


Unit 7 Integrating data

IBM Training

IBM



Integrating data

IBM SPSS Modeler (v18)

© Copyright IBM Corporation 2016
Course materials may not be reproduced in whole or in part without the written permission of IBM.

Unit objectives

- Integrate data by appending records from multiple datasets
- Integrate data by merging fields from multiple datasets
- Sample records

Integrating data

© Copyright IBM Corporation 2016

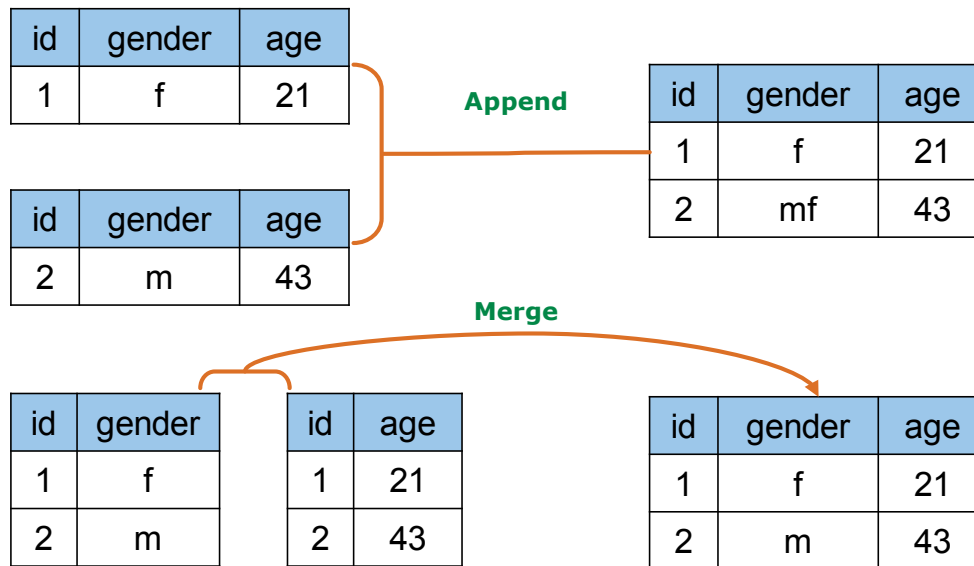
Unit objectives

Similar pieces of information may be stored in different datasets. These datasets must be combined into a single dataset for analyses. Typically, datasets are combined after the unit of analysis is set correctly for each of the datasets involved.

Combining datasets is referred to in the CRISP-DM process model as integrating data. Before reviewing this unit you should be familiar with:

- CRISP-DM
- IBM SPSS Modeler streams, nodes and palettes
- methods to collect initial data
- methods to explore the data
- methods to set the unit of analysis

Methods to integrate data



Integrating data

© Copyright IBM Corporation 2016

Methods to integrate data

There are two ways to combine datasets:

- Append: Add records from one dataset to another
- Merge: Add fields from one dataset to another

This unit presents these two methods of combining datasets.

Append records

- Similar pieces of information for different groups of records are stored in different datasets.
- For analysis and modeling, create a single dataset.
- Use the Append node (Record Ops).



Integrating data

© Copyright IBM Corporation 2016

Append records

Similar pieces of information for different groups of records may be stored in different datasets. Examples of this include:

- fraud information for different local offices
- bank account information for different financial years
- examination results for different academic years
- transaction data for different weeks

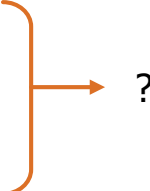
There is often a need to collectively analyze such data, possibly to compare performance over subsequent years or to discover group differences. To analyze such information the datasets must be combined into a single dataset.

Use the Append node (located in the Record Ops palette) to append datasets.

Options to append records

id	gender	age
1	f	21

id	gender	mar- ried
2	m	no



- Options:
 - Main dataset: one dataset is leading
 - All datasets: equal role of the datasets

Integrating data

© Copyright IBM Corporation 2016

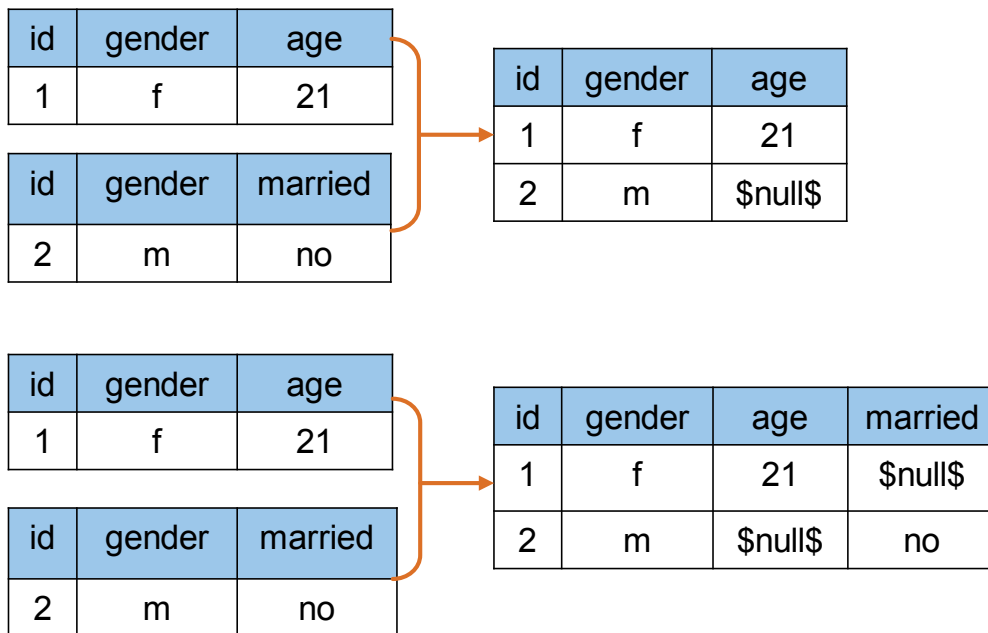
Options to append records

In appending records from one dataset to another, various options are available for the situation where not all fields are the same.

Two choices can be made when appending records from datasets: to treat one of the datasets as the main dataset or to let the datasets play an equal role.

Using one of the datasets as the main dataset means that only the fields present in the main dataset will be output to the new dataset. Fields only present in the secondary dataset are lost.

When the datasets play an equal role, all fields from all datasets will be output to the new dataset.

Append records illustrated

Integrating data

© Copyright IBM Corporation 2016

Append records illustrated

The first figure on this slide gives an example, with the upper dataset as the main dataset. Only the fields id, gender, and age are output and the married field is lost.

The second figure illustrates the situation where the datasets play an equal role.

When you append records from different datasets, some records may have undefined values, regardless of the method used. For example, in the first example id 2 will have an undefined value for age, because age was not observed for this person. Likewise, id 1 has an undefined value for married and id 2 has an undefined value for age in the second example.

Merge fields

- Different pieces of information for the same records are stored in different datasets.
- Create a single dataset for analyses.
- Use the Merge node (Record Ops).



Integrating data

© Copyright IBM Corporation 2016

Merge fields

In many organizations, different pieces of information for individuals are held in separate locations. Examples of this include:

- customer information that is held separately from purchase information
- account details that is held in a database separate from transactions
- a housing organization may hold information at an individual and property level

To analyze such data, for example to find patterns in purchase behavior and link them back to demographics, the individual datasets have to be combined into one single dataset. The way to join these datasets is not by adding records from one dataset to another, but by adding fields.

You can use the Merge node, located in the Record Ops palette) to combine fields from different datasets into a single dataset.

Options to merge fields

id	gender			id	age		
1	f	+		2	25	= ?	
2	m			3	43		

- Options:
 - Inner join
 - Full outer join
 - Partial outer join
 - Anti-join

Integrating data

© Copyright IBM Corporation 2016

Options to merge fields

When datasets do not have the same records, three choices can be made for the merge:

1. Only matching records are passed through the Merge node, known as an inner join.
2. Records from all datasets are passed, known as an outer join.
3. One dataset is leading in the merge and enriched with fields from other datasets, but no records are added from other datasets. This is known as a partial outer join.

There is also an option to retain only the records that are unique to a specific dataset. This is known as an anti join.

Merge fields illustrated: Inner and outer join

id	gender
1	f
2	m

+

id	age
2	25
3	43

=

id	gender	age
2	m	25

id	gender
1	f
2	m

+

id	age
2	25
3	43

=

id	gender	age
1	f	\$null\$
2	m	25
3	\$null\$	43

Integrating data

© Copyright IBM Corporation 2016

Merge fields illustrated: Inner and outer join

In an inner join (the upper figure) only data from records present in all source datasets will be merged. In this example only id 2 will be output.

In an outer join (the lower figure) every record from each dataset is passed through the Merge node. The undefined (\$null\$) value is added to the missing fields. In this example id 1 will have the undefined value for the age field, and id 3 will have the undefined value for gender.

Merge fields illustrated: Partial join and anti join

id	gender		id	age	=	id	gender	age
1	f		2	25		1	f	\$null\$
2	m	+	3	43		2	m	25

id	gender		id	age	=	id	gender
1	f		2	25		1	f
2	m	+	3	43			

Integrating data

© Copyright IBM Corporation 2016

Merge fields illustrated: Partial join and anti join

In a partial outer join (the upper figure) one of the datasets is leading in the merge, in the sense that all records from the leading dataset will be in the output dataset and only information from matching records in the secondary datasets will be added. In this example, taking the left dataset as the leading dataset, id's 1 and 2 will be output, with id 1 having the undefined (\$null\$) values for age.

An anti join (the lower figure) takes one dataset as leading and retains only records from that dataset that are not present in the secondary datasets. No fields will be added. In this example, with the left dataset as the leading dataset, this will only keep id 1.

A many-to-one merge

id	age	zip code		zip code	% with car		id	age	zip code	% with car
1	43	105	+	105	25	=	1	43	105	25
2	22	105		481	43		2	22	105	25
3	51	481		585	12		3	51	481	43
4	63	481					4	63	481	43

Integrating data

© Copyright IBM Corporation 2016

A many-to-one merge

In the previous examples the situation was one of a 1-to-1 match: one record in one dataset matches one (or no) record in the other dataset.

The Merge node accommodates as well a 1-to-many match. This slide shows an example, which is frequently encountered in database marketing. A customer dataset can be enriched with aggregated data by using the zipcode as key field for the merge. Typically, a partial outer join will be used in this situation, with the customer dataset leading.

Sample records

- Processing big amounts of data can be inefficient in terms of time and memory.
- Sample the records and if the stream does the job, use all records.
- Use the Sample node (Record Ops).



Sample records

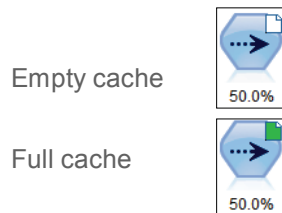
It is not uncommon to have hundreds of thousands, if not millions, of records available in one or more datasets. When appending or merging this size of datasets, data processing can take a huge amount of time.

In general, using all the records can be quite inefficient in terms of processing time and memory. To build and test the stream it is recommended to first sample the records and if the stream does the job as required, have a final test with all records.

To sample records, use the Sample node (located in the Record Ops palette).

Cache data

- IBM SPSS Modeler processes data starting at the source nodes.
- With huge amounts of data, cache the data.
 - Right-click the node, then select Cache\Enable.
- Cached nodes:



Integrating data

© Copyright IBM Corporation 2016

Cache data

When you sample records, the Sample node pulls data from the data source. This process repeats every time that the stream is run. This means that each time data must be read and records must be sampled (and in case of a fixed seed value, the same records will be sampled). This is highly inefficient in the case of huge amounts of data. Therefore, IBM SPSS Modeler provides the option to cache the data. Caching creates a temporary file behind the scenes, storing the data.

When a node carries a cache (enabled by selecting Cache\Enable from the context menu on a node), a document icon is displayed at the top right corner. At first, the icon will be empty, but when data flow through the cached node, the cache fills and the icon turns green. If changes are made upstream from a cached node, the cache will become emptied, as its contents will no longer be valid. It will refill when data are passed through the node.

Data can be cached at any non-terminal node. A data cache is also useful at time-consuming nodes such as a Sort, Append, Merge, or Aggregate. Caching on or after these nodes can substantially improve performance.