BIRMINGHAM ROAD","GC14346","GlobalCoUS"

"MARY GARDENER","127 BORDER ST","GC13933","GlobalCoUS"

"SARA PEARS","45 ALCESTER WAY","GC14263","GlobalCoUS"[Copy](javascript:void(0);)

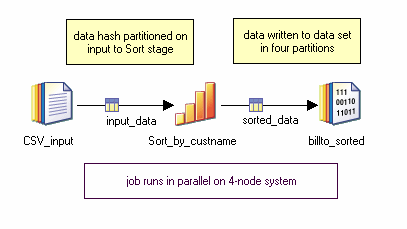
This example shows the Sort stage in parallel and creates a number of partitions.

This example uses the same job and the same data as the last previous example, but this time you are going to run the Sort stage in parallel and create a number of partitions.

In the Sort stage you specify parallel execution in the Stage page Advanced tab. In the Input page Partitioning tab you specify a partitioning type of Hash, and specify the column SOURCE as the hash key. Because the partitioning takes place on the input link, the data is partitioned before the sort stage actually tries to sort it. You hash partition to ensure that customers from the same country end up in the same partition. The data is then sorted within those partitions.

The job is run on a four-node system, so you end up with a data set comprising four partitions.

*Figure 1. Sort stage*



The following is a sample of the data in partition 2 after partitioning, but before sorting:

"JEAN DUPONT","576 RUE DE PARIS","GC20002","GlobalCoFR"

"MARIE TISON","14 AVENUE DE CALAIS","GC20012","GlobalCoFR"

"PIERRE FOURNIER","321 RUE VOLTAIRE","GC20021","GlobalCoFR"

"LOUIS LEROY","3 RUE DES TREUILS","GC20032","GlobalCoFR"

"NICOLE GIRARD","1234 QUAI DE LA TOURNELLE","GC20040","GlobalCoFR"

"DANIELLE BLANC","987 BOULEVARD AUXERRE","GC20049","GlobalCoFR"[Copy](javascript:void(0);)

And here is a sample of the data in partition 2 after it has been processed by the sort stage:

"DANIELLE BLANC","987 BOULEVARD AUXERRE","GC20049","GlobalCoFR"

"JEAN DUPONT","576 RUE DE PARIS","GC20002","GlobalCoFR"

"LOUIS LEROY","3 RUE DES TREUILS","GC20032","GlobalCoFR"

"MARIE TISON","14 AVENUE DE CALAIS","GC20012","GlobalCoFR"

"NICOLE GIRARD","1234 QUAI DE LA TOURNELLE","GC20040","GlobalCoFR"

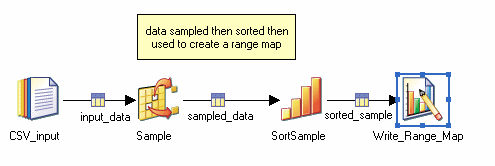
"PIERRE FOURNIER","321 RUE VOLTAIRE","GC20021","GlobalCoFR"[Copy](javascript:void(0);)

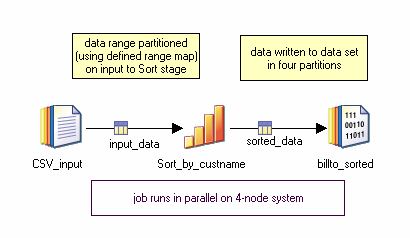
If you want to bring the data back together into a single partition, for example to write to another sequential file, you need to be mindful of how it is collected, or you will lose the benefit of the sort. If we use the sort/merge collection method, specifying the CUST\_NAME column as the collection key, you will end up with a totally sorted data set.

You can perform a total sort on a parallel data set, such that the data is ordered within each partition and the partitions themselves are ordered.

A total sort requires that all similar and duplicate records are located in the same partition of the data set. Similarity is based on the key fields in a record. The partitions also need to be approximately the same size so that no one node becomes a processing bottleneck.

In order to meet these two requirements, the input data is partitioned using the range partitioner. This guarantees that all records with the same key fields are in the same partition, and calculates the partition boundaries based on the key field to ensure fairly even distribution. In order to use the range partitioner you must first take a sample of your input data, sort it, then use it to build a range partition map as described in [Write Range Map stage](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/c_deeref_Write_Range_Map_Stage.html?view=kc), You then specify this map when setting up the range partitioner in the Input page Partitioning tab of your Sort stage.





When you run the job it will produce a totally sorted data set across the four partitions.

This section specifies the minimum steps to take to get a Sort stage functioning.

**About this task**

InfoSphere® DataStage® has many defaults which means that it can be very easy to include Sort stages in a job. InfoSphere DataStage provides a versatile user interface, and there are many shortcuts to achieving a particular end, this section describes the basic method, you will learn where the shortcuts are when you get familiar with the product.

To use a Sort stage:

* In the Stage page [**Properties Tab**](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Properties_Tab_sort_stage.html?view=kc), under the Sorting Keys category:
  + specify the key that you are sorting on. Repeat the property to specify a composite key.
* In the Output page [**Mapping Tab**](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Mapping_Tab_sort_stage.html?view=kc), specify how the output columns are derived.

You can specify aspects of the Sort stage from the Sort stage: Stage page.

The General tab allows you to specify an optional description of the stage. The Properties tab lets you specify what the stage does. The Advanced tab allows you to specify how the stage executes. The NLS Locale tab appears if your have NLS enabled on your system. It allows you to select a locale other than the project default to determine collating rules.

Use the Properties tab to specify how the Sort stage operates.

The Properties tab allows you to specify properties which determine what the stage actually does. Some of the properties are mandatory, although many have default settings. Properties without default settings appear in the warning color (red by default) and turn black when you supply a value for them.

The following table gives a quick reference list of the properties and their attributes. A more detailed description of each property follows.

| **Category/Property** | **Values** | **Default** | **Mandatory?** | **Repeats?** | **Dependent of** |
| --- | --- | --- | --- | --- | --- |
| Sorting Keys/[Key](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Sorting_Keys_Category_sort_stage.html?view=kc) | Input Column | N/A | Y | Y | N/A |
| Sorting Keys/[Sort Order](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Sorting_Keys_Category_sort_stage.html?view=kc) | Ascending/Descending | Ascending | Y | N | Key |
| Sorting Keys/[Nulls position](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Sorting_Keys_Category_sort_stage.html?view=kc) (only available for Sort Utility = DataStage®) | First/Last | First | N | N | Key |
| Sorting Keys/[Sort as EBCDIC](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Sorting_Keys_Category_sort_stage.html?view=kc) | True/False | False | N | N | Key |
| Sorting Keys/[Case Sensitive](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Sorting_Keys_Category_sort_stage.html?view=kc) | True/False | True | N | N | Key |
| Sorting Keys/[Sort Key Mode](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Sorting_Keys_Category_sort_stage.html?view=kc) (only available for Sort Utility = DataStage) | Sort/Don't Sort (Previously Grouped)/Don't Sort (Previously Sorted) | Sort | Y | N | Key |
| Options/[Sort Utility](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Options_Category_sort_stage.html?view=kc) | DataStage/ UNIX | DataStage | Y | N | N/A |
| Options/[Stable Sort](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Options_Category_sort_stage.html?view=kc) | True/False | True for Sort Utility = DataStage, False otherwise | Y | N | N/A |
| Options/[Allow Duplicates](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Options_Category_sort_stage.html?view=kc) (not available for Sort Utility = UNIX) | True/False | True | Y | N | N/A |
| Options/[Output Statistics](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Options_Category_sort_stage.html?view=kc) | True/False | False | Y | N | N/A |
| Options/[Create Cluster Key Change Column](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Options_Category_sort_stage.html?view=kc) (only available for Sort Utility = DataStage) | True/False | False | N | N | N/A |
| Options/[Create Key Change Column](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Options_Category_sort_stage.html?view=kc) | True/False | False | N | N | N/A |
| Options/[Restrict Memory Usage](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Options_Category_sort_stage.html?view=kc) | number MB | 20 | N | N | N/A |
| Options/[Workspace](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/r_deeref_Options_Category_sort_stage.html?view=kc) | string | N/A | N | N | N/A |
| *Table 1. Properties* | | | | | |

Use the Sorting keys category to specify how the Sort stage operates.

**Key**

Specifies the key column for sorting. This property can be repeated to specify multiple key columns. You can use the Column Selection dialog box to select several keys at once if required. Key has dependent properties depending on the Sort Utility chosen:

* **Sort Order**

All sort types. Choose Ascending or Descending. The default is Ascending.

* **Nulls position**

This property appears for sort type DataStage® and is optional. By default columns containing null values appear first in the sorted data set. To override this default so that columns containing null values appear last in the sorted data set, select Last.

* **Sort as EBCDIC**

To sort as in the EBCDIC character set, choose True.

* **Case Sensitive**

All sort types. This property is optional. Use this to specify whether each group key is case sensitive or not, this is set to True by default, that is, the values "CASE" and "case" would not be judged equivalent.

* **Sort Key Mode**

This property appears for sort type DataStage. It is set to Sort by default and this sorts on all the specified key columns.

Set to Don't Sort (Previously Sorted) to specify that input records are already sorted by this column. The Sort stage will then sort on secondary key columns, if any. This option can increase the speed of the sort and reduce the amount of temporary disk space when your records are already sorted by the primary key column(s) because you only need to sort your data on the secondary key column(s).

Set to Don't Sort (Previously Grouped) to specify that input records are already grouped by this column, but not sorted. The operator will then sort on any secondary key columns. This option is useful when your records are already grouped by the primary key column(s), but not necessarily sorted, and you want to sort your data only on the secondary key column(s) within each group

Use the Options category to specify how the Sort stage operates.

**Sort utility**

The type of sort the stage will carry out. Choose from:

* **DataStage®**. The default. This uses the built-in InfoSphere® DataStage sorter, you do not require any additional software to use this option.
* **UNIX**. This specifies that the UNIX *sort* command is used to perform the sort.

**Stable sort**

Applies to a Sort Utility type of DataStage, the default is True. It is set to True to guarantee that this sort operation will not rearrange records that are already in a properly sorted data set. If set to False no prior ordering of records is guaranteed to be preserved by the sorting operation.

**Allow duplicates**

Set to True by default. If False, specifies that, if multiple records have identical sorting key values, only one record is retained. If Stable Sort is True, then the first record is retained. This property is not available for the UNIX sort type.

**Output statistics**

Set False by default. If True it causes the sort operation to output statistics. This property is not available for the UNIX sort type.

**Create cluster key change column**

This property appears for sort type DataStage and is optional. It is set False by default. If set True it tells the Sort stage to create the column **clusterKeyChange** in each output record. The **clusterKeyChange** column is set to 1 for the first record in each group where groups are defined by using a Sort Key Mode of Don't Sort (Previously Sorted) or Don't Sort (Previously Grouped). Subsequent records in the group have the **clusterKeyChange** column set to 0.

**Create key change column**

This property appears for sort type DataStage and is optional. It is set False by default. If set True it tells the Sort stage to create the column **KeyChange** in each output record. The **KeyChange** column is set to 1 for the first record in each group where the value of the sort key changes. Subsequent records in the group have the **KeyChange** column set to 0.

**Restrict memory usage**

This is set to 20 by default. It causes the Sort stage to restrict itself to the specified number of megabytes of virtual memory on a processing node.

The number of megabytes specified should be smaller than the amount of physical memory on a processing node. For Windows systems, the value for **Restrict Memory Usage** should not exceed 500.

**Workspace**

This property appears for sort type UNIX only. Optionally specifies the workspace used by the stage.

This tab allows you to specify options.

This tab allows you to specify the following:

* **Execution Mode**. The stage can execute in parallel mode or sequential mode. In parallel mode the input data is processed by the available nodes as specified in the Configuration file, and by any node constraints specified on the **Advanced** tab. In Sequential mode the entire data set is processed by the conductor node.
* **Preserve partitioning**. This is **Set** by default. You can explicitly select **Set** or **Clear**. Select **Set** to request the next stage in the job should attempt to maintain the partitioning.
* **Combinability mode**. This is Auto by default, which allows InfoSphere® DataStage® to combine the operators that underlie parallel stages so that they run in the same process if it is sensible for this type of stage.
* **Node pool and resource constraints**. Select this option to constrain parallel execution to the node pool or pools or resource pool or pools specified in the grid. The grid allows you to make choices from drop down lists populated from the Configuration file.
* **Node map constraint**. Select this option to constrain parallel execution to the nodes in a defined node map. You can define a node map by typing node numbers into the text box or by clicking the browse button to open the **Available Nodes** dialog box and selecting nodes from there. You are effectively defining a new node pool for this stage (in addition to any node pools defined in the Configuration file).

**Note**In the **Node map constraint** text box, you can enter jobs parameters as well as numbers. You can enter a single parameter, for example #testnode#, or you can enter a comma separated lists of parameters, for example #testnode#, #testnode2#. The browse button next to the text box will display a list of the node names from the last configuration file that was referenced by the job, but the browse button will not display the node names that were specified by the job parameters.

For the Sort stage, the NLS Locale tab appears if you have NLS enabled on your system.

If you are using the DataStage® sort type, it lets you view the current default collate convention, and select a different one for this stage if required (for UNIX sorts, it is blank). You can also use a job parameter to specify the locale, or browse for a file that defines custom collate rules. The collate convention defines the order in which characters are collated. The Sort stage uses this when it is determining the order of the sorted fields. Select a locale from the list, or click the arrow button next to the list to use a job parameter or browse for a collate file.

The input page allows you to specify details about the data coming in to be sorted. The Sort stage can have only one input link.

The input page allows you to specify details about the data coming in to be sorted. The Sort stage can have only one input link.

The General tab allows you to specify an optional description of the link. The Partitioning tab allows you to specify how incoming data on the source data set link is partitioned. The Columns tab specifies the column definitions of incoming data. The Advanced tab allows you to change the default buffering settings for the input link.

Details about Sort stage partitioning are given in the following section. See ["Stage Editors,"](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/c_deeref_Stage_Editors.html?view=kc) for a general description of the other tabs.

The Partitioning tab allows you to specify details about how the incoming data is partitioned or collected before the sort is performed.

By default the stage uses the auto partitioning method. If the Preserve Partitioning option has been set on the previous stage in the job the stage will warn you when the job runs if it cannot preserve the partitioning of the incoming data.

If the Sort Set stage is operating in sequential mode, it will first collect the data before writing it to the file using the default auto collection method.

The Partitioning tab allows you to override this default behavior. The exact operation of this tab depends on:

* Whether the Sort stage is set to execute in parallel or sequential mode.
* Whether the preceding stage in the job is set to execute in parallel or sequential mode.

If the Sort stage is set to execute in parallel, then you can set a partitioning method by selecting from the **Partition type** drop-down list. This will override any current partitioning.

If the Sort stage is set to execute in sequential mode, but the preceding stage is executing in parallel, then you can set a collection method from the **Collector type** drop-down list. This will override the default auto collection method.

The following partitioning methods are available:

* **(Auto)**. InfoSphere® DataStage® attempts to work out the best partitioning method depending on execution modes of current and preceding stages and how many nodes are specified in the Configuration file. This is the default method for the Sort stage.
* **Entire**. Each file written to receives the entire data set.
* **Hash**. The records are hashed into partitions based on the value of a key column or columns selected from the **Available**list.
* **Modulus**. The records are partitioned using a modulus function on the key column selected from the **Available** list. This is commonly used to partition on tag fields.
* **Random**. The records are partitioned randomly, based on the output of a random number generator.
* **Round Robin**. The records are partitioned on a round robin basis as they enter the stage.
* **Same**. Preserves the partitioning already in place.
* **Db2®**. Replicates the Db2 partitioning method of a specific Db2 table. Requires extra properties to be set. Access these properties by clicking the properties button.
* **Range**. Divides a data set into approximately equal size partitions based on one or more partitioning keys. Range partitioning is often a preprocessing step to performing a total sort on a data set. Requires extra properties to be set. Access these properties by clicking the properties button.

The following Collection methods are available:

* **(Auto)**. This is the default collection method for the Sort stage. Normally, when you are using Auto mode, InfoSphere DataStage will eagerly read any row from any input partition as it becomes available.
* **Ordered**. Reads all records from the first partition, then all records from the second partition, and so on.
* **Round Robin**. Reads a record from the first input partition, then from the second partition, and so on. After reaching the last partition, the operator starts over.
* **Sort Merge**. Reads records in an order based on one or more columns of the record. This requires you to select a collecting key column from the **Available** list.

The Partitioning tab also allows you to specify that data arriving on the input link should be sorted before the Sort is performed. This is a standard feature of the stage editors, if you make use of it you will be running a simple sort before the main Sort operation that the stage provides. The sort is always carried out within data partitions. If the stage is partitioning incoming data the sort occurs after the partitioning. If the stage is collecting data, the sort occurs before the collection. The availability of sorting depends on the partitioning or collecting method chosen (it is not available with the default auto methods).

Select the check boxes as follows:

* **Perform Sort**. Select this to specify that data coming in on the link should be sorted. Select the column or columns to sort on from the **Available** list.
* **Stable**. Select this if you want to preserve previously sorted data sets. This is the default.
* **Unique**. Select this to specify that, if multiple records have identical sorting key values, only one record is retained. If stable sort is also set, the first record is retained.

If NLS is enabled an additional button opens a dialog box allowing you to select a locale specifying the collate convention for the sort.

You can also specify sort direction, case sensitivity, whether sorted as ASCII or EBCDIC, and whether null columns will appear first or last for each column. Where you are using a keyed partitioning method, you can also specify whether the column is used as a key for sorting, for partitioning, or for both. Select the column in the **Selected** list and right-click to invoke the shortcut menu.

The Output page allows you to specify details about data output from the Sort stage.

The Sort stage can have only one output link.

The General tab allows you to specify an optional description of the output link. The Columns tab specifies the column definitions of the data. The Mapping tab allows you to specify the relationship between the columns being input to the Sort stage and the Output columns. The Advanced tab allows you to change the default buffering settings for the output link.

Details about Sort stage mapping is given in the following section. See ["Stage Editors,"](https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.ds.parjob.dev.doc/topics/c_deeref_Stage_Editors.html?view=kc) for a general description of the other tabs.

For the Sort stage, the Mapping tab allows you to specify how the output columns are derived, that is, what input columns map onto them.

The left pane shows the columns of the sorted data. These are read only and cannot be modified on this tab. This shows the meta data from the input link.

The right pane shows the output columns for the output link. This has a **Derivations** field where you can specify how the column is derived. You can fill it in by dragging input columns over, or by using the Auto-match facility.

In the above example, the left pane represents the incoming data after the sort has been performed. The right pane represents the data being output by the stage after the sort operation. In this example the data has been mapped straight across.