Advanced Data Preparation Using IBM SPSS Modeler (v16)

Student Guide

Course Code: 0A055

Advanced Data Preparation Using IBM SPSS Modeler (v16)
0A055
ERC: 1.0
Published June 2014

This material is meant for IBM Academic Initiative purposes.

Contents

Course Overview

Course Overview

Advanced Data Preparation Using IBM SPSS MODELER (v16) is a one day course that covers advanced topics to aid in the preparation of data for a successful data mining project. You will learn how to use functions, deal with missing values, use advanced field operations, handle sequence data, apply advanced sampling methods, and improve efficiency.

Intended Audience

IBM SPSS Modeler Analysts and IBM SPSS Modeler Data Experts who want to become familiar with the full range of techniques available in IBM SPSS Modeler for data manipulation.

Topics Covered

- Using Functions
- Data Transformations
- Working with Sequence Data
- Sampling Records
- Improving Efficiency

Course Prerequisites

Participants should have:

- General computer literacy
- Some experience using IBM SPSS Modeler including familiarity with the Modeler environment, creating streams, reading data files, and doing simple data exploration and manipulation using the Derive node.
- Prior completion of Introduction to IBM SPSS Modeler and Data Mining (v16) is recommended.

Document Conventions

Conventions used in this guide follow Microsoft Windows application standards, where applicable. As well, the following conventions are observed:

Bold

Bold style is used in demo and workshop step-by-step solutions to indicate either:

- actionable items

(Point to **Sort**, and then click **Ascending**.)

- text to type or keys to press

(Type **Sales Report**, and then press **Enter**.)

- UI elements that are the focus of attention

(In the **Format** pane, click **Data**)

Italic

Used to reference book titles.

CAPITALIZATION

All file names, table names, column names, and folder names appear in this guide exactly as they appear in the application.

To keep capitalization consistent with this guide, type text exactly as shown.

MODELER

Used to reference IBM SPSS Modeler version 16.

Workshops

Workshop Format

Workshops are designed to allow you to work according to your own pace. Content contained in a workshop is not fully scripted out to provide an additional challenge. Refer back to demonstrations if you need assistance with a particular task. The workshops are structured as follows:

The Business Question Section

This section presents a business-type question followed by a series of tasks. These tasks provide additional information to help guide you through the workshop. Within each task, there may be numbered questions relating to the task. Complete the tasks by using the skills you learned in the module. If you need more assistance, you can refer to the Task and Results section for more detailed instruction.

The Task and Results Section

This section provides a task based set of instructions that presents the question as a series of numbered tasks to be accomplished. The information in the tasks expands on the business case, providing more details on how to accomplish a task. Screen captures are also provided at the end of some tasks and at the end of the workshop to show the expected results.

Additional Training Resources

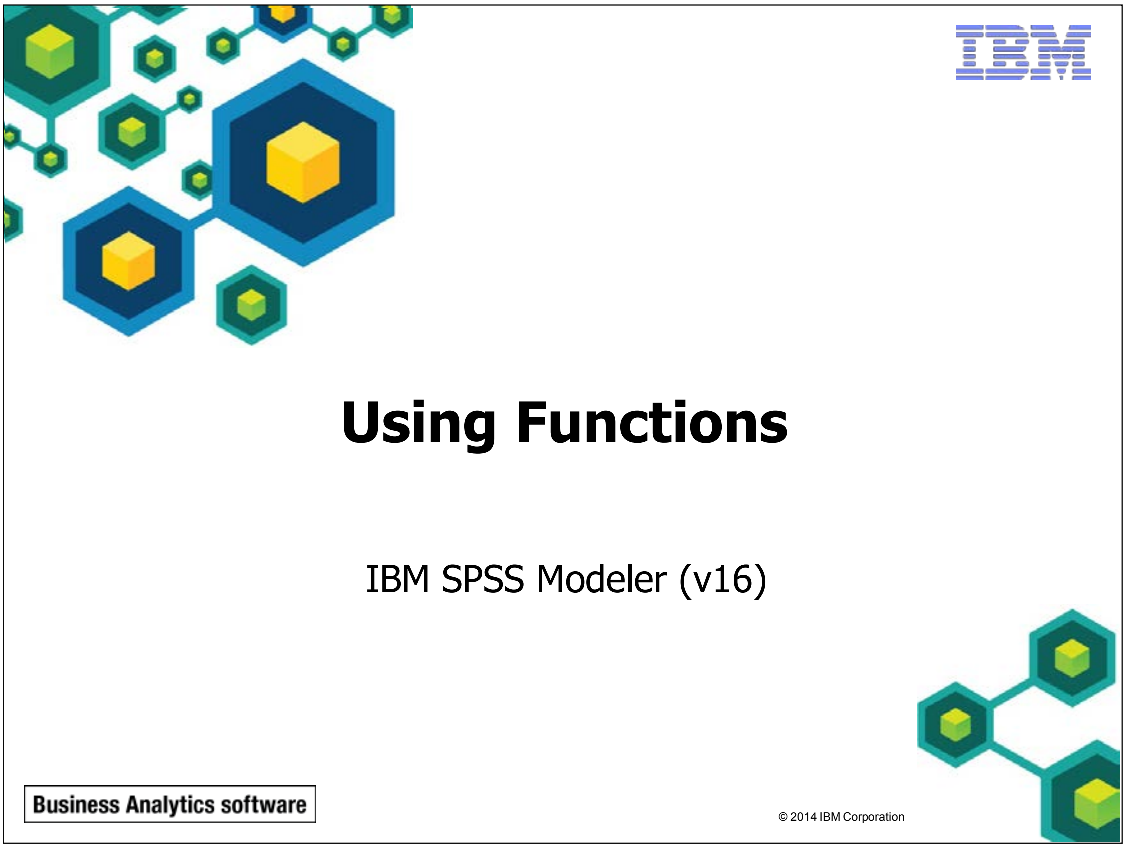
Bookmark Business Analytics Product Training http://www-01.ibm.com/software/analytics/training-and-certification/ for details on:

- instructor-led training in a classroom or online
- self-paced training that fits your needs and schedule
- comprehensive curricula and training paths that help you identify the courses that are right for you
- IBM Business Analytics Certification program
- other resources that will enhance your success with IBM Business Analytics Software

IBM Product Help

Help type	When to use	Location
Task-oriented	You are working in the product and you need specific task-oriented help.	*IBM Product* - Help link
Books for Printing (.pdf)	You want to use search engines to find information. You can then print out selected pages, a section, or the whole book. Use Step-by-Step online books (.pdf) if you want to know how to complete a task but prefer to read about it in a book. The Step-by-Step online books contain the same information as the online help, but the method of presentation is different.	Start/Programs/*IBM Product*/Documentation
IBM on the Web	You want to access any of the following:	
	• Training and Certification Web site	• http://www-01.ibm.com/software/analytics/training-and-certification/
	• Online support	• http://www-947.ibm.com/support/entry/portal/Overview/Software
	• IBM Web site	• http://www.ibm.com

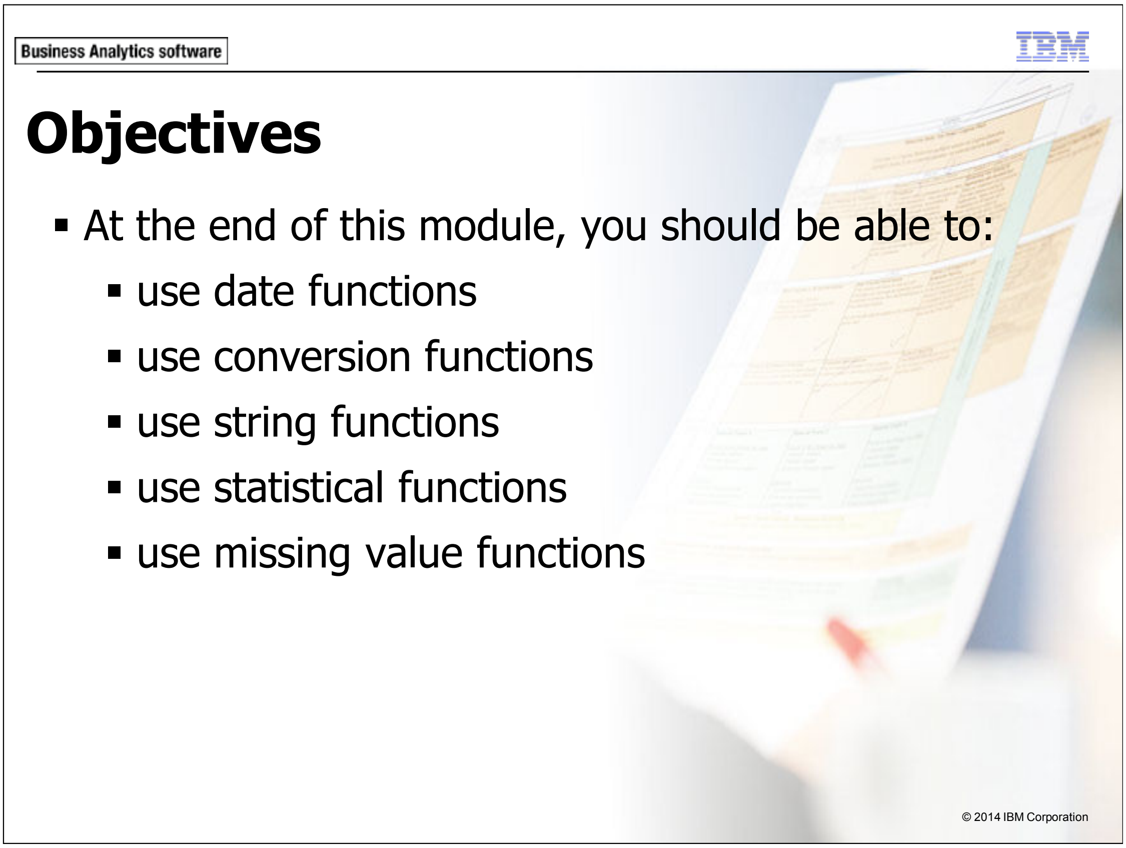
This material is meant for IBM Academic Initiative purposes.



Using Functions

IBM SPSS Modeler (v16)

Business Analytics software



Business Analytics software

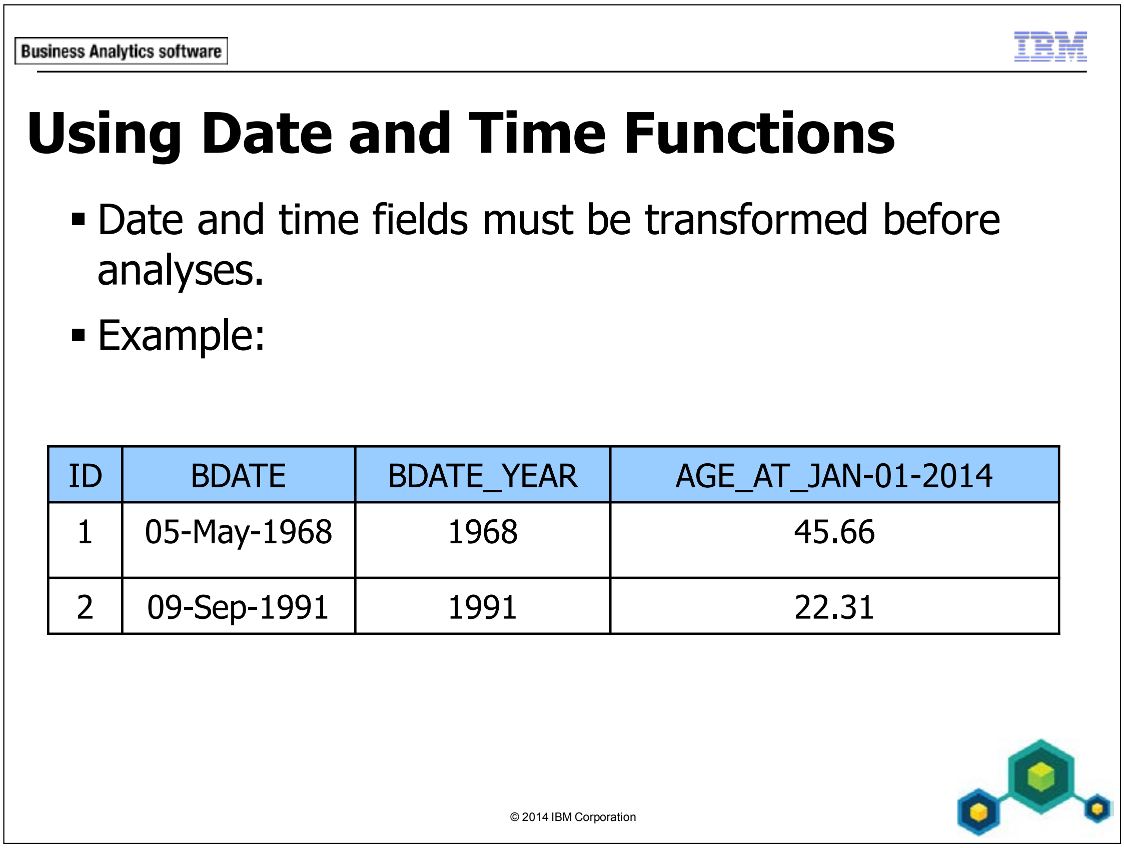
IBM

Objectives

- At the end of this module, you should be able to:
  - use date functions
  - use conversion functions
  - use string functions
  - use statistical functions
  - use missing value functions

Before reviewing this module you should be familiar with:

- working with MODELER (streams, nodes, SuperNodes, palettes)
- importing data (Var. File node)
- defining measurement levels and roles, blanks, and instantiating data (Type node)
- selecting fields and records (Filter node, Select node)
- CLEM and the Expression Builder
- deriving fields (Derive node)
- examining the data with graphs (Distribution node, Histogram node) and tables (Table node, Data Audit node, Statistics node, Matrix node, Means node)



Business Analytics software

Using Date and Time Functions

- Date and time fields must be transformed before analyses.
- Example:

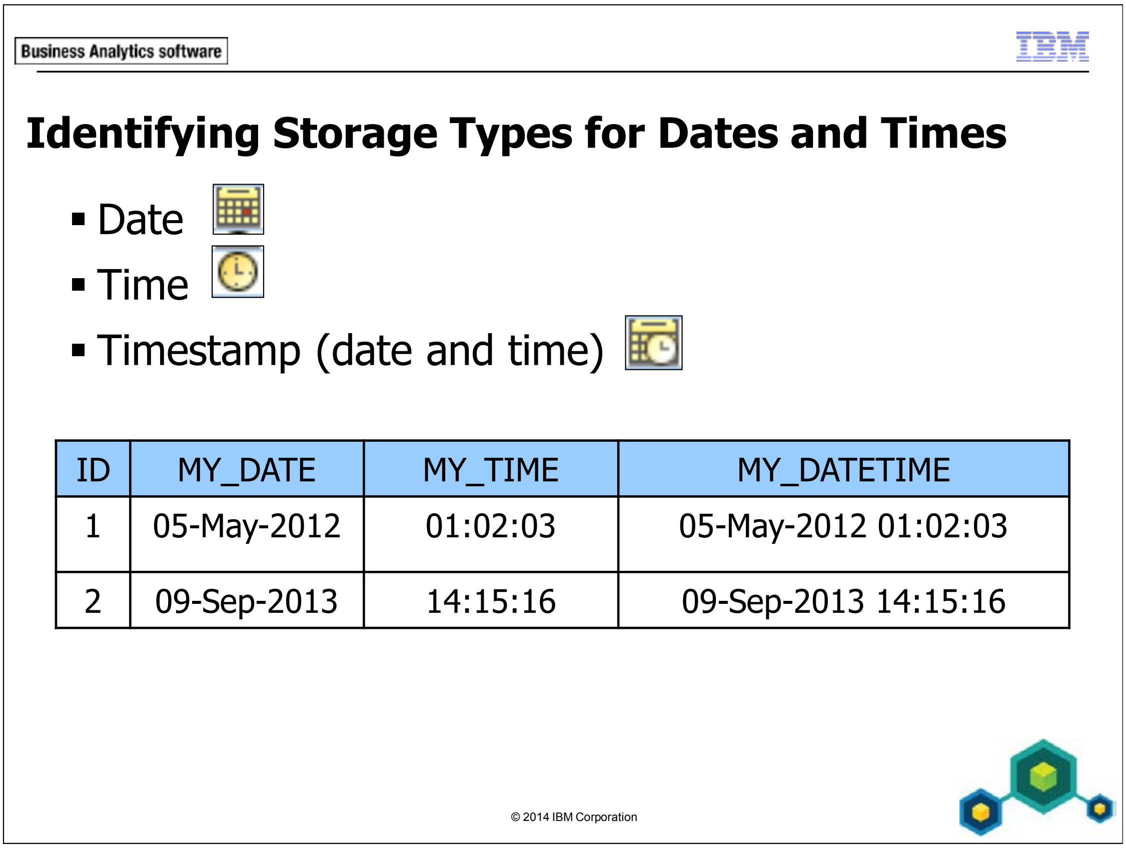
ID	BDATE	BDATE_YEAR	AGE_AT_JAN-01-2014
1	05-May-1968	1968	45.66
2	09-Sep-1991	1991	22.31

Manipulations involving date and time fields are a commonplace task with data.

The date and time functions in MODELER address issues such as:

- extracting a part from a date or time field such as the year out of a date
- calculating age from date of birth
- calculating the length of time that someone has been a customer
- creating a date field from information contained in separate fields

This slide shows an example, with two fields that are derived from BDATE. The first field gives the year of birth, the second field AGE at Jan 1st, 2014.



Business Analytics software

Identifying Storage Types for Dates and Times

- Date
- Time
- Timestamp (date and time)

ID	MY_DATE	MY_TIME	MY_DATETIME
1	05-May-2012	01:02:03	05-May-2012 01:02:03
2	09-Sep-2013	14:15:16	09-Sep-2013 14:15:16

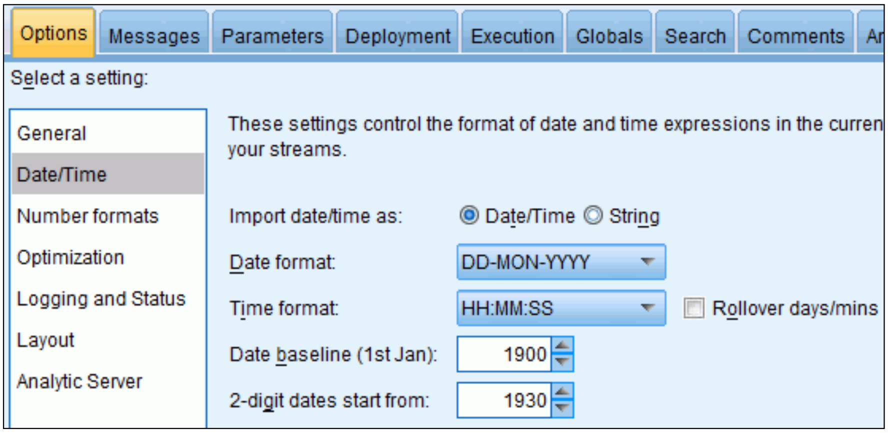
Dates and times are numeric fields with special storage types to display the information they contain. MODELER distinguishes between:

- Date: A date such as 05-May-2012.
- Time: A time such as 01:02:03.
- Timestamp: A combination of date and time.

Business Analytics software

Setting Options for Dates and Times

- Defaults for display, defaults for calculations in **Tools\Stream Properties\Options**



These settings control the format of date and time expressions in the curren your streams.

Import date/time as: ● Date/Time ○ String

Date format: DD-MON-YYYY

Time format: HH:MM:SS □ Rollover days/mins

Date baseline (1st Jan): 1900

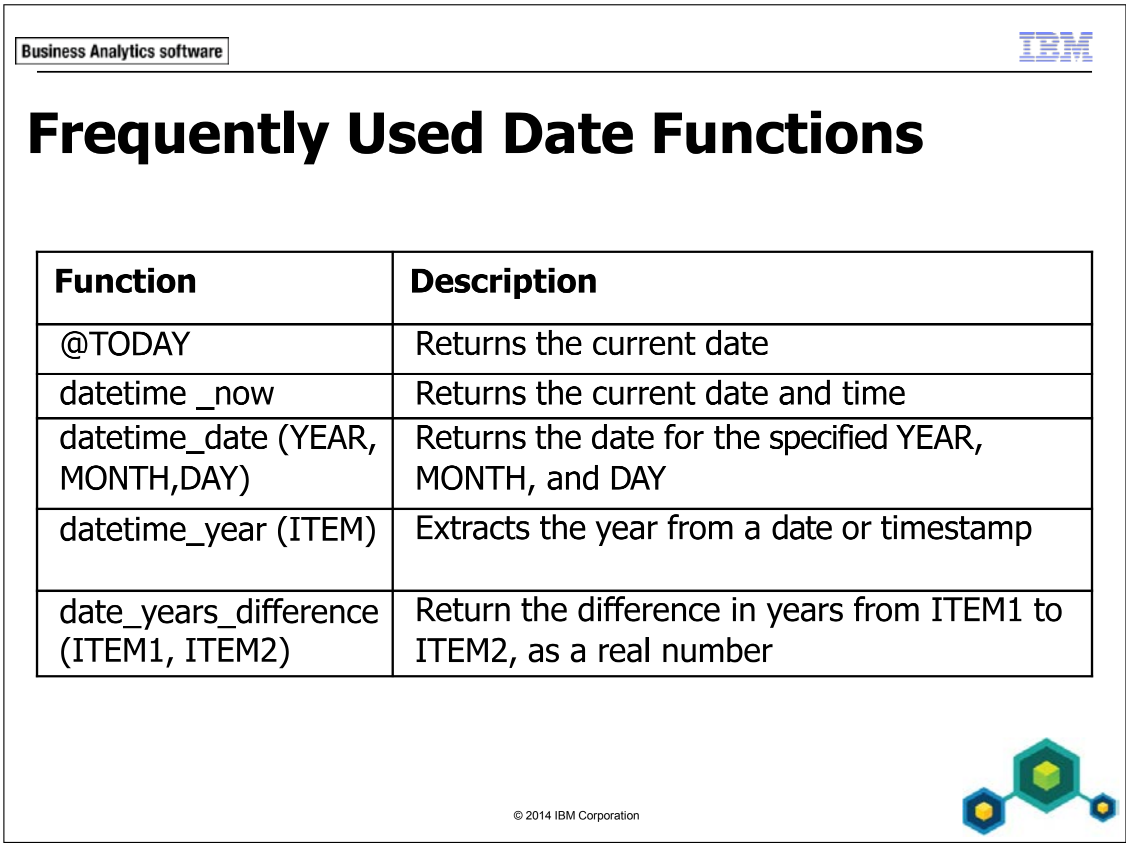
2-digit dates start from: 1930

© 2014 IBM Corporation

How dates and times are displayed depends on MODELER's settings. This can be a different format from what it is in the data source. For example, a date field can be stored in a text file in the DD/MM/YYYY format, but once the field is imported in MODELER it will be displayed using the format defined on the Options tab in the Stream Properties dialog box.

For computations involving dates, the Date baseline option sets the baseline year (the baseline month and day is always January 1st). For example, MODELER’s date_in_years function returns the time in years from the baseline date, so date_in_years (DOB) returns 78 for DOB 01-JAN-1978 if the baseline date is 1900.

The 2-digits dates start from option sets the cutoff year to add century digits for years denoted with 2 digits. If 1930 is set as the cutoff year, a 2 digit year greater than or equal to 30 will be assumed to be in the 20th century. For example a 2 digit date such as 01-JAN-78 will be set to January 1st of the year 1978. The same setting will use the 21th century for years less than 30; thus 01-JAN-14 evaluates to January 1st 2014.



Business Analytics software

Frequently Used Date Functions

Function	Description
@TODAY	Returns the current date
datetime _now	Returns the current date and time
datetime_date (YEAR, MONTH,DAY)	Returns the date for the specified YEAR, MONTH, and DAY
datetime_year (ITEM)	Extracts the year from a date or timestamp
date_years_difference (ITEM1, ITEM2)	Return the difference in years from ITEM1 to ITEM2, as a real number

This table lists a number of commonly used functions for date and timestamp fields.

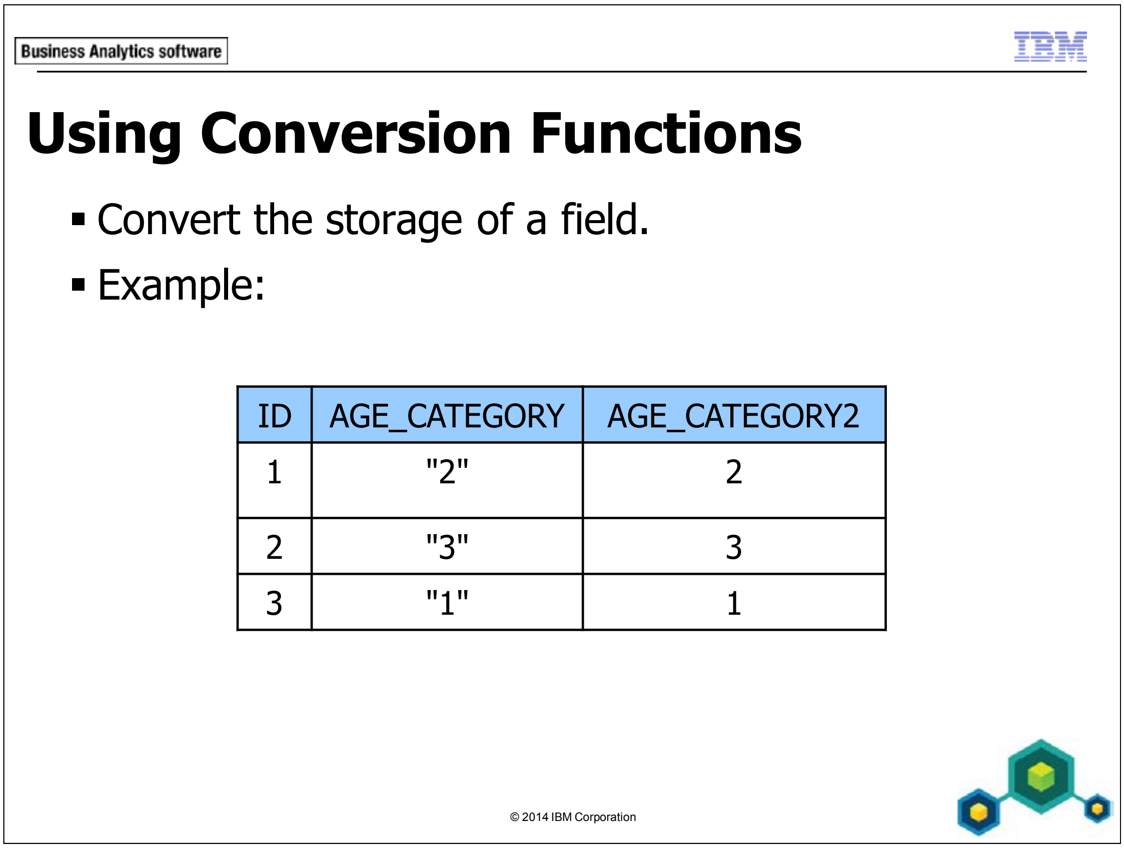
Apart from extracting the year from a date you can extract the month (as a number, long month name, or abbreviated month name) or the day of the week (as a number, long day name, or abbreviated day name). Analogous to the date_years_difference function you can compute the difference in days, weeks or months.

Date functions, like all functions, can be nested and can be used in conjunction with non-date functions. For example, the following expression computes the age, truncated to an integer because of the intof function, at January 1st, 2014:

intof (date_years_difference (DOB, datetime_date (2014, 1, 1)))

Functions analogous to the ones presented on this slide are available for time, such as datetime_hour which extracts the hour from a time or timestamp field, and time_hours_difference which gives the difference in hours between times/timestamps.

Date and time functions are available under the Data and Time functions in the Expression Builder.



Business Analytics software

Using Conversion Functions

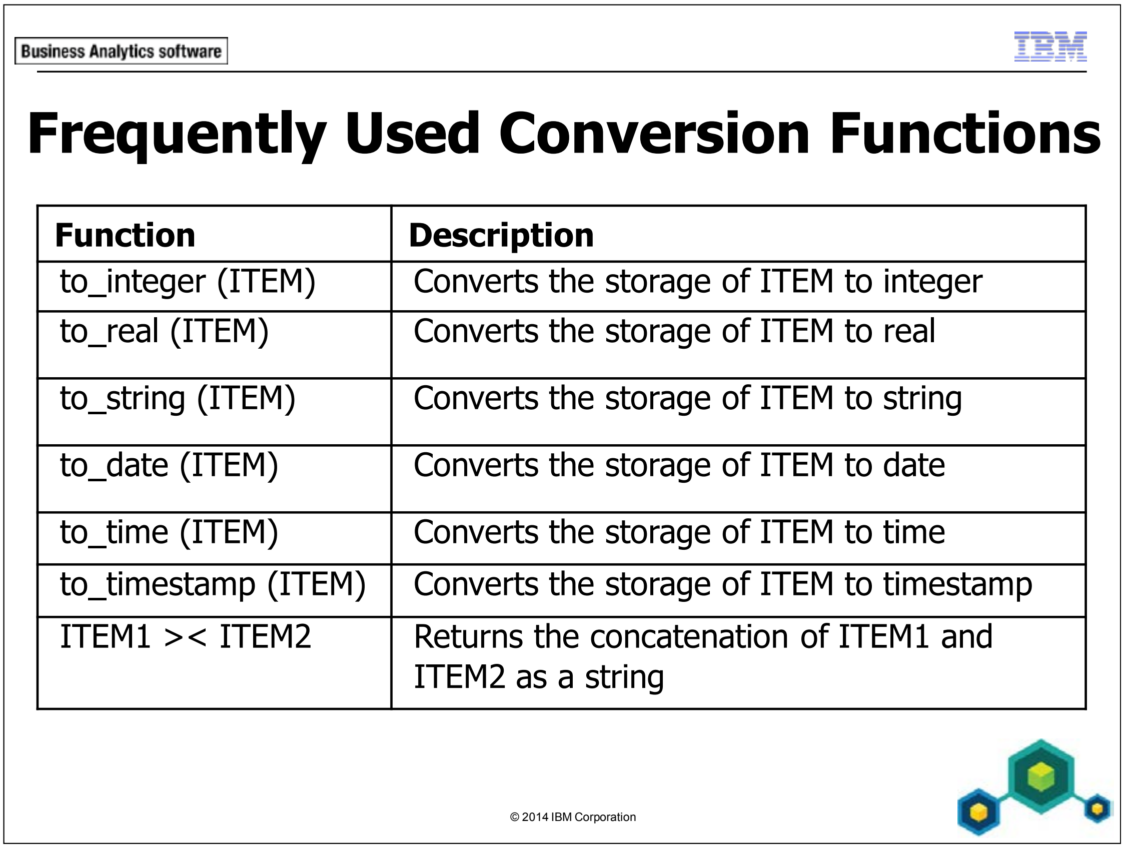
- Convert the storage of a field.
- Example:

ID	AGE_CATEGORY	AGE_CATEGORY2
1	"2"	2
2	"3"	3
3	"1"	1

Sometimes a field has the wrong storage for analyses. This slide shows an example. The AGE_CATEGORY field stores the string values "1", "2", and "3", while these values must be integers so that the field can be used as an ordinal field in modeling tasks. The AGE_CATEGORY2 field stores the values as integers.

Other situations where a conversion function is required are:

- convert a real field to an integer field
- convert a timestamp field to a date field



Business Analytics software

Frequently Used Conversion Functions

Function	Description
to_integer (ITEM)	Converts the storage of ITEM to integer
to_real (ITEM)	Converts the storage of ITEM to real
to_string (ITEM)	Converts the storage of ITEM to string
to_date (ITEM)	Converts the storage of ITEM to date
to_time (ITEM)	Converts the storage of ITEM to time
to_timestamp (ITEM)	Converts the storage of ITEM to timestamp
ITEM1 >< ITEM2	Returns the concatenation of ITEM1 and ITEM2 as a string

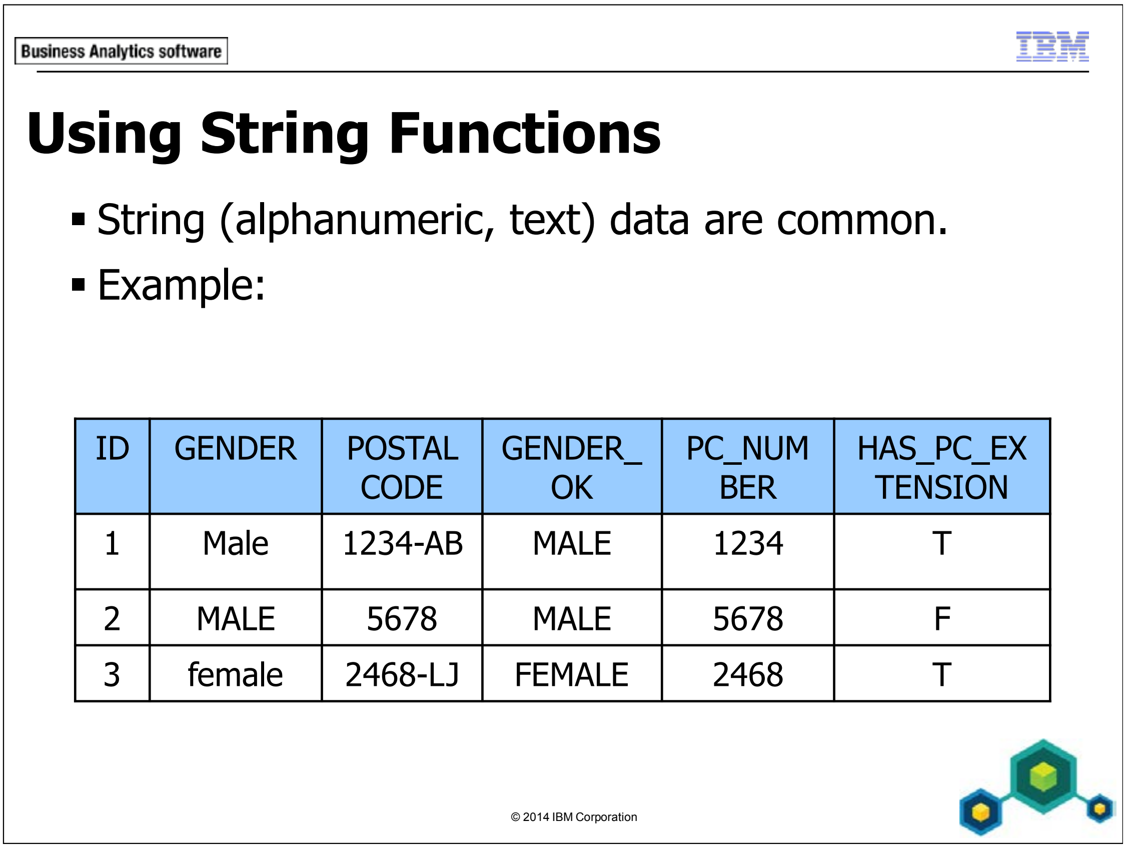
The table on this slide lists frequently used conversion functions. The concatenation function>< is also considered to be a conversion function, since this function will combine fields of any storage into a string field. For example, consider the following expression, with ZIPCODE_1 and ZIPCODE_2 as integer fields, and DOB as date field:

ZIPCODE_1 >< "-" >< ZIPCODE_2 >< "-" >< DOB

This expression returns a string field although none of the source fields is a string field.

When you want to change the storage of a field you can either create a new field or overwrite the field. You can use the Derive node or Filler node to create a new field or to overwrite the filed, respectively. The Filler node is presented in the *Data Transformations* module in this course.

Conversion functions are available under the Conversion functions in the Expression Builder.



Business Analytics software

IBM

Using String Functions

- String (alphanumeric, text) data are common.
- Example:

ID	GENDER	POSTAL CODE	GENDER_ OK	PC_NUM BER	HAS_PC_EX TENSION
1	Male	1234-AB	MALE	1234	T
2	MALE	5678	MALE	5678	F
3	female	2468-LJ	FEMALE	2468	T

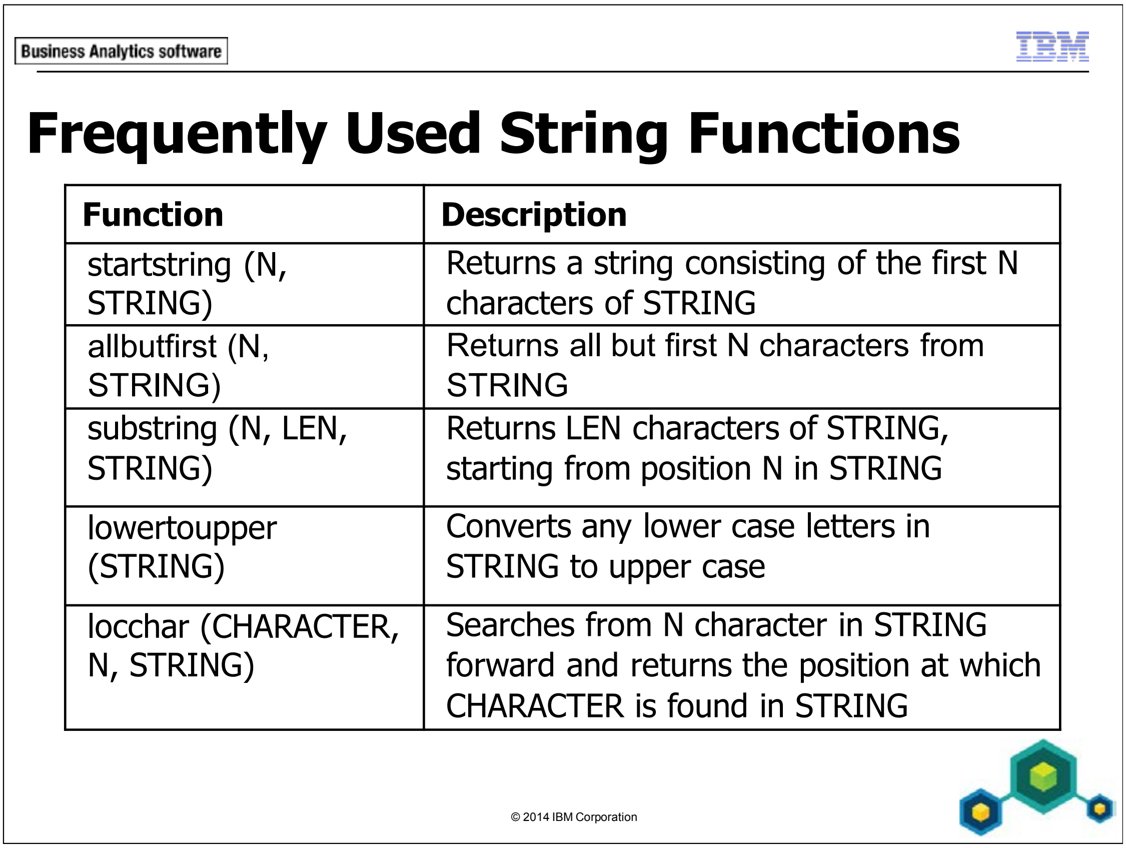
© 2014 IBM Corporation

String (or alphanumeric or text) fields, such as names, zip postal codes, and telephone numbers are common in datasets, especially in databases.

MODELER offers many functions to accommodate working with strings. Tasks that can be completed include:

- correct spelling
- extract a portion of a postal code
- concatenate the first name and the surname into a single field
- extract keywords from a set of text responses

This slide shows an example. The GENDER_OK field replaces the different spellings in GENDER with upper case characters. The PC_NUMBER field extracts the numeric part out of the POSTALCODE field, as an integer. The HAS_PC_EXTENSION field flags if the POSTALCODE field has an extension.



Business Analytics software

Frequently Used String Functions

Function	Description
startstring (N, STRING)	Returns a string consisting of the first N characters of STRING
allbutfirst (N, STRING)	Returns all but first N characters from STRING
substring (N, LEN, STRING)	Returns LEN characters of STRING, starting from position N in STRING
lowertoupper (STRING)	Converts any lower case letters in STRING to upper case
locchar (CHARACTER, N, STRING)	Searches from N character in STRING forward and returns the position at which CHARACTER is found in STRING

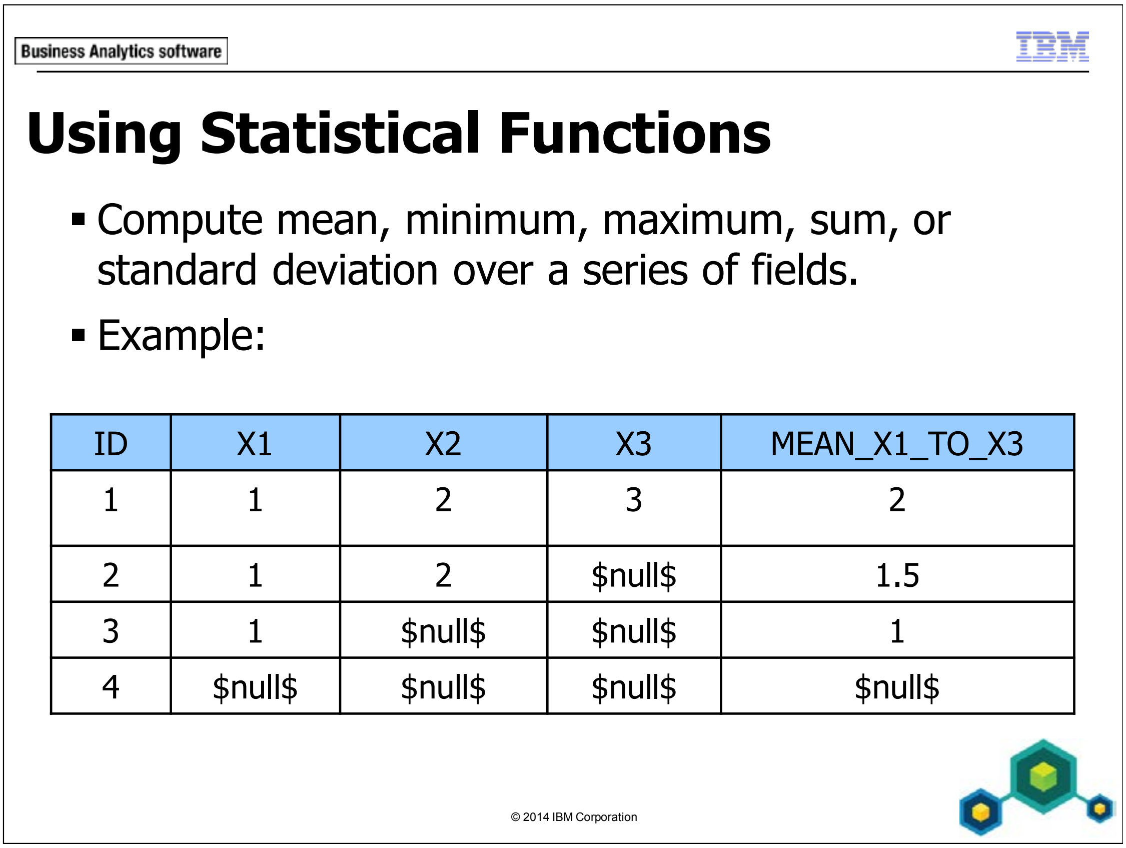
© 2014 IBM Corporation

This slide lists only a few of the many string functions that are commonly used. Also frequently used are functions such as:

- substring_between (N1, N2, STRING): Returns the substring of STRING, which begins at subscript N1 and ends at subscript N2.
- count_substring (STRING, SUBSTRING): Returns the number of instances of SUBSTRING in STRING.
- length (STRING): Returns the number of characters of STRING.
- replace (SUBSTRING1, SUB STRING2, STRING): Replaces SUB STRING1 with SUBSTRING2 in STRING.

When you have text in a function, such as in replace ("Ms", "Miss", NAME), enter the text in double quotes. Important exceptions to this rule are the locchar, locchar_back, stripchar and skipchar functions, where you specify one character which has to be enclosed within single back quotes. For example: locchar (`@`, E_MAIL_ADDRESS).

String functions are available under the String functions in the Expression Builder.



Business Analytics software

IBM

Using Statistical Functions

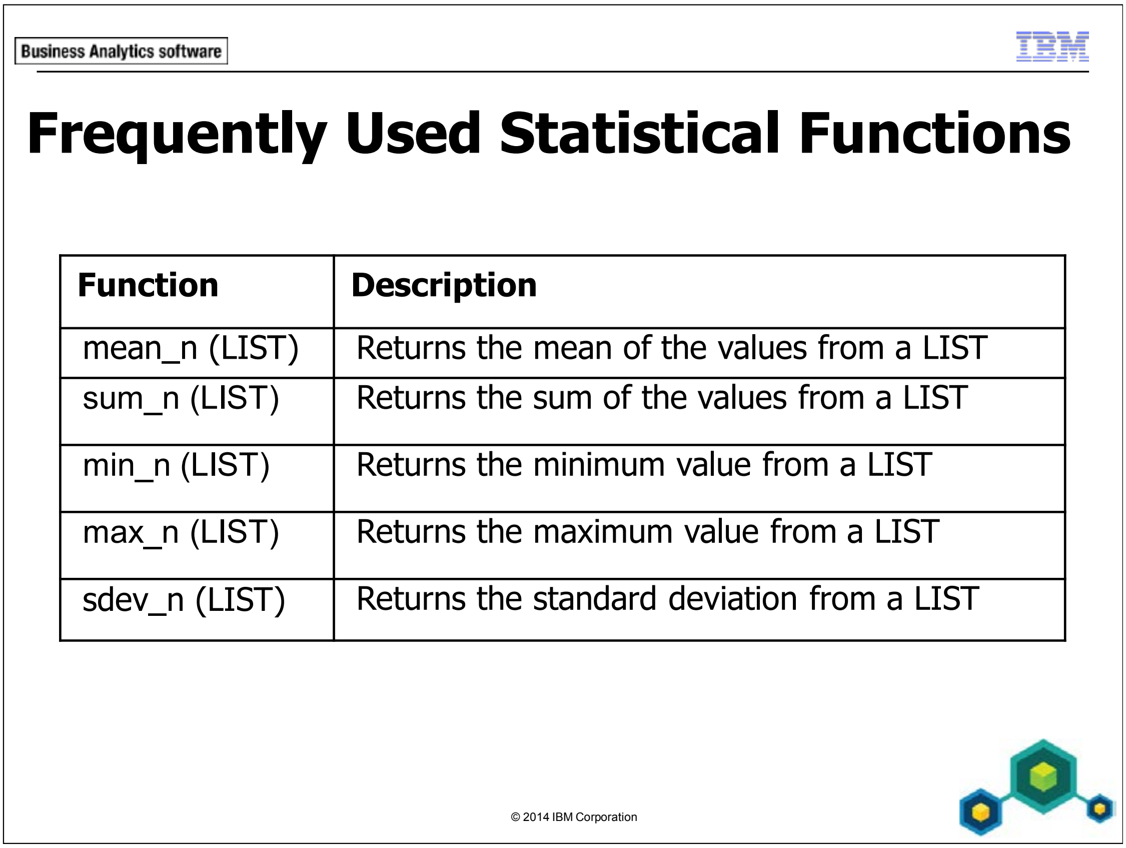
- Compute mean, minimum, maximum, sum, or standard deviation over a series of fields.
- Example:

ID	X1	X2	X3	MEAN_X1_TO_X3
1	1	2	3	2
2	1	2	$null$	1.5
3	1	$null$	$null$	1
4	$null$	$null$	$null$	$null$

Statistical functions are used to calculate a statistic such as the mean or the sum of a series of fields. It is important to note that these functions compute the result per record, rather than computing the mean or sum over all records. For the latter you can use the Aggregate node (if you want to have the statistics in your dataset) or the Statistics node (if you only want to know the statistics).

Rather than using a statistical function you could use an arithmetic expression, but the results of using an arithmetic expression can be different from those obtained when a function is used. The calculation of the mean on this slide illustrates the difference. Missing values are accounted for when you use a statistical function. For example, using the function to compute the mean returns 1.5 for the second record, although this record has a $null$ value. When you compute the mean by the formula (X1 + X2 + X3) / 3 the outcome for the second record will be undefined ($null$).

This example shows that statistical functions and arithmetic expressions do not return the same result when you have missing values.



Business Analytics software

IBM

Frequently Used Statistical Functions

Function	Description
mean_n (LIST)	Returns the mean of the values from a LIST
sum_n (LIST)	Returns the sum of the values from a LIST
min_n (LIST)	Returns the minimum value from a LIST
max_n (LIST)	Returns the maximum value from a LIST
sdev_n (LIST)	Returns the standard deviation from a LIST

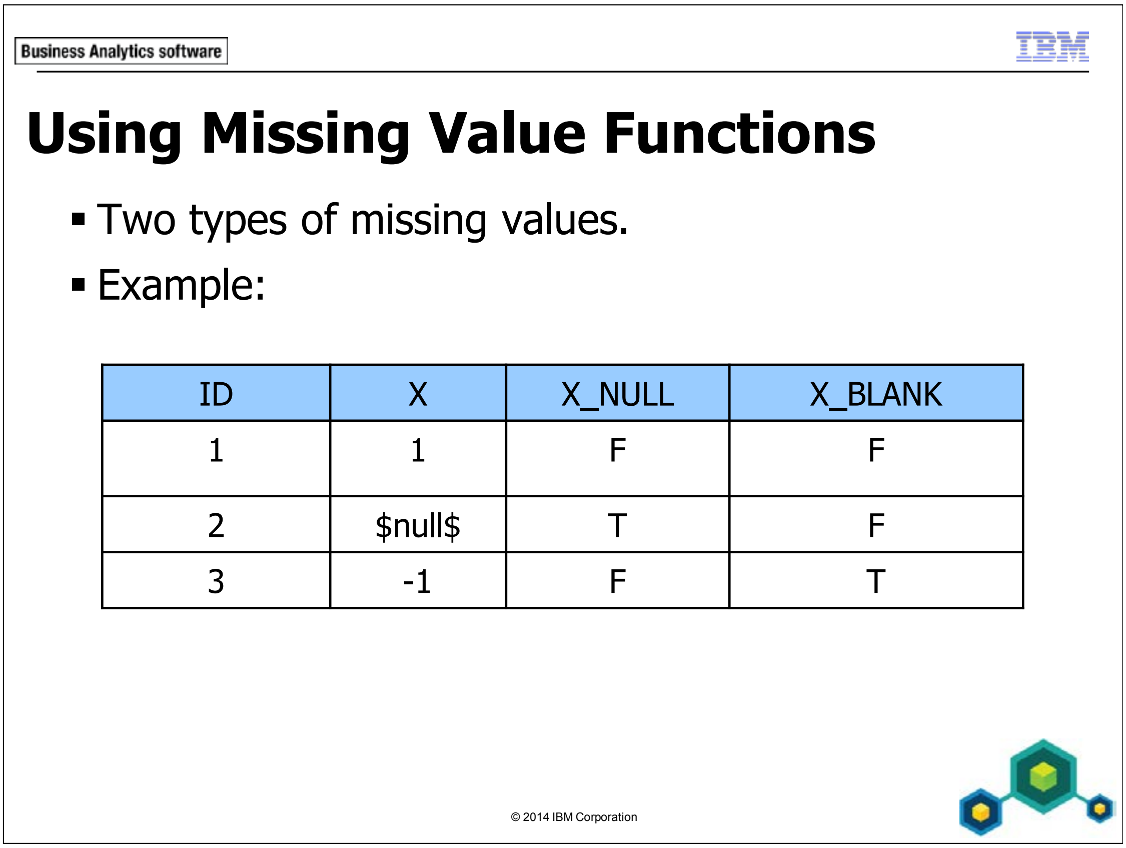
Statistical functions require a list of fields from which to compute the statistic. There are several ways to specify the list:

- Specify the fields and enclose the list in square brackets.
- Use the @FIELDS_BETWEEN function, and then specify the first field and the last field in the list, separated by a comma.
- Use the @FIELDS_MATCHING function, defining all fields whose name match the supplied pattern, where a question mark (?) in the pattern matches one character and an asterisk (*) matches 0 or more characters.

For example, the following three expressions compute the mean for X1 to X3 (assuming that X1, X2 and X3 are adjacent in the dataset and that there are no other X fields in the dataset):

```
mean_n ([X1 X2 X3])
mean_n (@FIELDS_BETWEEN (X1, X3))
mean_n (@FIELDS_MATCHING ("X?")
```

The mean_n, sum_n, and sdev_n functions are available under Numeric in the Expression Builder. The min_n and max_n functions are located under Comparison.



Business Analytics software

Using Missing Value Functions

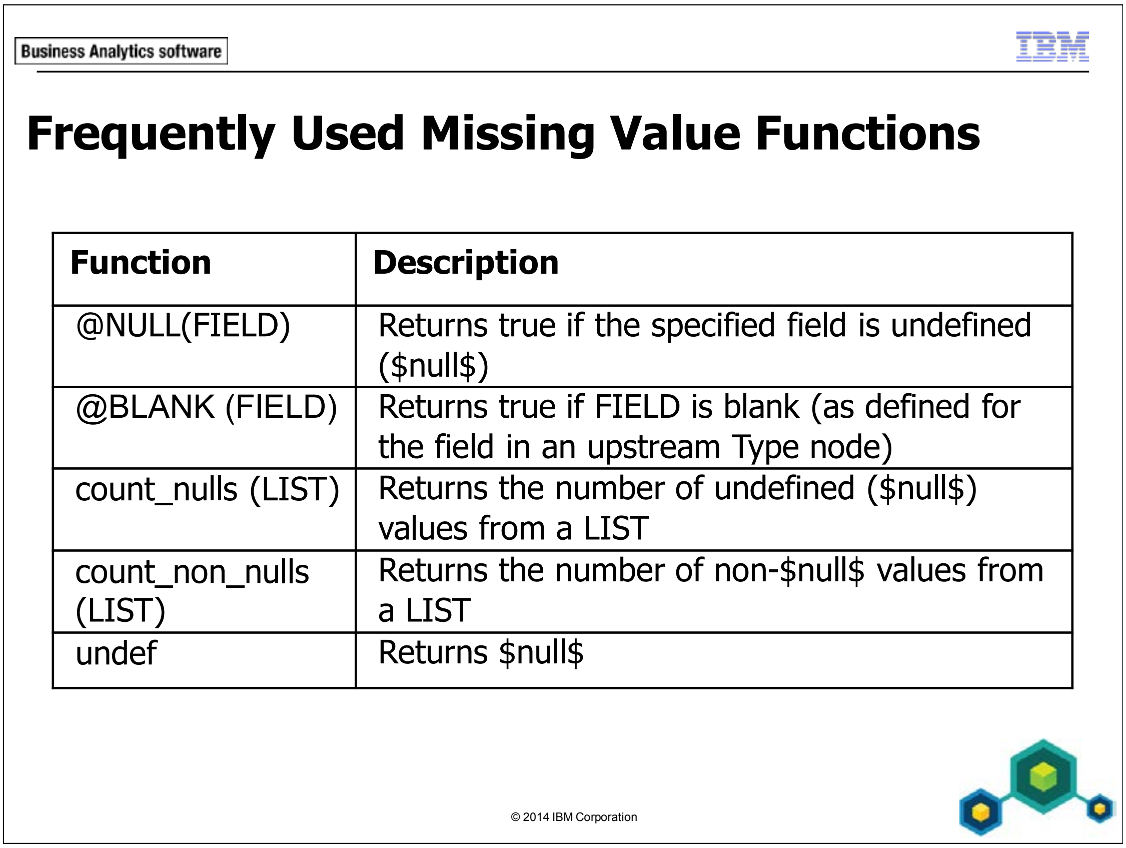
- Two types of missing values.
- Example:

ID	X	X_NULL	X_BLANK
1	1	F	F
2	$null$	T	F
3	-1	F	T

In any data mining project you will encounter missing values. MODELER distinguishes between undefined ($null$) values and blank values, also referred to as user-defined missing values, such as a value -1 for an AGE field.

By default $null$ values are not regarded as blank values, but you can instruct Modeler to handle $null$ values as blank values in a Type node. Refer to the *Introduction to IBM SPSS Modeler and Data Mining (v16)* course for the details.

In the example on this slide,-1 was declared as blank for X, but the $null$ value was not declared as blank for this field. The X_NULL field and the X_BLANK field flag if X field is undefined ($null$) or blank, respectively. The X_NULL field is only true for the second record. The X_BLANK field is only true for the third record. When the undefined ($null$) value would have been declared as blank value for X, X_BLANK would also have returned true for the second record.



Business Analytics software

Frequently Used Missing Value Functions

Function	Description
@NULL(FIELD)	Returns true if the specified field is undefined ($null$)
@BLANK (FIELD)	Returns true if FIELD is blank (as defined for the field in an upstream Type node)
count_nulls (LIST)	Returns the number of undefined ($null$) values from a LIST
count_non_nulls (LIST)	Returns the number of non-$null$ values from a LIST
undef	Returns $null$

The @NULL function and the @BLANK function flag if the specified field is undefined ($null$) or blank, respectively. As illustrated on the previous slide, the @BLANK function will also return true for undefined ($null$) values, provided that undefined values are declared as blank.

The count_nulls function and count_non_nulls function return the number of $null$ values and the number of non-$null$ values, respectively. There is no function to count the number of blanks on a series of fields.

The @NULL, @BLANK and undef function are located under Blanks and Null in the Expression Builder . The count_nulls function and the count_non_nulls function are available under Comparison.

Business Analytics software

Demo 1

Using Functions

© 2014 IBM Corporation

The following (synthetic) file coming from a (fictitious) telecommunications firm is used to demonstrate how you use functions:

- **telco x subset.dat**: Text file that stores demographic and churn data on the company's customers. The file is located in **C:\Train\0A055**.

Before you begin with the demo, ensure that:

- You have started MODELER.
- You have set MODELER's working folder. (In MODELER, select **File\Set Directory**. When you are working in an IBM operated environment, browse to the folder **C:\Train\0A055** and then click **Set** to select it. When you are not working in an IBM operated environment, ask your trainer for the folder name.)

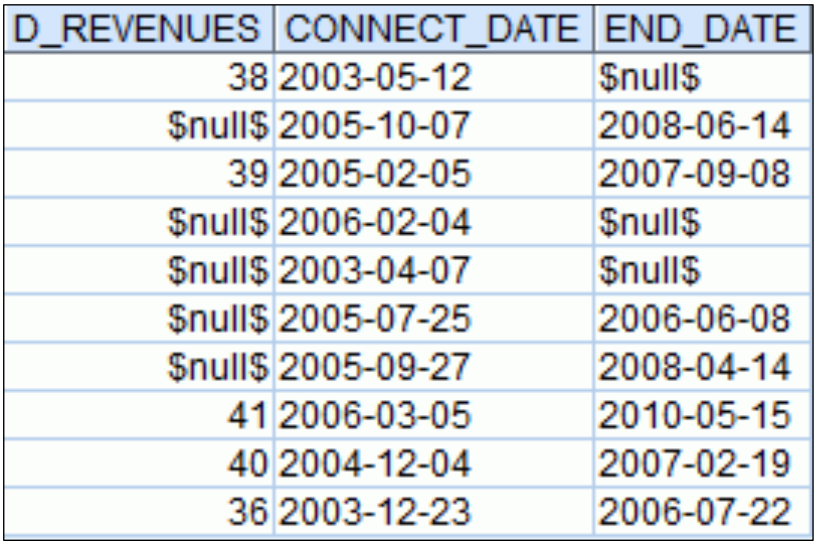
Demo 1: Using Functions

Purpose:
You work for a telecommunications firm where you have to cleanse and enrich a dataset, so that better models can be built later.

Task 1. Using date functions to derive fields.

1. Use the **Var. File** node to import data from the comma-separated text file **telco x subset.dat**, click **Preview** in the **Var. File** dialog box, and then scroll to the last fields in the **Preview** output window.

   A section of the results appear as follows:



D_REVENUES	CONNECT_DATE	END_DATE
38	2003-05-12	$null$
$null$	2005-10-07	2008-06-14
39	2005-02-05	2007-09-08
$null$	2006-02-04	$null$
$null$	2003-04-07	$null$
$null$	2005-07-25	2006-06-08
$null$	2005-09-27	2008-04-14
41	2006-03-05	2010-05-15
40	2004-12-04	2007-02-19
36	2003-12-23	2006-07-22

   The CONNECT_DATE field gives the date that the customer was connected to the company's network. The END_DATE stores the date that the customer ended his or her subscription and is undefined when the customer did not end the subscription.

2. Close the **Preview** output window, and then close the **Var. File** dialog box.

   You will compute the length of stay, in months.

3. Add a **Derive** node downstream from the **Var. File** node.

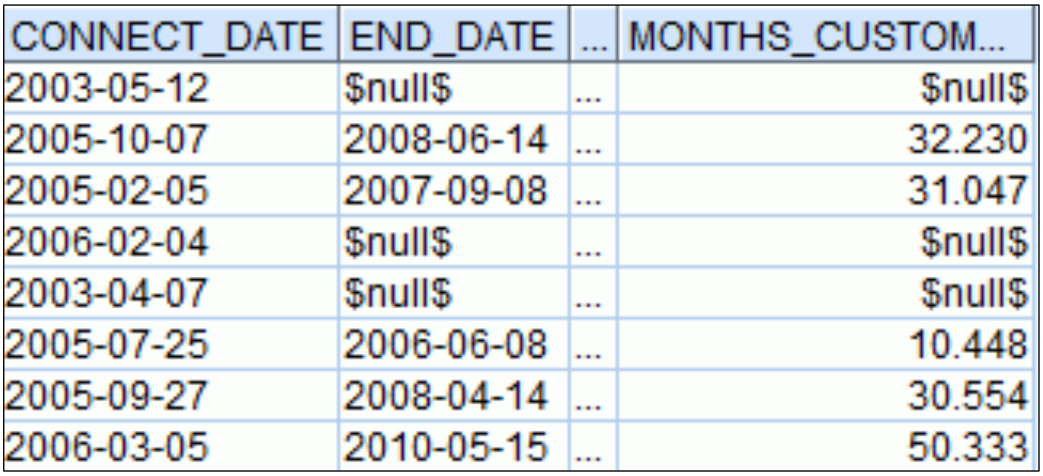
4. Edit the **Derive** node, and then:

   - for **Derive field**, enter **MONTHS_CUSTOMER**
   - for **Formula**, enter **date_months_difference (CONNECT_DATE, END_DATE)**

   Note: Type the expression or use the Expression Builder to construct the expression. In this course "enter" refers to typing or using the Expression Builder, according to your preference.

5. Click **Preview**, and then scroll to the last fields in the **Preview** output window.

   A section of the results appear as follows:



CONNECT_DATE	END_DATE	...	MONTHS_CUSTOM...
2003-05-12	$null$	...	$null$
2005-10-07	2008-06-14	...	32.230
2005-02-05	2007-09-08	...	31.047
2006-02-04	$null$	...	$null$
2003-04-07	$null$	...	$null$
2005-07-25	2006-06-08	...	10.448
2005-09-27	2008-04-14	...	30.554
2006-03-05	2010-05-15	...	50.333

   The new field gives the number of months that elapsed between the two dates, stored as a real number. If you want the result in whole months you can use a function such as round, intof or to_integer to derive the field as integer.

   Notice that the date_months_difference function returns the undefined ($null$) value when END_DATE is undefined ($null$).

6. Close the **Preview** output window, and then close the **Derive** dialog box.

   You will derive a field that gives the month when the customer was connected, and a field that gives the month when the customer ended the subscription. Furthermore, the month must be returned as a string, not as a number. For example, the result must be January rather than 1.

   Because the same function can be applied you will compute the two fields in a single Derive node.

7. Add a second **Derive** node downstream from the **Derive** node named **MONTHS_CUSTOMERS**.

8. Edit the **Derive** node, and then set the **Mode** to **Multiple**.

   The Derive dialog box reflects the change. The Derive field text box is replaced with a Derive from box in which the fields must be selected on which the calculations are based.

9. Continue editing the **Derive** node, and then:

   - for **Derive from**, select **CONNECT_DATE** and **END_DATE**.
   - for **Field name extension**, type **_MONTH**

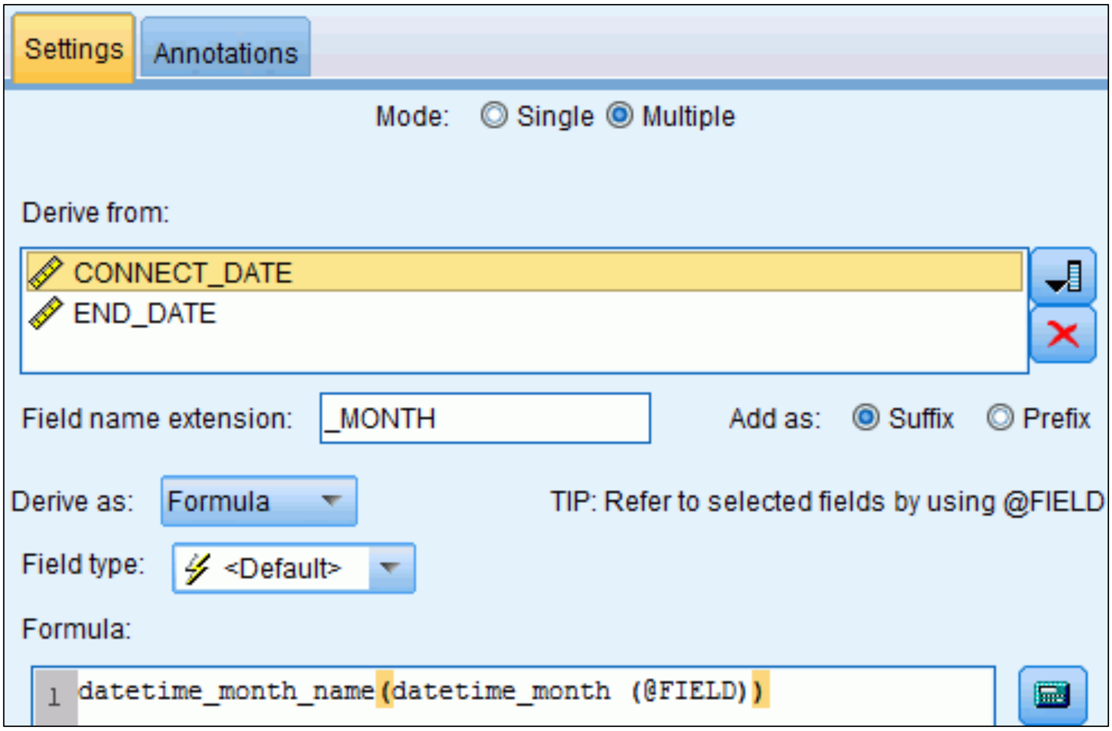
   You need the function datetime_month_name (MONTH NUMBER) to get the name of the month. This function cannot be applied directly to a date because this function's argument must be an integer in the range from 1 to 12. You will first extract the month's number from the date using the datetime_month function and then apply the datetime_month_name function to arrive at the name of the month.

   Because the formula must be applied to both fields, you will use @FIELD as a substitute for the field names.

10. Continue editing the **Derive** node:

   - for **Formula**, enter **datetime_month_name (datetime_month (@FIELD))**

   A section of the results appear as follows:



Mode: ○ Single ● Multiple

Derive from:
CONNECT_DATE
END_DATE

Field name extension: _MONTH

Add as: ● Suffix ○ Prefix

Derive as: Formula

TIP: Refer to selected fields by using @FIELD

Field type: <Default>

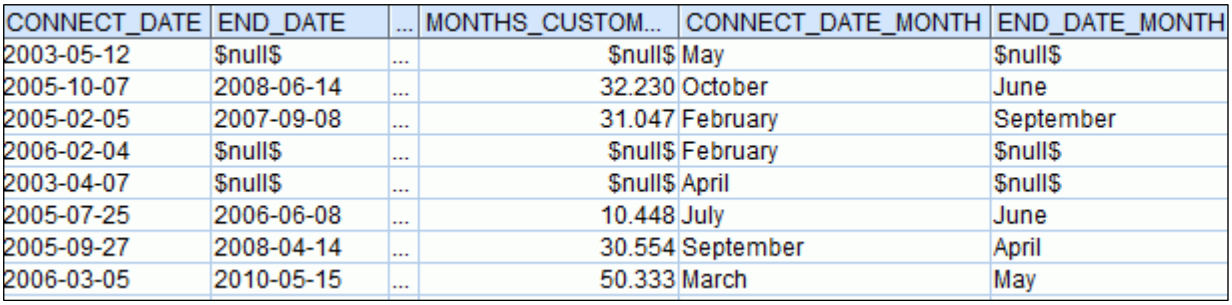
Formula:

```
datetime_month_name(datetime_month (@FIELD))
```

This material is meant for IBM Academic Initiative purposes.

11. Click **Preview**, and then scroll to the last fields in the **Preview** output window.

    A section of the results appear as follows:



CONNECT_DATE	END_DATE	...	MONTHS_CUSTOM...	CONNECT_DATE_MONTH	END_DATE_MONTH
2003-05-12	$null$	...	$null$	May	$null$
2005-10-07	2008-06-14	...	32.230	October	June
2005-02-05	2007-09-08	...	31.047	February	September
2006-02-04	$null$	...	$null$	February	$null$
2003-04-07	$null$	...	$null$	April	$null$
2005-07-25	2006-06-08	...	10.448	July	June
2005-09-27	2008-04-14	...	30.554	September	April
2006-03-05	2010-05-15	...	50.333	March	May

    Two fields have been created in a single Derive node. One field stores the month (as a name) in which the person was connected to the company's network, the other stores the month in which the person ended the subscription (undefined when END_DATE is undefined).

12. Close the **Preview** output window, and then close the **Derive** dialog box.

    Leave the stream open for the next task.

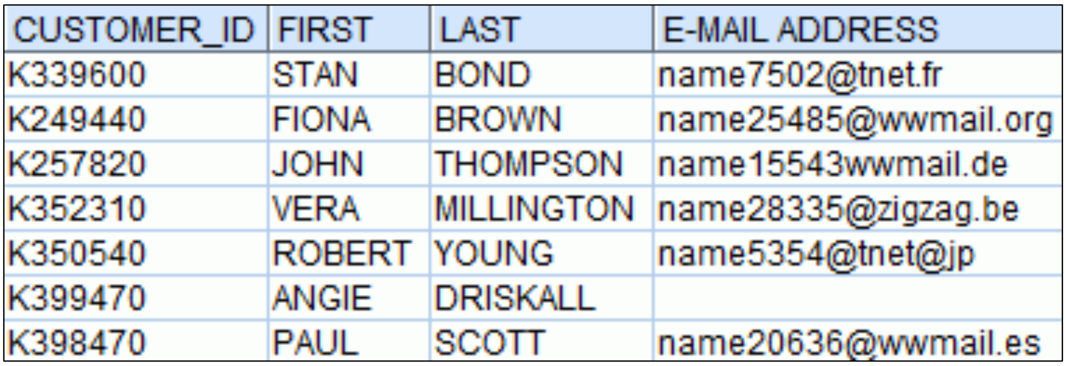
Task 2. Using string functions to derive fields.

In this task you will perform checks on the e-mail address and you will derive a field that gives the domain name.

You will build from the stream that you created in the previous task.

1. Add a **Table** node downstream from the **Derive** node named **_MONTH**, and then run the **Table** node.

    A section of the results appear as follows:



CUSTOMER_ID	FIRST	LAST	E-MAIL ADDRESS
K339600	STAN	BOND	name7502@tnet.fr
K249440	FIONA	BROWN	name25485@wwmail.org
K257820	JOHN	THOMPSON	name15543wwmail.de
K352310	VERA	MILLINGTON	name28335@zigzag.be
K350540	ROBERT	YOUNG	name5354@tnet@jp
K399470	ANGIE	DRISKALL	
K398470	PAUL	SCOTT	name20636@wwmail.es

    The values in the E-MAIL_ADDRESS field are not always representing a correct e-mail address. For example, the third record (CUSTOMER_ID K257820) misses the at sign @ in the address, while the fifth record (CUSTOMER_ID K350540) has two. The e-mail address for the sixth record

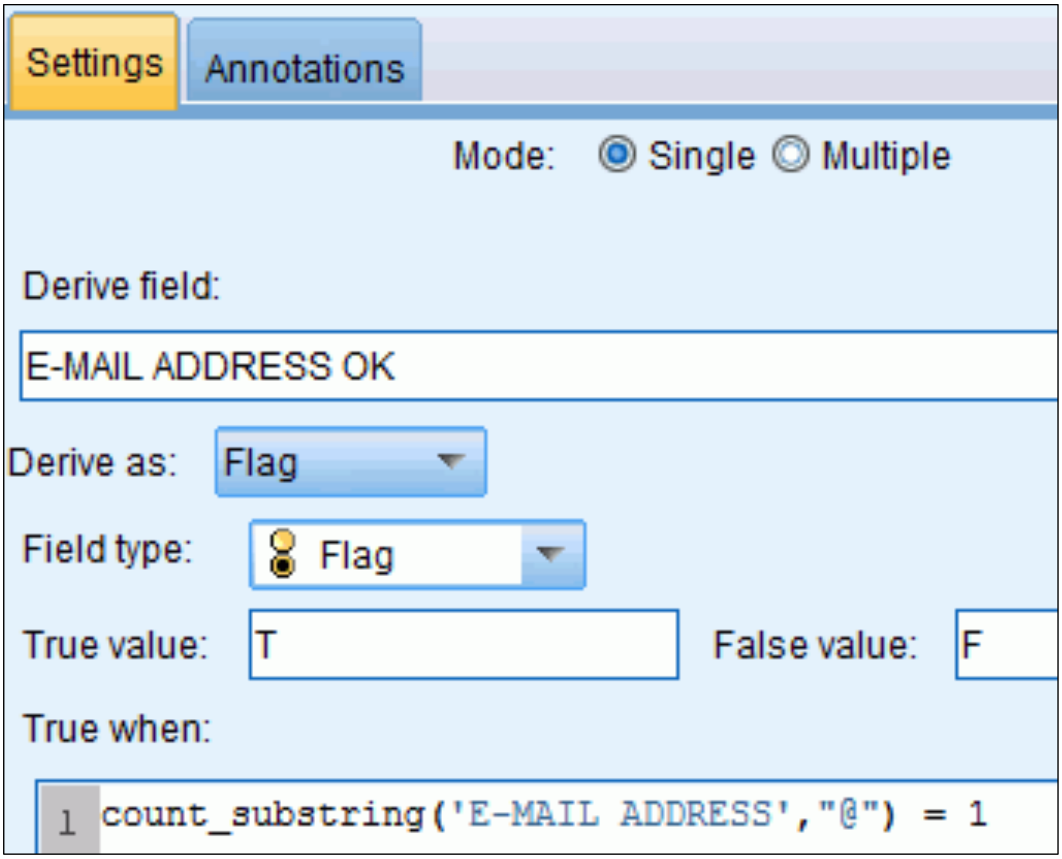
(CUSTOMER_ID K399470) appears as empty space, thus also misses the at sign. A correct e-mail address should have exactly one at sign.

2. Close the **Table** output window.

   You will derive a field named E-MAIL ADDRESS OK which returns true if the address has exactly one at sign. To derive this field you will need the count_substring (STRING, SUBSTRING) function. This function returns the number of instances of SUBSTRING that are found in STRING.

3. Add a **Derive** node downstream from the **Derive** node named **_MONTH**.
4. Edit the **Derive** node, and then:
   - for **Derive field** type **E-MAIL ADDRESS OK**
   - for **Derive as**, select **Flag**
   - for **True when**, enter **count_substring('E-MAIL ADDRESS', "@") = 1**

A section of the results appear as follows:



Mode: ● Single ○ Multiple

Derive field:

E-MAIL ADDRESS OK

Derive as: Flag

Field type: Flag

True value: T

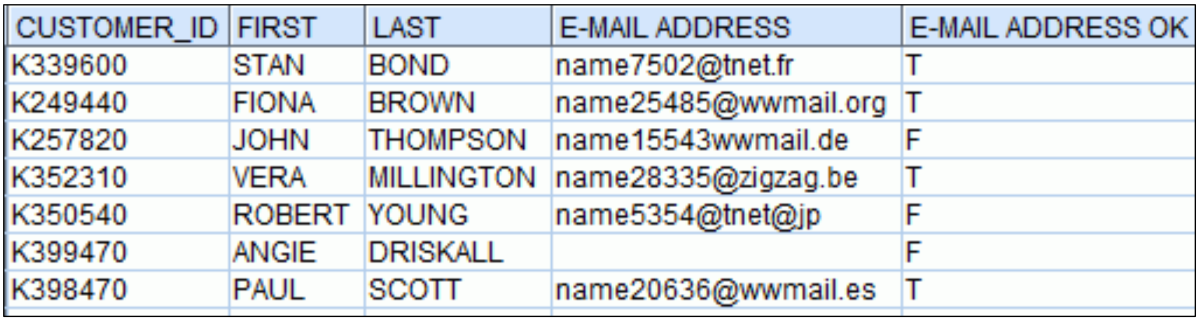
False value: F

True when:

```
count_substring('E-MAIL ADDRESS',"@") = 1
```

5. Click **Preview**, and then move **E-MAIL ADDRESS OK** next to **E-MAIL_ADDRESS** in the **Preview** output window.

   A section of the results appear as follows:



CUSTOMER_ID	FIRST	LAST	E-MAIL ADDRESS	E-MAIL ADDRESS OK
K339600	STAN	BOND	name7502@tnet.fr	T
K249440	FIONA	BROWN	name25485@wwmail.org	T
K257820	JOHN	THOMPSON	name15543wwmail.de	F
K352310	VERA	MILLINGTON	name28335@zigzag.be	T
K350540	ROBERT	YOUNG	name5354@tnet@jp	F
K399470	ANGIE	DRISKALL		F
K398470	PAUL	SCOTT	name20636@wwmail.es	T

   Records having exactly one at sign in their e-mail address are true for the derived field, records which have no at sign or which have more than 1 at sign in their e-mail address are false.

6. Close the **Preview** output window, and then close the **Derive** dialog box.

   Suppose that records with an empty e-mail address are of special interest and that a field must be derived that flags if there is no address at all. For example, this field must return true for the sixth record (CUSTOMER_ID K399470).

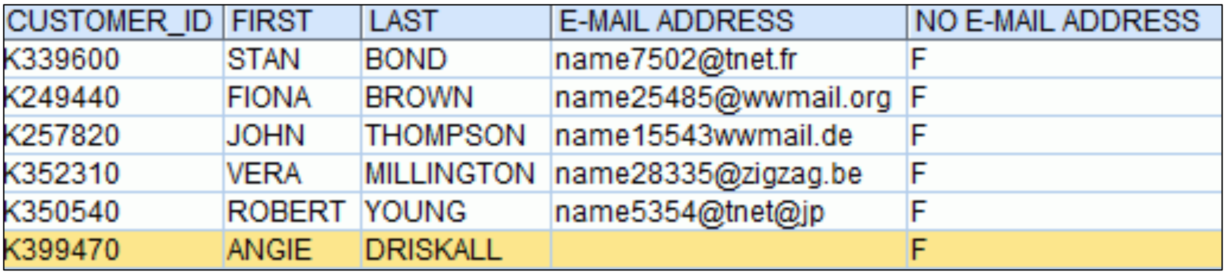
   There are several ways to accomplish this. One option is to use the length function to check the length of the e-mail address and to have the function return true if the length is 0. You will explore this option.

7. Add a **Derive** node downstream from the **Derive** node named **E-MAIL ADDRESS OK**.

8. Edit the **Derive** node, and then:

   - for **Derive field** type **NO E-MAIL ADDRESS**
   - for **Derive as**, select **Flag**
   - for **True when**, enter **length ('E-MAIL ADDRESS') = 0**
   - click **Preview**, and move **NO E-MAIL ADDRESS** next to **E-MAIL ADDRESS**

A section of the results appear as follows:

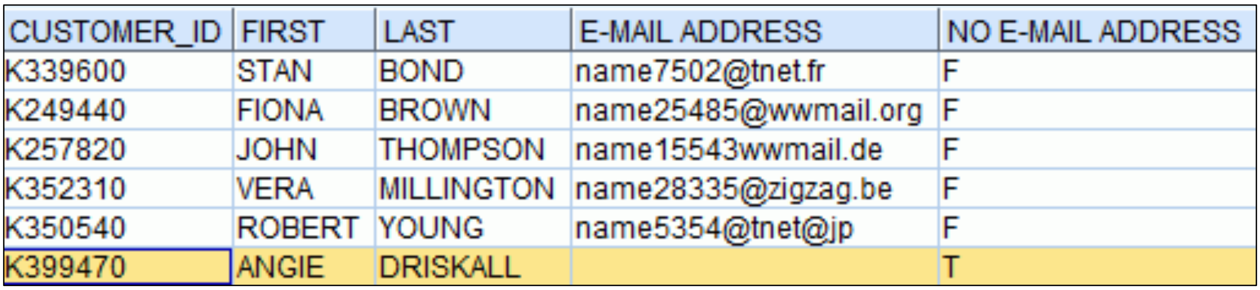


CUSTOMER_ID	FIRST	LAST	E-MAIL ADDRESS	NO E-MAIL ADDRESS
K339600	STAN	BOND	name7502@tnet.fr	F
K249440	FIONA	BROWN	name25485@wwmail.org	F
K257820	JOHN	THOMPSON	name15543wwmail.de	F
K352310	VERA	MILLINGTON	name28335@zigzag.be	F
K350540	ROBERT	YOUNG	name5354@tnet@jp	F
K399470	ANGIE	DRISKALL		F

Surprisingly enough, the sixth record is false for the derived field. Apparently, the length of the e-mail address is greater than 0 for this record. The only explanation can be that the e-mail address is comprised of a series of space characters. Thus, the space characters should be removed from the e-mail address first. The trim function is designed for this and you will combine this function with the length function to arrive at the required results.

9. Close the **Preview** output window.
10. Continue editing the **Derive** node, and then:
    - for **True when**, enter **length (trim ('E-MAIL ADDRESS')) = 0**
    - **Preview** the data, and then move **NO E-MAIL ADDRESS** next to **E-MAIL ADDRESS**

    A section of the results appear as follows:



CUSTOMER_ID	FIRST	LAST	E-MAIL ADDRESS	NO E-MAIL ADDRESS
K339600	STAN	BOND	name7502@tnet.fr	F
K249440	FIONA	BROWN	name25485@wwmail.org	F
K257820	JOHN	THOMPSON	name15543wwmail.de	F
K352310	VERA	MILLINGTON	name28335@zigzag.be	F
K350540	ROBERT	YOUNG	name5354@tnet@jp	F
K399470	ANGIE	DRISKALL		T

The sixth record has the true value for the derived field, as required.

11. Close the **Preview** output window, and then close the **Derive** dialog box.

Rather than using the length function combined with the trim function, you can use the iswhitespace function to arrive at the same results. The iswhitespace function tests whether the specified string is all spaces (zero or more space characters). This is left as an exercise. It should be noted that the iswhitespace function is not listed as one of the functions in the Expression Builder, so you would have to type the name of the function.

As a last, and more advanced, example of applying string functions you will derive a field that gives the domain name. The domain name is the last part of the e-mail address. For example, for an e-mail address such as my_name@tnet.fr it must return fr.

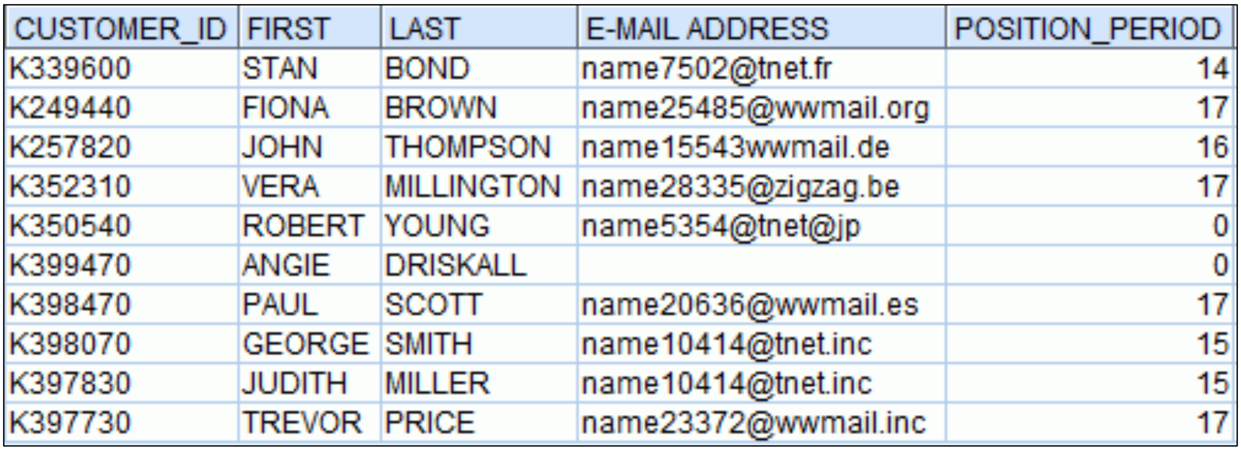
The strategy for extracting the domain name is based on the idea that the domain name is the part after the last period. Thus, you will first have to locate the period, starting at the end of the e mail address and then search backwards. The locchar_back (CHAR, N, STRING) is designed for this. This function searches from the Nth character of STRING backwards and returns the position at which CHAR is found in STRING. In the locchar function, you will specify the period as character to look for and E-MAIL ADDRESS as string to look in. The value of N, the starting position for searching backwards, is not a fixed number, but varies depending on the length of the string. In other words, the value for N must be equal to the length of the e-mail address. The length function does exactly that, as demonstrated in the previous task.

Thus, you first will derive a field that gives the position of the last period in the e-mail address.

12. Add a **Derive** node downstream from the **Derive** node named **NO E-MAIL ADDRESS**.
13. Edit the **Derive** node, and then:
    - for **Derive field** type **POSITION_PERIOD**
    - for **Formula**, enter **locchar_back(`.`, length ('E-MAIL ADDRESS'), 'E-MAIL ADDRESS')**

      Note: The period must be enclosed within back quotes.
    - click **Preview**, and then move **POSITION_PERIOD** next to **E-MAIL ADDRESS**

A section of the results appear as follows:



CUSTOMER_ID	FIRST	LAST	E-MAIL ADDRESS	POSITION_PERIOD
K339600	STAN	BOND	name7502@tnet.fr	14
K249440	FIONA	BROWN	name25485@wwmail.org	17
K257820	JOHN	THOMPSON	name15543wwmail.de	16
K352310	VERA	MILLINGTON	name28335@zigzag.be	17
K350540	ROBERT	YOUNG	name5354@tnet@jp	0
K399470	ANGIE	DRISKALL		0
K398470	PAUL	SCOTT	name20636@wwmail.es	17
K398070	GEORGE	SMITH	name10414@tnet.inc	15
K397830	JUDITH	MILLER	name10414@tnet.inc	15
K397730	TREVOR	PRICE	name23372@wwmail.inc	17

The new field stores the position of the last period, except for e-mail addresses that do not have a period in it, in which case the position equals 0.

14. Close the **Preview** output window, and then close the **Derive** dialog box.

    Knowing the position of the last period, there are several ways to proceed. You can use a substring function or function such as endstring. Here you will use the substring_between (N1, N2, STRING) function. This function returns the substring of STRING beginning at subscript N1 and ending at subscript N2. In this expression, N1 is the position of the period plus 1, and N2 is the length of the e-mail address.

    Notice that the computation must be conditional on whether the e-mail address has a period in it, in which case POSITION_PERIOD is greater than 0.

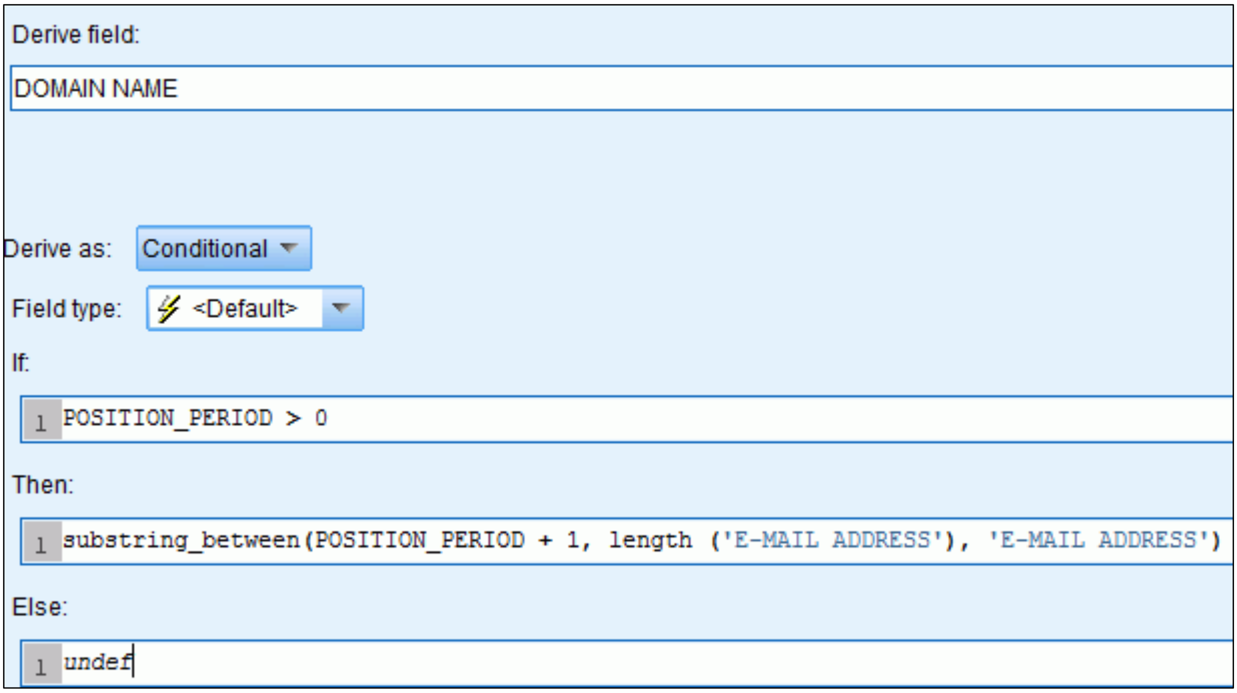
15. Add a **Derive** node downstream from the **Derive** node named **POSITION_PERIOD**.

16. Edit the **Derive** node, and then:

    - for **Derive field** type **DOMAIN NAME**
    - for **Derive as**, select **Conditional**
    - for **If**, enter **POSITION_PERIOD > 0**
    - for **Then**, enter **substring_between (POSITION_PERIOD + 1, length ('E-MAIL ADDRESS'), 'E-MAIL ADDRESS')**
    - for **Else**, type **undef**

This material is meant for IBM Academic Initiative purposes.

A section of the results appear as follows:



Derive field:

DOMAIN NAME

Derive as: Conditional

Field type: <Default>

If:

```
POSITION_PERIOD > 0
```

Then:

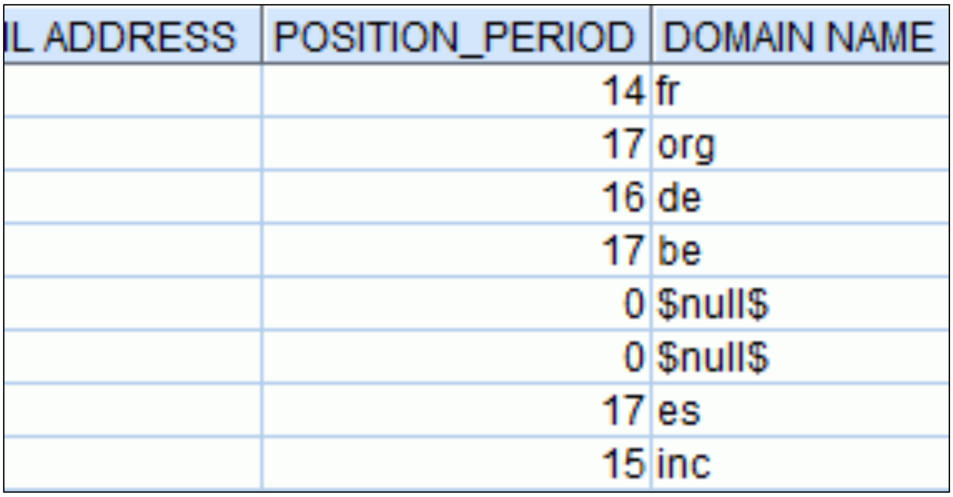
```
substring_between(POSITION_PERIOD + 1, length ('E-MAIL ADDRESS'), 'E-MAIL ADDRESS')
```

Else:

```
undef
```

17. Click **Preview**, and then scroll to the last fields in the **Preview** output window.

    A section of the results appear as follows:



IL ADDRESS	POSITION_PERIOD	DOMAIN NAME
	14	fr
	17	org
	16	de
	17	be
	0	$null$
	0	$null$
	17	es
	15	inc

The field DOMAIN NAME stores the domain name and is undefined when the e-mail address does not have a period, as required.

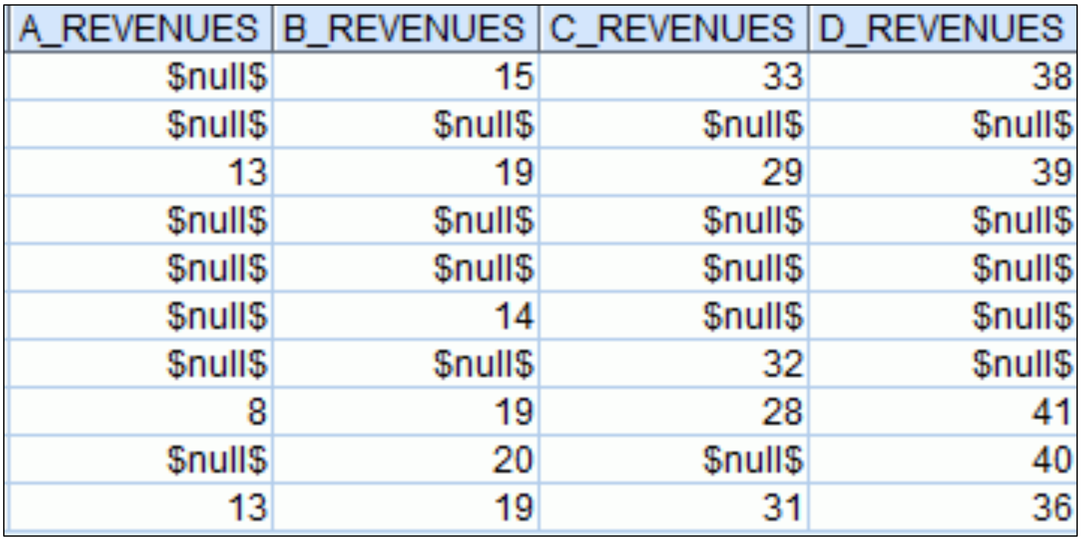
18. Close the **Preview** output window, and then close the **Derive** dialog box.

    As noticed there are alternative ways to arrive at the same result. You can also use the textsplit function which splits a string at a specified character and takes the 1st part or 2nd part (which is one of the arguments of this function). This, however, assumes that there is only one period in an e-mail address which may be a risky assumption.

    Leave the stream open for the next task.

Task 3. Using statistical functions to derive fields.

In this task you will use statistical functions to compute the mean revenues per record and the sum of revenues per record, based on A_REVENUES to D_REVENUES.



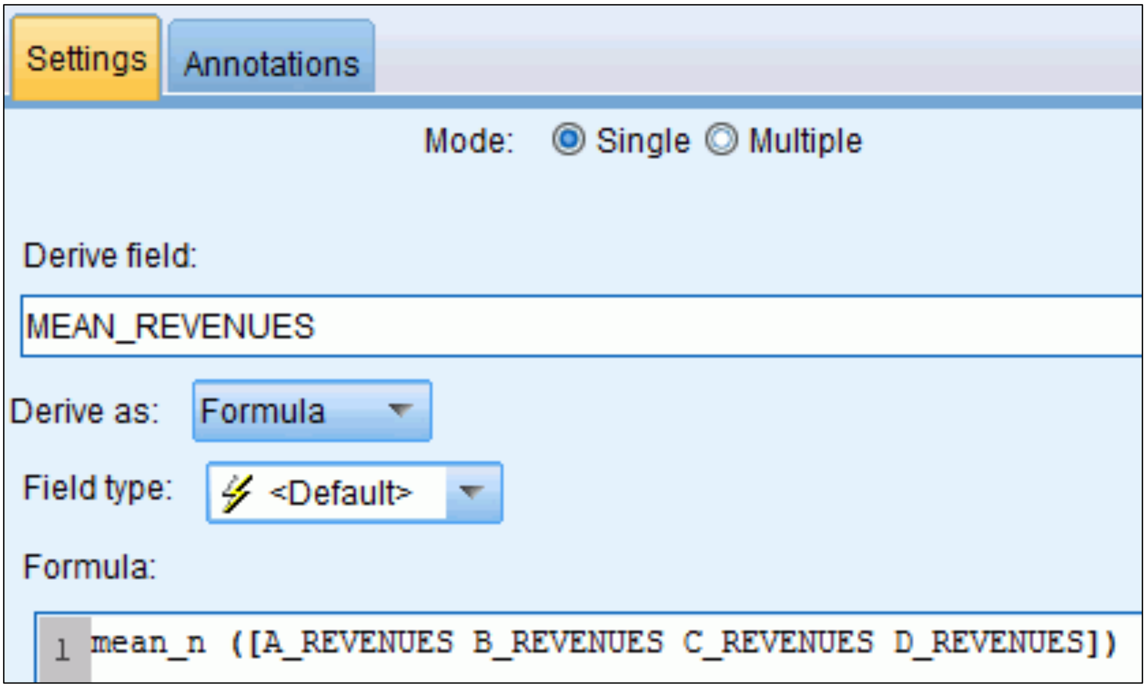
A_REVENUES	B_REVENUES	C_REVENUES	D_REVENUES
$null$	15	33	38
$null$	$null$	$null$	$null$
13	19	29	39
$null$	$null$	$null$	$null$
$null$	$null$	$null$	$null$
$null$	14	$null$	$null$
$null$	$null$	32	$null$
8	19	28	41
$null$	20	$null$	40
13	19	31	36

For example, the third record has mean revenues (13 + 19 + 29 + 39) / 4=25.

You will build from the stream that you created in the previous task.

1. Add a **Derive** node downstream from the **Derive** node named **DOMAIN NAME**.
2. Edit the **Derive** node, and then:
   - for **Derive field**, enter **MEAN_REVENUES**
   - for **Formula**, enter **mean_n ([A_REVENUES B_REVENUES C_REVENUES D_REVENUES])**

A section of the results appear as follows:



Mode: ● Single ○ Multiple

Derive field:

MEAN_REVENUES

Derive as: Formula

Field type: <Default>

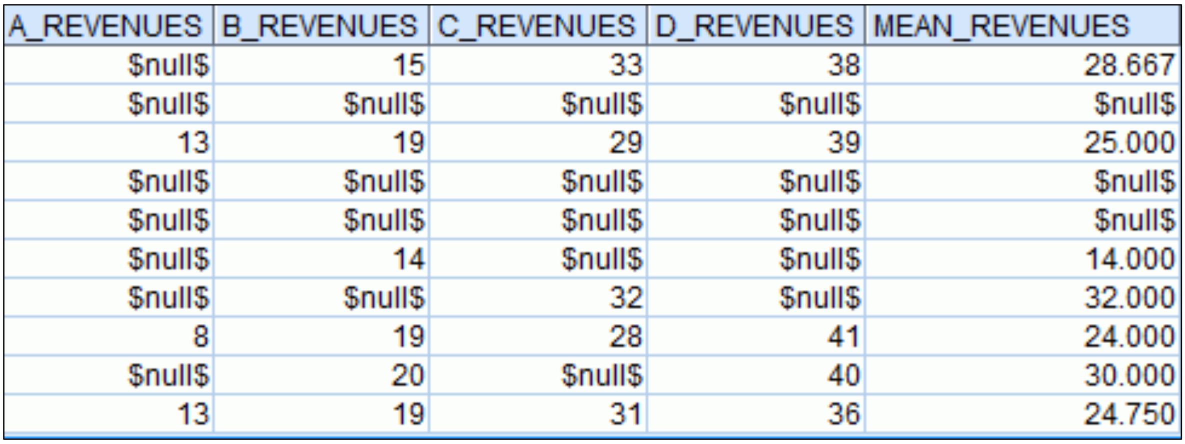
Formula:

```
mean_n ([A_REVENUES B_REVENUES C_REVENUES D_REVENUES])
```

Notice that the fields in the list are enclosed within square brackets.

3. Click **Preview**, and then move the **MEAN_REVENUES** next to **D_REVENUES**.

A section of the results appear as follows:



A_REVENUES	B_REVENUES	C_REVENUES	D_REVENUES	MEAN_REVENUES
$null$	15	33	38	28.667
$null$	$null$	$null$	$null$	$null$
13	19	29	39	25.000
$null$	$null$	$null$	$null$	$null$
$null$	$null$	$null$	$null$	$null$
$null$	14	$null$	$null$	14.000
$null$	$null$	32	$null$	32.000
8	19	28	41	24.000
$null$	20	$null$	40	30.000
13	19	31	36	24.750

The mean_n function returned the mean computed over the valid values. For example, the first record's average is (15+33+38)/3 = 28.667. Thus, missing values are accounted for by the mean_n function. The mean_n function will only return an undefined value when all source fields are missing, as for the second record.

4. Close the **Preview** output window, and then close the **Derive** dialog box.

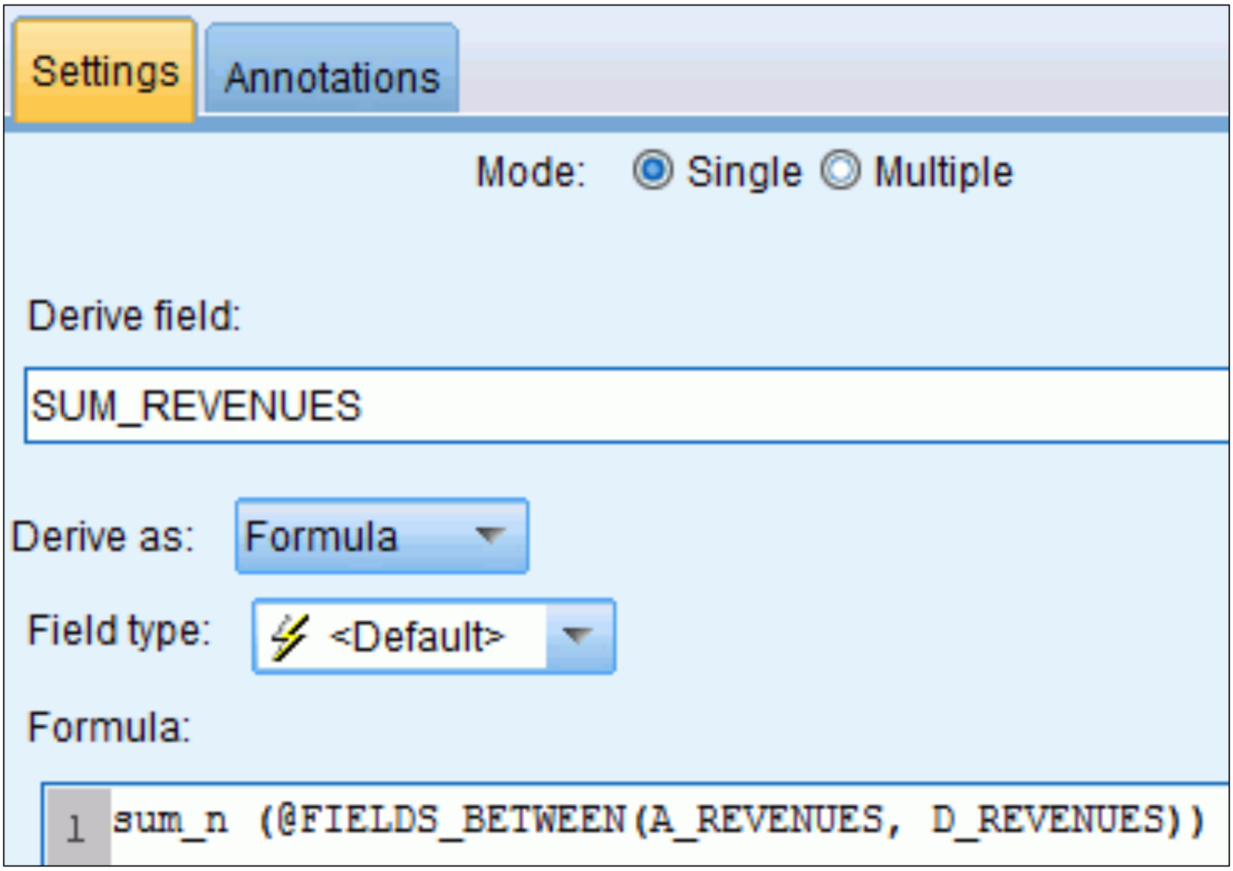
   Next to computing the mean you will compute the sum. Rather than specifying the source fields by name, you will use the @FIELD_BETWEEN function to reference them.

5. Add a **Derive** node downstream from the **Derive** node named **MEAN_REVENUES**.

6. Edit the **Derive** node, and then:

   - for **Derive field**, type **SUM_REVENUES**
   - for **Formula**, enter **sum_n (@FIELDS_BETWEEN(A_REVENUES, D_REVENUES))**

   A section of the results appear as follows:



Mode: ● Single ○ Multiple

Derive field:

SUM_REVENUES

Derive as: Formula

Field type: <Default>

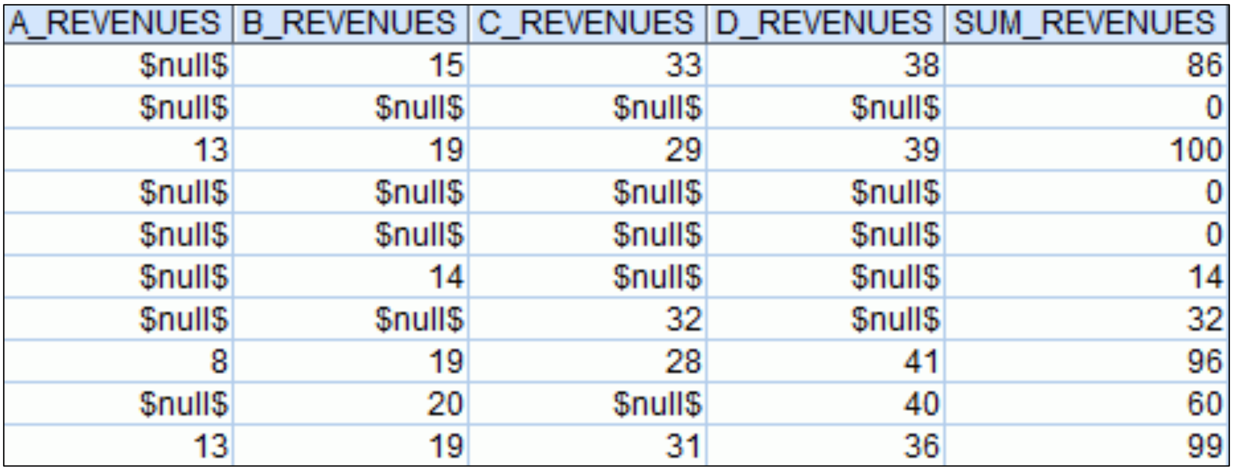
Formula:

```
sum_n (@FIELDS_BETWEEN(A_REVENUES, D_REVENUES))
```

Notice that the fields are not enclosed in square brackets when you use the @FIELDS_BETWEEN function.

7. Click **Preview**, and then move **SUM_REVENUES** next to **D_REVENUES**.

   A section of the results appear as follows:



A_REVENUES	B_REVENUES	C_REVENUES	D_REVENUES	SUM_REVENUES
$null$	15	33	38	86
$null$	$null$	$null$	$null$	0
13	19	29	39	100
$null$	$null$	$null$	$null$	0
$null$	$null$	$null$	$null$	0
$null$	14	$null$	$null$	14
$null$	$null$	32	$null$	32
8	19	28	41	96
$null$	20	$null$	40	60
13	19	31	36	99

   Notice that the sum_n function returns 0 when all source fields are undefined, whereas the mean_n function returned the undefined value. In the next task you will compute the sum conditionally and assign the undefined value to the sum when all source fields are undefined.

8. Close the **Preview** output window, and then close the **Derive** dialog box.

   Leave the stream open for the next task.

Task 4. Using missing values functions to derive fields.

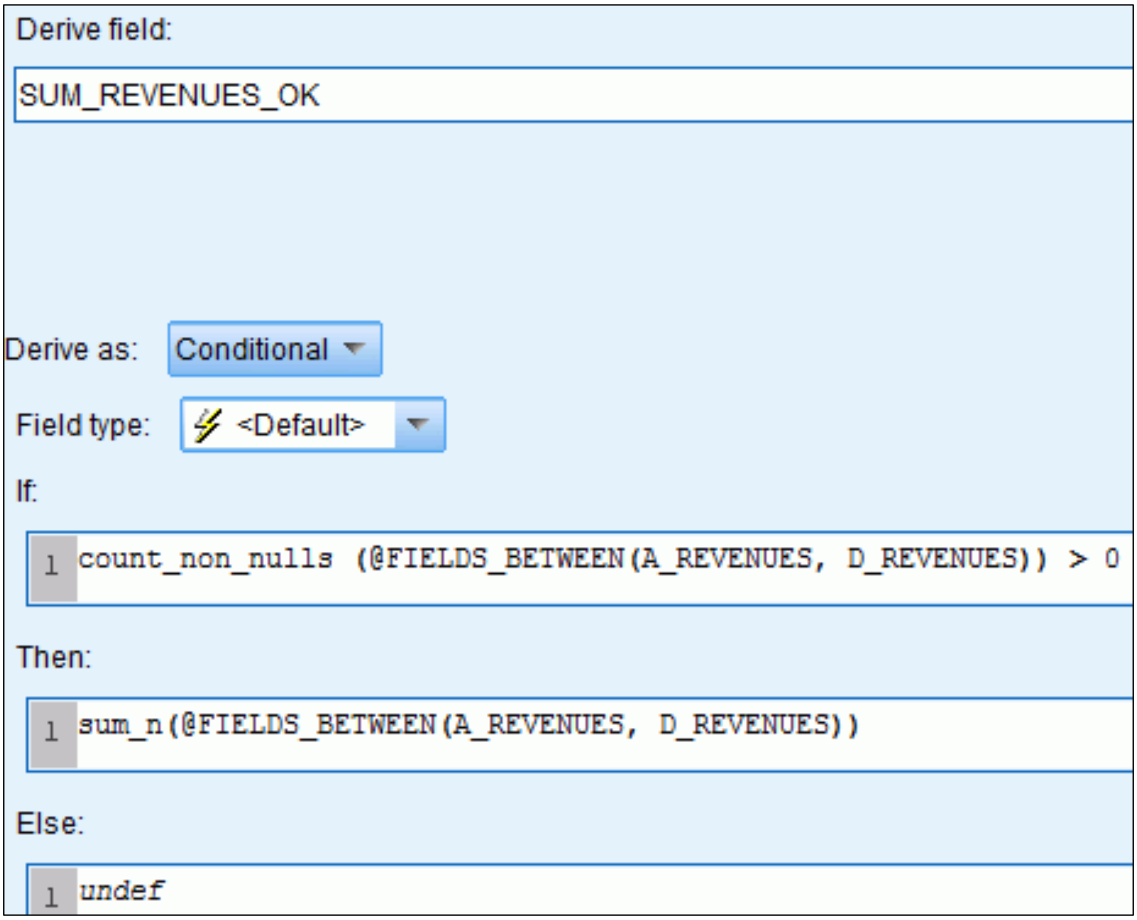
In this task you will compute the sum of A_REVENUES to D_REVENUES so that the result is undefined when all source fields are undefined. This means that the sum only has to be computed when the number of non-null values is greater than 0, for which the count_non_nulls function can be used. If the number of non-null values is not greater than 0 you will assign the undefined ($null$) value to the new field.

You will build from the stream that you created in the previous task.

1. Add a **Derive** node downstream from the **Derive** node named **SUM_REVENUES**.

2. Edit the **Derive** node, and then:
   - for **Derive field**, enter **SUM_REVENUES_OK**
   - for **Derive As**, select **Conditional**
   - for **If**, enter **count_non_nulls (@FIELDS_BETWEEN(A_REVENUES, D_REVENUES)) > 0**
   - for **Then**, enter **sum_n(@FIELDS_BETWEEN(A_REVENUES, D_REVENUES))**
   - for **Else**, enter **undef**

   A section of the results appear as follows:



Derive field: SUM_REVENUES_OK

Derive as: Conditional

Field type: <Default>

If:

```
count_non_nulls (@FIELDS_BETWEEN(A_REVENUES, D_REVENUES)) > 0
```

Then:

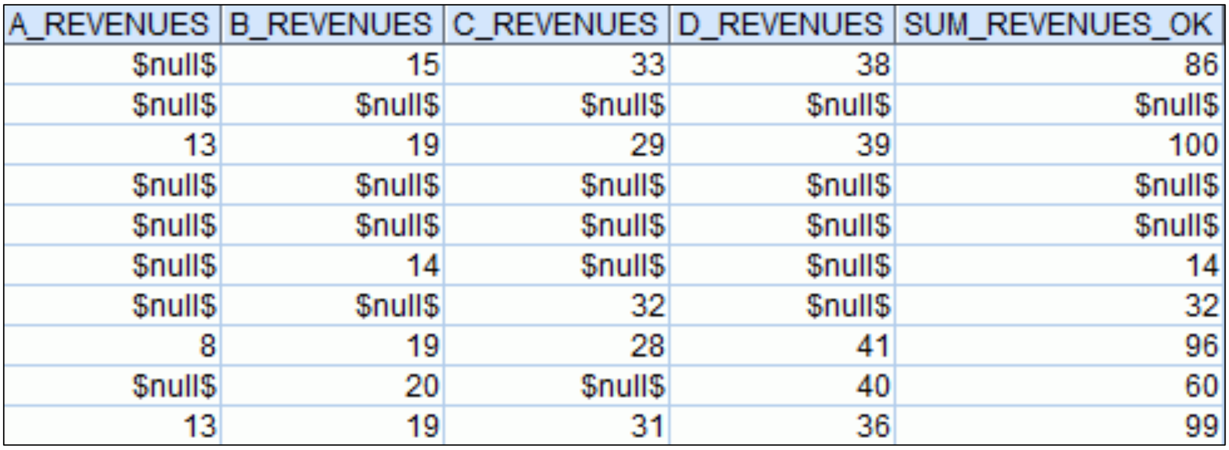
```
sum_n(@FIELDS_BETWEEN(A_REVENUES, D_REVENUES))
```

Else:

```
undef
```

3. Click **Preview**, and move **SUM_REVENUES_OK** next to **D_REVENUES**.

   A section of the results appear as follows:



A_REVENUES	B_REVENUES	C_REVENUES	D_REVENUES	SUM_REVENUES_OK
$null$	15	33	38	86
$null$	$null$	$null$	$null$	$null$
13	19	29	39	100
$null$	$null$	$null$	$null$	$null$
$null$	$null$	$null$	$null$	$null$
$null$	14	$null$	$null$	14
$null$	$null$	32	$null$	32
8	19	28	41	96
$null$	20	$null$	40	60
13	19	31	36	99

   The new field is undefined when all source fields are undefined, as required.

4. Close the **Preview** output window, and then close the **Derive** dialog box.

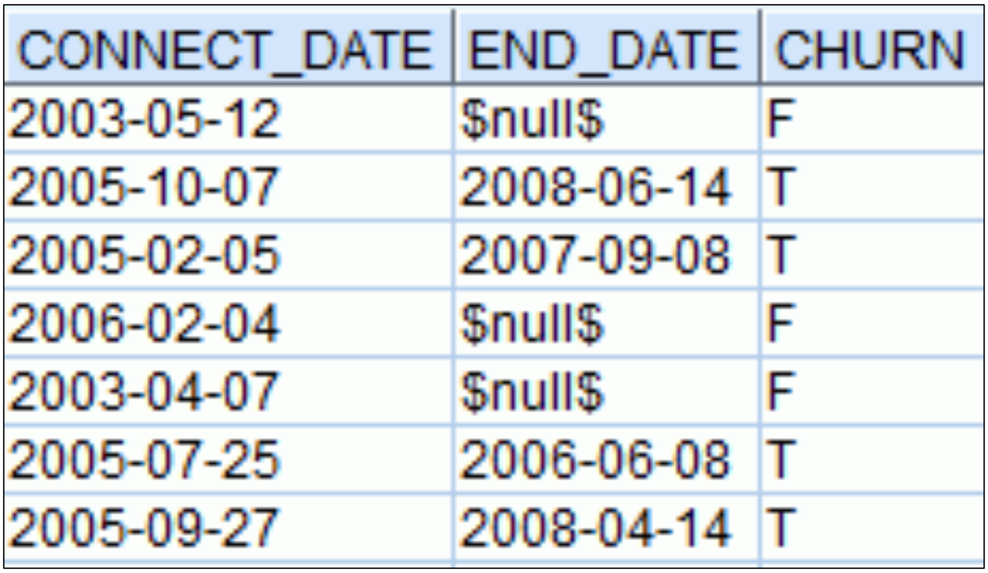
   As a last example of using missing value functions you will derive a field CHURN which flags whether the customer has ended his or her subscription. Notice that END_DATE is not undefined when the customer has churned.

5. Add a **Derive** node downstream from the **Derive** node named **SUM_REVENUES_OK**.

6. Edit the **Derive** node, and then:

   - for **Derive field**, enter **CHURN**
   - for **Derive As**, select **Flag**
   - for **True when**, enter **not (@NULL (END_DATE))**
   - click **Preview**, and then move **CHURN** next to **END_DATE**

A section of the results appear as follows:



CONNECT_DATE	END_DATE	CHURN
2003-05-12	$null$	F
2005-10-07	2008-06-14	T
2005-02-05	2007-09-08	T
2006-02-04	$null$	F
2003-04-07	$null$	F
2005-07-25	2006-06-08	T
2005-09-27	2008-04-14	T

The CHURN field is true for those having an end date, and false for those having no end date, as required.

7. Close the **Preview** output window, and then close the **Derive** dialog box.

This completes the demo for this module. You will find the solution results in **demo_using_functions_completed.str**, located in the **01-Using_Functions\Solutions** sub folder.

Results:
You have cleansed and enriched your dataset, so that better models can be built later.

Apply Your Knowledge

Use the questions in this section to test your knowledge of the course material.

Question 1: What does the expression **date_years_difference (DOB, datetime_date (1, 1, 2014))** compute? (The DOB field gives the date of birth.)

A. Age at JAN-01-2014.

B. Age at today's date.

C. It produces undefined ($null$) values because the order of arguments in the datetime_date function is incorrect.

D. It produces undefined ($null$) values because the order of two functions must be reversed.

Note: The following functions are used:

- **datetime_date (YEAR, MONTH, DAY)**. Returns the date value for the given YEAR, MONTH, and DAY.
- **date_years_difference (DATE1, DATE2)**. Returns the difference in years from DATE1 to DATE2.

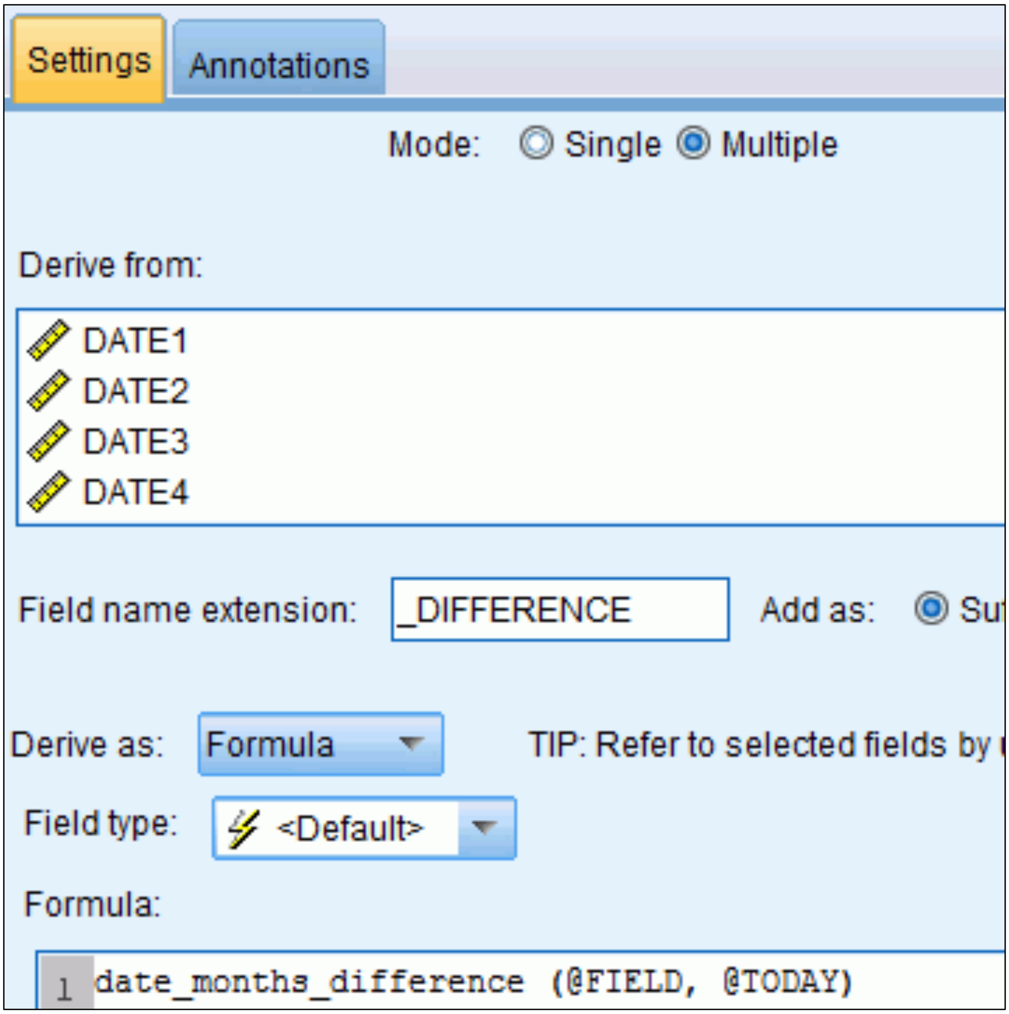
Question 2: Which of the following statements are correct? Refer to the figure below.

A. Four fields are derived, giving the difference in months between DATE1 and today's date, DATE2 and today's date, DATE3 and today's date, and DATE4 and today's date.

B. The term @FIELD in the expression references all fields listed in the Derive from: area.

C. The new fields will be named DATE1_DIFFERENCE, DATE2_DIFFERENCE, DATE3_DIFFERENCE, and DATE4_DIFFERENCE.

D. The expression will issue an error message, because the arguments of the function @TODAY and @FIELD and must be in lower case.



Mode: ○ Single ● Multiple

Derive from:

DATE1
DATE2
DATE3
DATE4

Field name extension: _DIFFERENCE

Add as: ● Su

Derive as: Formula

TIP: Refer to selected fields by

Field type: <Default>

Formula:

```
date_months_difference (@FIELD, @TODAY)
```

This material is meant for IBM Academic Initiative purposes.

Question 3: A dataset comprises a field E_MAIL_ADDRESS. Which of the following expressions derive a flag field which returns true if the E_MAIL_ADDRESS contains at least one at sign ("@")?

A. locchar(`@`, 1, E_MAIL_ADDRESS) > 0

B. count_substring (E_MAIL_ADDRESS, "@") > 0

C. issubstring("@",E_MAIL_ADDRESS) > 0

D. E_MAIL_ADDRESS matches "*@*"

Note: The following functions are used:

- **locchar(CHAR,N, STRING)**. Returns the subscript at which CHAR is found in STRING. Searches from Nth character of STRING forward.
- **count_substring (STRING, SUBSTRING)**. Returns the number of instances of SUBSTRING that can be found in STRING.
- **issubstring (SUBSTRING, STRING)**. Searches STRING for SUBSTRING. If found, returns the starting subscript.
- **STRING1 matches STRING2**. Returns true if STRING1 matches the pattern defined by STRING2. "?" matches 1 character, "*" matches 0 or more characters.

Question 4: Suppose that a field named BIRTHDATE has storage string, and is formatted as MMYYYYDD (for example 01201104 represents January 4th, 2011). Which expression converts this string field to a date field?

A. datetime_date(substring (to_integer (BIRTHDATE)))

B. datetime_date(substring (5, 4, BIRTHDATE), substring (1, 2, BIRTHDATE), substring (7, 2, BIRTHDATE))

C. datetime_date(to_integer (substring (3, 4, BIRTHDATE)), to_integer (substring (1, 2, BIRTHDATE)), to_integer (substring (7, 2, BIRTHDATE)))

Note: The following functions are used:

- **datetime_date (YEAR, MONTH, DAY)**. Returns the date value for the given YEAR, MONTH, and DAY. The arguments must be integers.
- **substring (N, LENGTH, STRING)**. Returns a string, which consists of LEN characters of STRING, starting from the character at N.
- **to_integer (ITEM)**. Converts ITEM to an integer. ITEM must be a string, or a number.

Question 5: Is the following statement true or false? The functions @BLANK and @NULL will always return the same outcome.

A. True

B. False

Question 6: Is the following statement true or false? The mean_n will return 0 when it is applied to a list of fields X1 to X4, and all these fields are undefined.

A. True

B. False

Question 7: Given is an integer field X and a real field Y. What is the storage of the field Z, which is derived as X >< Y?

A. Integer

B. Real

C. String

D. Date

Answers to questions:

Answer 1: C. The expression will yield undefined ($null$) values. The datetime_date function needs arguments year, month, and day, in that order. Thus, the correct specification is datetime_date (2014, 1, 1)) rather than datetime_date (1, 1, 2014)).

Answer 2: A, B, C. The dialog box shows a multiple derive where four fields are derived. The formula computes the difference in months between today and the respective source fields, which are referenced by @FIELD. The new field name will be the concatenation of the original field name and _DIFFERENCE, as suffix. The functions @TODAY and @FIELD must be in upper case.

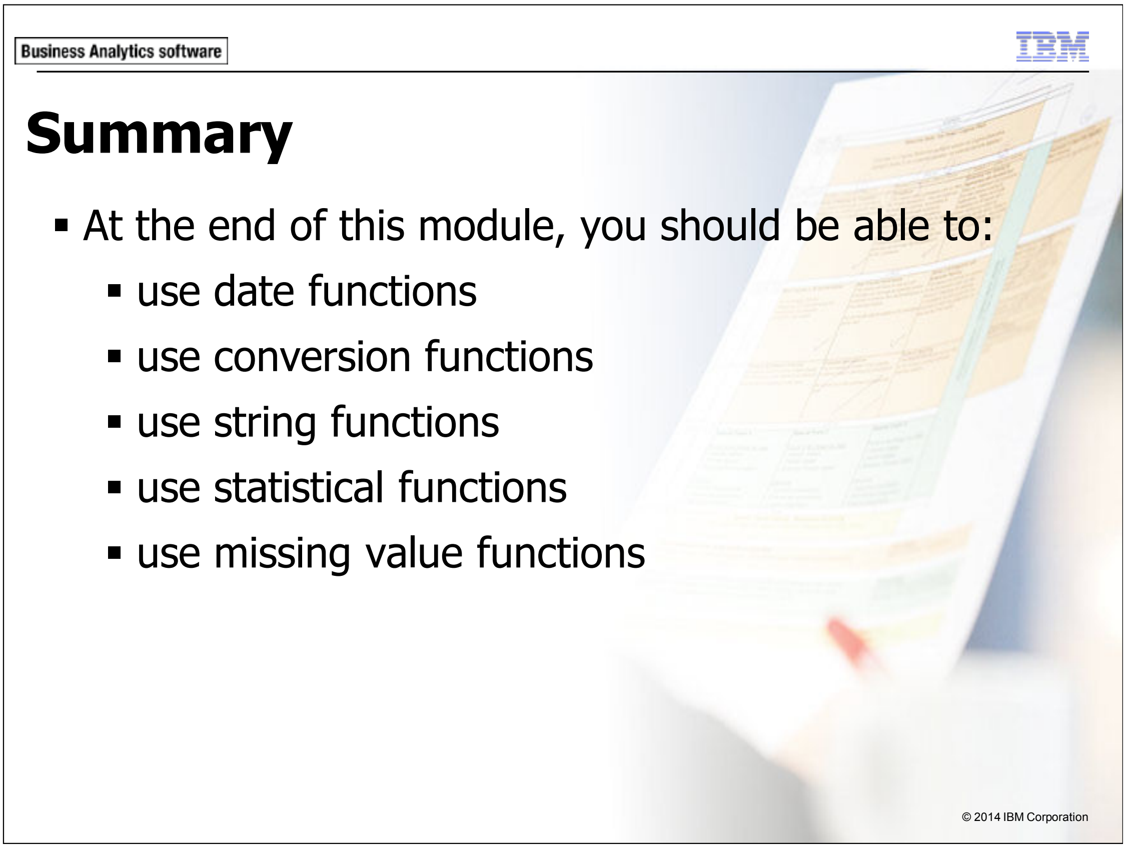
Answer 3: A, B, C, D. All specifications are correct. Notice that the locchar function needs back quotes, while the other functions need double quotes.

Answer 4: C. Year, month and day have to be extracted, as integers. For example, to_integer(substring (1, 2, BIRTHDATE) extracts the month, as an integer. The datetime_date function is then used to convert the separate parts to a date format.

Answer 5: B. False. When undefined values are not declared as blanks, @BLANK will return false for an undefined ($null$) value, while @NULL will return true.

Answer 6: B. False. The mean_n function will return the undefined value when all the source fields are undefined.

Answer 7: C. The >< function will always return a string field, no matter the storage of the source fields.

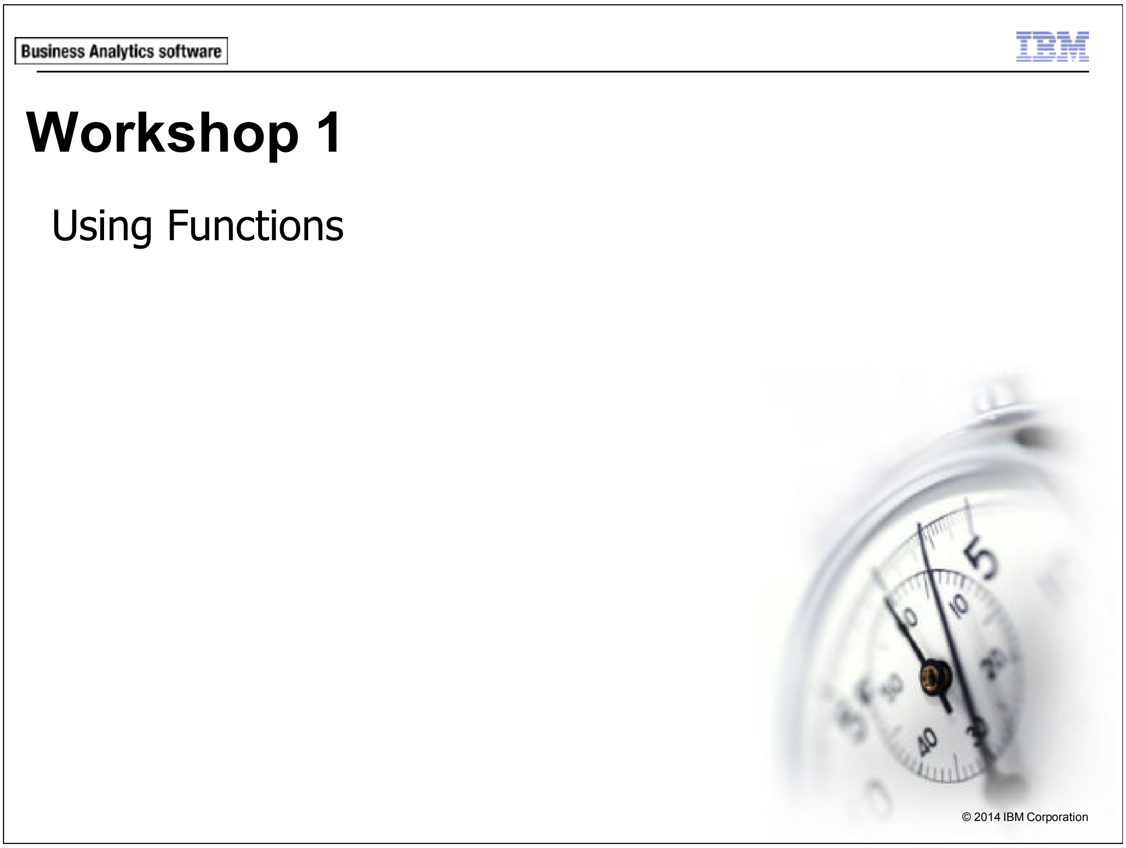


Business Analytics software

IBM

Summary

- At the end of this module, you should be able to:
  - use date functions
  - use conversion functions
  - use string functions
  - use statistical functions
  - use missing value functions



Business Analytics software

Workshop 1

Using Functions

© 2014 IBM Corporation

The following (synthetic) file is used in this workshop:

- **customers_and_holidays.dat**: A text file storing data on holiday destinations for 411 customers of a travel agency. The file is located in **C:\Train\0A055**.

Before you begin with the workshop, ensure that:

- You have started MODELER.
- You have set MODELER's working folder. (In MODELER, select **File\Set Directory**. When you are working in an IBM operated environment, browse to the folder **C:\Train\0A055** and then click **Set** to select it. When you are not working in an IBM operated environment, ask your trainer for the folder name.)

Workshop 1: Using Functions

In this workshop you will work with data on customers and their holiday destinations. To answer questions such as "What is the mean age of the customers?", "What was the most popular month to travel?", "What was the most popular destination?" and "What was the mean amount of money spent?", you will derive new fields using various functions.Use a **Var. File** node to import the data from **customers_and_holidays.dat** (a text file), add a **Type** node downstream from the data source, edit the **Type** node, declare **-999** as blank for **DISTANCE_TO_BEACH**, and then click **Read Values** to instantiate the data.

- Create a new field, named **AGE**, giving the customer's age (in years, as a real number) on the date that he or she traveled.

  Note: The DOB field stores the date of birth; the TRAVDATE field stores the date that the customer travelled.

  What is the mean age?

- Create two fields: **YEAR_DOB** and **YEAR_TRAVDATE**, using only a single Derive node.

- Derive a field **COUNTRY_OK**, which equals COUNTRY, but starts with an upper case letter, followed by lower case letters(for example: "SPAIN" should become "Spain").

  Check your results with a Matrix node.

  What was the most popular country for holiday?

- Create a new field **CODE_HOLIDAY** which returns the integer from the HOLCODE field. For example, the result for HOLCODE "CAF3105" should be CODE_HOLIDAY 3105 (as an integer).

  Note: You may assume that HOLCODE is comprised of three letters, followed by the code.

  Check your results with a Matrix node (Output palette).

- Compute a new field named **SPENT_TOTAL**, which is the total amount spent on SPENT_TRAVEL, SPENT_HOTEL and SPENT_LEISURE. Note, however, that the total should only be computed when there are at least two non-null values on these three fields.

  What was the highest total amount spent?

- The DISTANCE_TO_BEACH field stores the distance to the beaches in miles. Create a new field named **DISTANCE_KM** that returns the distance to the beach in kilometers (1 mile = 1.61 kilometers).

  What is the mean distance in kilometers?

  Note: When the distance to the beach is unknown, the value for DISTANCE_TO_BEACH equals -999 (which was declared as blank in the first task).

For more information about where to work and the workshop results, refer to the Task and Results section that follows. If you need more information to complete a task, refer to earlier demos for detailed steps.

Workshop 1: Tasks and Results

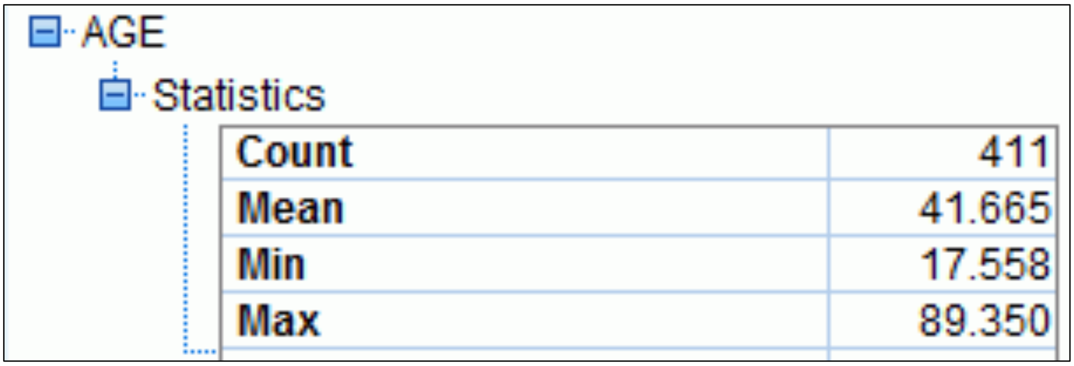
Task 1. Import and instantiate the data.

- Use a **Var. File** node to import the data from **customers_and_holidays.dat**.
- Add a **Type** node downstream from the **Var. File** node.
- Edit the **Type** node, and then:
  - declare **-999** as blank for **DISTANCE_TO_BEACH**
  - click **Read Values** to instantiate the data
  - close the **Type** node

Task 2. Compute an AGE field.

- Add a **Derive** node downstream from the **Type** node.
- Edit the **Derive** node, and then:
  - for **Derive field**, enter **AGE**
  - for **Formula**, enter **date_years_difference(DOB, TRAVDATE)**
  - close the **Derive** dialog box
- Add a **Statistics** node downstream from the **Derive** node, select **AGE**, and then run the **Statistics** node.

  A section of the results appear as follows:



AGE

Statistics

Count	411
Mean	41.665
Min	17.558
Max	89.350

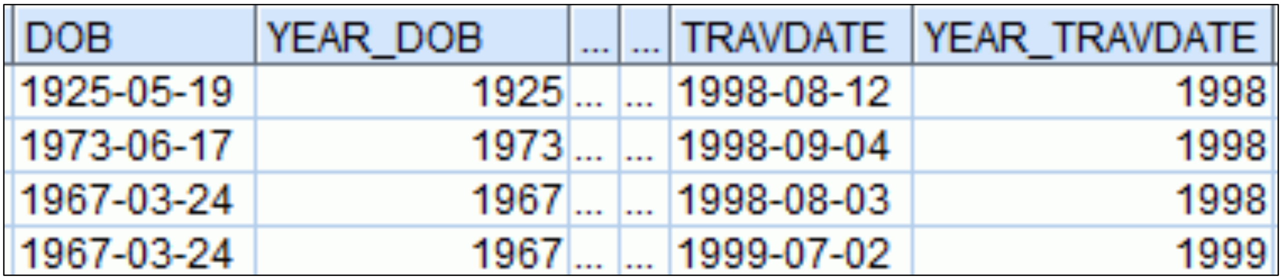
The mean age is 41.665.

This material is meant for IBM Academic Initiative purposes.

Task 3. Create fields that hold the year.

- Add a **Derive** node downstream from the **Derive** node named **AGE**.
- Edit the **Derive** node, and then:
  - for **Mode**, select **Multiple**
  - for **Derive from**, select **DOB** and **TRAVDATE**
  - for **Field name extension**, type **YEAR_**
  - for **Add as**, select **Prefix**
  - for **Formula**, type **datetime_year(@FIELD)**
  - close the **Derive** dialog box
- Run a **Table** node downstream from the **Derive** node to check your results. Rearrange the fields to better view the results.

  A section of the results appear as follows:



DOB	YEAR_DOB	...	...	TRAVDATE	YEAR_TRAVDATE
1925-05-19	1925	...	...	1998-08-12	1998
1973-06-17	1973	...	...	1998-09-04	1998
1967-03-24	1967	...	...	1998-08-03	1998
1967-03-24	1967	...	...	1999-07-02	1999

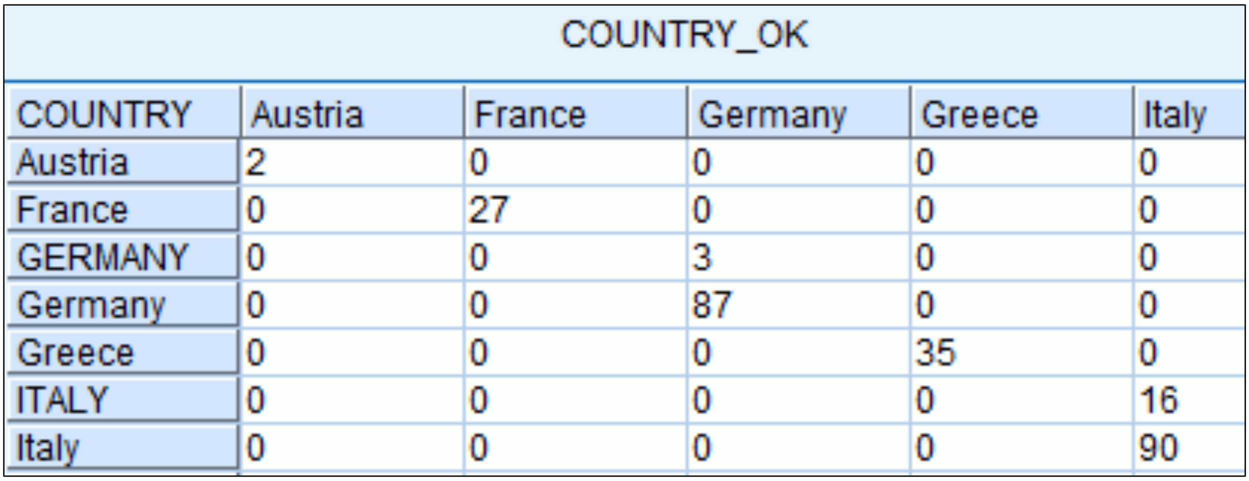
  The YEAR_DOB stores the year of birth, the field YEAR_TRAVDATE stores the year that one travelled, as required.

Task 4. Cleanse the values for COUNTRY.

- Add a **Derive** node downstream from the **Derive** node named **YEAR_**.
- Edit the **Derive** node, and then:
  - for **Derive field**, enter **COUNTRY_OK**
  - for **Formula**, enter **lowertoupper (startstring(1,COUNTRY)) >< uppertolower (allbutfirst(1,COUNTRY))**
  - close the **Derive** dialog box

- Add a **Matrix** node downstream from the **Derive** node.
- Edit the **Matrix** node, and then:
  - for **Rows** select **COUNTRY**
  - for **Columns** select **COUNTRY_OK**
  - run the **Matrix** node

A section of the results appear as follows:

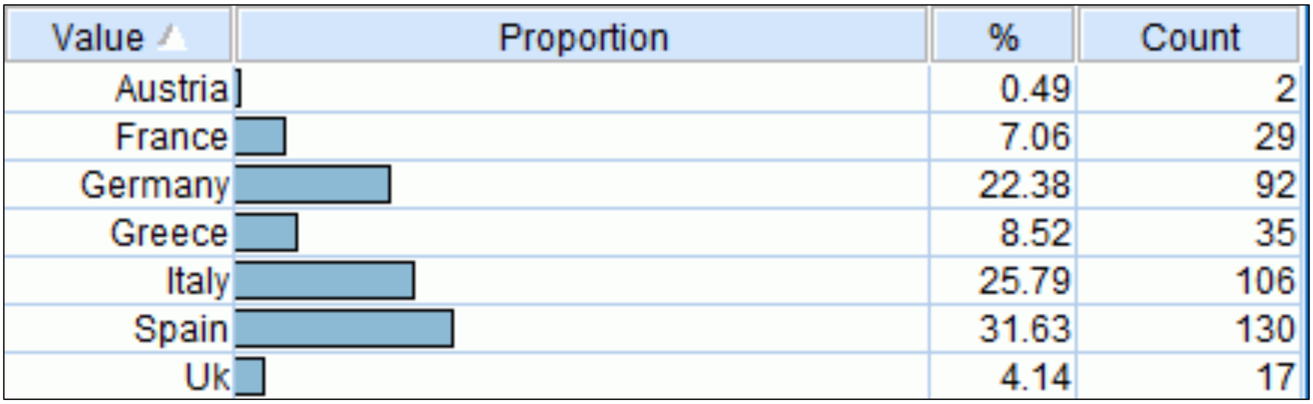


	COUNTRY_OK				
COUNTRY	Austria	France	Germany	Greece	Italy
Austria	2	0	0	0	0
France	0	27	0	0	0
GERMANY	0	0	3	0	0
Germany	0	0	87	0	0
Greece	0	0	0	35	0
ITALY	0	0	0	0	16
Italy	0	0	0	0	90

The countries names are as required.

- Add a **Distribution** node downstream from the **Derive** node.
- Edit the **Distribution**, and then:
  - select **COUNTRY_OK**
  - run the **Distribution** node

A section of the results appear as follows:



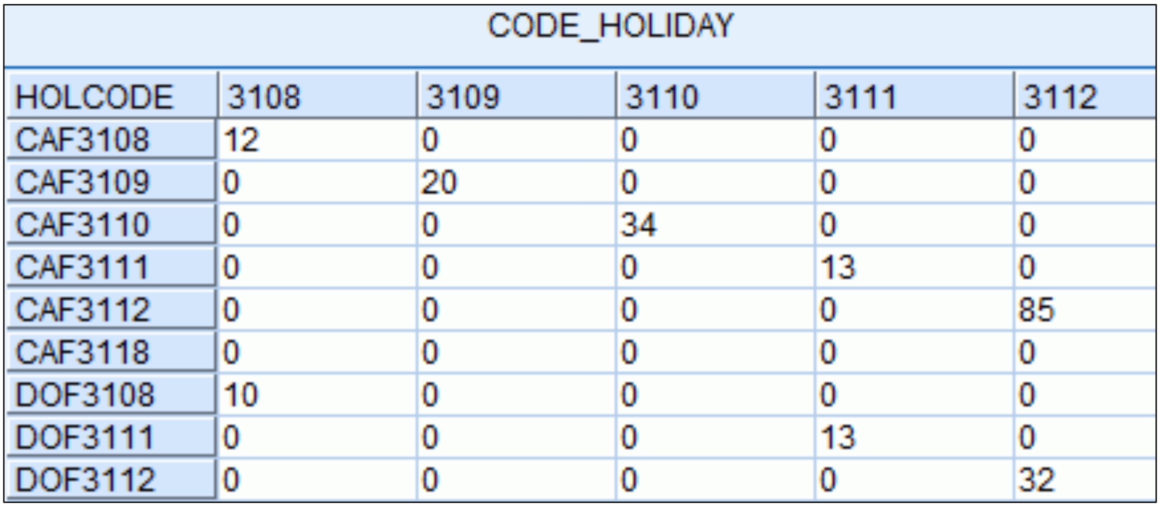
Value	Proportion	%	Count
Austria		0.49	2
France		7.06	29
Germany		22.38	92
Greece		8.52	35
Italy		25.79	106
Spain		31.63	130
Uk		4.14	17

Spain is the most popular holiday destination.

Task 5. Extract a number out of the string HOLCODE.

- Add a **Derive** node downstream from the **Derive** node named **COUNTRY_OK**.
- Edit the **Derive** node, and then:
    - for **Derive field**, enter **CODE_HOLIDAY**
    - for **Formula**, enter **to_integer (allbutfirst (3, HOLCODE))**
    - close the **Derive** dialog box
- Add a **Matrix** node downstream from the **Derive** node.
- Edit the **Matrix** node, and then:
    - for **Rows** select **HOLCODE**
    - for **Columns** select **CODE_HOLIDAY**
    - run the **Matrix** node

A section of the results appear as follows:



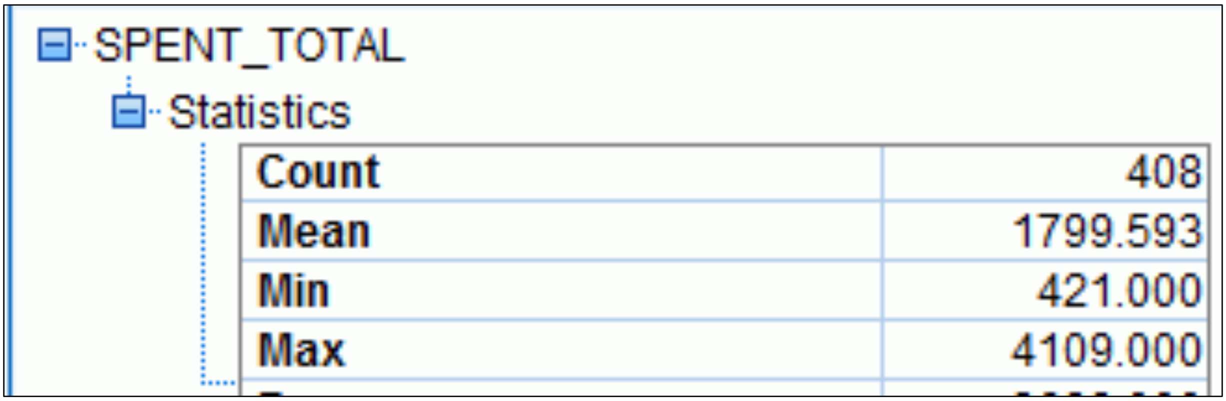
	CODE_HOLIDAY				
HOLCODE	3108	3109	3110	3111	3112
CAF3108	12	0	0	0	0
CAF3109	0	20	0	0	0
CAF3110	0	0	34	0	0
CAF3111	0	0	0	13	0
CAF3112	0	0	0	0	85
CAF3118	0	0	0	0	0
DOF3108	10	0	0	0	0
DOF3111	0	0	0	13	0
DOF3112	0	0	0	0	32

The results are as required.

Task 6. Conditionally compute the sum over a series of fields.

- Add a **Derive** node downstream from the **Derive** node named **CODE_HOLIDAY**.
- Edit the **Derive** node, and then:
    - for **Derive field**, enter **SPENT_TOTAL**
    - for **Derive as**, select **Conditional**
    - for **If**, enter **count_non_nulls ([SPENT_TRAVEL SPENT_HOTEL SPENT_LEISURE])>= 2**
    - for **Then**, enter **sum_n ([SPENT_TRAVEL SPENT_HOTEL SPENT_LEISURE])**
    - for **Else**, enter **undef**
    - close the **Derive** dialog box
- Add a **Statistics** node downstream from the **Derive** node.
- Edit the **Statistics** node, and then:
    - select **SPENT_TOTAL**
    - run the **Statistics** node

A section of the results appear as follows:



SPENT_TOTAL

Statistics

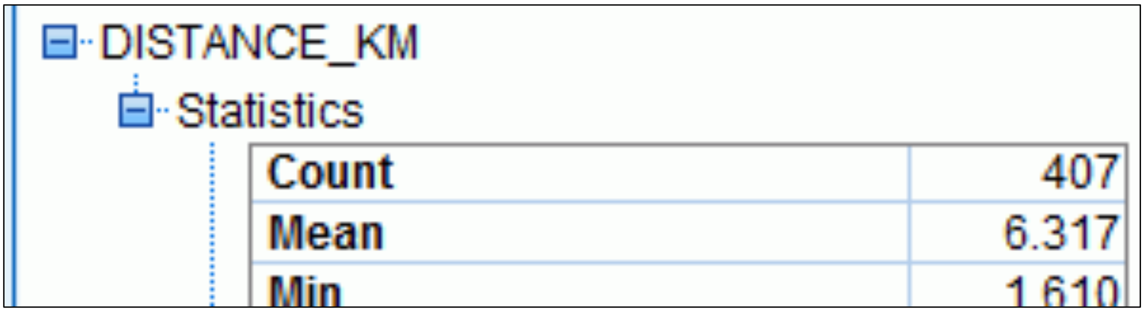
Count	408
Mean	1799.593
Min	421.000
Max	4109.000

The highest amount spent is 4109.

Task 7. Derive a field taking blank values into account.

- Add a **Derive** node downstream from the **Derive** node named **SPENT_TOTAL**.
- Edit the **Derive** node, and then:
  - for **Derive field**, enter **DISTANCE_KM**
  - for **Derive as**, select **Conditional**
  - for **If**, enter **not (@BLANK (DISTANCE_TO_BEACH))**
  - for **Then**, enter **DISTANCE_TO_BEACH * 1.61**
  - for **Else**, enter **undef**
  - close the **Derive** dialog box
- Add a **Statistics** node downstream from the **Derive** node.
- Edit the **Statistics** node, and then:
  - select **DISTANCE_KM**
  - run the **Statistics** node

A section of the results appear as follows:



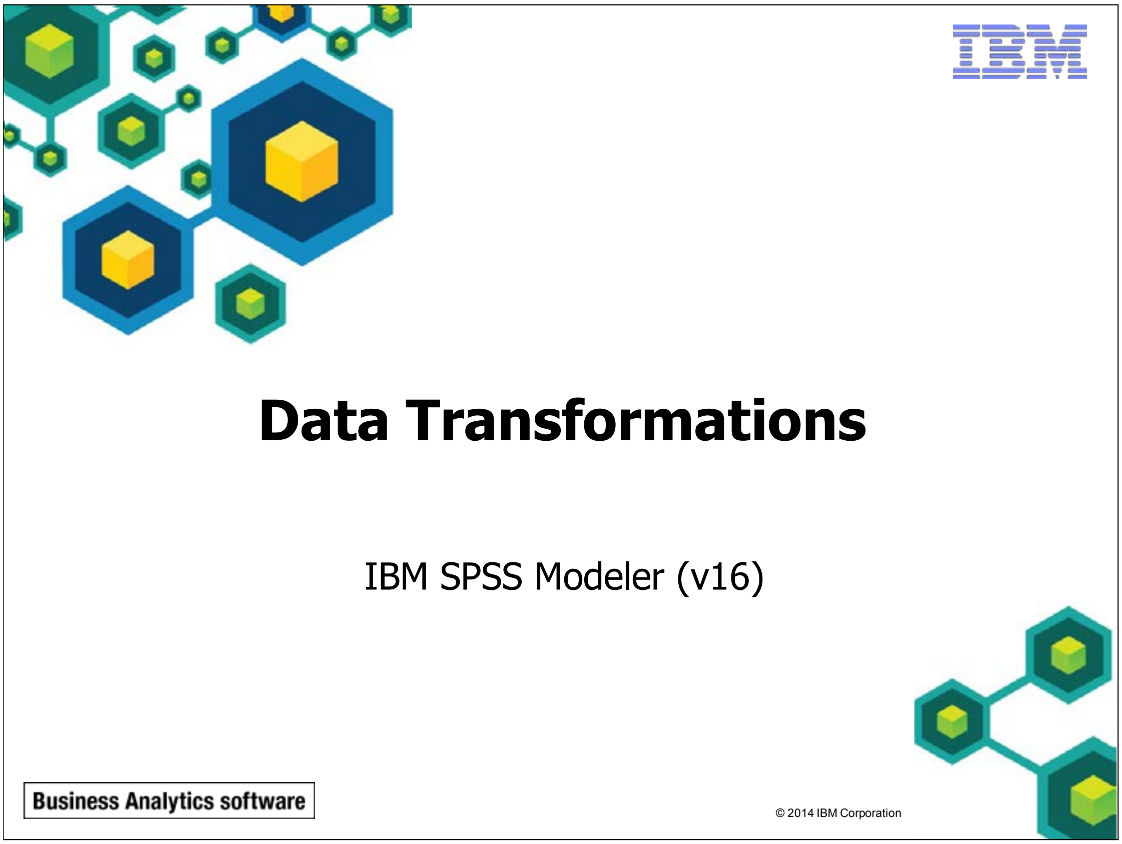
DISTANCE_KM

Statistics

Count	407
Mean	6.317
Min	1.610

The mean distance in kilometers is 6.317. Notice, that there are 407 records in computing the statistics (the total number of records in the dataset is 411). Four records were assigned the undefined ($null$) value because they had the blank value -999 on the source field DISTANCE_TO_BEACH.

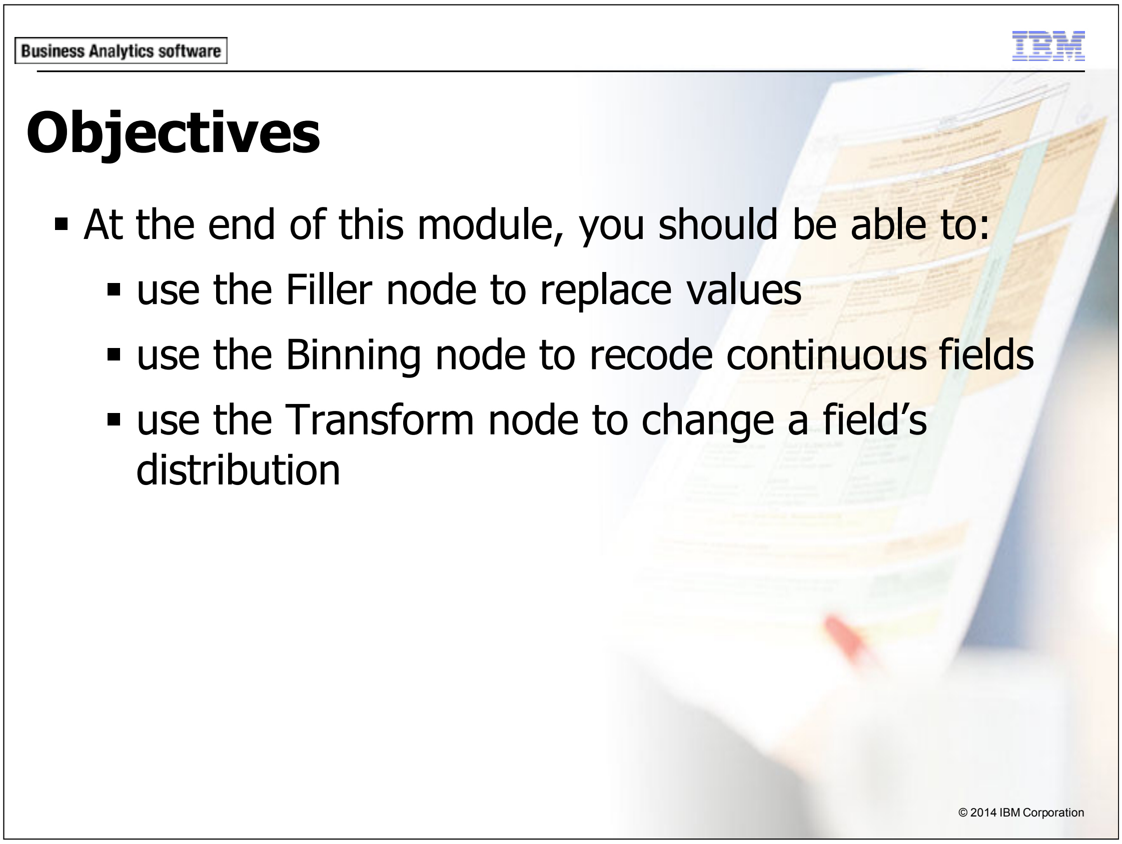
Note: The stream **workshop_using_functions_completed**, located in the **01-Using_Functions\Solutions** sub folder, provides a solution to the workshop exercises.



Data Transformations

IBM SPSS Modeler (v16)

Business Analytics software



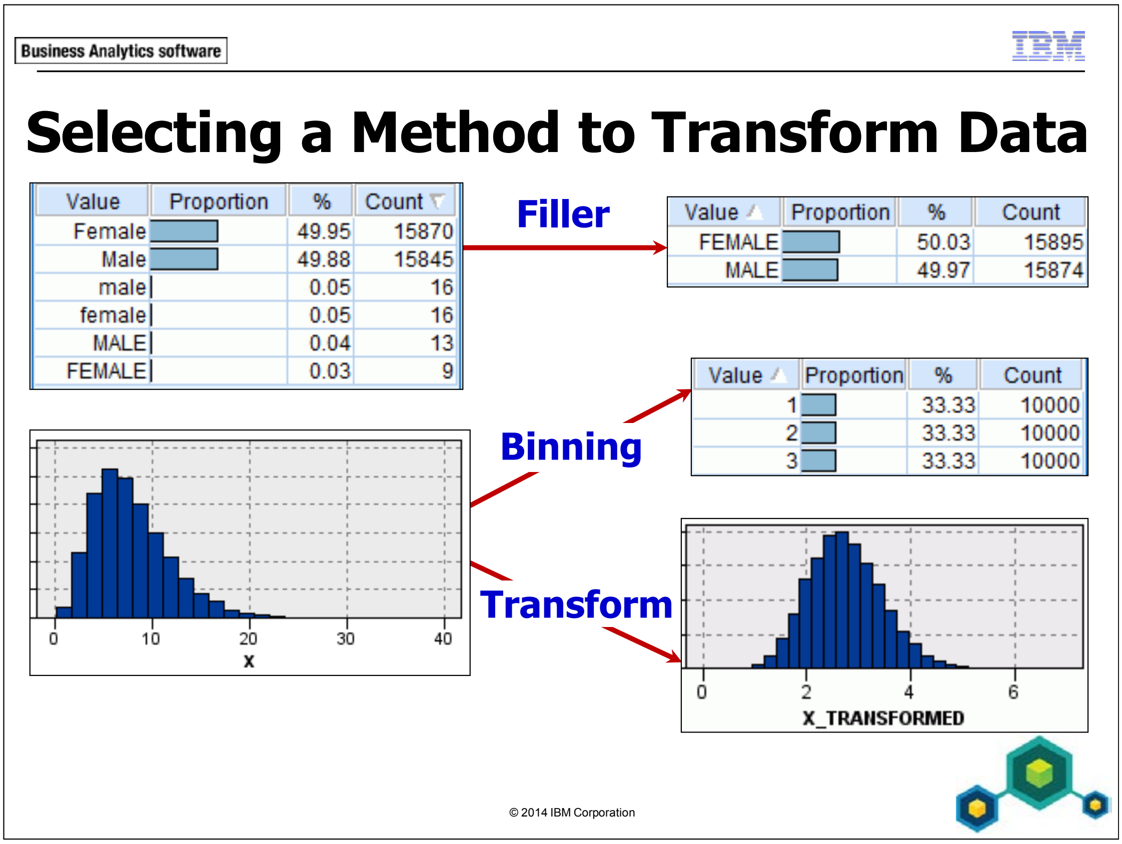
Business Analytics software

Objectives

- At the end of this module, you should be able to:
  - use the Filler node to replace values
  - use the Binning node to recode continuous fields
  - use the Transform node to change a field's distribution

Before reviewing this module you should be familiar with:

- working with MODELER (streams, nodes, SuperNodes, palettes)
- importing data (Var. File node)
- defining measurement levels, roles, blanks, and instantiating data (Type node)
- selecting fields and records (Filter node, Select node)
- CLEM and the Expression Builder
- deriving fields (Derive node)
- date and time-, conversion-, string-, statistical- and missing value functions
- examining the data with graphs (Distribution node, Histogram node) and tables (Table node, Data Audit node, Statistics node, Matrix node, Means node)



Business Analytics software

Selecting a Method to Transform Data

Value	Proportion	%	Count
Female	▬	49.95	15870
Male	▬	49.88	15845
male	|	0.05	16
female	|	0.05	16
MALE	|	0.04	13
FEMALE	|	0.03	9

Filler

Value	Proportion	%	Count
FEMALE	▬	50.03	15895
MALE	▬	49.97	15874

Binning

Value	Proportion	%	Count
1	▬	33.33	10000
2	▬	33.33	10000
3	▬	33.33	10000

Transform

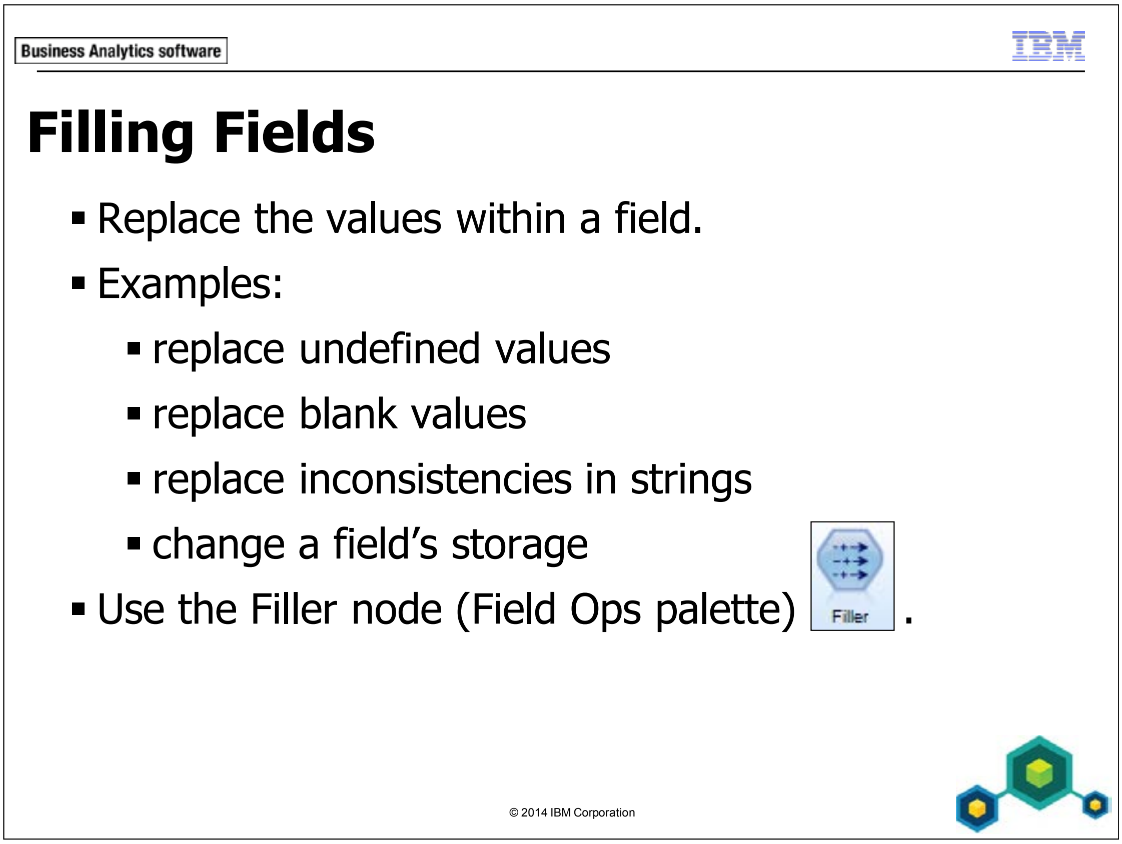
© 2014 IBM Corporation

Preparing data for modeling often involves cleaning existing fields, or deriving new ones. In the *Introduction to IBM SPSS Modeler and Data Mining* course two such field operations were discussed, Derive and Reclassify. In this module further nodes are introduced to modify fields, giving you more options to prepare your data for modeling.

In the first place, whereas Derive always creates a new field, it is often more efficient to replace the values in the field itself. The Filler node offers this capability.

Secondly, although both Derive and Reclassify can be used to recode a field into a number of categories, there is often the need to do this more efficiently. In particular, a data driven recode is required where the thresholds come out of the data itself. The Binning node provides this functionality.

Thirdly, for modeling purposes, it may be needed to transform a field so that its distribution conforms to the assumptions of the modeling technique. A field's distribution can be explored and, if desired, changed with the Transform node.



Business Analytics software

Filling Fields

- Replace the values within a field.
- Examples:
  - replace undefined values
  - replace blank values
  - replace inconsistencies in strings
  - change a field's storage
- Use the Filler node (Field Ops palette).

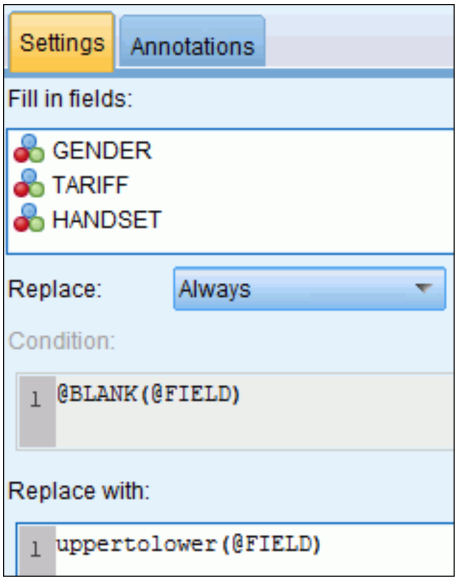
Sometimes there is a need to replace a field's values. Examples are:

- Replace the undefined ($null$) value with a valid value. For example, set the undefined ($null$) value for a flag field to F.
- Replace user-defined blanks with the $null$ value. For example, to prevent that user-defined blanks are included in the calculation of aggregate statistics you can replace the user-defined blanks with the undefined ($null$) value.
- Remove inconsistencies in string values by replacing them with lower case or upper case letters. For example, if gender stores values FEMALE, female, Female, MALE, Male, male, replace the values with FEMALE and MALE.
- Change the storage of a field. For example, when a field age has values 21.00, 22.00, and so forth, convert the field to integer values 21, 22, and so forth.

The Filler node, located in the Field Ops palette, is used to replace field values and to change storage. The Filler node uses CLEM to specify the expressions.

Business Analytics software

Exploring the Filler Dialog Box



Fill in fields:

- GENDER
- TARIFF
- HANDSET

Replace: Always

Condition:

```
@BLANK(@FIELD)
```

Replace with:

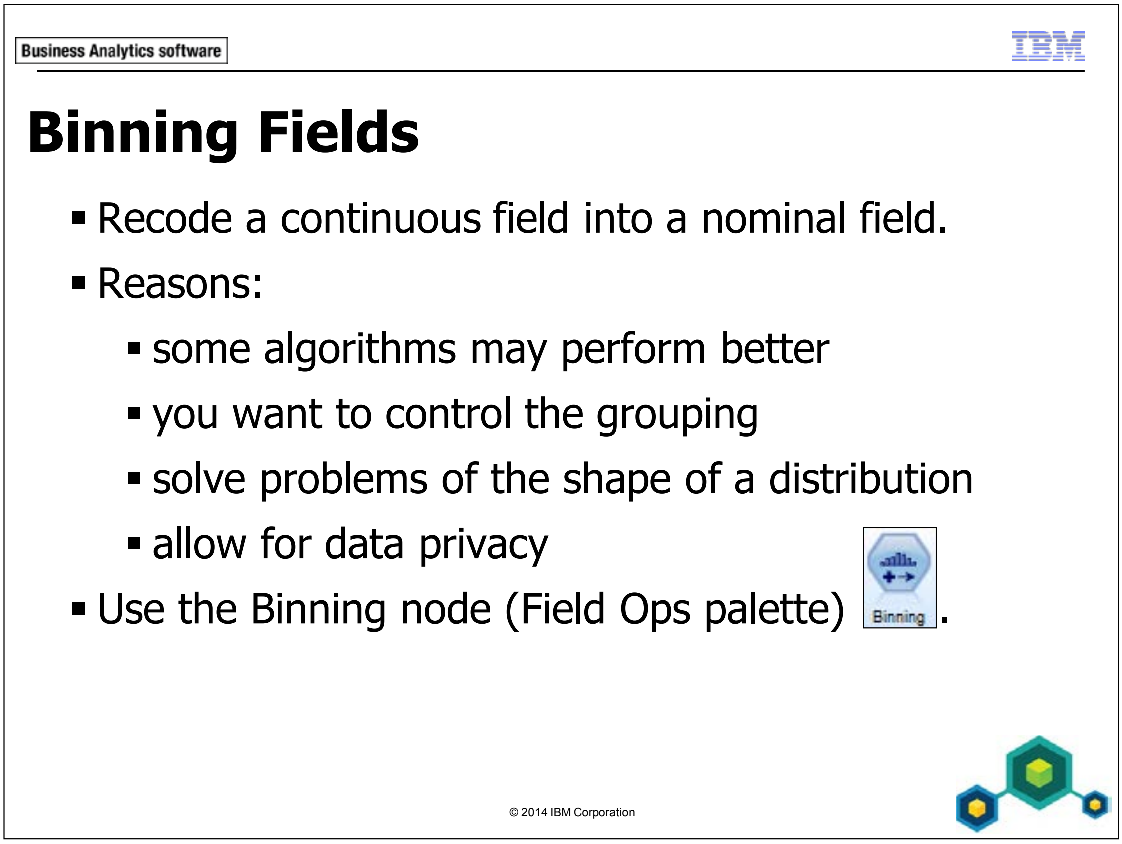
```
uppertolower(@FIELD)
```

In the Filler dialog box, in the Fill in fields area, specify the fields whose values must be replaced. For Replace, select one of the following options, determining when values will be replaced:

- when a specific condition is met
- always
- when the value is a blank
- when the value is undefined ($null$)
- when the value is a blank or undefined ($null$)

You specify the expression for the new value in the Replace with area. You can use @FIELD to reference the selected fields. If desired, use the Expression builder to enter the expression.

In the example shown here, the values for three (string) fields are converted to lower case.



Business Analytics software

Binning Fields

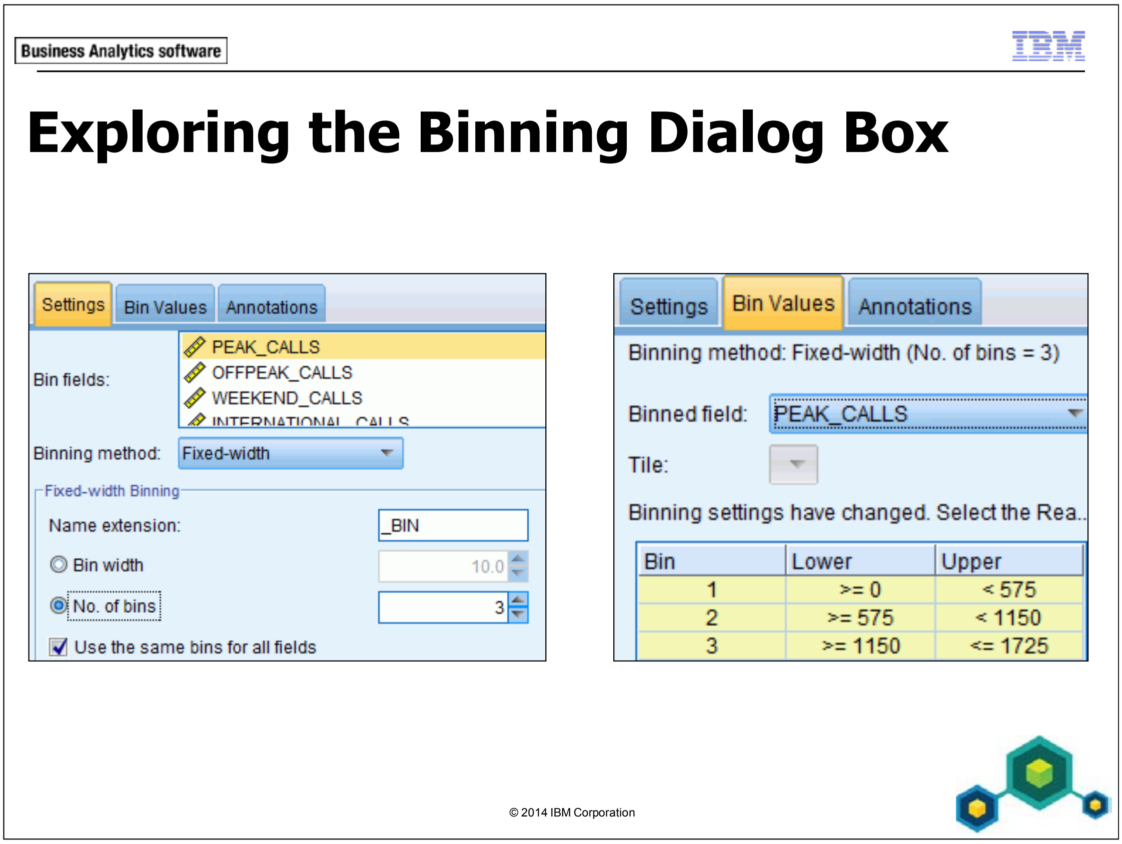
- Recode a continuous field into a nominal field.
- Reasons:
  - some algorithms may perform better
  - you want to control the grouping
  - solve problems of the shape of a distribution
  - allow for data privacy
- Use the Binning node (Field Ops palette) .

© 2014 IBM Corporation

Binning will automatically recode a continuous field into a nominal field. There are several reasons why you need to bin a continuous field:

- Some algorithms such as decision trees may perform better if an input has fewer categories.
- Even when an algorithm groups values of a continuous field, you may wish to control the grouping beforehand to create meaningful categories rather than rely on the grouping method of the algorithm.
- The effect of outliers can be reduced by binning, so that all the outliers and extreme cases are placed in one bin.
- Binning solves problems of the shape of a distribution since the continuous field is turned into a nominal field.
- Binning can allow for data privacy by reporting such things as salaries or bonuses in ranges rather than the actual values.

The Binning node is located in the Field Ops palette.



Exploring the Binning Dialog Box

Bin fields: PEAK_CALLS, OFFPEAK_CALLS, WEEKEND_CALLS, INTERNATIONAL_CALLS

Binning method: Fixed-width

Fixed-width Binning

Name extension: _BIN

○ Bin width 10.0

● No. of bins 3

☑ Use the same bins for all fields

Binning method: Fixed-width (No. of bins = 3)

Binned field: PEAK_CALLS

Tile:

Binning settings have changed. Select the Rea..

Bin	Lower	Upper
1	>= 0	< 575
2	>= 575	< 1150
3	>= 1150	<= 1725

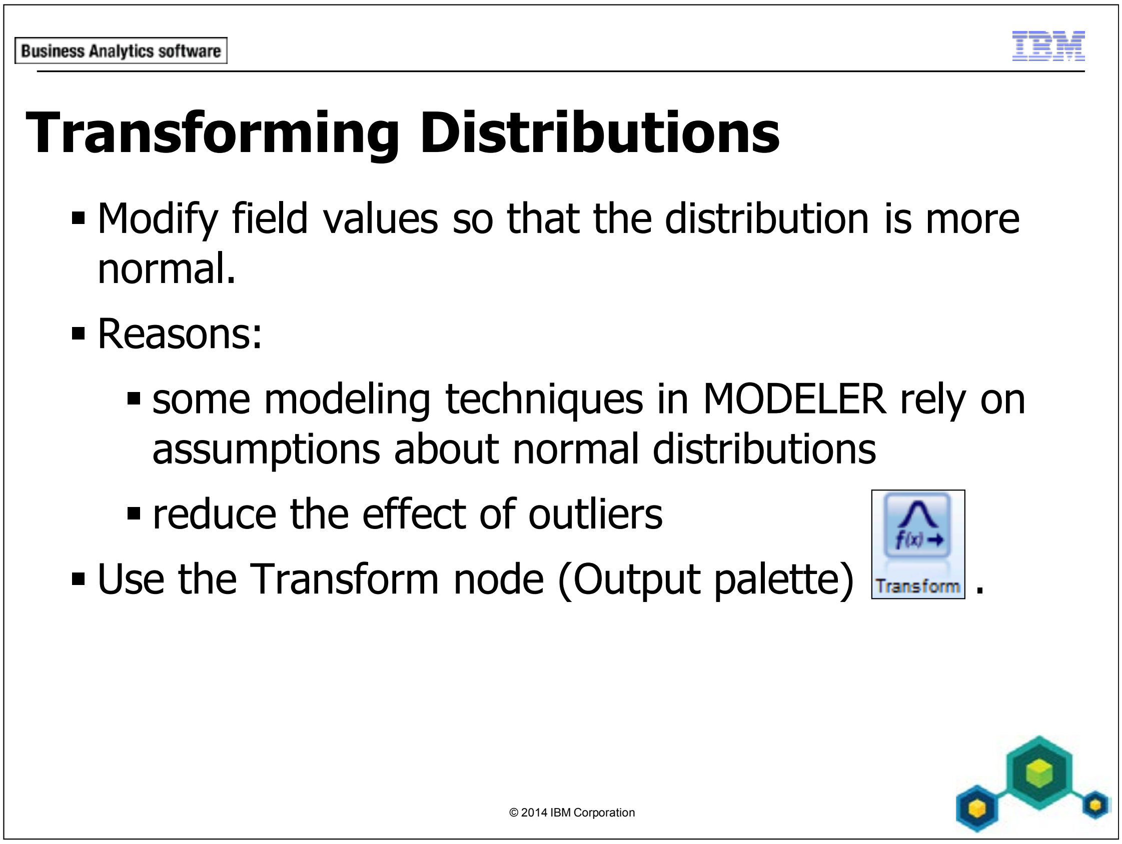
In the Binning dialog box, on the Settings tab, select the fields to bin, and select a binning method. MODELER provides the following methods to bin a field.

- Fixed-width binning: This option will group the values of the original field into equal ranges, such as age in groups of 20-29, 30-39, 40-49 and so forth.Tiles: This option creates groups based on percentiles. For example, the choice of quartiles will create four groups of equal numbers of records. Another option within this method is to create groups so that the sum of values in each group is approximately the same.
- Ranks: This option transforms a field into ranks, from 1 to N, where N is the number of distinct values in the original field.
- Mean and standard deviation: This option groups the values based on the number of standard deviations below and above the mean.
- Optimal binning: This method transforms a field based on a supervising field. The binning is done so that the binned field has the strongest relationship with the supervising field.

Which binning method you choose is a business decision. You can use more than one method and run the model with the respective fields that you created.

The lower part of the dialog box will reflect the method that is selected. On this slide, with the method Fixed-width selected, specify either the bin width or the number of bins.

There are two options for the thresholds that are used for binning. You can use the thresholds values that are already computed in a previous run, or you can recompute the threshold values when data pass through the Binning node. The Bin Values tab shows these thresholds and enables you to recompute the thresholds by enforcing a data pass.



Business Analytics software

Transforming Distributions

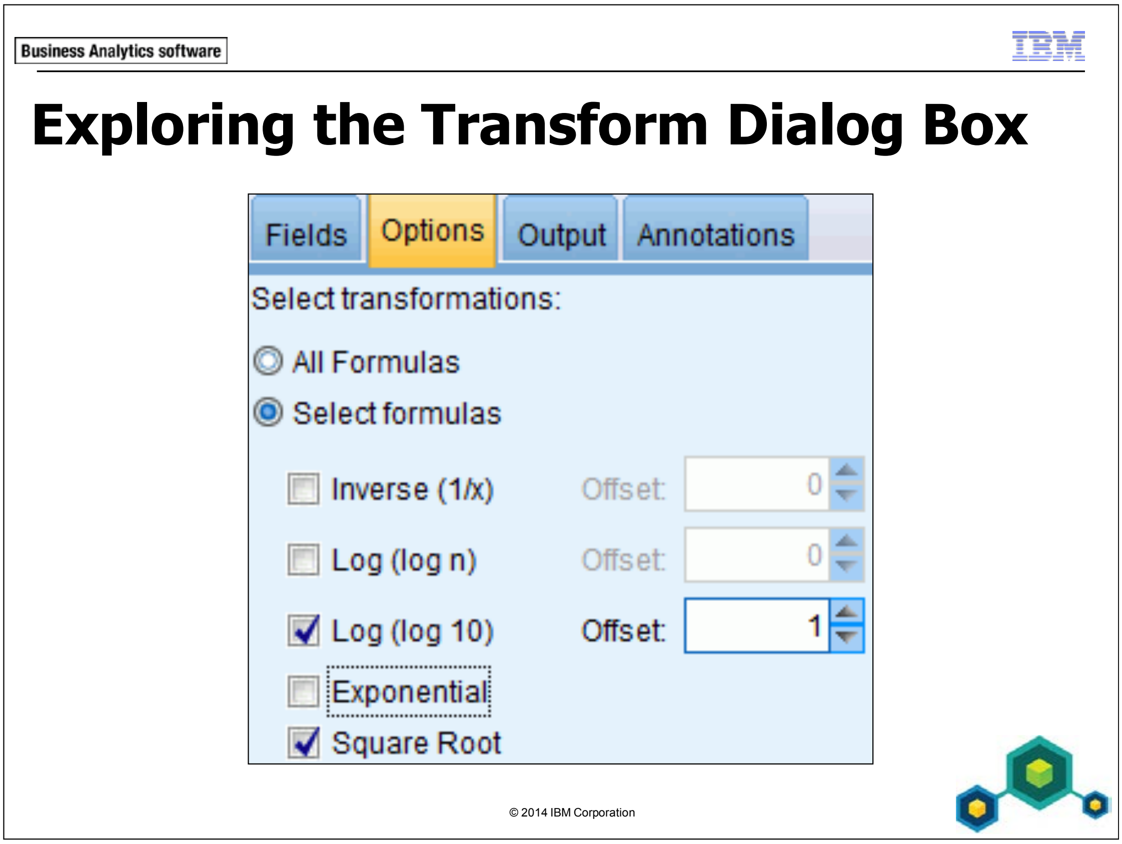
- Modify field values so that the distribution is more normal.
- Reasons:
  - some modeling techniques in MODELER rely on assumptions about normal distributions
  - reduce the effect of outliers
- Use the Transform node (Output palette) Transform .

When the distribution of a continuous field is skewed rather than normal (bell shaped), it can cause problems when you build models:

Several of the modeling techniques in MODELER that are based on traditional statistical theory function best with normal data, including regression, logistic regression, and discriminant analysis. These techniques rely on assumptions about normal distributions of data that may not often be true for real world data.

Also, skewed fields will typically have outliers, because the skewness, in part, is caused by the outlying values. Reducing the effect of outliers may lead to better models.

One approach to handle skewed distributions is to apply a transformation that modifies a field's values so that the overall distribution is more normal. The Transform node (located in the Output palette, not in the Field Ops palette) accomplishes this.



Business Analytics software

Exploring the Transform Dialog Box

Fields | Options | Output | Annotations

Select transformations:

○ All Formulas

● Select formulas

□ Inverse (1/x) Offset: 0

□ Log (log n) Offset: 0

☑ Log (log 10) Offset: 1

□ Exponential

☑ Square Root

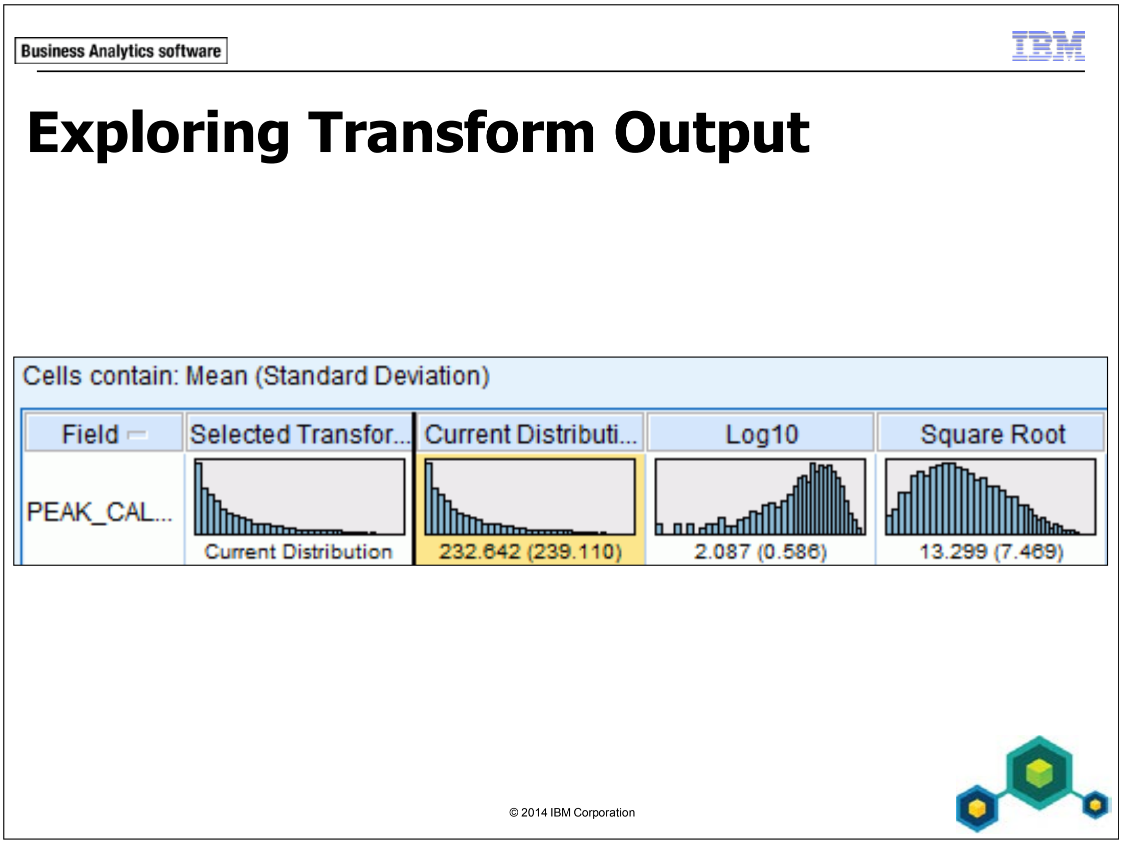
© 2014 IBM Corporation

Fields are selected on the Fields tab. By default, all transformations are applied and you can fine tune the transformations on the Options tab.

An offset value can be specified for the Inverse and the two Log functions. The offset value will be added to the original scores. The option to specify an offset value is available because the inverse function is not defined for 0 and the two log functions are not defined for values less than or equal to 0. For example, when a field's value equals 0, the inverse function and the two log functions return the undefined ($null$) values, and by using an offset such as 1 these functions will return a non-null value.

The offset value can be quite small, although when more than one field is specified at the same time, the offset will apply to all.

Note: The log n transformation takes the natural log, which is the log function with base e (the number 2.718). The log 10 transformation is the log function with base 10.



Business Analytics software

Exploring Transform Output

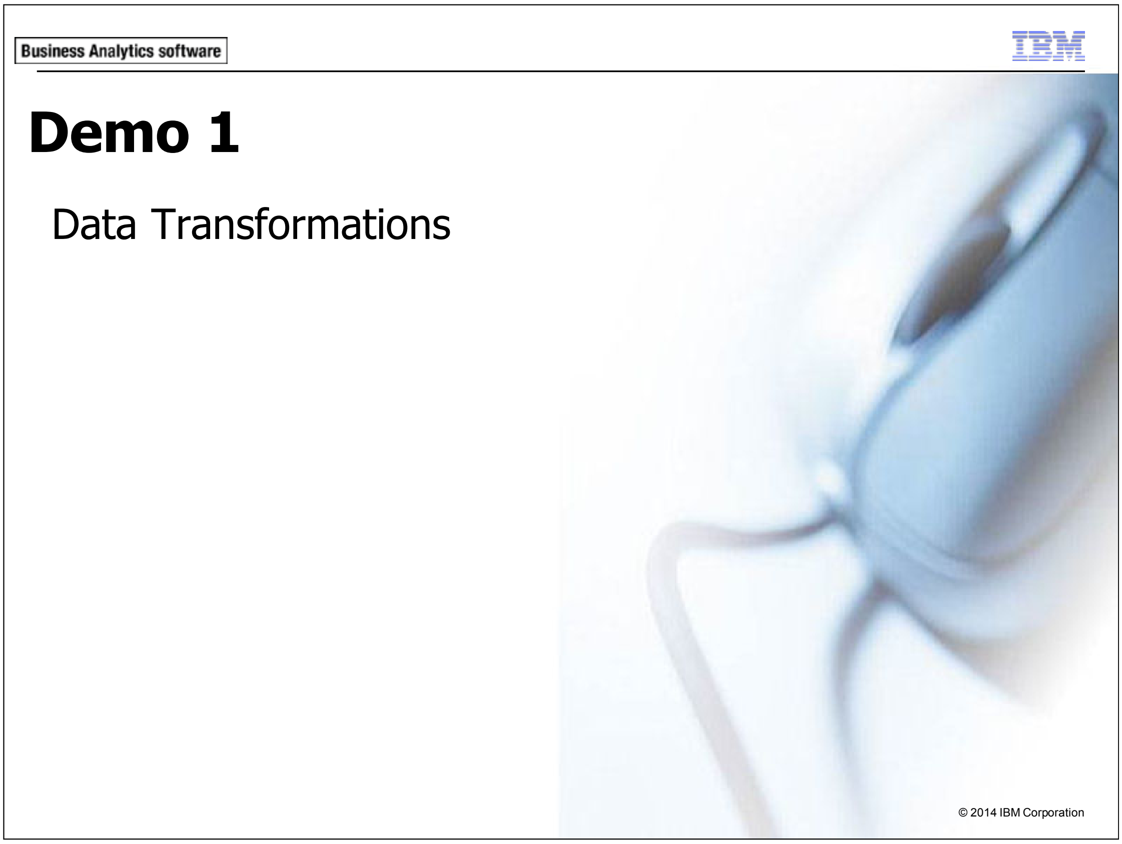
Cells contain: Mean (Standard Deviation)

Field	Selected Transfor...	Current Distributi...	Log10	Square Root
PEAK_CAL...	Current Distribution	232.642 (239.110)	2.087 (0.586)	13.299 (7.469)

The Transform output window shows thumbnails with the histograms of the transformed fields. Double-clicking a thumbnail will show the full histogram, overlaid with the best fitting normal distribution.

If you want to create a field with the transformed scores you can select either Generate\Derive Node or Generate\Filler Node to generate a new field or to overwrite a field's values, respectively. In both cases a SuperNode will be generated which comprises the transformations.

As a further option, the transformed scores can be standardized. Standardized scores express how many standard deviations a score is above or below the mean. Refer to the *Efficiency* module in this course for information on standardized scores.



Business Analytics software

Demo 1

Data Transformations

© 2014 IBM Corporation

The following (synthetic) file coming from a (fictitious) telecommunications firm is used to demonstrate how you can transform your data using the Filler, Binning and Transform node:

- **telco x churn data.dat**: A text file storing demographic and churn data on the company's customers. The file is located in **C:\Train\0A055**.

Before you begin with the demo, ensure that:

- You have started MODELER.
- You have set MODELER's working folder. (In MODELER, select **File\Set Directory**. When you are working in an IBM operated environment, browse to the folder **C:\Train\0A055** and then click **Set** to select it. When you are not working in an IBM operated environment, ask your trainer for the folder name.)

Demo 1: Data Transformations

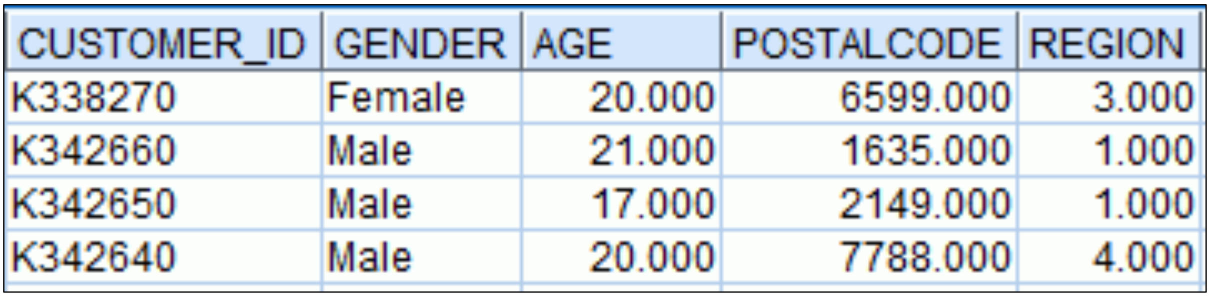
Purpose:
In order to build better models later you will cleanse your data, bin fields, and transform fields so that their distribution is similar to the normal distribution. You will use the Filler node, Binning node and Transform node to accomplish this.

Task 1. Using the Filler node to change storage.

In this task you will you the Filler node to change the storage of a number of fields.

1. Use a **Var. File** node to import data from the text file **telco x churn data.dat**.
2. Add a **Type** node downstream from the **Var. File** node.
3. Edit the **Type** node, and then:
   - click **Read Values**
   - click **Preview**

   A section of the results appear as follows:



CUSTOMER_ID	GENDER	AGE	POSTALCODE	REGION
K338270	Female	20.000	6599.000	3.000
K342660	Male	21.000	1635.000	1.000
K342650	Male	17.000	2149.000	1.000
K342640	Male	20.000	7788.000	4.000

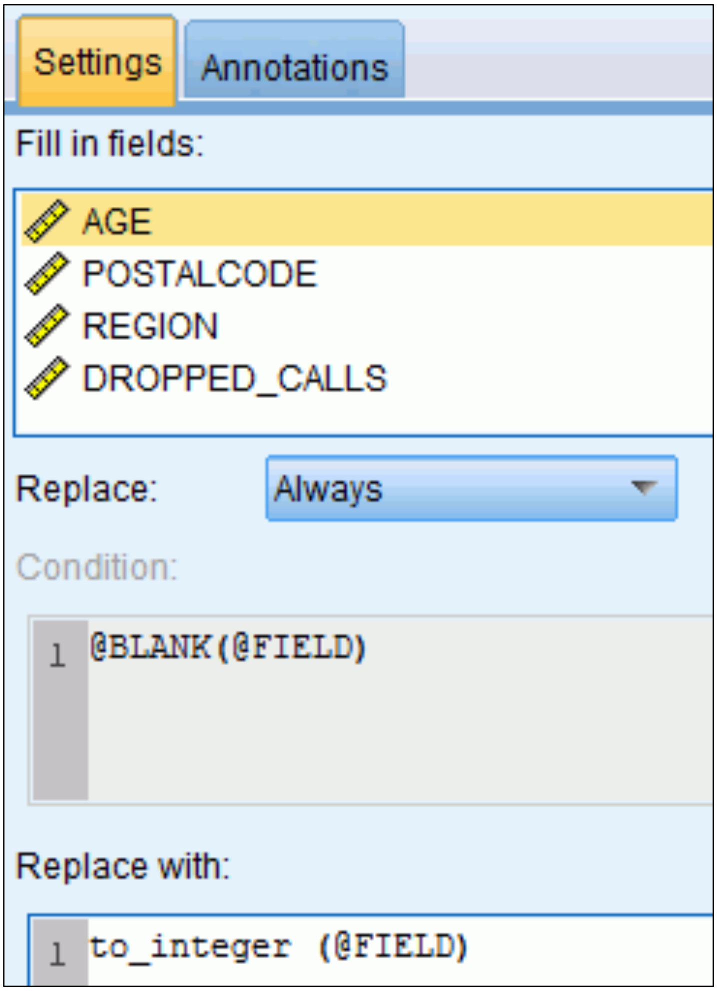
   Notice that AGE, POSTALCODE, and REGION, amongst others, are stored as reals, while the values can only be integer. You will use a Filler node to change the storage of these fields.

4. Close the **Preview** output window, and then close the **Type** dialog box.
5. Add a **Filler** node downstream from the **Type** node.

   Note: Be careful to select the Filler node, not the Filter node.

6. Edit the **Filler** node, and then:
   - for **Fill in fields**, select **AGE**, **POSTALCODE**, **REGION**, and **DROPPED_CALLS**
   - for **Replace**, select **Always**
   - for **Replace with**, enter **to_integer (@FIELD)**

   A section of the results appear as follows:



Fill in fields:

- AGE
- POSTALCODE
- REGION
- DROPPED_CALLS

Replace: Always

Condition:

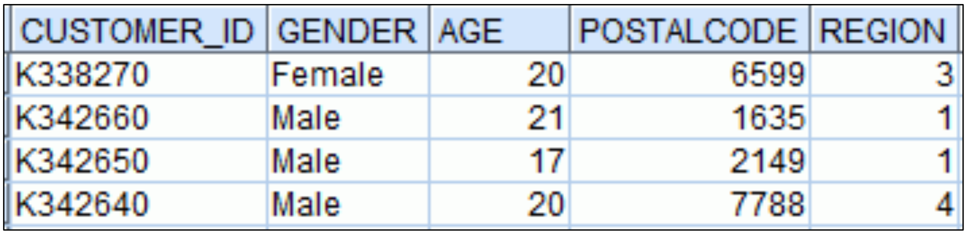
```
@BLANK(@FIELD)
```

Replace with:

```
to_integer (@FIELD)
```

7. Click **Preview**.

   A section of the results appear as follows:



CUSTOMER_ID	GENDER	AGE	POSTALCODE	REGION
K338270	Female	20	6599	3
K342660	Male	21	1635	1
K342650	Male	17	2149	1
K342640	Male	20	7788	4

The values are integers, as required.

8. Close the **Preview** output window, and then close the **Filler** dialog box. Leave the stream open for the next task.

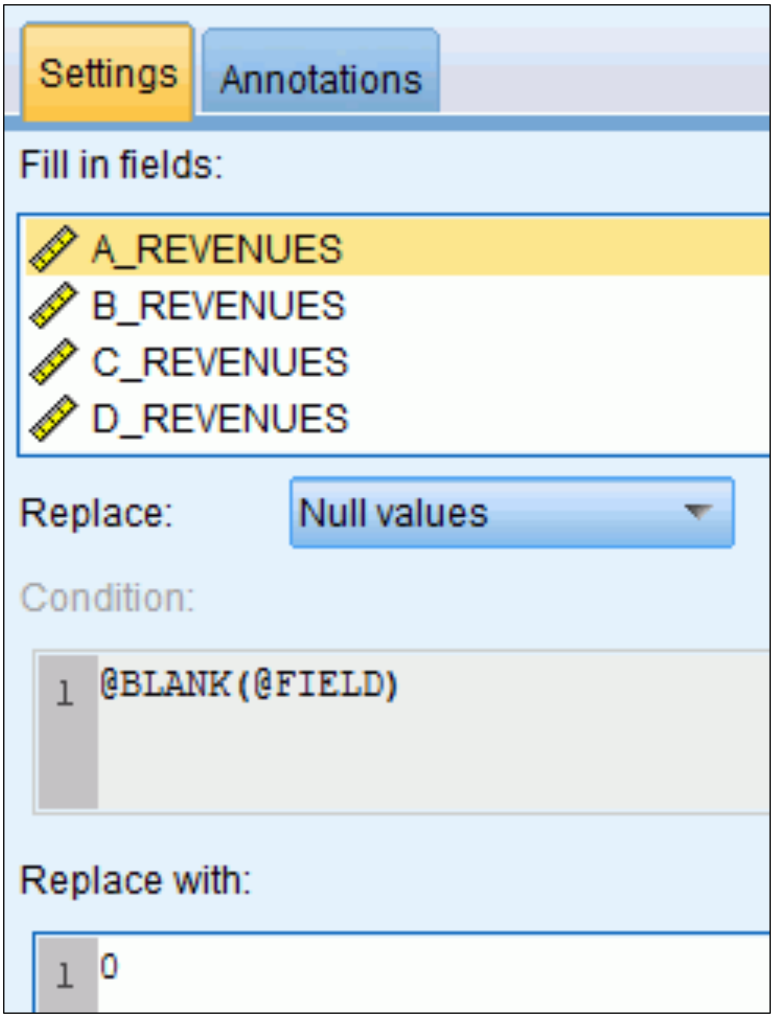
Task 2. Using the Filler node to replace null values.

Consider the fields that store the revenues for various products, A_REVENUES to L_REVENUES. These fields are undefined ($null$) when one does not possess the product. However, these undefined values should be replaced with 0, because missing revenues means that revenues equals 0.

You will build from the stream that you created in the previous task.

1. Add a **second Filler** node downstream from to the **first Filler** node.
2. Edit the second **Filler** node, and then:
   - for **Fill in fields**, select **A_REVENUES** to **L_ REVENUES**
   - for **Replace**, select **Null values**
   - for **Replace with**, ensure that the value is **0**

   A section of the results appear as follows:



Fill in fields:
A_REVENUES
B_REVENUES
C_REVENUES
D_REVENUES

Replace: Null values

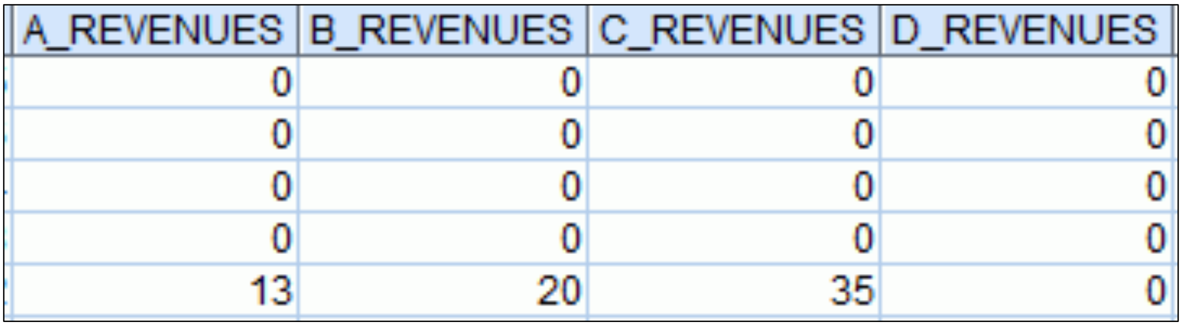
Condition:
```
@BLANK(@FIELD)
```

Replace with:
```
0
```

This material is meant for IBM Academic Initiative purposes.

3. Click **Preview**, and then scroll to **A_REVENUES** in the output window.

   A section of the results appear as follows:

A_REVENUES	B_REVENUES	C_REVENUES	D_REVENUES
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
13	20	35	0

   The $null$ values have been replaced with 0, as required.

4. Close the **Preview** output window, and then close the **Filler** dialog box.

   Leave the stream open for the next task.

Task 3. Using the Filler node to replace strings.

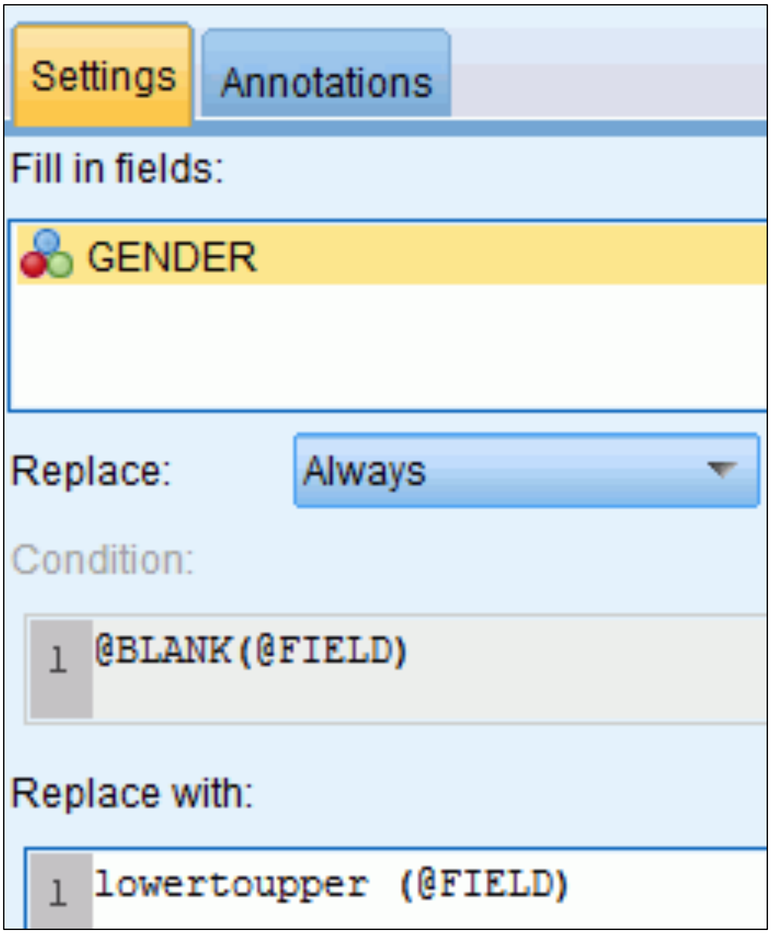
As a last example of data cleaning using the Filler node, consider the field GENDER with values female, Female, FEMALE, male, Male and MALE. Al these different spellings must be replaced with uppercase letters.

You will build from the stream that you created in the previous task.

1. Add a **third Filler** node downstream from the **second Filler** node.
2. Edit the **Filler** node, and then:
   - for **Fill in fields**, select **GENDER**
   - for **Replace**, select **Always**
   - for **Replace with**, enter **lowertoupper (@FIELD)** (alternatively, you can use lowertoupper (GENDER), but using @FIELD rather than the field name is more general and will work when the transformation has to be applied to multiple fields)

This material is meant for IBM Academic Initiative purposes.

A section of the results appear as follows:



Fill in fields:

GENDER

Replace: Always

Condition:

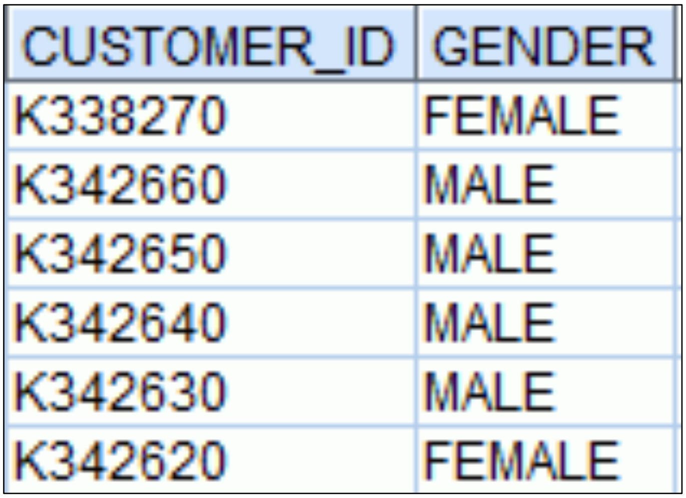
```
@BLANK(@FIELD)
```

Replace with:

```
lowertoupper (@FIELD)
```

3. Click **Preview**.

   A section of the results appear as follows:



CUSTOMER_ID	GENDER
K338270	FEMALE
K342660	MALE
K342650	MALE
K342640	MALE
K342630	MALE
K342620	FEMALE

   The values are in upper case, as required.

4. Close the **Preview** output window, and then close the **Filler** dialog box.

   Leave the stream open for the next task.

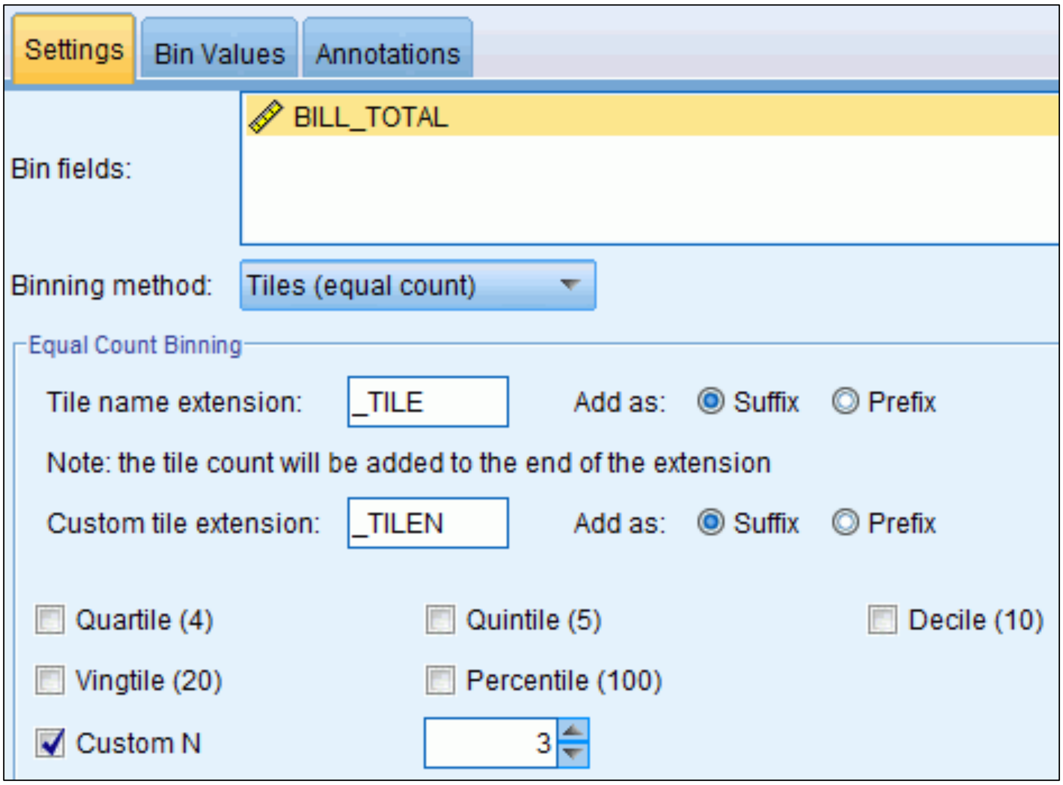
Task 4. Binning with equal counts.

You will bin BILL_TOTAL into three categories, so that each category has the same numbers of records in it. The first category stores the 33.3% records with the smallest values for BILL_TOTAL, and the third category stores the 33.3% records with largest values for BILL_TOTAL. In other words, a new field is created with three categories that could be labeled as bronze customers, silver customers, and gold customers.

You will build from the stream that you created in the previous task.

1. Add a **Binning** node downstream from the **third Filler** node.
2. Edit the **Binning** node, and:
   - for **Bin fields**, select **BILL_TOTAL**
   - for **Binning method**, select **Tiles (equal count)**
   - disable the **Decile (10)** option
   - select **Custom N**, and type **3**

   A section of the results appear as follows:



Bin fields: BILL_TOTAL

Binning method: Tiles (equal count)

Equal Count Binning

Tile name extension: _TILE Add as: ● Suffix ○ Prefix

Note: the tile count will be added to the end of the extension

Custom tile extension: _TILEN Add as: ● Suffix ○ Prefix

□ Quartile (4) □ Quintile (5) □ Decile (10)

□ Vingtile (20) □ Percentile (100)

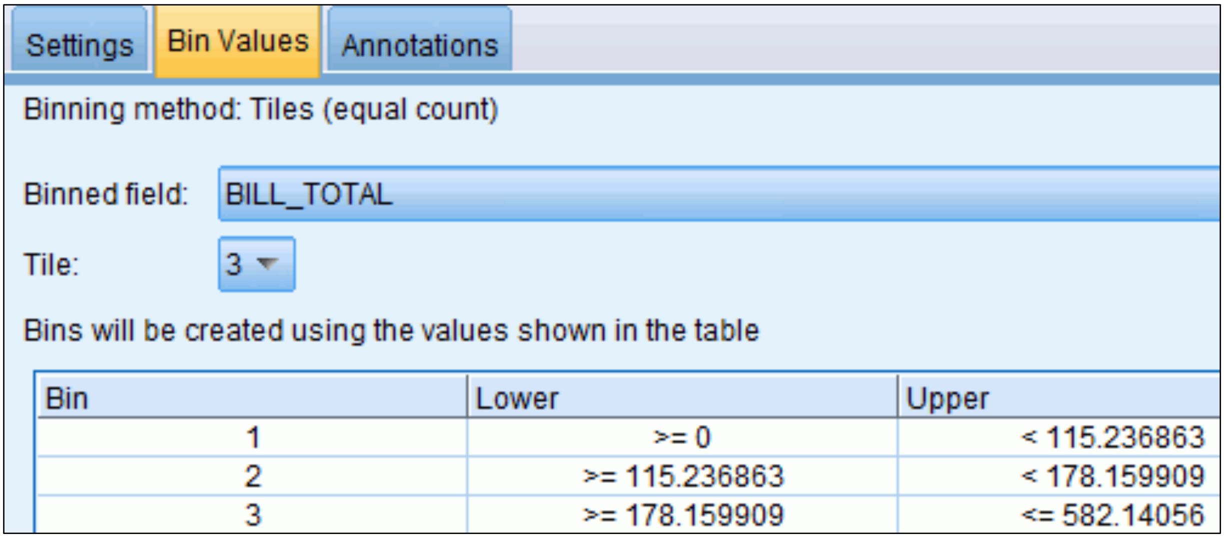
☑ Custom N 3

The name for the new field will be BILL_TOTAL_TILEN, the source field name extended with the custom tile extension, _TILEN.

At this point you will examine the thresholds that will be used for binning the field.

3. Click the **Bin Values** tab, and then click the **Read Values** button.

A section of the results appear as follows:



Binning method: Tiles (equal count)

Binned field: BILL_TOTAL

Tile: 3

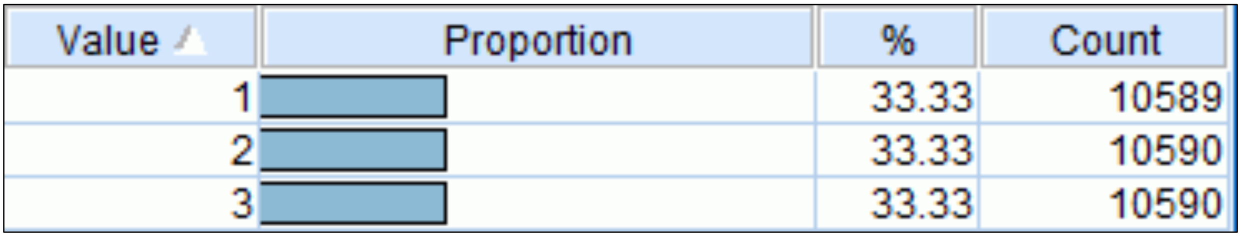
Bins will be created using the values shown in the table

Bin	Lower	Upper
1	>= 0	< 115.236863
2	>= 115.236863	< 178.159909
3	>= 178.159909	<= 582.14056

The first threshold is 115.23. If a record has a BILL_TOTAL less than 115.23, it will be assigned to the first category of the new field.

4. Close the **Binning** dialog box.
5. Add a **Distribution** node downstream from the **Binning** node.
6. Edit the **Distribution** node, and then:
   - select **BILL_TOTAL_TILEN**
   - click **Run**

A section of the results appears as follows:



Value	Proportion	%	Count
1		33.33	10589
2		33.33	10590
3		33.33	10590

A nominal field has been created with three categories, each, approximately, containing the same number of records.

7. Close the **Distribution** output window.

Leave the stream open for the next task.

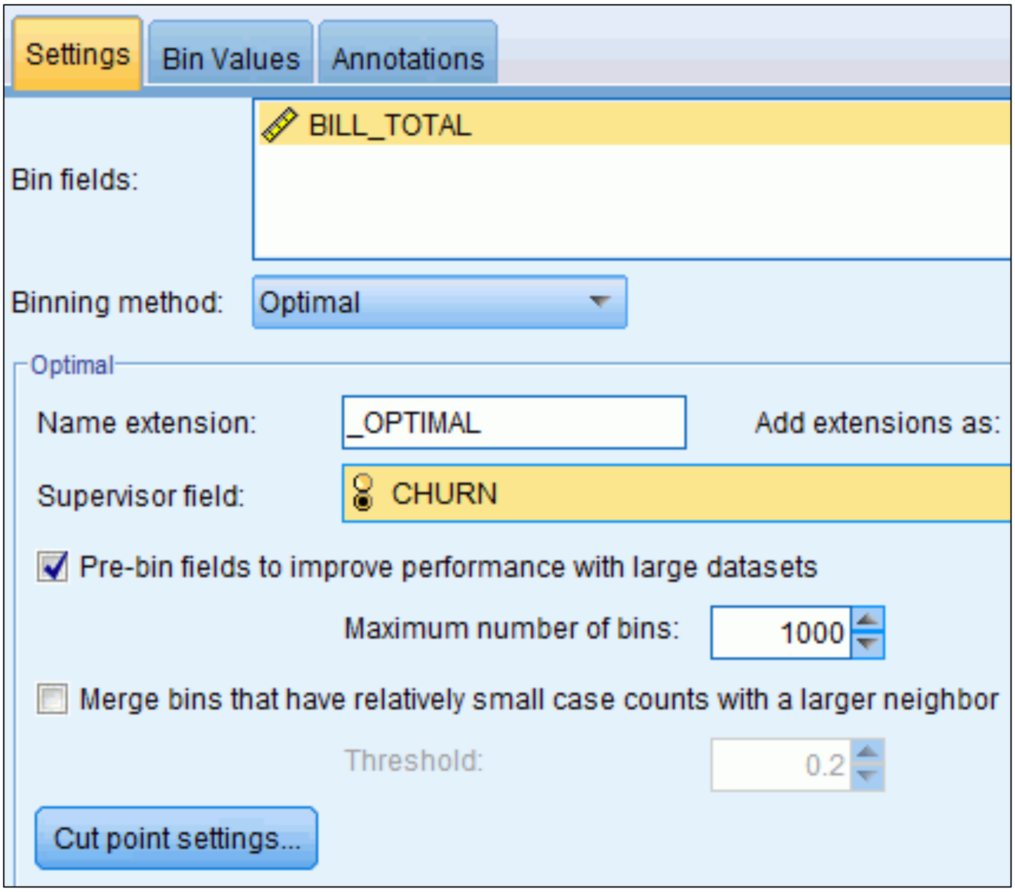
Task 5. Binning using a supervisor field.

You will bin the BILL_TOTAL field in such a way that its relationship with CHURN (flagging whether the customer has churned) is maximized.

You will build from the stream that you created in the previous task.

1. Add a **second Binning** node downstream from the **first Binning** node.
2. Edit the **second Binning** node, and then:
   - for **Bin fields**, select **BILL_TOTAL**
   - for **Binning method**, select **Optimal**
   - for **Supervisor field**, select **CHURN**

   A section of the results appear as follows:



Bin fields: BILL_TOTAL

Binning method: Optimal

Optimal

Name extension: _OPTIMAL

Add extensions as:

Supervisor field: CHURN

☑ Pre-bin fields to improve performance with large datasets

Maximum number of bins: 1000

□ Merge bins that have relatively small case counts with a larger neighbor

Threshold: 0.2

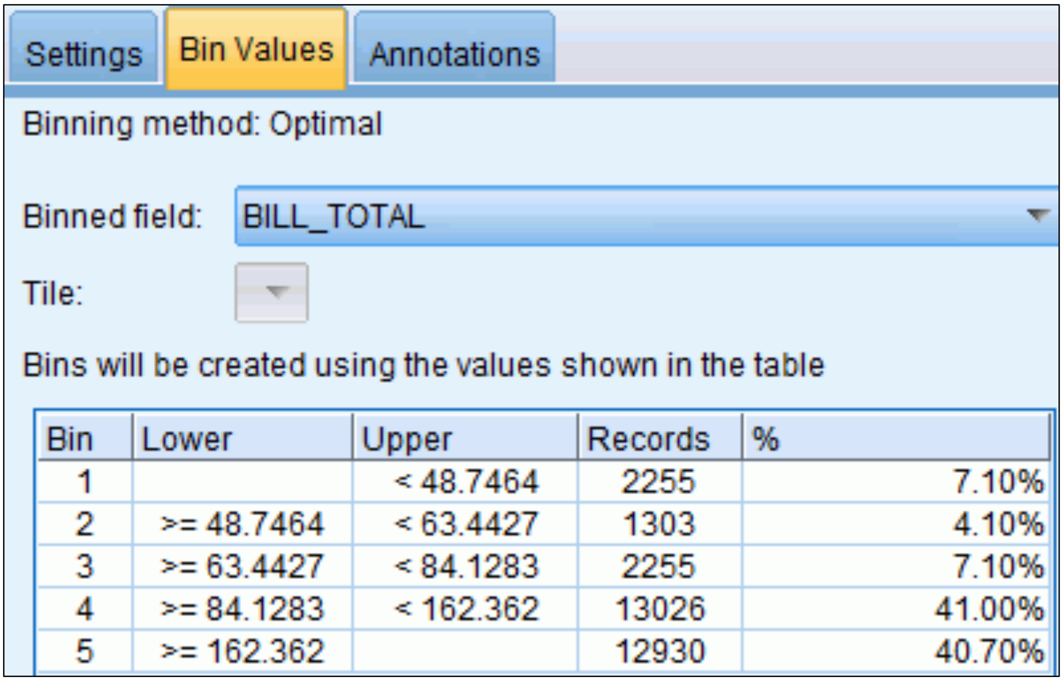
Notice that the new field name is the source field name, extended with _OPTIMAL as a suffix.

This material is meant for IBM Academic Initiative purposes.

At this point you will examine the thresholds used for binning.

3. Click the **Bin Values** tab, and then click **Read Values**.

   A section of the results appear as follows:



Binning method: Optimal

Binned field: BILL_TOTAL

Tile:

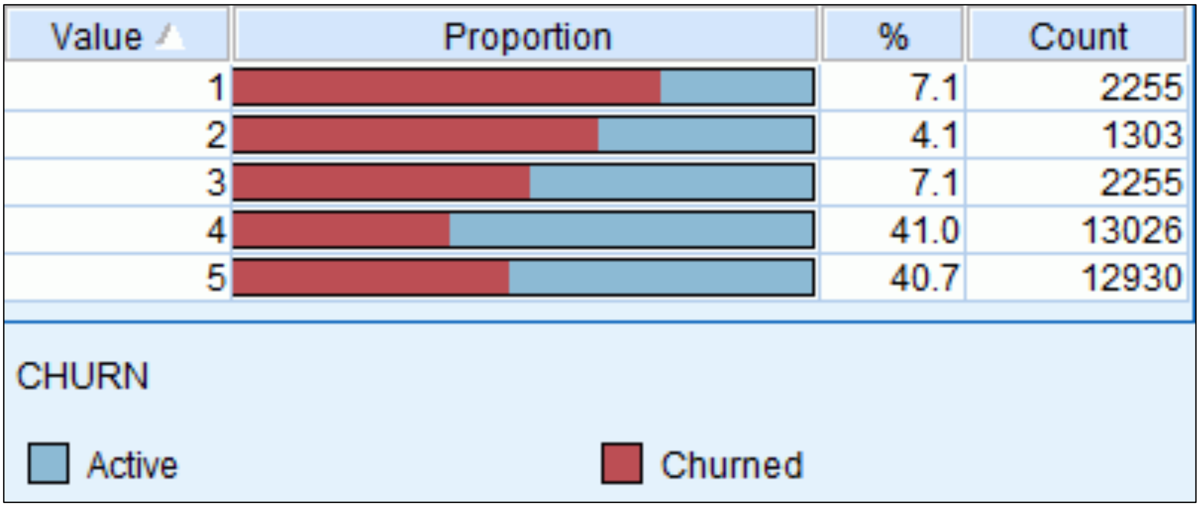
Bins will be created using the values shown in the table

Bin	Lower	Upper	Records	%
1		< 48.7464	2255	7.10%
2	>= 48.7464	< 63.4427	1303	4.10%
3	>= 63.4427	< 84.1283	2255	7.10%
4	>= 84.1283	< 162.362	13026	41.00%
5	>= 162.362		12930	40.70%

   Optimal binning will create a nominal field with 5 categories, the first threshold being 48.7464.

4. Close the **Binning** dialog box.
5. Add a **Distribution** node downstream from the **second Binning** node.
6. Edit the **Distribution** node, and then:
   - for **Field**, select **BILL_TOTAL_OPTIMAL**
   - for **Color**, select **CHURN**
   - enable the option **Normalize by color**
   - click **Run**

A section of the results appear as follows:



Value	Proportion	%	Count
1		7.1	2255
2		4.1	1303
3		7.1	2255
4		41.0	13026
5		40.7	12930

CHURN

🟦 Active 🟥 Churned

The highest churn rate is found in the first category; these records have the lowest BILL_TOTAL values. The customers in the last category show a high percentage of churners, which may be a point of concern to the company.

7. Close the **Distribution** output window.

   Leave the stream open for the next task.

Task 6. Transform a field so that its distribution is less skewed.

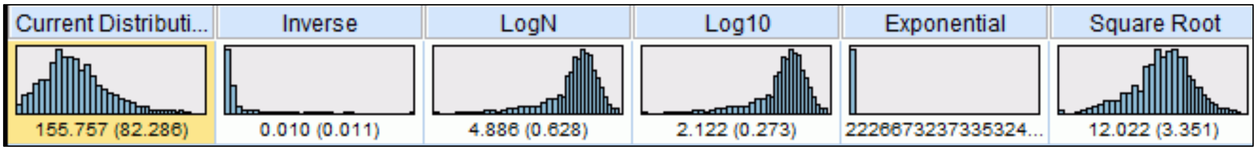
In this task you will examine the distribution of BILL_TOTAL and apply a transformation so that its distribution is less skewed (more normal).

You will build from the stream that you created in the previous task.

1. Add a **Transform** node (Output palette) downstream from the **second Binning** node.
2. Edit the **Transform** node, and then:
   - select **BILL_TOTAL**
   - click the **Options** tab, and ensure that all formulas are applied
   - click **Run**

This material is meant for IBM Academic Initiative purposes.

A section of the results appear as follows:

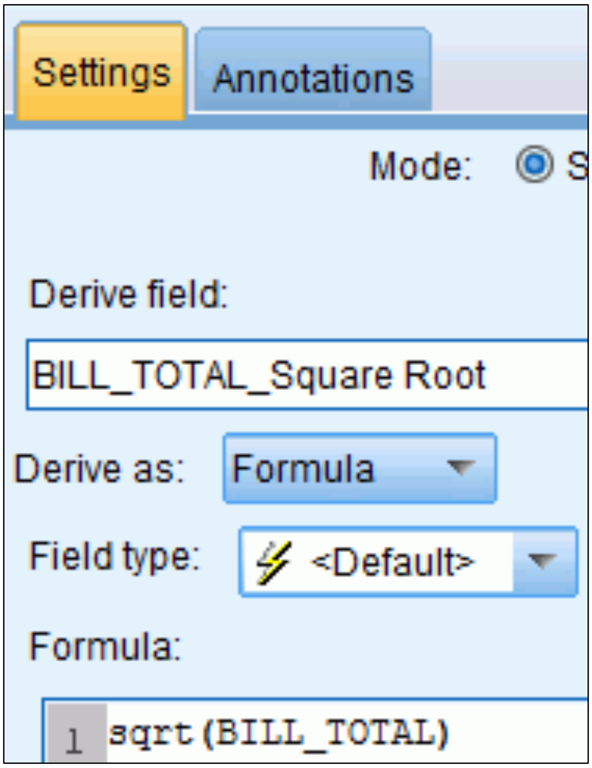


Current Distributi...	Inverse	LogN	Log10	Exponential	Square Root
155.757 (82.286)	0.010 (0.011)	4.886 (0.628)	2.122 (0.273)	2226673237335324...	12.022 (3.351)

Only the square root transformation appears to make the distribution less skewed. You will derive a new field storing the transformed scores.

3. Click the **Square Root thumbnail** so that it is selected.
4. Select **Generate\Derive Node**, and click **OK** in the dialog box that displays (you will compute non-standardized scores).
5. Close the **Transform** output window.
6. Add the generated **SuperNode**, located in the left upper corner on the stream canvas, downstream from the **second Binning** node.
7. Edit the **SuperNode**, click **Zoom in**, and then edit the node named **BILL_TOTAL_Square Root**.

   A section of the results appear as follows:



Mode: ● S

Derive field: BILL_TOTAL_Square Root

Derive as: Formula

Field type: <Default>

Formula:

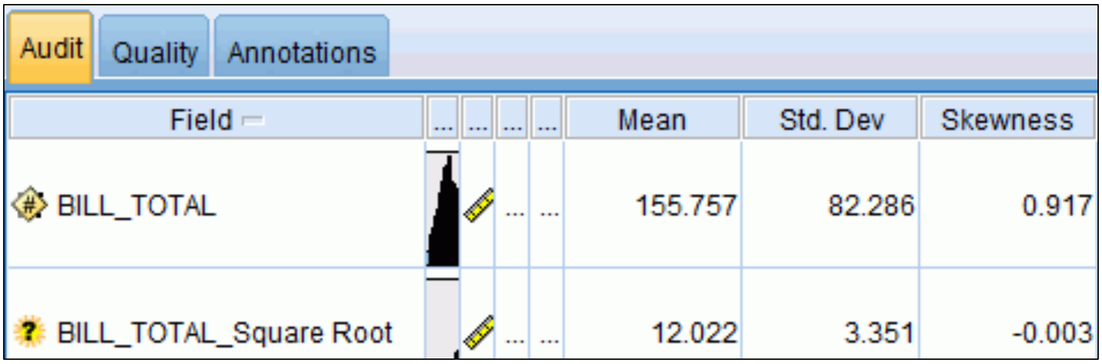
```
sqrt(BILL_TOTAL)
```

   A new field is derived, named BILL_TOTAL_Square Root, by applying the square root function to BILL_TOTAL.

8. Close the **Derive** dialog box, **Zoom out**, and then close the **SuperNode** dialog box.

9. Add a **Data Audit** node downstream from the **SuperNode**.
10. Edit the **Data Audit** node, and then:
    - select **Use custom fields**
    - select **BILL_TOTAL** and **BILL_TOTAL_Square Root**
    - click **Run**

    A section of the results appear as follows:



Field	...	...	...	...	Mean	Std. Dev	Skewness
BILL_TOTAL			...	...	155.757	82.286	0.917
BILL_TOTAL_Square Root			...	...	12.022	3.351	-0.003

The skewness for the original field is 0.917, indicating a significant departure from normality. The transformed field is very close to a symmetric distribution given its skewness of -0.003. This can be enough motivation to use the transformed field in modeling rather than the original field.

11. Close the **Data Audit** output window.

This completes the demo for this module. You will find the solution results in **demo_data_transformations_completed.str**, located in the **02-Data_Transformations\Solutions** sub folder.

Results:
You have cleansed your data using the Filler node and added new fields using the Binning node and the Transform node. Now that you have added these fields to your dataset you will be able to build better models later.

Apply Your Knowledge

Use the questions in this section to test your knowledge of the course material.

Question 1: Is the following statement true or false? The Filler node will create a new field.

A. True

B. False

Question 2: Suppose that a string field stores the values "1", "2", etc. For modeling purposes you need to convert the string values to integers 1, 2, etc. The node to use is:

A. Filler

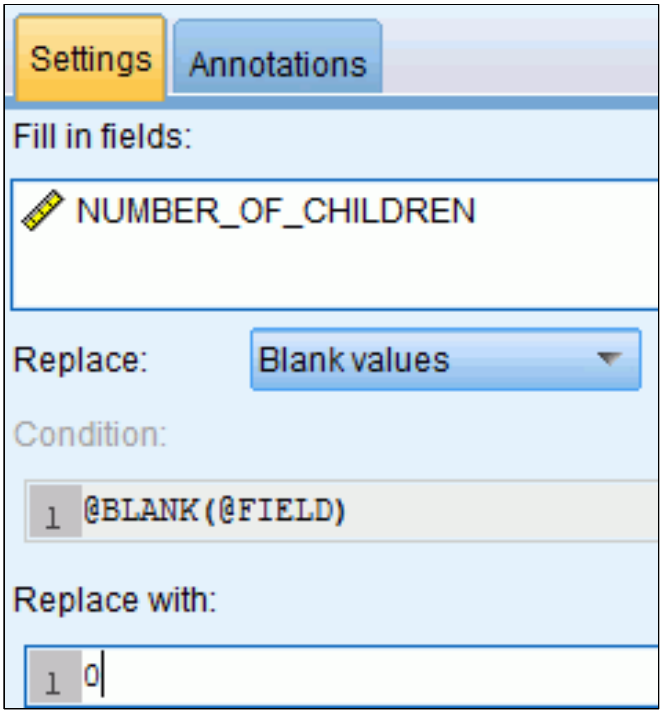
B. Binning

C. Transform

Question 3: Is the following statement true or false? Given is a field NUMBER_OF_CHILDREN, for which the undefined value ($null$) is declared as a blank value. The specifications shown in the figure below will replace the $null$ value with 0 for NUMBER_OF_CHILDREN.

A. True

B. False



Fill in fields:

NUMBER_OF_CHILDREN

Replace: Blank values

Condition:

```
@BLANK(@FIELD)
```

Replace with:

```
0
```

Question 4: There are a number of reasons why one should or should not use the Binning node. Which of the following statements regarding the rationale behind binning are correct?

A. The effect of outliers can be reduced by binning because the outliers will be placed into the same bin.

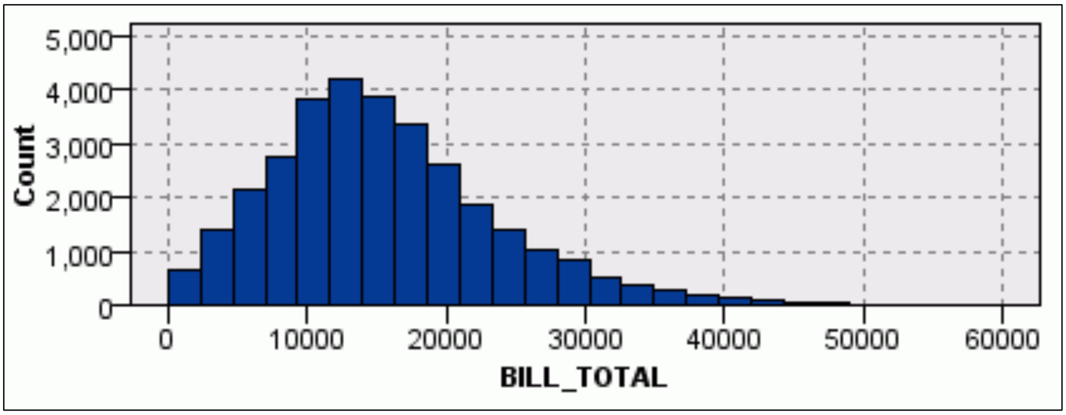
B. Binning has the advantage of enhancing the amount of information about the field.

C. Before modeling you may want to create custom groups ahead of time rather than rely on the way values are grouped by the modeling algorithm.

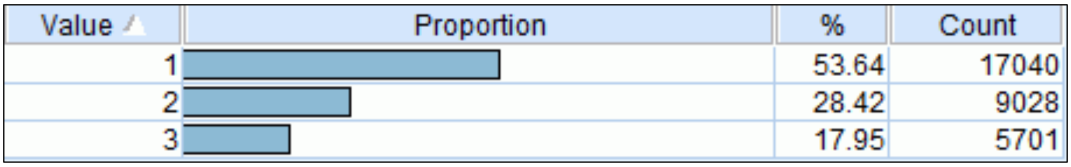
D. Binning reduces the number of undefined ($null$) values on a field.

Question 5: Which of the following is the correct statement? Refer to the figures that follow. What were the settings to create the new field?

Original field:



New field using the Binning node:



Value	Proportion	%	Count
1		53.64	17040
2		28.42	9028
3		17.95	5701

A. Binning method: Tiles using Tiling method: Tiles: Record count.

B. Binning method: Tiles using Tiling method: Tiles: Sum of values.

C. Binning method: Fixed Width using Bin width: 500

D. Binning method: Fixed Width using Bin width: 5000

Question 6: Is the following statement true or false? Optimal binning always needs a supervisor field.

A. True

B. False

Question 7: Is the following statement true or false? The Transform node is located in the Field Ops palette.

A. True

B. False

Question 8: Which of the following is the correct statement? Suppose that a field has values between -100 and -1 (including boundary values -100 and -1), and that you want to explore how a Log10 transformation will change the field's distribution. What is the best value for the offset?

A. -100

B. 1

C. 100

D. 101

Answers to questions:

Answer 1: B. False. The Filler node replaces values in an existing field.

Answer 2: A. Only the Filler node enables you to change the storage of a field.

Answer 3: A. True. Undefined ($null$) values are declared as blanks and when you replace blank values you will also replace the undefined ($null$) values.

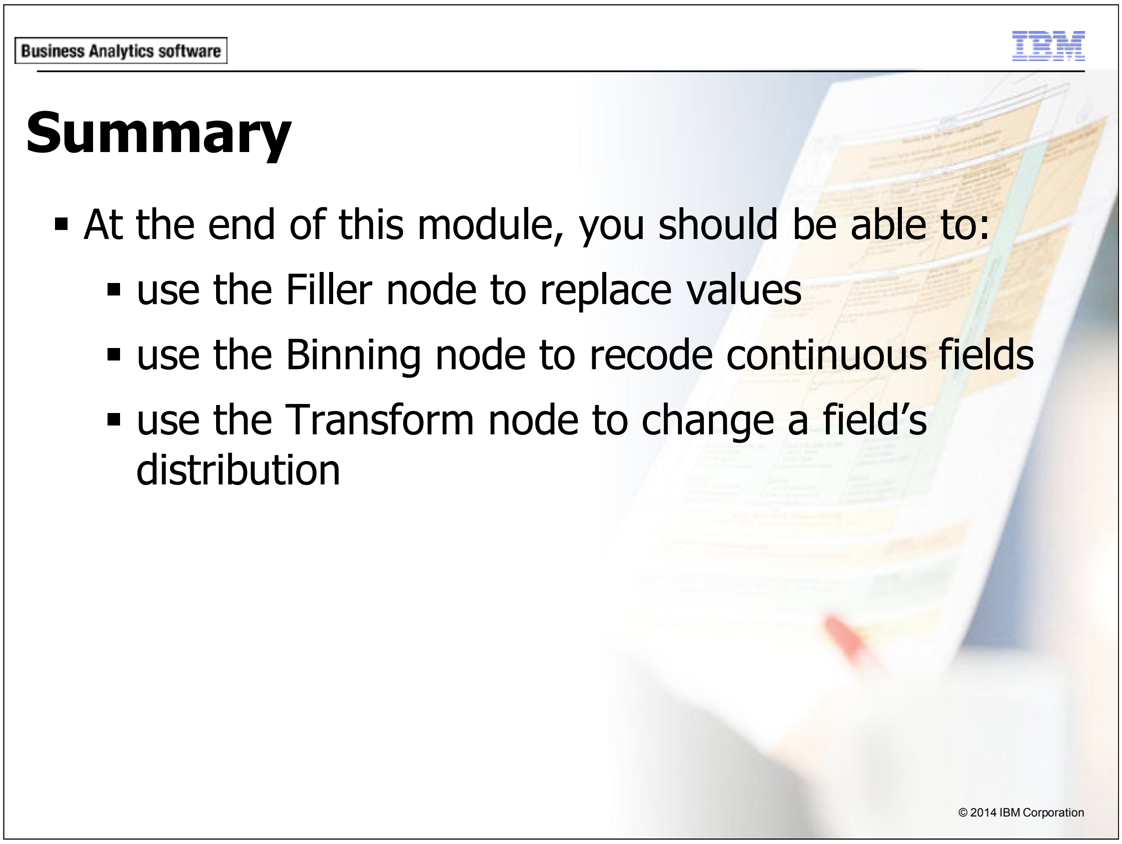
Answer 4: A, C. Outliers will be binned into the same category, thus reducing their effect. Also, binning enables you to create your own categories rather than being dependent on how a modeling algorithm bins values. Binning will reduce the amount of information, not enhance it. The number of undefined ($null$) values will not change, because $null$ values on the source field will be $null$ on the binned field also.

Answer 5: B. The binning method did not use intervals of width 500 or 5000, because then there would have been more than 3 categories. Also, the number of records is not the same in the categories, so that dismisses the tiling method of an equal record count and leaves the tiling binning method that is based on the sum of values.

Answer 6: A. True. Optimal binning requires a supervisor field by definition.

Answer 7: B. The Transform node is located in the Output palette, not in the Field Ops palette.

Answer 8: D. Only 101 will make the values positive, for which the Log 10 transformation is defined.

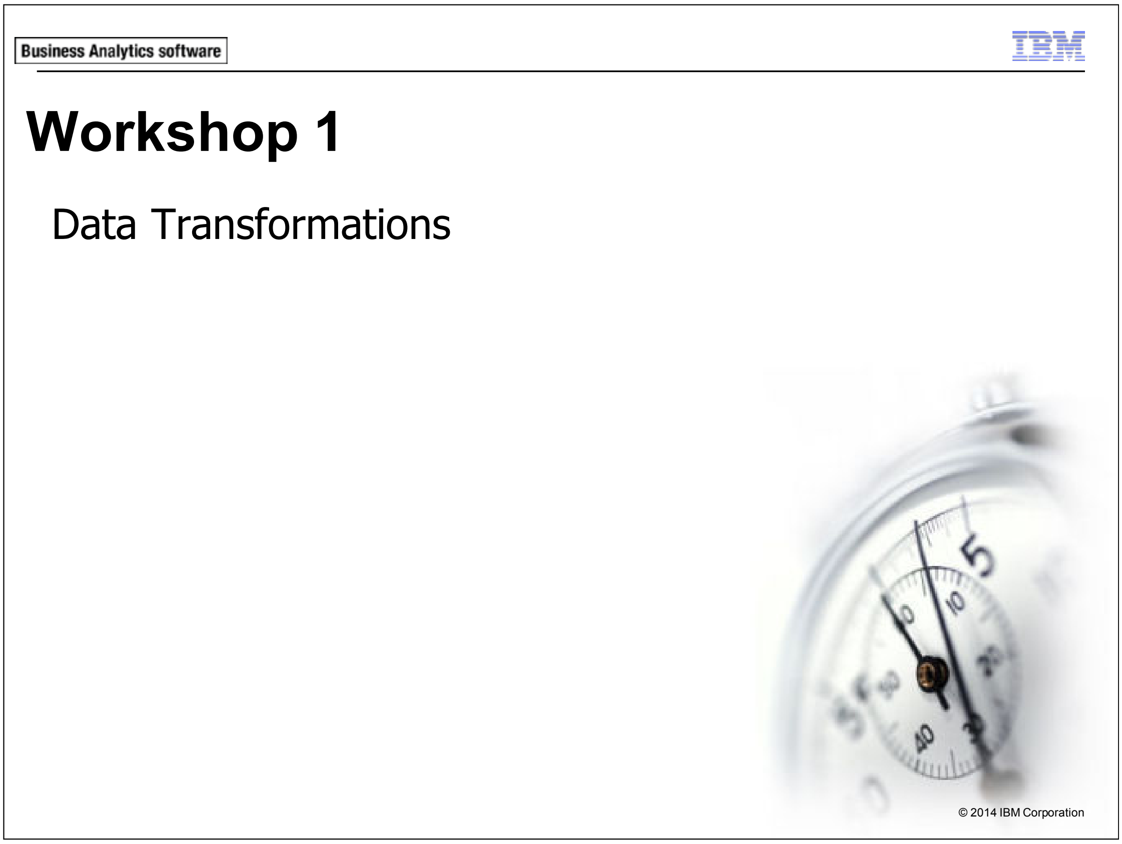


Business Analytics software

IBM

Summary

- At the end of this module, you should be able to:
  - use the Filler node to replace values
  - use the Binning node to recode continuous fields
  - use the Transform node to change a field’s distribution



Business Analytics software

IBM

Workshop 1

Data Transformations

© 2014 IBM Corporation

The following (synthetic) file is used in this workshop:

- **customers_and_holidays.dat**: A text file that stores data on holiday destinations for 411 customers of a travel agency. The file is located in **C:\Train\0A055**.

Before you begin with the workshop, ensure that:

- You have started MODELER.
- You have set MODELER's working folder. (In MODELER, select **File\Set Directory**. When you are working in an IBM operated environment, browse to the folder **C:\Train\0A055** and then click **Set** to select it. When you are not working in an IBM operated environment, ask your trainer for the folder name.)

Workshop 1: Data Transformations

- Import the data from the text file **customers_and_holidays.dat**, add a **Type** node downstream from the data source node, edit the **Type** node, declare **-999** as blank for **DISTANCE_TO_BEACH**, and then instantiate the data.
- Replace the values for **REGION** and **COUNTRY** so that the region and country name are always in upper case (for example: "East Anglia" should become "EAST ANGLIA" and "spain" should become "SPAIN").
- Replace blank values in **DISTANCE_TO_BEACH** with undefined ($null$) values.
- Recode **SPENT_LEISURE** into a new field named **SPENT_LEISURE_CATEGORY** that has three categories, with the same sum of **SPENT_LEISURE** in each of the categories.

  Which category has the smallest number of records in it (apart from the $null$ category)?
- Bin **SPENT_LEISURE** so that the relationship between **SPENT_LEISURE** and **SATISFIED** is optimal.

  Which category has the highest percentage of satisfied customers?
- Transform **SPENT_HOTEL** into a new field so that the distribution for this new field is less skewed than for the source field **SPENT_HOTEL**.

  What is the skewness for **SPENT_HOTEL** and what is the skewness for the new field?

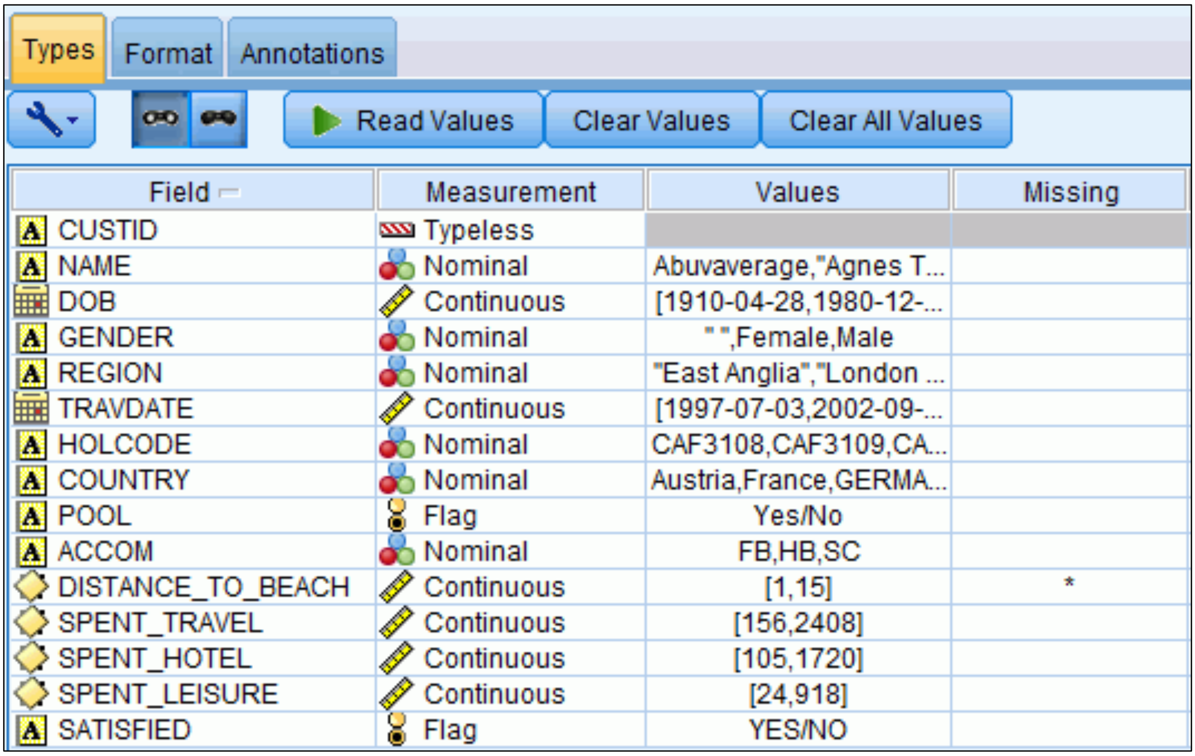
For more information about where to work and the workshop results, refer to the Task and Results section that follows. If you need more information to complete a task, refer to earlier demos for detailed steps.

Workshop 1: Tasks and Results

Task 1. Import and instantiate the data.

- Use a **Var. File** node to import the data from **customers_and_holidays.dat**.
- Add a **Type** node downstream from the **Var. file** node, edit the **Type** node, declare **-999** as blank for **DISTANCE_TO_BEACH**, and then click **Read Values**.

  A section of the results appear as follows:



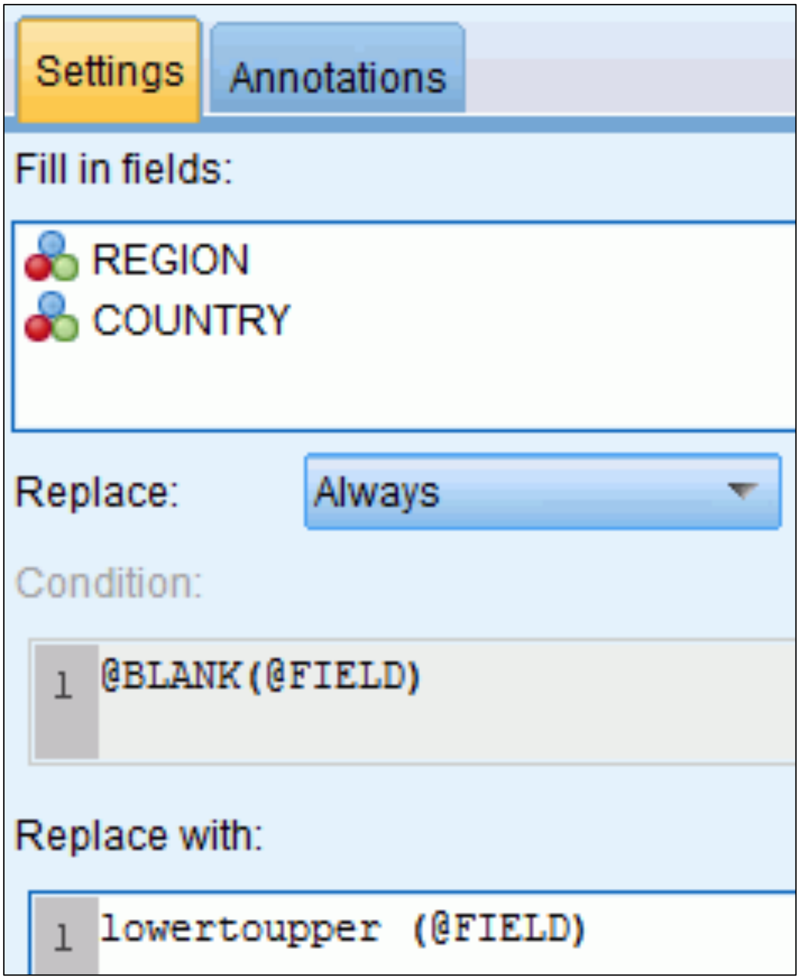
Field	Measurement	Values	Missing
CUSTID	Typeless		
NAME	Nominal	Abuvaverage,"Agnes T...	
DOB	Continuous	[1910-04-28,1980-12-...	
GENDER	Nominal	" ",Female,Male	
REGION	Nominal	"East Anglia","London ...	
TRAVDATE	Continuous	[1997-07-03,2002-09-...	
HOLCODE	Nominal	CAF3108,CAF3109,CA...	
COUNTRY	Nominal	Austria,France,GERMA...	
POOL	Flag	Yes/No	
ACCOM	Nominal	FB,HB,SC	
DISTANCE_TO_BEACH	Continuous	[1,15]	*
SPENT_TRAVEL	Continuous	[156,2408]	
SPENT_HOTEL	Continuous	[105,1720]	
SPENT_LEISURE	Continuous	[24,918]	
SATISFIED	Flag	YES/NO	

- Close the **Type** dialog box.

Task 2. Correct the spelling.

- Add a **Filler** node downstream from the **Type** node.
- Edit the **Filler** node, and then:
  - for **Fill in fields**, select **REGION** and **COUNTRY**
  - for **Replace**, select **Always**
  - for **Replace with**, enter **lowertoupper (@FIELD))**

  A section of the results appear as follows:



Fill in fields:

REGION
COUNTRY

Replace: Always

Condition:

```
@BLANK(@FIELD)
```

Replace with:

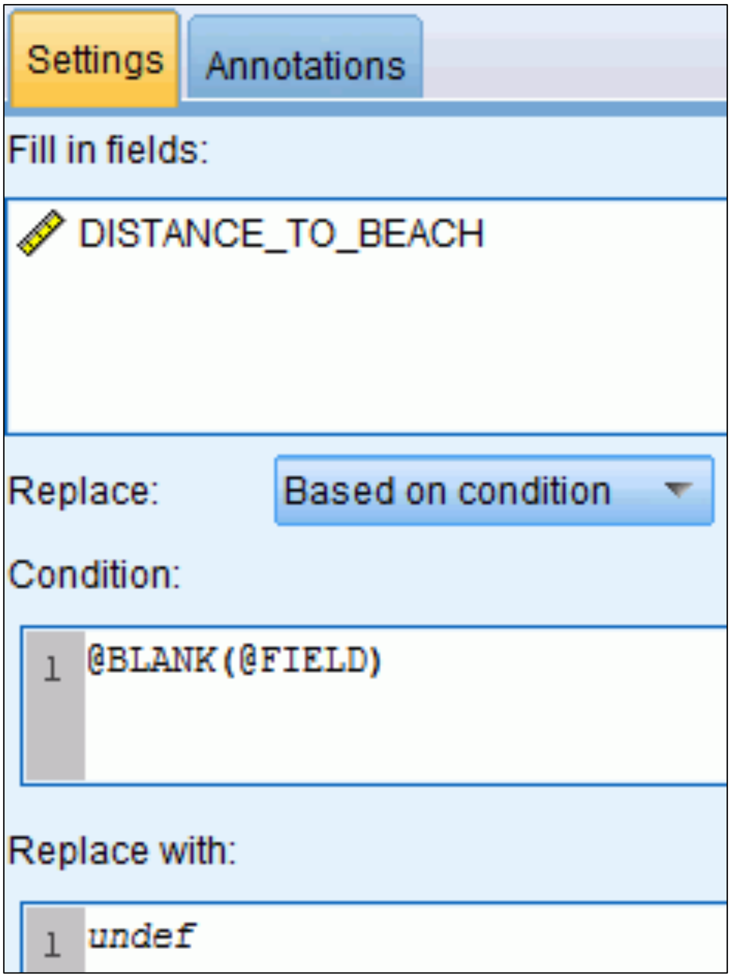
```
lowertoupper (@FIELD)
```

- Close the **Filler** dialog box.

Task 3. Replace blanks with undefined values.

- Add a **Filler** node downstream from the first **Filler** node.
- Edit the **Filler** node, and then:
  - for **Fill in fields**, select **DISTANCE_TO_BEACH**
  - for **Replace**, select **Based on condition** (the default)
  - for **Condition**, ensure that the expression is **@BLANK(@FIELD)** (the default; alternatively, select **Blank values** for **Replace**)
  - for **Replace with**, type **undef**

A section of the results appear as follows:



Fill in fields:
DISTANCE_TO_BEACH

Replace: Based on condition

Condition:

```
@BLANK(@FIELD)
```

Replace with:

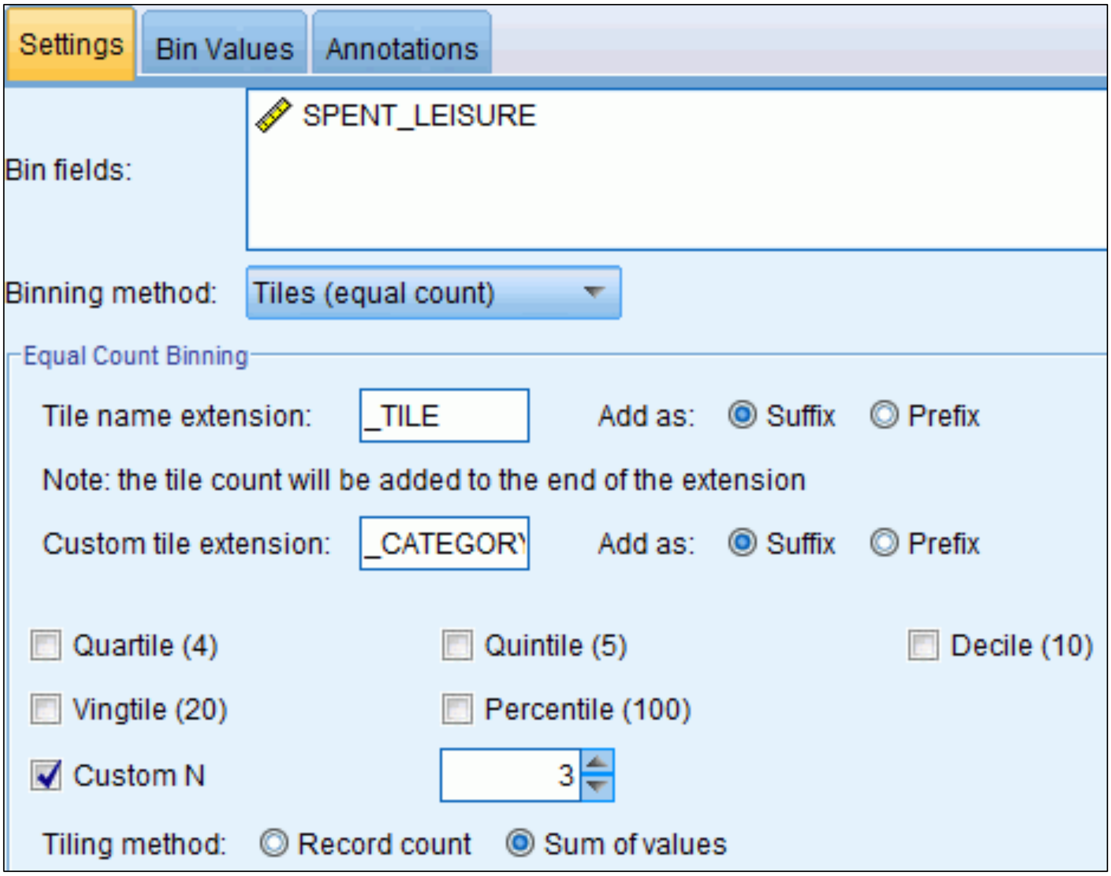
```
undef
```

- Close the **Filler** dialog box.

Task 4. Recode a field into three categories, based on the sum of values.

- Add a **Binning** node downstream from the second **Filler** node.
- Edit the **Binning** node, and then:
  - for **Bin fields**, select **SPENT_LEISURE**
  - for **Binning method**, select **Tiles (equal count)**
  - for **Custom tile extension**, type **_CATEGORY**
  - disable the **Decile (10)** option
  - enable the **Custom N** option, and type **3**
  - for **Tiling** method, select **Sum of values**

A section of the results appear as follows:



Bin fields: SPENT_LEISURE

Binning method: Tiles (equal count)

Equal Count Binning

Tile name extension: _TILE Add as: ● Suffix ○ Prefix

Note: the tile count will be added to the end of the extension

Custom tile extension: _CATEGORY Add as: ● Suffix ○ Prefix

□ Quartile (4) □ Quintile (5) □ Decile (10)

□ Vingtile (20) □ Percentile (100)

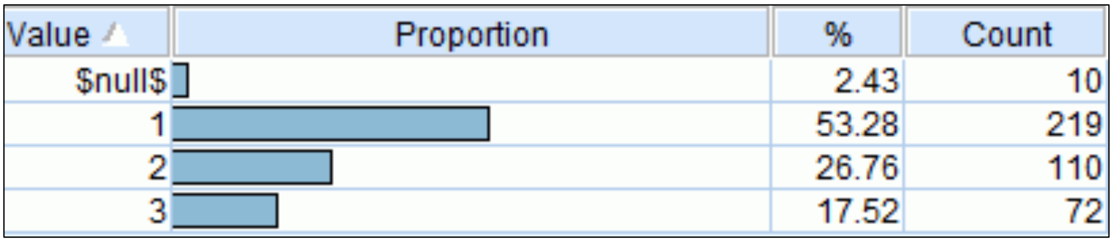
☑ Custom N 3

Tiling method: ○ Record count ● Sum of values

- Close the **Binning** dialog box.

- Add a **Distribution** node downstream from the **Binning** node, edit the **Distribution** node, select **SPENT_LEISURE_CATEGORY**, and then run the **Distribution** node.

  A section of the results appear as follows:

Value	Proportion	%	Count
$null$		2.43	10
1		53.28	219
2		26.76	110
3		17.52	72

  The third category has the smallest number of customers, as could be expected because the tiling method was such that the sum of values of SPENT_LEISURE was the same in the categories.

  Notice that 10 records have a $null$ value for the new field (they had an undefined value on the source field SPENT_LEISURE).

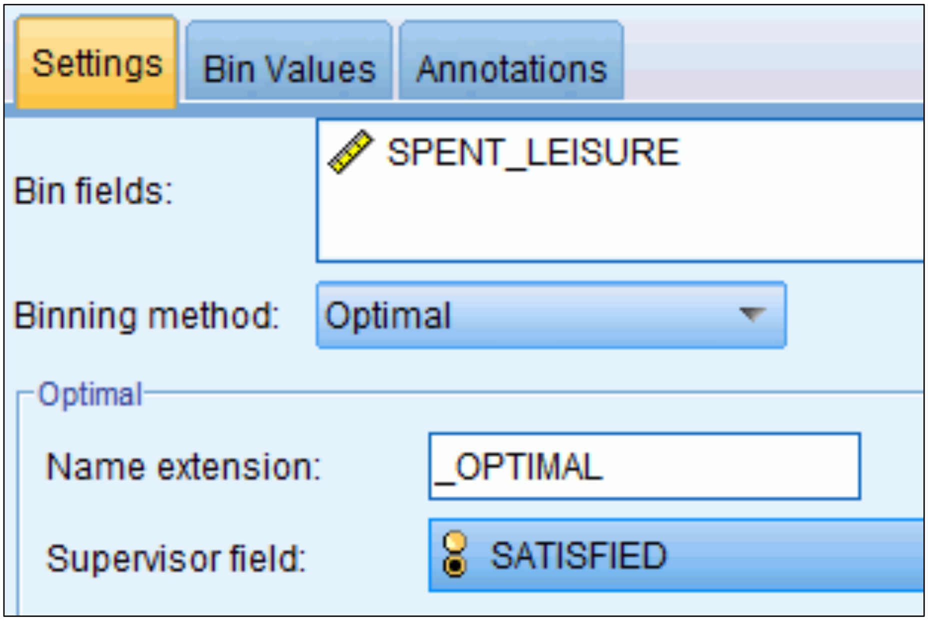
- Close the **Distribution** output window.

This material is meant for IBM Academic Initiative purposes.

Task 5. Bin a field optimally with respect to a target.

- Add a **Binning** node downstream from the first **Binning** node.
- Edit the **Binning** node, and then:
  - for **Bin fields**, select **SPENT_LEISURE**
  - for **Binning method**, select **_OPTIMAL**
  - for **Supervisor field**, select **SATISFIED**

  A section of the results appear as follows:



Bin fields: SPENT_LEISURE

Binning method: Optimal

Optimal

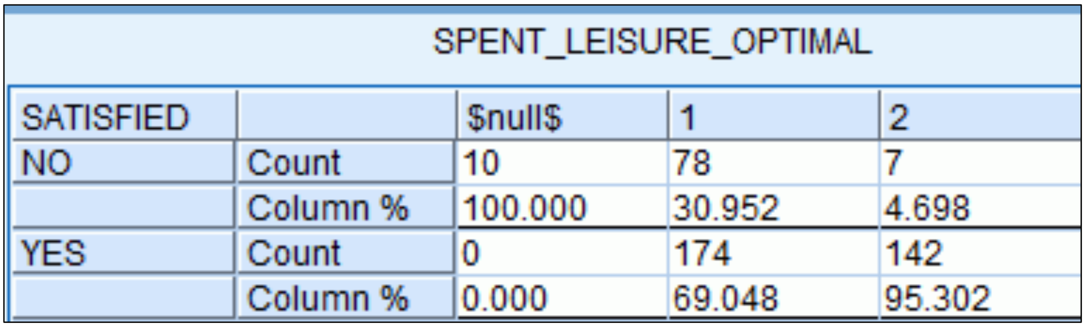
Name extension: _OPTIMAL

Supervisor field: SATISFIED

- Close the **Binning** dialog box.

- Add a **Matrix** downstream from the **Binning** node, edit the **Matrix** node, select **SATISFIED** in the row, **SPENT_LEISURE_OPTIMAL** in the column, request column percentages on the **Appearance** tab, and then run the **Matrix** node.

  A section of the results appear as follows:



		SPENT_LEISURE_OPTIMAL		
SATISFIED		$null$	1	2
NO	Count	10	78	7
	Column %	100.000	30.952	4.698
YES	Count	0	174	142
	Column %	0.000	69.048	95.302

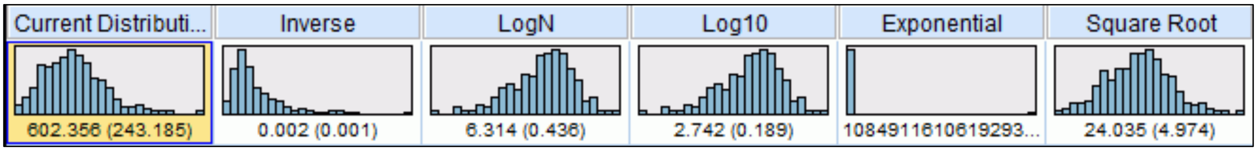
  Optimal binning recoded SPENT_LEISURE into two categories. The percentage satisfied customers is 69.048 in the first category and 95.302 in the second category.

- Close the **Matrix** output window.

Task 6. Transform a field to change its distribution.

- Add a **Transform** node (Output palette) downstream from the **second Binning** node.
- Edit the **Transform** node, and then:
  - for **Fields**, select **SPENT_HOTEL**
  - run the **Transform** node

  A section of the results appear as follows:

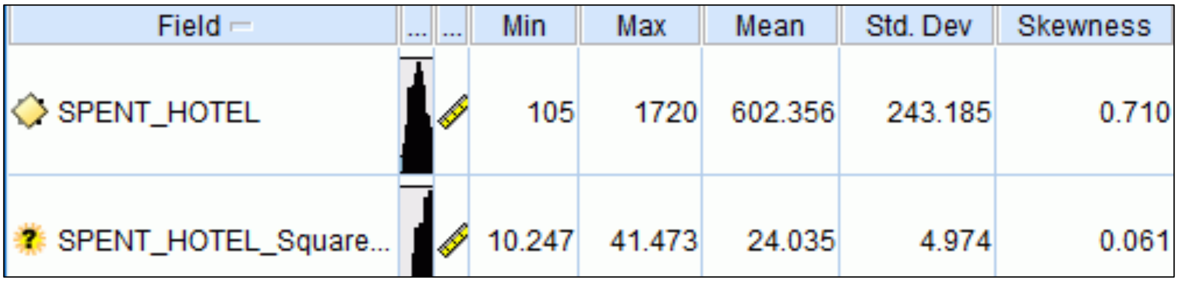


  The square root transformation appears to make the distribution less skewed.

- Click the **Square Root** thumbnail.
- Select **Generate\Derive node** from the main menu.
- Close the **Transform** output window.

- Add the generated **Supernode_Transform** node downstream from the second **Binning** node.
- Add a **Data Audit** node downstream from the **Supernode_Transform** node, edit the **Data Audit** node, select **Use custom fields**, select **SPENT_HOTEL** and **SPENT_HOTEL_Square_Root**, and then run the **Data Audit** node

  A section of the results appear as follows:

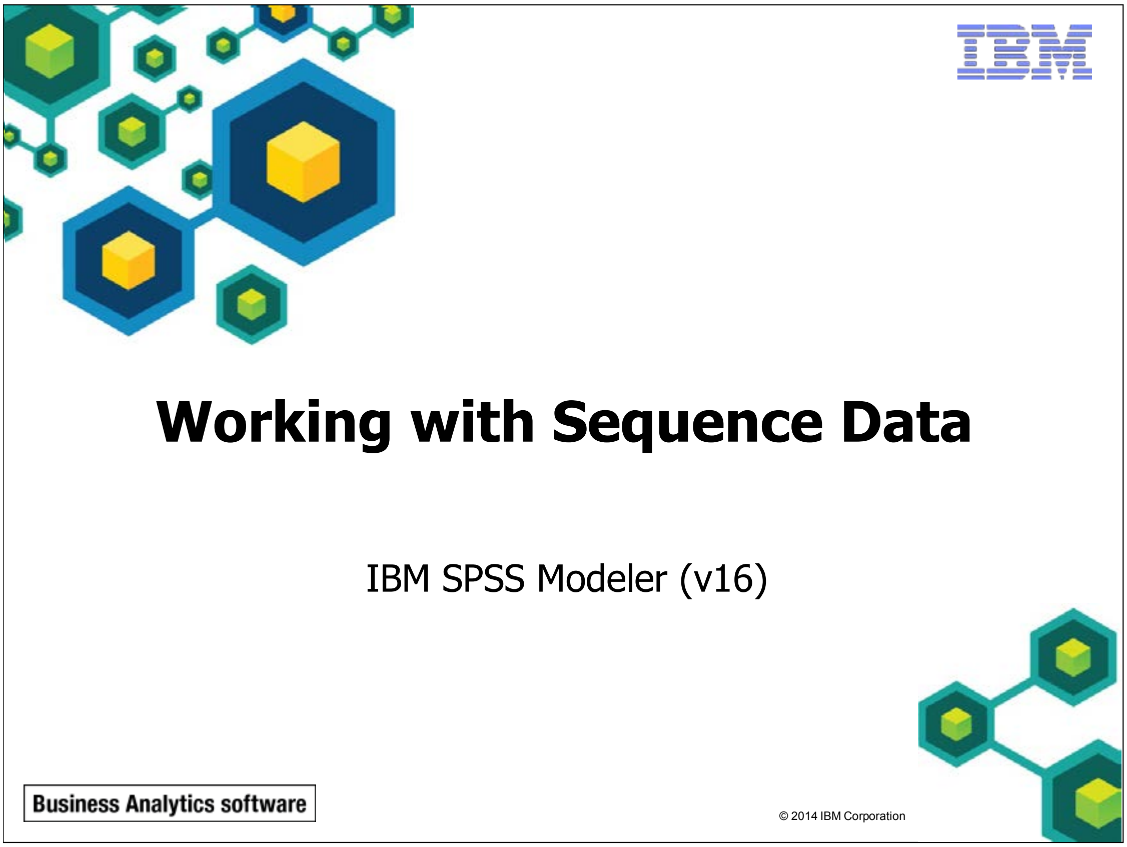
  

Field	...	...	Min	Max	Mean	Std. Dev	Skewness
SPENT_HOTEL			105	1720	602.356	243.185	0.710
SPENT_HOTEL_Square...			10.247	41.473	24.035	4.974	0.061

  The skewness for SPENT_HOTEL changed from 0.710 to 0.061 by applying the square root transformation.

- Close the **Data Audit** output window.

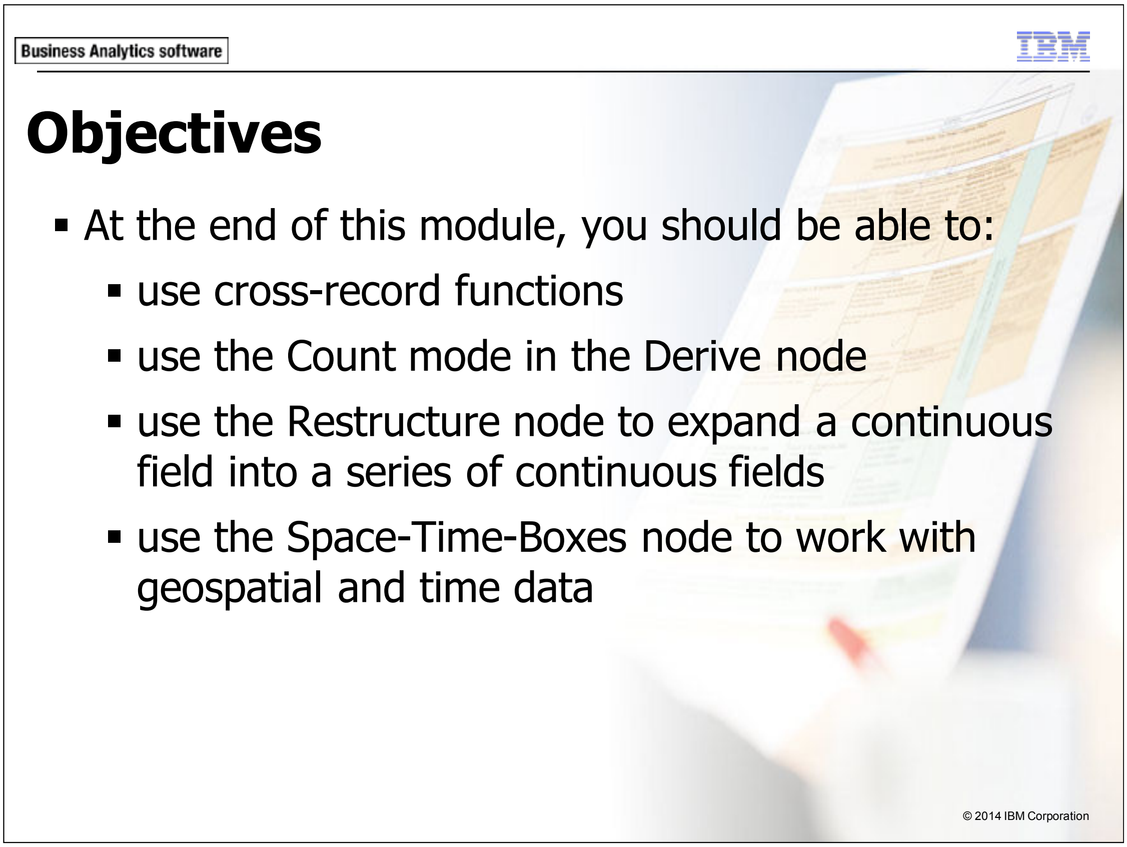
Note: The stream **workshop_data_transformations_completed.str**, located in the **02-Data_Transformations\Solutions** sub folder, provides a solution to the workshop.



Working with Sequence Data

IBM SPSS Modeler (v16)

Business Analytics software



Business Analytics software

IBM

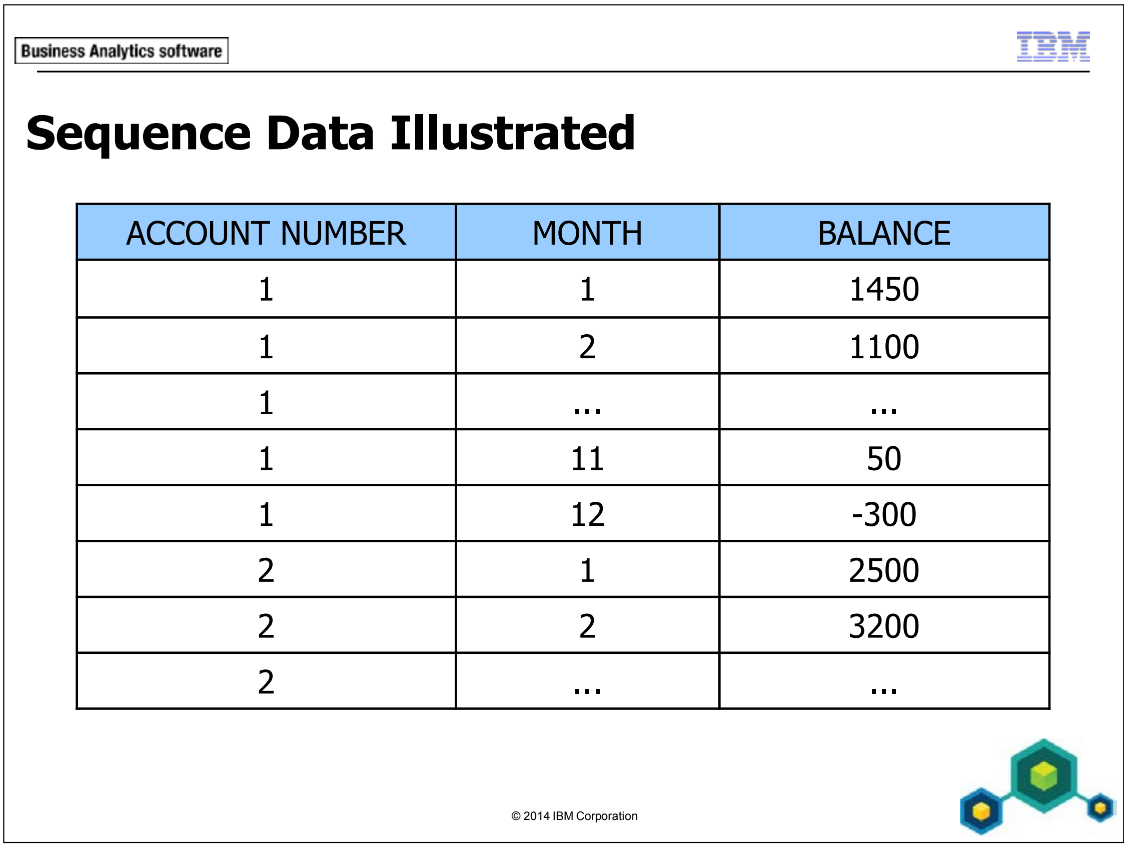
Objectives

- At the end of this module, you should be able to:
  - use cross-record functions
  - use the Count mode in the Derive node
  - use the Restructure node to expand a continuous field into a series of continuous fields
  - use the Space-Time-Boxes node to work with geospatial and time data

© 2014 IBM Corporation

Before reviewing this module you should be familiar with:

- working with MODELER (streams, nodes, SuperNodes, palettes)
- importing data (Var. File node)
- defining measurement levels, roles, blanks, and instantiating data (Type node)
- selecting fields and records (Filter node, Select node)
- CLEM and the Expression Builder
- deriving fields (Derive node)
- date and time-, conversion-, string-, statistical- and missing value functions
- the Aggregate node
- the SetToFlag node
- examining the data with graphs (Distribution node, Histogram node) and tables (Table node, Data Audit node, Statistics node, Matrix node, Means node)



Business Analytics software

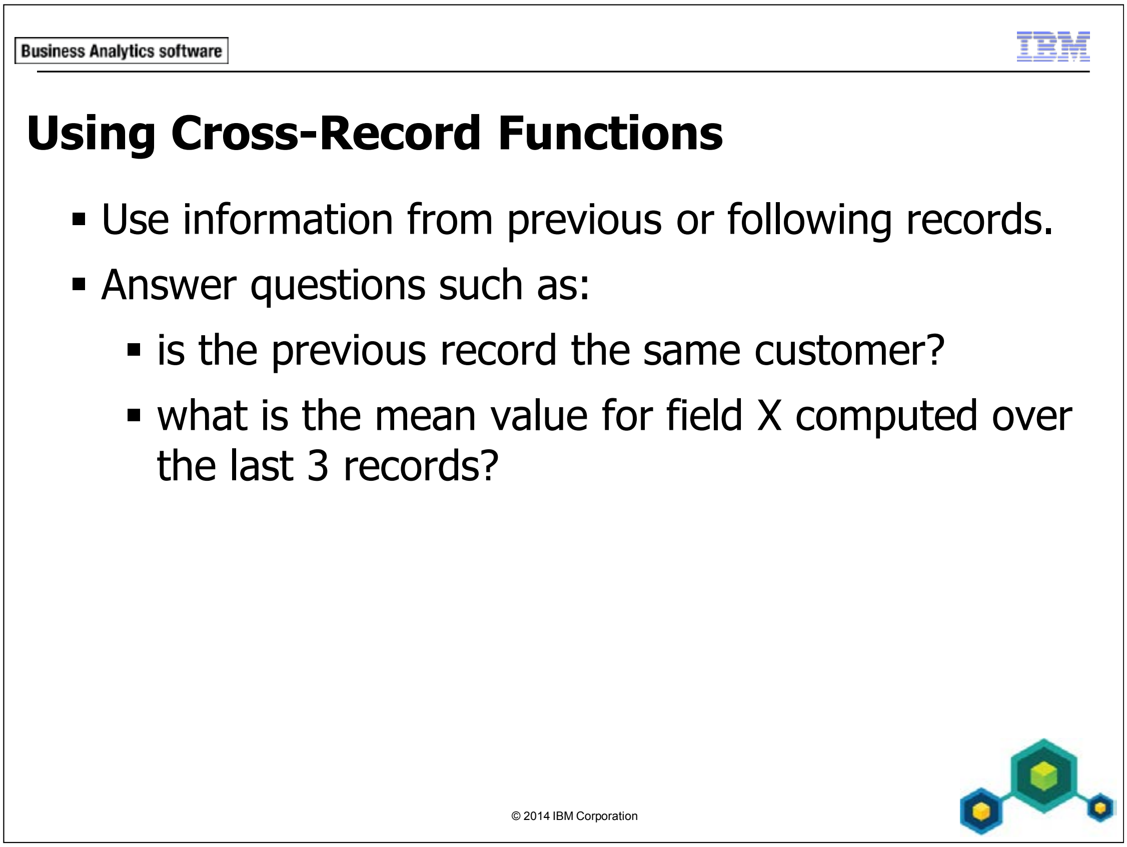
IBM

Sequence Data Illustrated

ACCOUNT NUMBER	MONTH	BALANCE
1	1	1450
1	2	1100
1	...	...
1	11	50
1	12	-300
2	1	2500
2	2	3200
2	...	...

In many situations each record in a dataset can be considered as an individual case, independent of all others. In such instances the order of records is unimportant for analyses. However, in some datasets the sequence of records is extremely important. This often occurs with time-structured data in which the order of the records represents a sequence of events. In this situation each record can be thought of as representing a snapshot at a particular instant in time. The instantaneous values may be of interest, or the way in which they vary over time.

This slide shows an example of sequence data: a bank holds data for individual accounts, with each month making up a record.



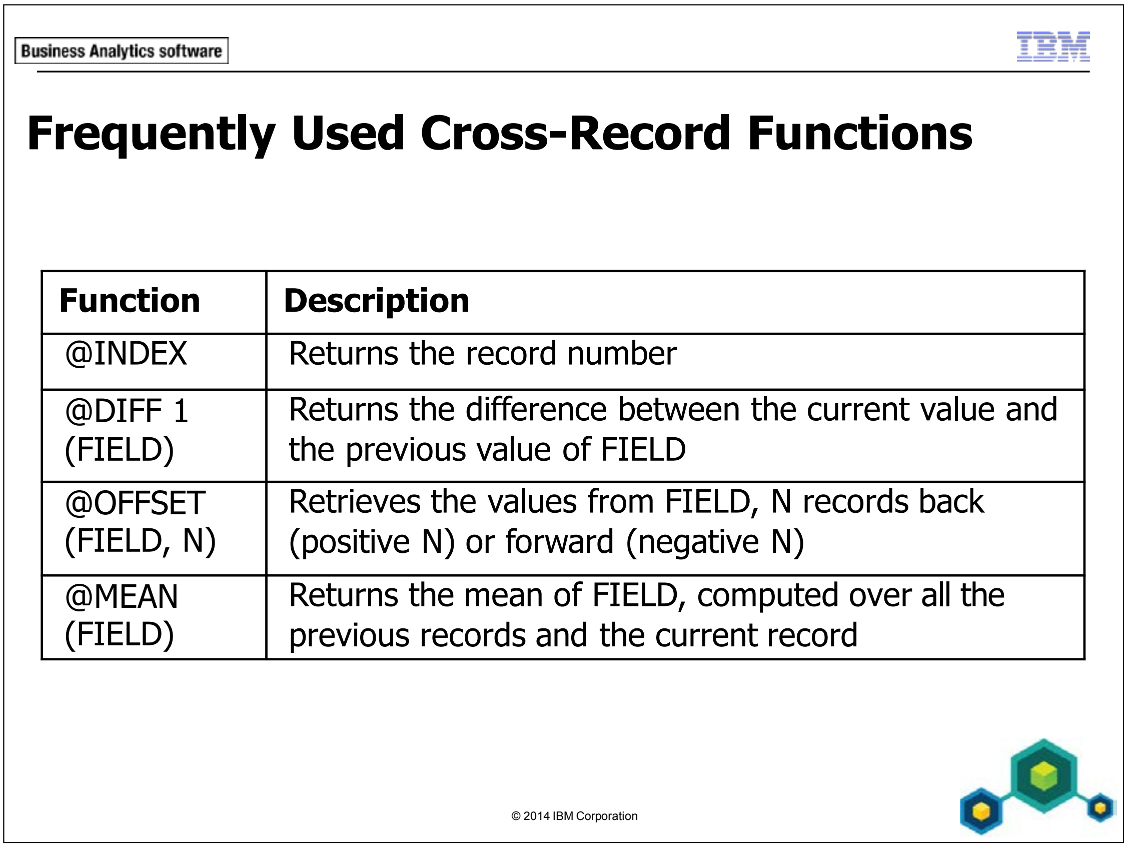
Business Analytics software

Using Cross-Record Functions

- Use information from previous or following records.
- Answer questions such as:
  - is the previous record the same customer?
  - what is the mean value for field X computed over the last 3 records?

Cross-record functions use information from previous or following records.

Typically these functions are used to check if the previous record is the same customer as the current record, or to compute a so-called moving average over the last N records. The latter is especially relevant when you have time series data. For example, a company may collect data on their monthly revenues and create a moving average time series that smooths (flattens the peaks and troughs) the original time series to see the trend more clearly.



Business Analytics software

IBM

Frequently Used Cross-Record Functions

Function	Description
@INDEX	Returns the record number
@DIFF 1 (FIELD)	Returns the difference between the current value and the previous value of FIELD
@OFFSET (FIELD, N)	Retrieves the values from FIELD, N records back (positive N) or forward (negative N)
@MEAN (FIELD)	Returns the mean of FIELD, computed over all the previous records and the current record

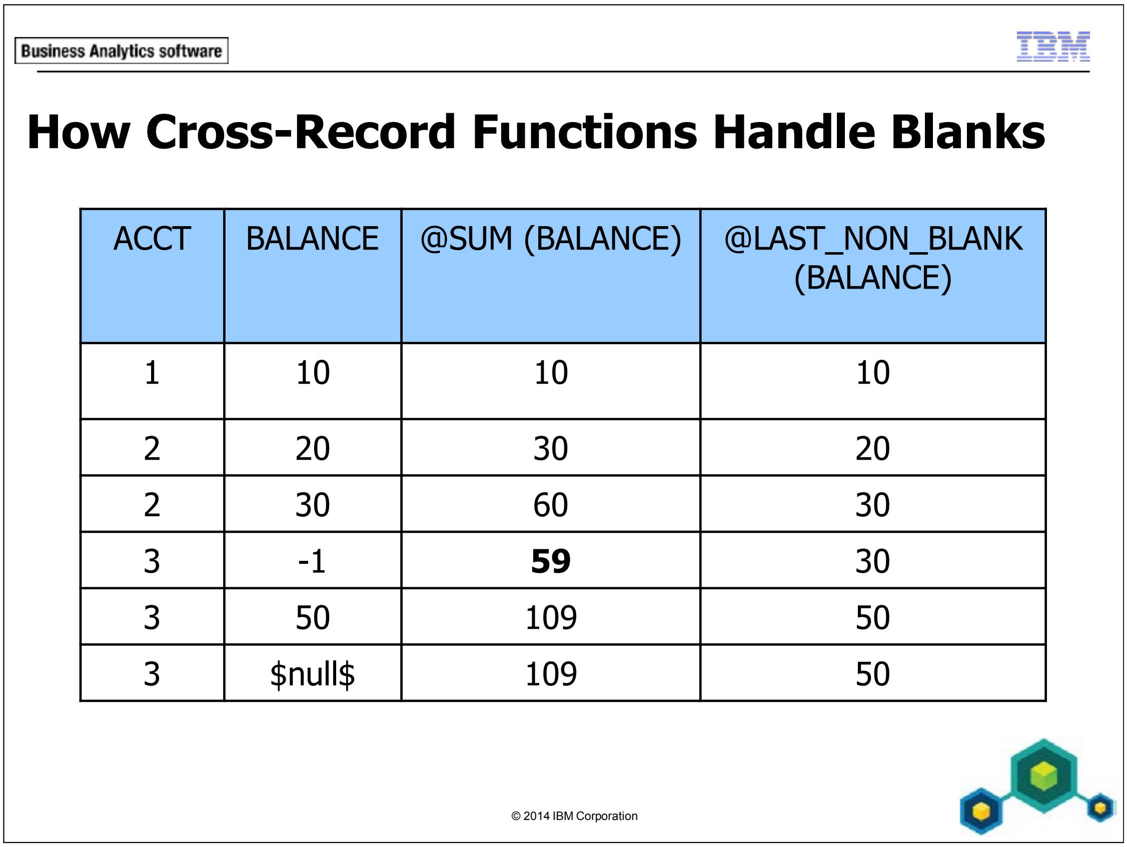
Cross-record functions are prefixed with the at sign (@) and are in upper case.

The simplest cross-record function is @INDEX, which returns the consecutive number of the record in the dataset, starting with 1. Creating a field storing the record number enables you to sort the records back into their original order at any moment downstream.

The @OFFSET function is used to retrieve values in previous or following records. The @OFFSET function requires a field name and an integer as arguments. The integer specifies how many records should be looked back (when N is a positive integer) or looked ahead (when N is a negative integer)

The @MEAN function returns the mean of a certain field, over all previous records and the current record, or over the previous N-1 records and the current record. In the latter case you need a second argument N for this function. The @MEAN function is frequently used to compute a moving average.

Similar to the MEAN function there are functions for the sum (@SUM) minimum (@MIN), maximum (@MAX), and standard deviation (@SDEV).



Business Analytics software

IBM

How Cross-Record Functions Handle Blanks

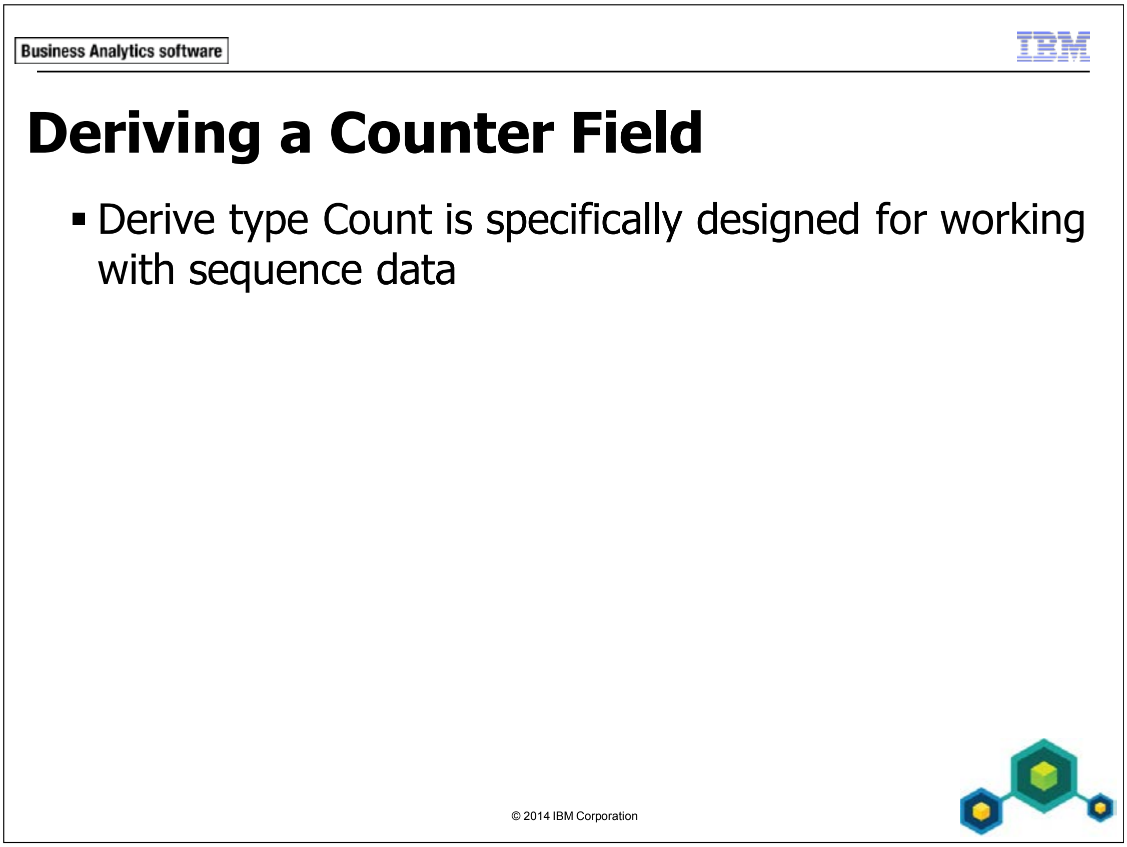
ACCT	BALANCE	@SUM (BALANCE)	@LAST_NON_BLANK (BALANCE)
1	10	10	10
2	20	30	20
2	30	60	30
3	-1	**59**	30
3	50	109	50
3	$null$	109	50

This slide shows an example of how cross-record functions handle blanks. The value -1 is declared as blank value for BALANCE. The undefined ($null$) value is also declared as blank for BALANCE.

The @SUM function does not take into account that -1 is declared as blank and includes the value in its computation. In general, you should be forewarned when you have user-defined blank values, because functions ignore the blank definition.

When the source field is undefined ($null$), the @SUM function will be equal to the sum of the previous record. This also suggests a solution for the issue that a cross-record function includes user-defined blanks in its computations: you can use the Filler node to replace the user-defined blank values with the undefined ($null$) value (refer to the *Data Transformations* module in this course for a presentation of the Filler node).

A cross-record function that specifically deals with blanks is @LAST_NON_BLANK. This function returns the last value that was not blank for a specified field. This function returns the value 30 for record #4, and 50 for record #6. If the undefined ($null$) value was not declared as blank for BALANCE, the result for record #6 would be $null$.



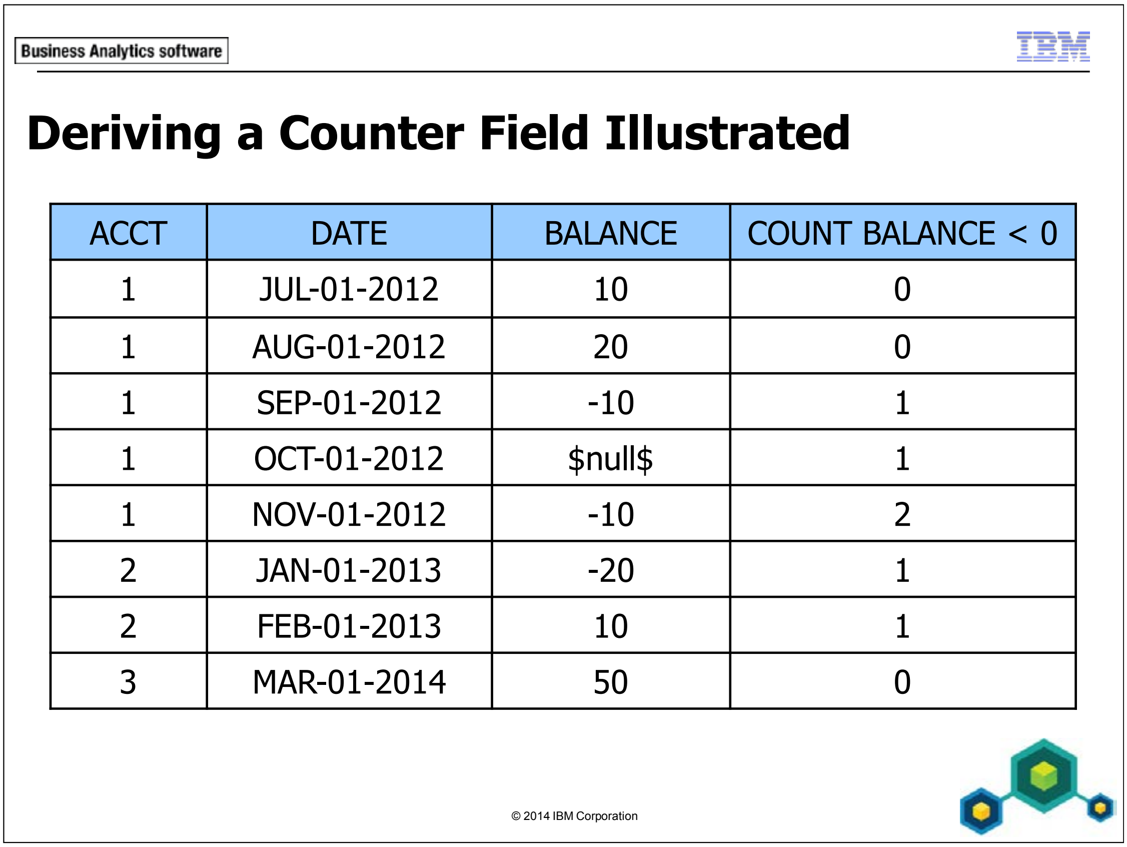
Business Analytics software

Deriving a Counter Field

- Derive type Count is specifically designed for working with sequence data

© 2014 IBM Corporation

The Derive type Count in the Derive node is specifically designed for working with sequence data. A field that is derived as Count calculates the number of times that a condition has been true.



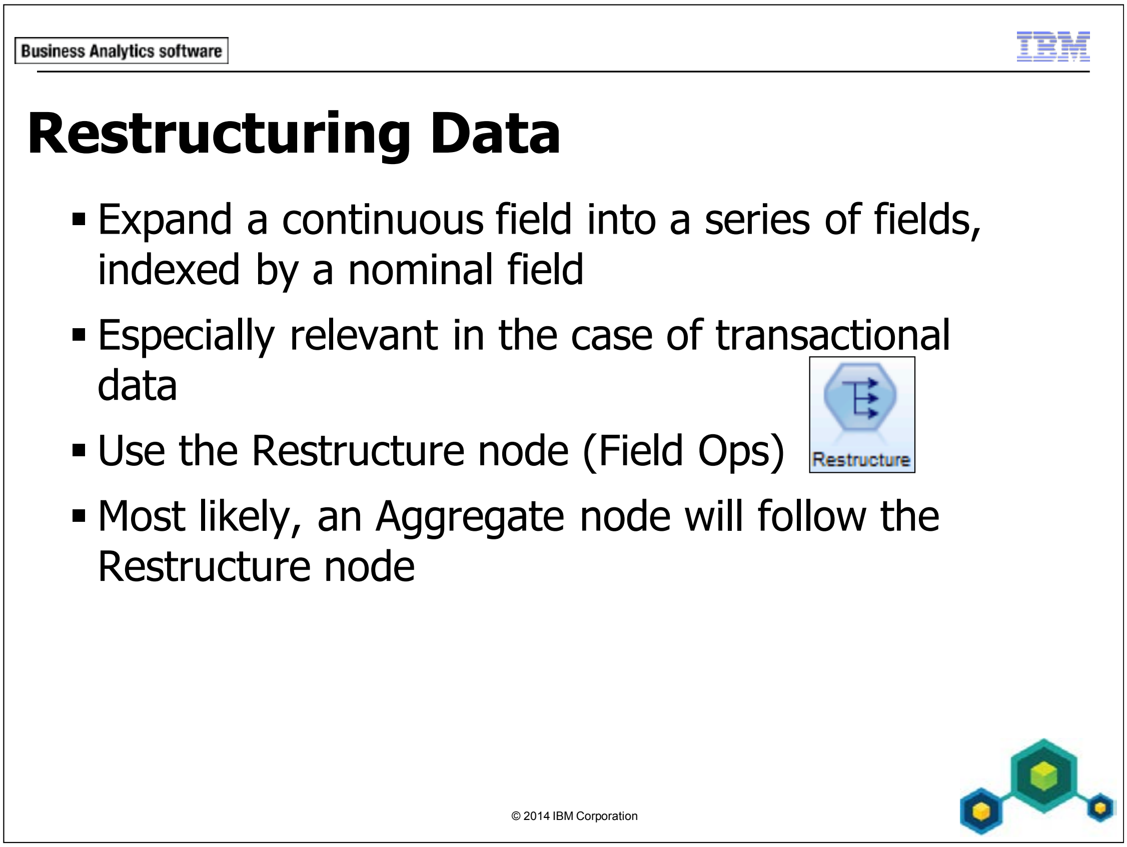
Business Analytics software

Deriving a Counter Field Illustrated

ACCT	DATE	BALANCE	COUNT BALANCE < 0
1	JUL-01-2012	10	0
1	AUG-01-2012	20	0
1	SEP-01-2012	-10	1
1	OCT-01-2012	$null$	1
1	NOV-01-2012	-10	2
2	JAN-01-2013	-20	1
2	FEB-01-2013	10	1
3	MAR-01-2014	50	0

The field COUNT BALANCE < 0 counts the number of times that the BALANCE is negative. The counter initializes at 0, and is incremented by 1 when a negative value is encountered. When the value is positive or undefined ($null$) the counter remains the same.

Notice that the field resets to 0 when a new account is encountered.



Business Analytics software

Restructuring Data

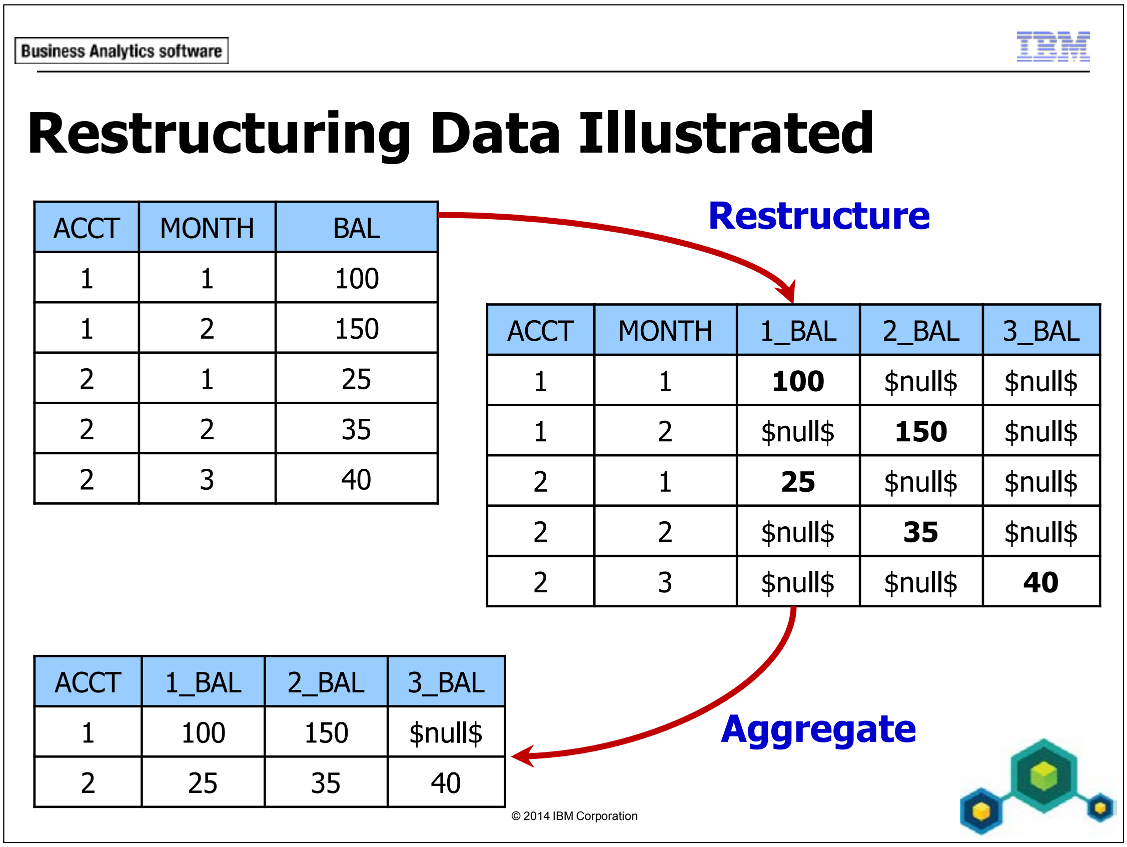
- Expand a continuous field into a series of fields, indexed by a nominal field
- Especially relevant in the case of transactional data
- Use the Restructure node (Field Ops)
- Most likely, an Aggregate node will follow the Restructure node

Restructure

Occasionally you will have a data structure that is not suited for the analyses that you want to run. For example, a customer may have purchased products at different times, and each purchase is a record. Datasets of this type, where not persons but transactions define unique records, are called transactional.

In transactional datasets you may want to know if the revenue for product A is related to the revenue for product B, but to answer this question you need a data structure that has only one record per customer, and the revenue for A and the revenue for B as fields.

The Restructure node, located in the Field Ops palette, is the first step to arrive at the required data structure. This node spreads the information contained in one field out into multiple fields. The next step is to aggregate the data. This cannot be done in the Restructure node itself, but requires a separate Aggregate node.



Business Analytics software

Restructuring Data Illustrated

ACCT	MONTH	BAL
1	1	100
1	2	150
2	1	25
2	2	35
2	3	40

Restructure

ACCT	MONTH	1_BAL	2_BAL	3_BAL
1	1	**100**	$null$	$null$
1	2	$null$	**150**	$null$
2	1	**25**	$null$	$null$
2	2	$null$	**35**	$null$
2	3	$null$	$null$	**40**

Aggregate

ACCT	1_BAL	2_BAL	3_BAL
1	100	150	$null$
2	25	35	40

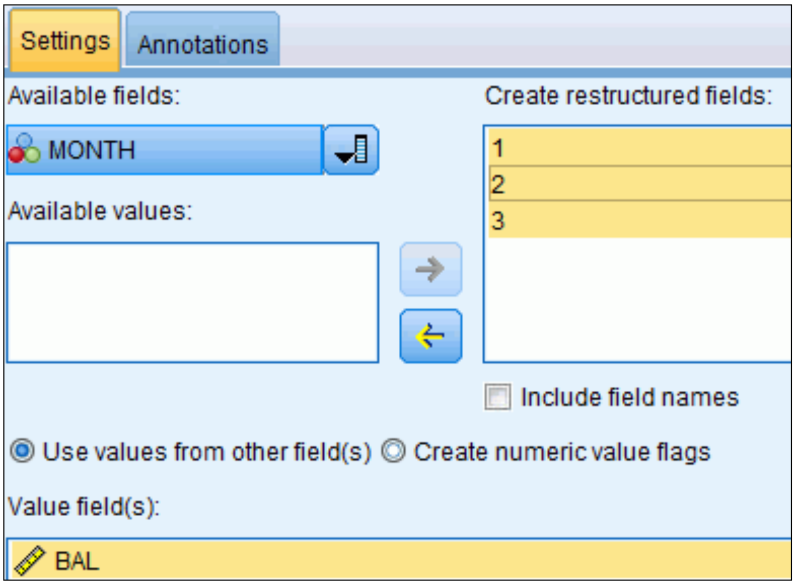
This slide illustrates how a transactional dataset sis transformed into a dataset with one record per customer.

The Restructure operation expands the BAL field into a number of fields, which are indexed by month. For example, 1_BAL stores the balance for the first month.

Restructure does not affect the data structure, because a customer still has as many records as he or she had transactions. Thus, an Aggregate node is required in the second step to create a dataset with one record per customer.

Business Analytics software

Exploring the Restructure Dialog Box



Available fields: MONTH

Available values:

Create restructured fields:
1
2
3

☐ Include field names

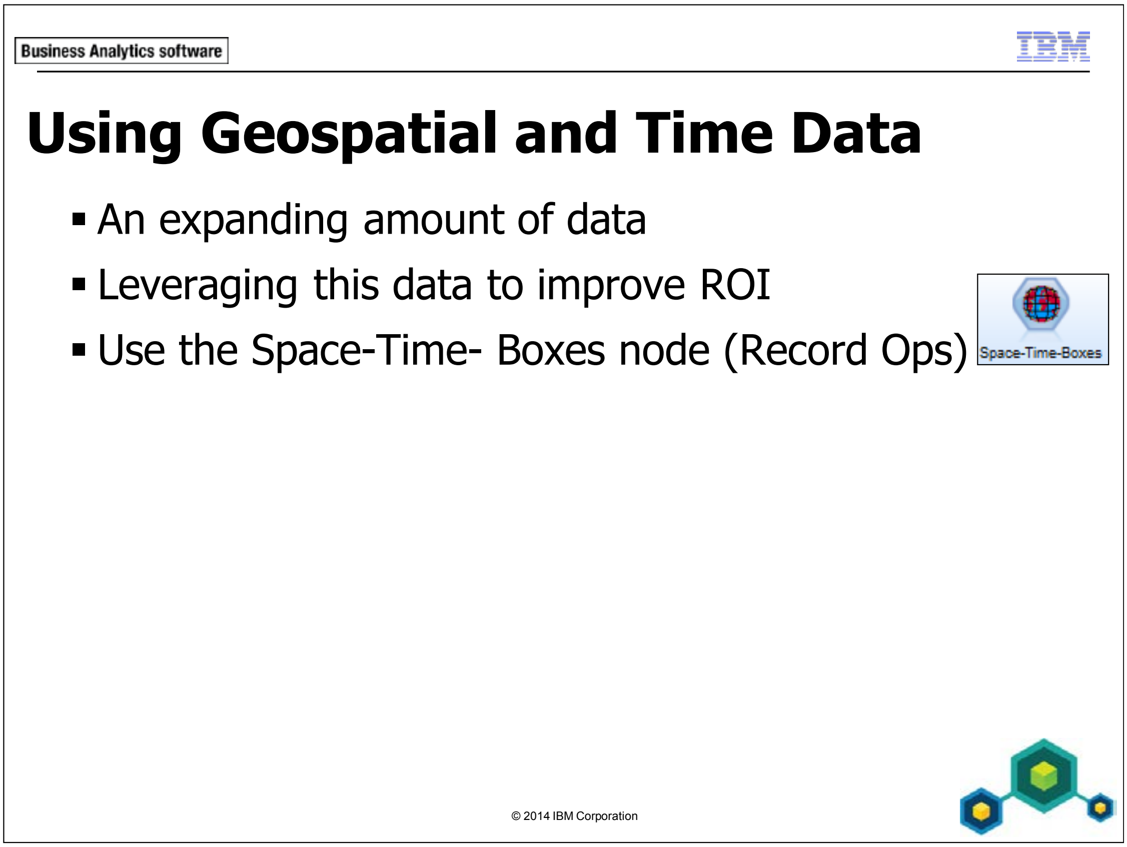
● Use values from other field(s) ○ Create numeric value flags

Value field(s): BAL

On the Settings tab, under Available fields, select the field that serves as index field for the restructure operation. Only categorical (flag, nominal, ordinal) fields are eligible. If you need a continuous field as index field, you need to change the field's measurement level to nominal in an upstream Type node.

The area under Available values will be populated with the categories of the selected categorical field, provided that the field is instantiated. When there are no available values you need to instantiate the field in an upstream Type node.

Move the values for which you want to create fields to the Create restructured fields area. You can enable the Include field names option when you want the name of the index field as a prefix for the new fields. The Restructure dialog box offers two modes. The Create numeric flags option creates flag field with integer storage. This feature mimics the SetToFlag node, but the SetToFlag node offers more flexibility, such as an aggregate in the node itself. Thus, you probably will prefer the SetToFlag node to create flag fields for the categories. Select the Use values from other field(s) option if a continuous field must be expanded into a series of new fields indexed by the values of the categorical field. In this case, select the continuous field(s) that must be expanded into a series of new continuous fields.



Business Analytics software

Using Geospatial and Time Data

- An expanding amount of data
- Leveraging this data to improve ROI
- Use the Space-Time- Boxes node (Record Ops)

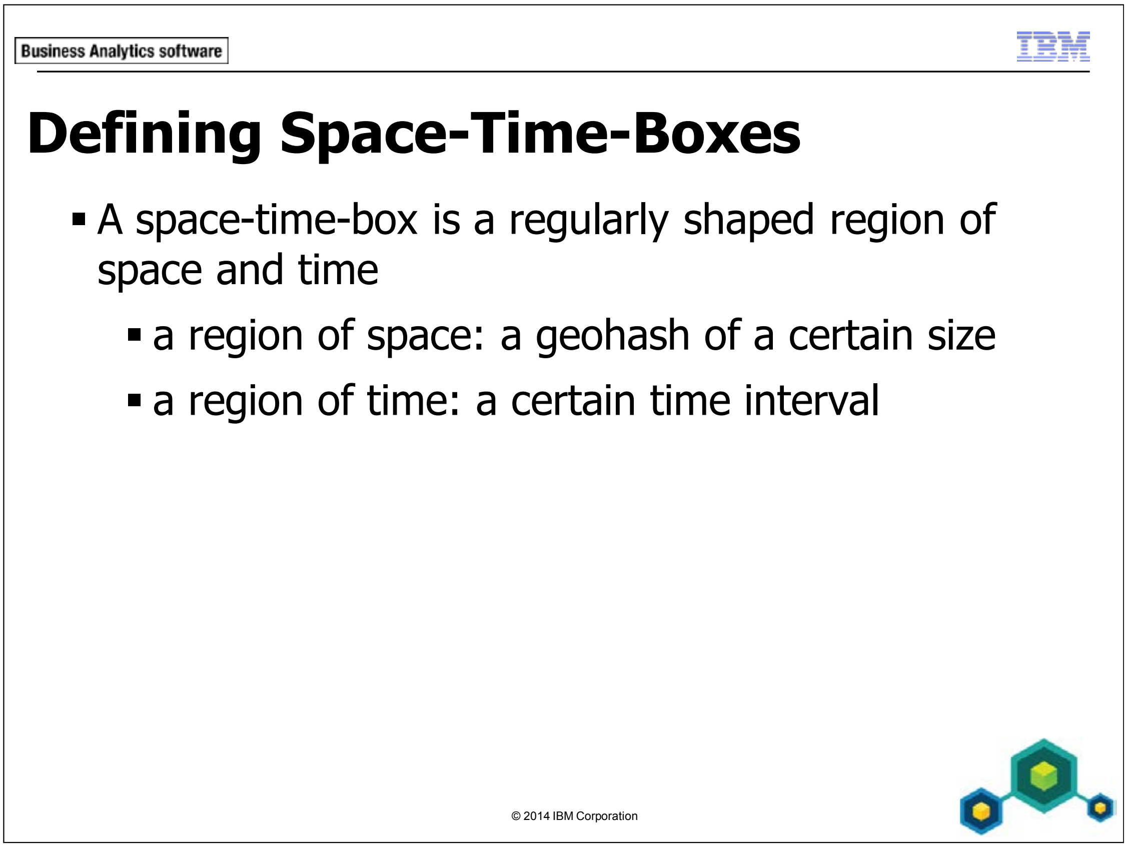
Space-Time-Boxes

Today, data are everywhere, and the analysis of big data is critical to a company's success.

As an example, it is estimated that mobile devices generate about 600 billion transactions a day. Every call, text message, e-mail and data transfer creates a data point with space/time coordinates. You could use this space and time information to improve confidence that two entities are the same person because they are virtually in the same place at the same time, and target this person in location-based promotions.

The first step to leveraging this information is to understand how geospatial and time data can be analyzed. MODELER's Space-Time-Boxes node is designed for this.

Note: There are a growing number of privacy laws related to location data emerging around the world. Thinking about the legal and policy aspects in such projects is considered a best practice. Privacy by Design (PbD) is an approach you may want to consider.



Business Analytics software

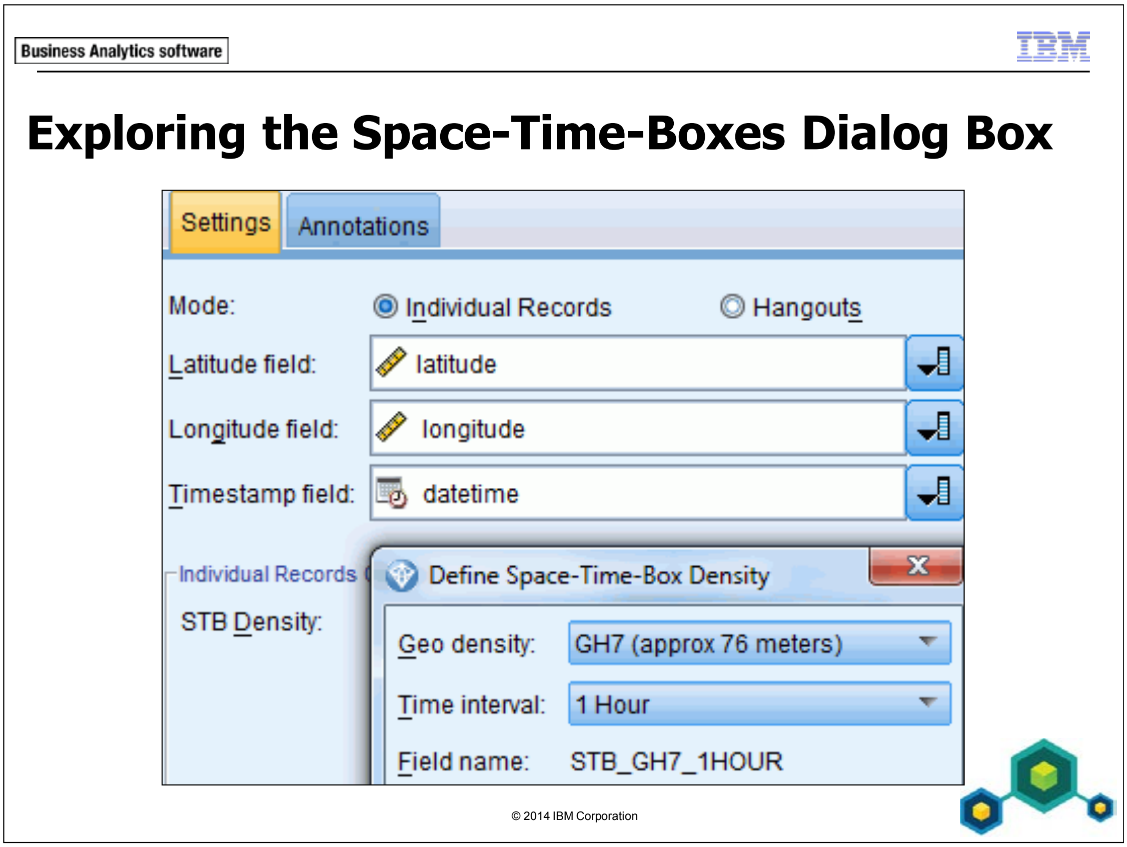
Defining Space-Time-Boxes

- A space-time-box is a regularly shaped region of space and time
  - a region of space: a geohash of a certain size
  - a region of time: a certain time interval

© 2014 IBM Corporation

Space-time-boxes are used to describe where and when an entity is, by combining a spatial grid with a time interval.

The spatial grid is defined by so-called geohashes. Geohashes define an area of a certain size and have a hierarchical structure. On the highest level earth is divided into 32 boxes, each of them is subdivided into 32 smaller boxes, and so forth. For example, geohash u is one of the 32 areas at the highest level and covers a part of Europe. Geohash u0 is one of the 32 boxes within geohash u and covers a great part of France, and geohash u09 covers Paris. At the deepest level, staying in Paris, u09tunqbwevy is the entrance of the Eiffel Tower in the south pillar. Thus, the longer the geohash is, the smaller is the size of the area and the more precise the location. In the same way, time can be thought of as being hierarchically divided into years, months, days, and so forth, down to the deepest level of seconds. A space-time-box encodes a location (given by latitude and longitude) and a timestamp into a geohash appended with the start and the end of the time interval. For example, coordinates [48.85767317994797, 02.2947075963020324] (the entrance of the Eiffel Tower in the south pillar), timestamp 2014-APR-02 12:23:58 is encoded into u09tunq|2014-04-02 12:00:00|2014-04-02 13:00:00 (using a density of approximately 76 meters for the geohash, and a one hour time interval).



Business Analytics software

Exploring the Space-Time-Boxes Dialog Box

Mode: ● Individual Records ○ Hangouts

Latitude field: latitude

Longitude field: longitude

Timestamp field: datetime

Individual Records

STB Density:

Define Space-Time-Box Density

Geo density: GH7 (approx 76 meters)

Time interval: 1 Hour

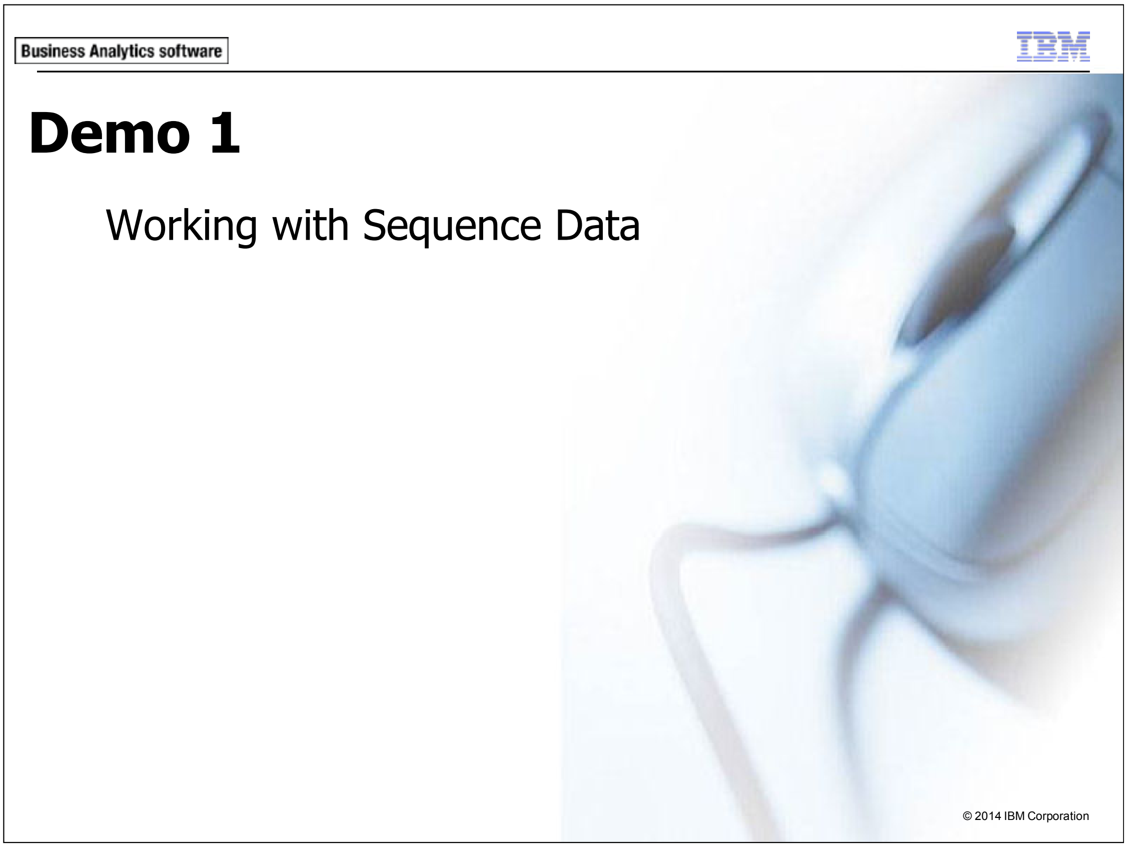
Field name: STB_GH7_1HOUR

The Space-Time-Boxes dialog box provides you with two modes:

- Individual Records. Based on the record's spatial coordinates and timestamp, a field is created with the geohash that encloses the location, and the start time and end time of the timestamp field. Typically, this mode is used in conjunction with the Aggregate node to determine how many entities exist at specific densities.
- Hangouts. A hangout can be thought of as a location and/or time in which an entity is continually or repeatedly found. This option will aggregate the data, with the combination of entity and space-time-boxes defining the records. (This is the reason why this node is located in the Record Ops palette.)

Both modes require the same basic specifications. The latitude and longitude field define the coordinates, the timestamp field defines the date and/or time. The STB Density menu enables you to set the geo density and the time interval.

When the Hangouts option is selected, the minimum number of events must be specified for the entity to be considered to be hanging out in a space-time-box. Also, the minimum dwell time (the minimum time that the entity stays in the same location) must be specified.



Business Analytics software

Demo 1

Working with Sequence Data

© 2014 IBM Corporation

The following (synthetic) files are used to demonstrate how you work with sequence data:

- **year_balances.txt**: A text file which contains end-of- month account balances for 12 customers. One year of data is held on each individual customer.
- **taxi cab customers app data.txt**: A text file storing for phones that have opted into a "smart-taxi" phone app.

Both files are located in **C:\Train\0A055**.

Before you begin with the demo, ensure that:

- You have started MODELER.
- You have set MODELER's working folder. (In MODELER, select **File\Set Directory**. When you are working in an IBM operated environment, browse to the folder **C:\Train\0A055** and then click **Set** to select it. When you are not working in an IBM operated environment, ask your trainer for the folder name.)

Demo 1: Working with Sequence Data

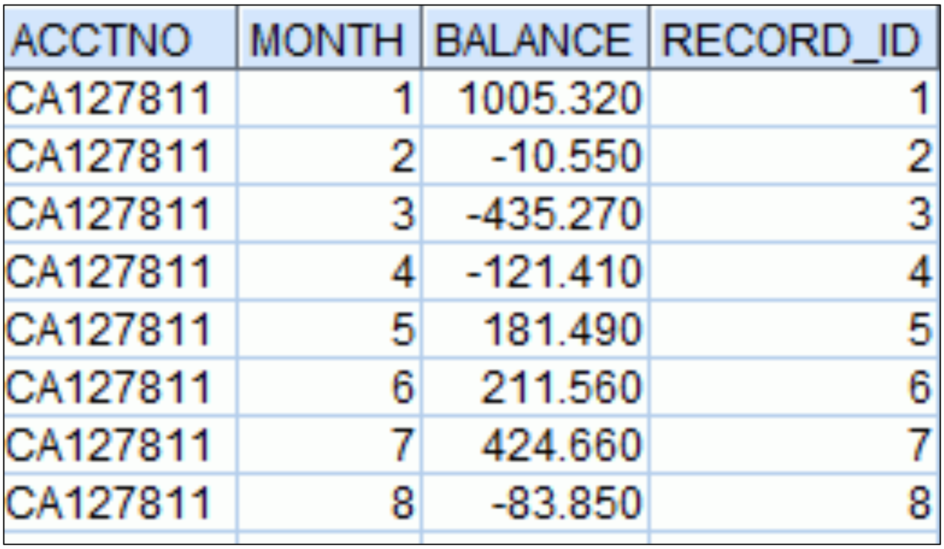
Purpose:
You will apply various transformations to sequence data, such as creating a record identifier and a moving average. You will restructure a transactional dataset into a dataset with one record per customer, which is the required data structure for correlational analyses. Finally, you will use the Space-Time-Boxes node to determine the most popular place and time where customers hang out who opted in on a cell phone app.

Task 1. Create a record identifier.

You will derive a field that serves as record identifier. Such a field can be used to resort the data downstream back into the original order.

1. Use a **Var. File** node to import data from **year_balances.txt**.
2. Add a **Derive** node downstream from the **Var. File** node.
3. Edit the **Derive** node, and then:
   - for **Derive field**, type **RECORD_ID**
   - for **Formula**, enter **@INDEX**
   - click **Preview**

   A section of the results appear as follows:



ACCTNO	MONTH	BALANCE	RECORD_ID
CA127811	1	1005.320	1
CA127811	2	-10.550	2
CA127811	3	-435.270	3
CA127811	4	-121.410	4
CA127811	5	181.490	5
CA127811	6	211.560	6
CA127811	7	424.660	7
CA127811	8	-83.850	8

A field has been added to the data that uniquely identifies the records.

4. Close the **Preview** output window, and then close the **Derive** dialog box.

   Leave the stream open for the next task.

Task 2. Create a moving average.

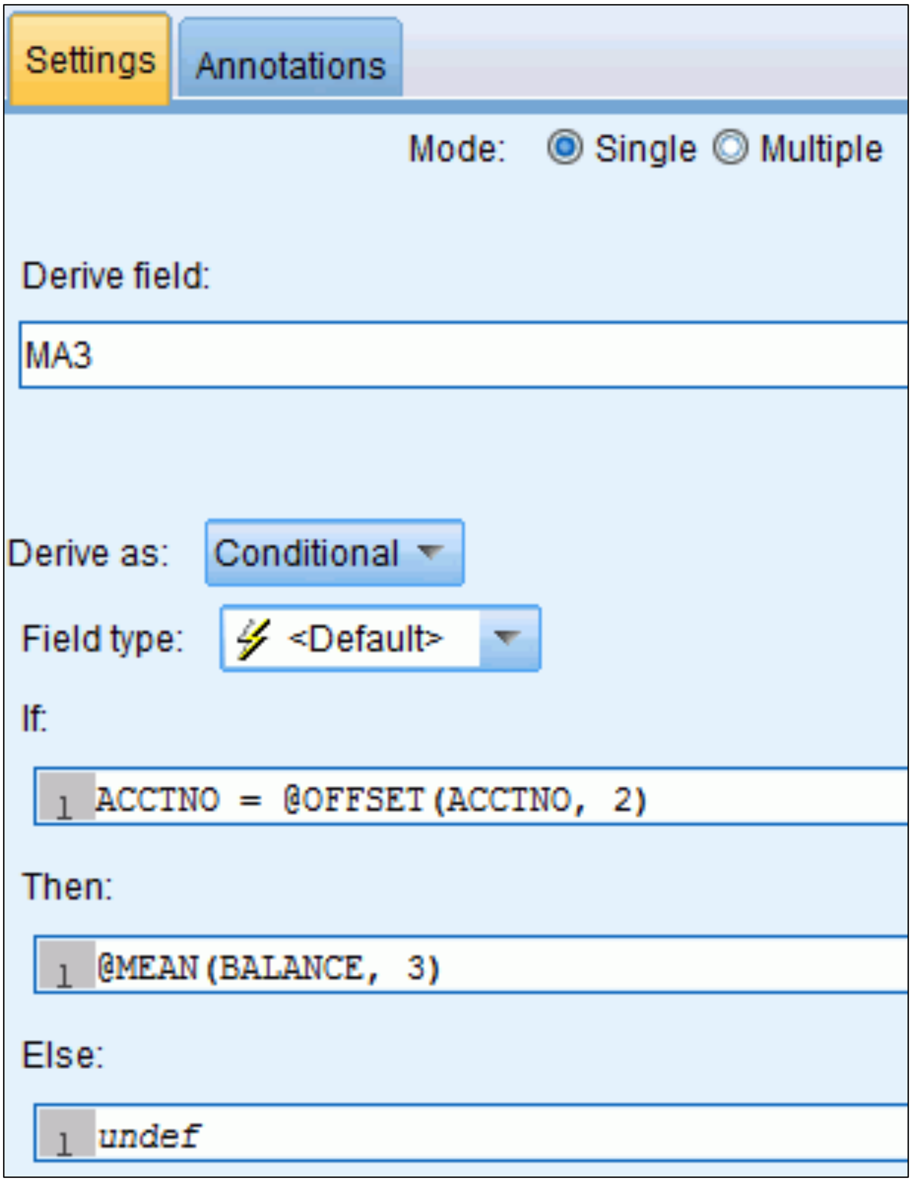
You will create a three-record moving average for BALANCE, which enables you to determine a trend (the latter is not pursued here). The value will reinitialize when a new account is encountered.

The mean will not be calculated unless the second record before the current one has the same account number. This implies that the mean will not be calculated for the first two months for each account, since the data are sorted by ACCTNO and MONTH.

You will build from the stream that you created in the previous task.

1. Add a **Derive** node downstream from the **Derive** node named **RECORD_ID**.
2. Edit the **Derive** node, and then:
   - for **Derive** field, type **MA3**
   - for **Derive as**, select **Conditional**
   - for **If**, enter **ACCTNO = @OFFSET(ACCTNO, 2)**
   - for **Then**, enter **@MEAN(BALANCE, 3)**
   - for **Else**, type **undef**

A section of the results appear as follows:



Mode: ● Single ○ Multiple

Derive field:

MA3

Derive as: Conditional

Field type: <Default>

If:

```
ACCTNO = @OFFSET(ACCTNO, 2)
```

Then:

```
@MEAN(BALANCE, 3)
```

Else:

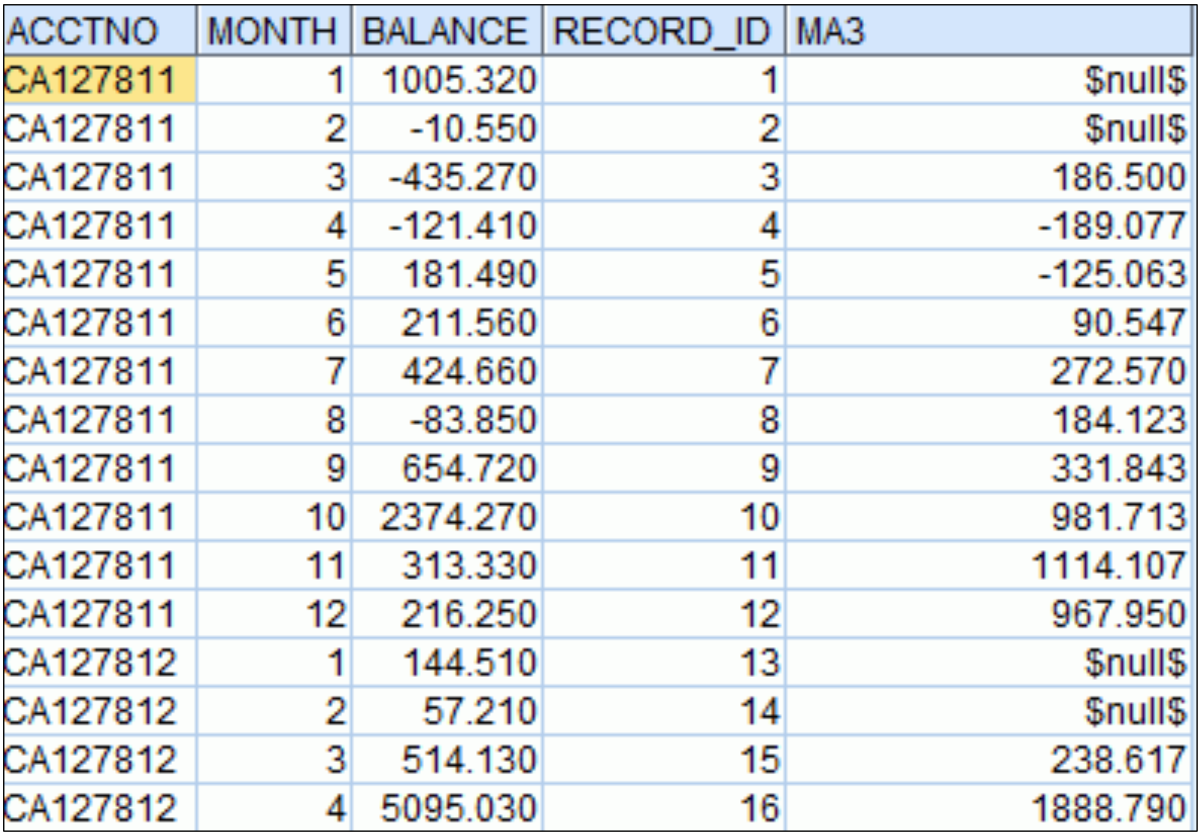
```
undef
```

3. Close the **Derive** dialog box.

This material is meant for IBM Academic Initiative purposes.

4. Add a **Table** node downstream from the **Derive** node, and then run the **Table** node.

   A section of the results appear as follows:



ACCTNO	MONTH	BALANCE	RECORD_ID	MA3
CA127811	1	1005.320	1	$null$
CA127811	2	-10.550	2	$null$
CA127811	3	-435.270	3	186.500
CA127811	4	-121.410	4	-189.077
CA127811	5	181.490	5	-125.063
CA127811	6	211.560	6	90.547
CA127811	7	424.660	7	272.570
CA127811	8	-83.850	8	184.123
CA127811	9	654.720	9	331.843
CA127811	10	2374.270	10	981.713
CA127811	11	313.330	11	1114.107
CA127811	12	216.250	12	967.950
CA127812	1	144.510	13	$null$
CA127812	2	57.210	14	$null$
CA127812	3	514.130	15	238.617
CA127812	4	5095.030	16	1888.790

   For each customer, the first two months are undefined ($null$,) and the records for month 3 through 12 store the moving average computed over the previous two months and the current month.

5. Close the **Table** output window.

   Leave the stream open for the next task.

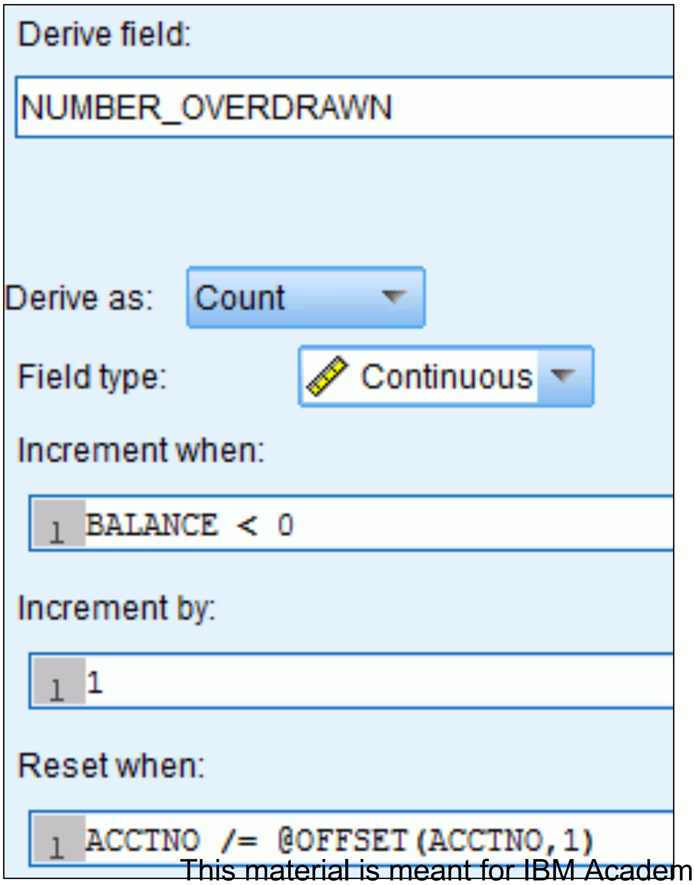
Task 3. Count the number of negative balances.

You will derive a field that counts the number of times to date that the account has been overdrawn (balance falls below 0). The count will be reset when the first entry of a new account is read, more specifically, if the account number differs from that of the previous record.

You will build from the stream that you created in the previous task.

1. Add a **Derive** node downstream from the **Derive** node named **MA3**.
2. Edit the **Derive** node, and then:
   - for **Derive field**, enter **NUMBER_OVERDRAWN**
   - for **Derive as**, select **Count**
   - for **Initial value**, ensure that the value is **0**
   - for **Increment when**, enter **BALANCE < 0**
   - for **Increment by**, ensure that the value is **1**
   - for **Reset when**, enter **ACCTNO /= @OFFSET(ACCTNO,1)**

   A section of the results appear as follows:



Derive field: NUMBER_OVERDRAWN

Derive as: Count

Field type: Continuous

Increment when:

```
1 BALANCE < 0
```

Increment by:

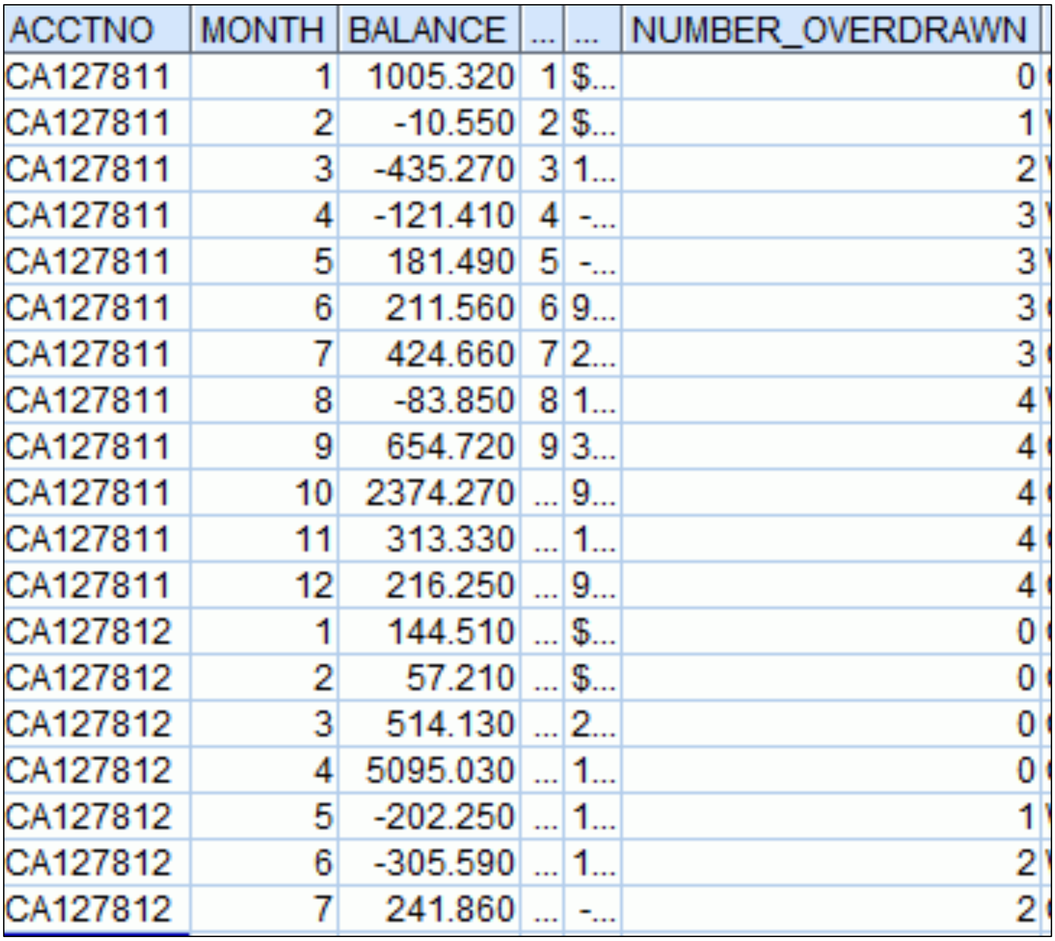
```
1 1
```

Reset when:

```
1 ACCTNO /= @OFFSET(ACCTNO,1)
```

3. Close the **Derive** dialog box.
4. Add a **Table** node downstream from the **Derive** node, and then run the **Table** node.

   A section of the results appear as follows:



ACCTNO	MONTH	BALANCE	...	...	NUMBER_OVERDRAWN
CA127811	1	1005.320	1	$...	0
CA127811	2	-10.550	2	$...	1
CA127811	3	-435.270	3	1...	2
CA127811	4	-121.410	4	-...	3
CA127811	5	181.490	5	-...	3
CA127811	6	211.560	6	9...	3
CA127811	7	424.660	7	2...	3
CA127811	8	-83.850	8	1...	4
CA127811	9	654.720	9	3...	4
CA127811	10	2374.270	...	9...	4
CA127811	11	313.330	...	1...	4
CA127811	12	216.250	...	9...	4
CA127812	1	144.510	...	$...	0
CA127812	2	57.210	...	$...	0
CA127812	3	514.130	...	2...	0
CA127812	4	5095.030	...	1...	0
CA127812	5	-202.250	...	1...	1
CA127812	6	-305.590	...	1...	2
CA127812	7	241.860	...	-...	2

   The first customer has overdrawn his/her account 4 times. The second account starts with 0 again.

   Note: It would be interesting to examine what the mean is for the new field. To answer this question, however, you need another data structure (one record per account), so you would need an Aggregate first, saving the maximum value of NUMBER_OVERDRAWN per account. This is left as an exercise.

5. Close the **Table** output window.

   Leave the stream open for the next task.

Task 4. Restructure data.

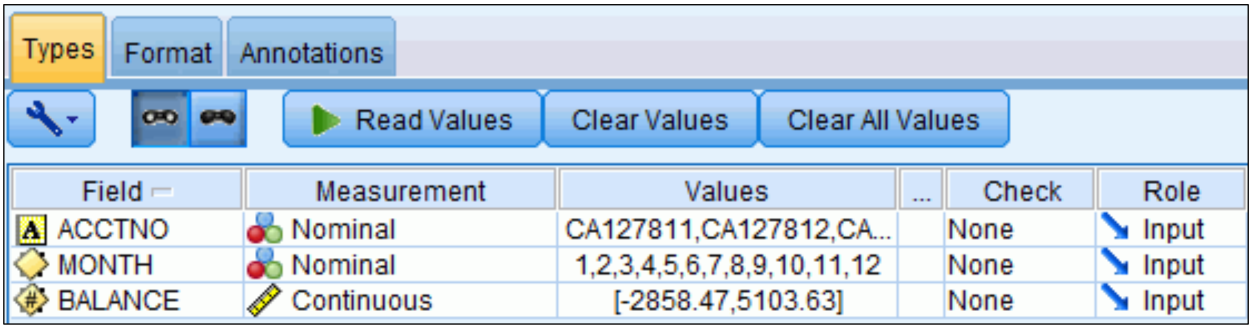
In this task you will create a dataset with one record per account and the monthly balances as fields. Hereto, you will start with a Restructure node. This node will spread out the records into fields, but will not group the records into one record per account. Thus, the Restructure node is followed an Aggregate node which brings the data to the required unit of analysis.

It is important to note that the index field that is used in the Restructure node to spread records into fields, MONTH, needs to be categorical. At this point, however, month is a continuous field because it has integer storage. Thus, you will first set the measurement level for MONTH to categorical.

You will build from the stream that you created in the previous task.

1. Add a **Type** node downstream from the **Derive** node named **NUMBER_OVERDRAWN**.
2. Edit the **Type** node, and then:
   - set the measurement level for **MONTH** to **Categorical**
   - click **Read Values**

   A section of the results appear as follows.

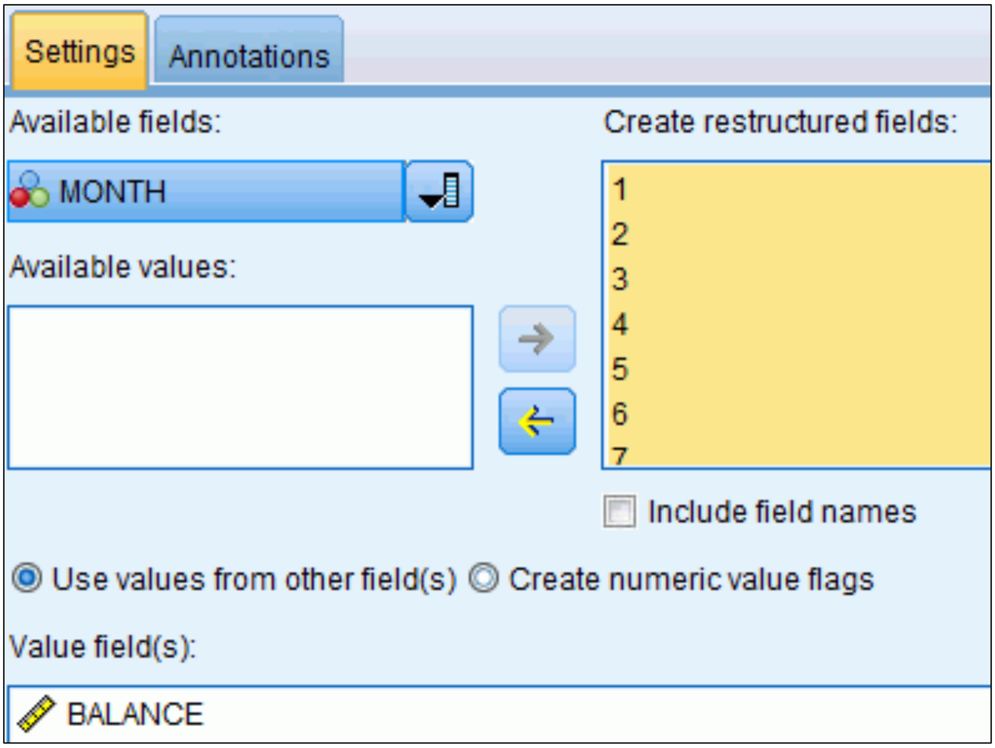


Field	Measurement	Values	...	Check	Role
A ACCTNO	Nominal	CA127811,CA127812,CA...		None	Input
MONTH	Nominal	1,2,3,4,5,6,7,8,9,10,11,12		None	Input
BALANCE	Continuous	[-2858.47,5103.63]		None	Input

   The MONTH field is instantiated as a nominal field, with values 1 through 12.
3. Close the **Type** dialog box.
4. Add a **Restructure** node (Field Ops) downstream from the **Type** node.

5. Edit the **Restructure** node, and then:
   - for **Available fields**, select **MONTH**
   - select all available values, and click them to **Create restructured fields**
   - disable the **Include field names** option
   - for **Value field(s)**, select **BALANCE**

   A section of the results appear as follows:



Available fields: MONTH

Available values:

Create restructured fields: 1 2 3 4 5 6 7

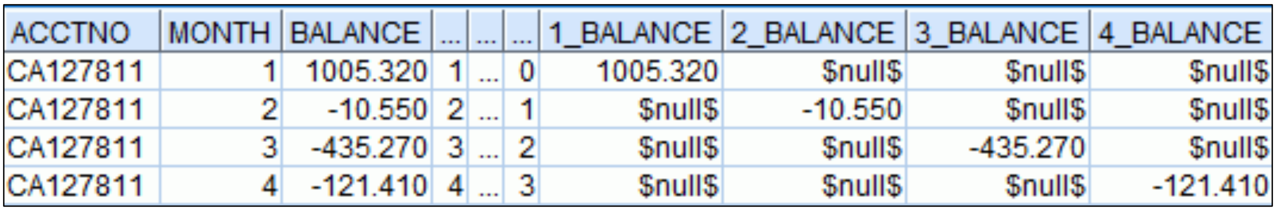
☐ Include field names

● Use values from other field(s) ○ Create numeric value flags

Value field(s): BALANCE

6. Click **Preview**.

   A section of the results appear as follows:



ACCTNO	MONTH	BALANCE	...	...	...	1_BALANCE	2_BALANCE	3_BALANCE	4_BALANCE
CA127811	1	1005.320	1	...	0	1005.320	$null$	$null$	$null$
CA127811	2	-10.550	2	...	1	$null$	-10.550	$null$	$null$
CA127811	3	-435.270	3	...	2	$null$	$null$	-435.270	$null$
CA127811	4	-121.410	4	...	3	$null$	$null$	$null$	-121.410

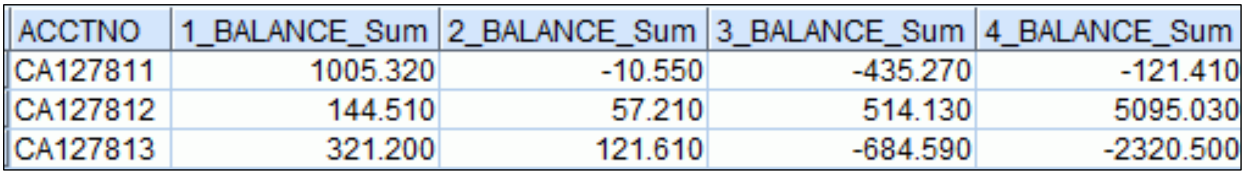
   The fields storing the monthly balances have been created, but there are still 12 records per account. Aggregating cannot be done in the Restructure node (in this respect it differs from the SetToFlag node), so you will need a separate Aggregate node downstream from the Restructure node.

7. Close the **Preview** output window, and then close the **Restructure** dialog box.

8. Add an **Aggregate** node (Record Ops) downstream from the **Restructure** node.
9. Edit the **Aggregate** node, and then:
   - for **Key fields**, select **ACCTNO**
   - under **Basic Aggregates**, **Aggregate fields**, select **1_BALANCE** to **12_BALANCE**
   - for **Aggregate fields**, check the **Sum** check boxes for **1_BALANCE** to **12_BALANCE**

     Note: The Sum, Mean, Min, and Max statistic will all give the same result, because there is only one valid value per ACCTNO for each month.
   - click **Preview**

   A section of the results appear as follows:



ACCTNO	1_BALANCE_Sum	2_BALANCE_Sum	3_BALANCE_Sum	4_BALANCE_Sum
CA127811	1005.320	-10.550	-435.270	-121.410
CA127812	144.510	57.210	514.130	5095.030
CA127813	321.200	121.610	-684.590	-2320.500

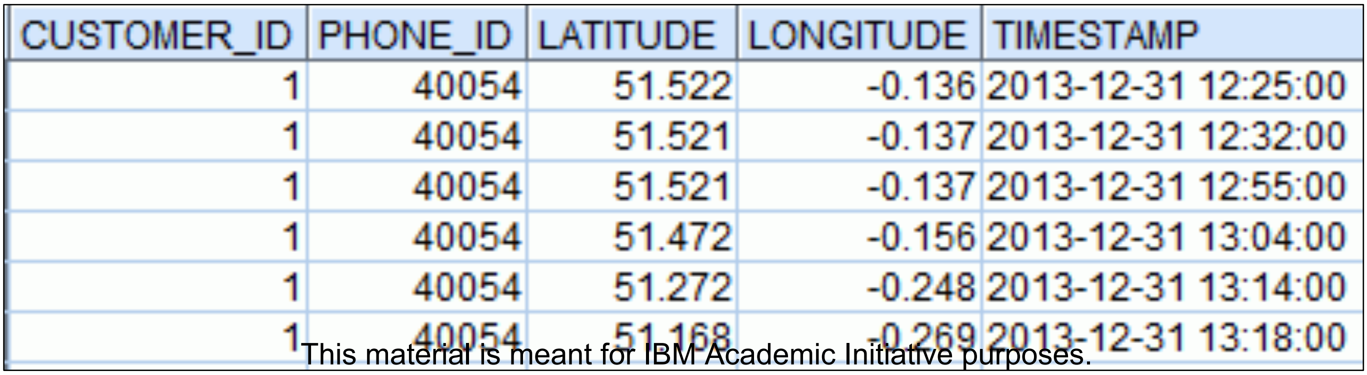
   There is one record per account, with 12 fields storing the monthly balances, as required.

10. Close the **Preview** output window, and then close the **Aggregate** dialog box.

    Leave the stream open for the next task.

Task 5. Analyzing geospatial and time data, using the Individual Records mode.

As an example of using geospatial and time data, consider a taxi cab company that wants to know the demand on New Year's Eve. The company has geospatial and time data for phones that have opted into their smart-taxi phone app. Each record is a geospatial ping from the app. The next figure shows the first records of the dataset.



CUSTOMER_ID	PHONE_ID	LATITUDE	LONGITUDE	TIMESTAMP
1	40054	51.522	-0.136	2013-12-31 12:25:00
1	40054	51.521	-0.137	2013-12-31 12:32:00
1	40054	51.521	-0.137	2013-12-31 12:55:00
1	40054	51.472	-0.156	2013-12-31 13:04:00
1	40054	51.272	-0.248	2013-12-31 13:14:00
1	40054	51.168	-0.269	2013-12-31 13:18:00

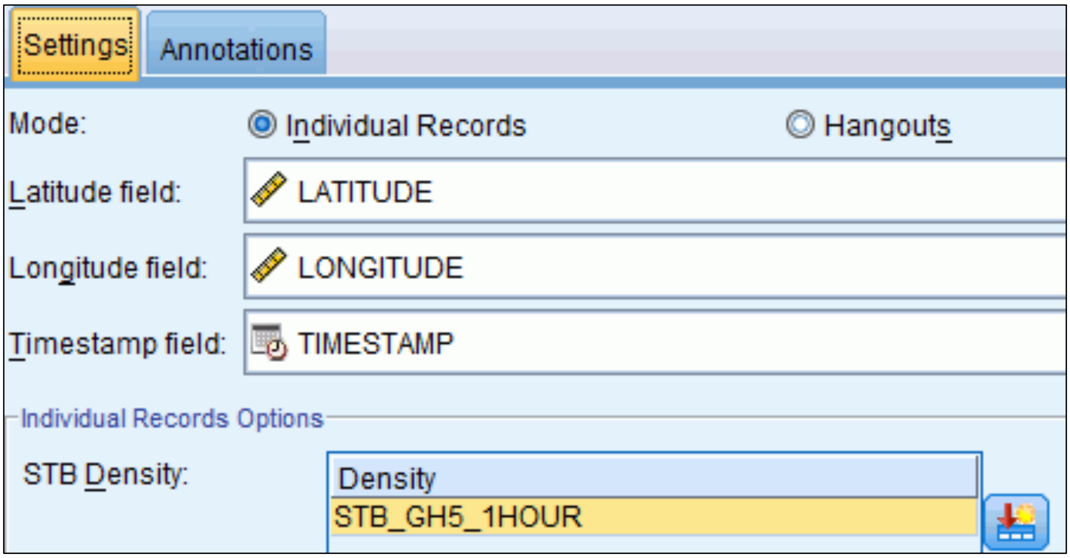
A customer is identified by an ID, linked to his or her PHONE_ID (so, a customer has only one phone id).

To determine the demand you will use the Space-Time-Boxes node to get the number of customers within a specific space and time density, for example within an area of 2.4 kilometers and a one hour time interval.

You can continue with working in the stream that you have created in the previous task.

1. Use a new **Var. File** node to import data from **taxi cab customers app data.txt**.
2. Add a **Space-Time-Boxes** node (Record Ops) downstream from the **Var. File** node.
3. Edit the **Space-Time-Boxes** node, and then:
   - for **Latitude field**, select **LATITUDE**
   - for **Longitude field**, select **LONGITUDE**
   - for **Timestamp field**, select **TIMESTAMP**
   - click the **Add density** button, and then for **Geo density** select **GH5 (approx. 2.4 kilometers)**, for **Time interval** select **1 Hour**
   - close the **Define Space-Time-Box Density** sub dialog box

   A section of the results appear as follows:



Mode: ● Individual Records ○ Hangouts

Latitude field: LATITUDE

Longitude field: LONGITUDE

Timestamp field: TIMESTAMP

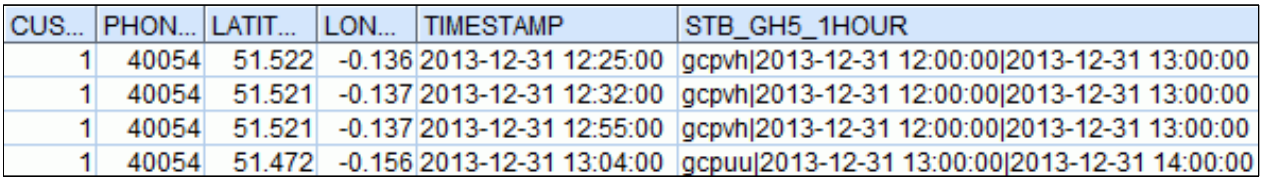
Individual Records Options

STB Density:

Density
STB_GH5_1HOUR

4. Click **Preview**.

   A section of the results appear as follows:



CUS...	PHON...	LATIT...	LON...	TIMESTAMP	STB_GH5_1HOUR
1	40054	51.522	-0.136	2013-12-31 12:25:00	gcpvh|2013-12-31 12:00:00|2013-12-31 13:00:00
1	40054	51.521	-0.137	2013-12-31 12:32:00	gcpvh|2013-12-31 12:00:00|2013-12-31 13:00:00
1	40054	51.521	-0.137	2013-12-31 12:55:00	gcpvh|2013-12-31 12:00:00|2013-12-31 13:00:00
1	40054	51.472	-0.156	2013-12-31 13:04:00	gcpuu|2013-12-31 13:00:00|2013-12-31 14:00:00

   The first three records, all from the same customer, are in the same space-time-box (geohash gcpvh, between 12:00h and 13:00h). The fourth record tells you that this customer is in another area in the next time interval.

5. Close the **Preview** output window, and then close the **Space-Time-Boxes** dialog box.

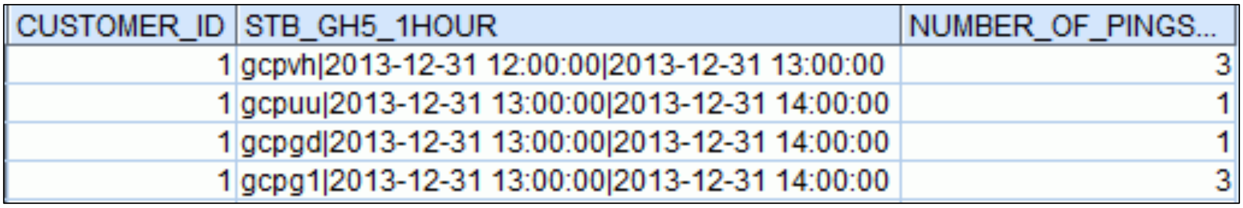
   You will aggregate the data to know the number of persons in a certain density. However, when you aggregate the data at this point using **STB_GH5_1HOUR** as the key field, you will get the number of pings in each space-time-box, not the number of persons. For example, the first customer has three records in the same space-time-box because he had three pings in that particular space-time-box, but you want this customer only to count once when you assess the demand. Thus, you will aggregate the data on both CUSTOMER_ID and STB_GH5_1HOUR. In this aggregate, you will save the number of pings. Although this is not necessary to answer the question, it will clarify that there is a difference between customers and pings.

6. Add an **Aggregate** node downstream from the **Space-Time-Boxes** node.

7. Edit the **Aggregate** node, and then:

   - for **Key fields**, select **CUSTOMER_ID** and **STB_GH5_1HOUR**
   - enable the option **Include record count in field**, and type **NUMBER_OF_PINGS_PER_CUSTOMER**
   - click **Preview**

   A section of the results appear as follows:



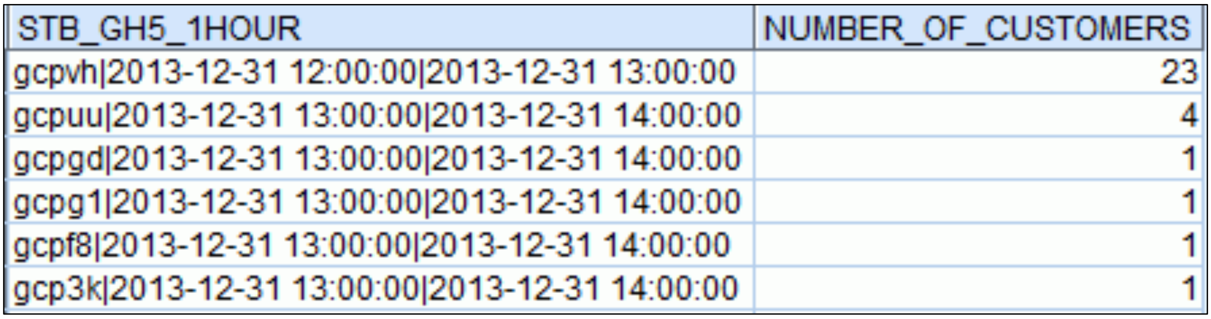
CUSTOMER_ID	STB_GH5_1HOUR	NUMBER_OF_PINGS...
1	gcpvh|2013-12-31 12:00:00|2013-12-31 13:00:00	3
1	gcpuu|2013-12-31 13:00:00|2013-12-31 14:00:00	1
1	gcpgd|2013-12-31 13:00:00|2013-12-31 14:00:00	1
1	gcpg1|2013-12-31 13:00:00|2013-12-31 14:00:00	3

   The first customer has three pings in geohash gcpvh, between 2013-12-31 12:00:00 and 2013-12-31 13:00:00, in agreement with previous findings.

To obtain the number of persons in each space-time-box, you will aggregate the data on **STB_GH5_1HOUR** and count the number of records (customers).

8. Close the **Preview** output window, and then close the **Aggregate** dialog box.
9. Add a **second Aggregate** node downstream from the **first Aggregate** node.
10. Edit the **Aggregate** node, and then:
    - for **Key fields**, select **STB_GH5_1HOUR**
    - ensure that the **Include record count in field** option is enabled, and type **NUMBER_OF_CUSTOMERS**
    - click **Preview**

    A section of the results appear as follows:



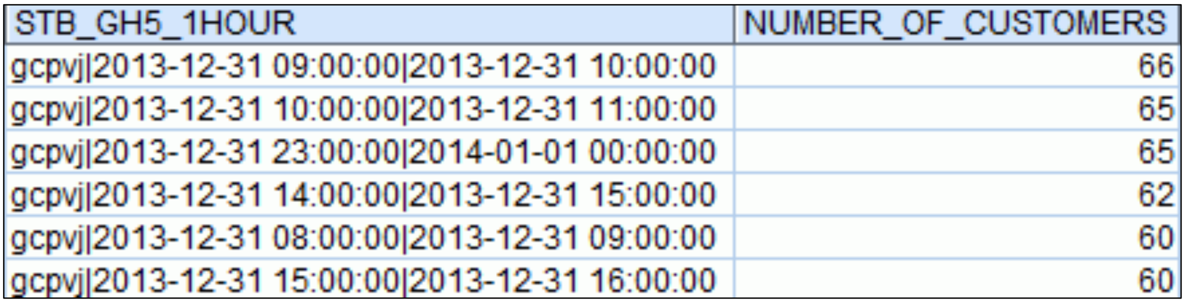
STB_GH5_1HOUR	NUMBER_OF_CUSTOMERS
gcpvh|2013-12-31 12:00:00|2013-12-31 13:00:00	23
gcpuu|2013-12-31 13:00:00|2013-12-31 14:00:00	4
gcpgd|2013-12-31 13:00:00|2013-12-31 14:00:00	1
gcpg1|2013-12-31 13:00:00|2013-12-31 14:00:00	1
gcpf8|2013-12-31 13:00:00|2013-12-31 14:00:00	1
gcp3k|2013-12-31 13:00:00|2013-12-31 14:00:00	1

There were 23 persons in geohash gcpvh, between 2013-12-31 12:00:00 and 2013-12-31 13:00:00.

To know in which area and time interval the demand is highest, you will sort the data descending on NUMBER_OF_CUSTOMERS.

11. Close the **Preview** output window, and then close the **Aggregate** dialog box.
12. Add a **Sort** node downstream from the **Aggregate** node.
13. Edit the **Sort** node, and then:
    - select **NUMBER_OF_CUSTOMERS**
    - under **Sort by:**, **Order** column change **Ascending** to **Descending** (not **Default sort order:** below Sort by: section.)
    - click **Preview**

A section of the results appear as follows:



STB_GH5_1HOUR	NUMBER_OF_CUSTOMERS
gcpvj|2013-12-31 09:00:00|2013-12-31 10:00:00	66
gcpvj|2013-12-31 10:00:00|2013-12-31 11:00:00	65
gcpvj|2013-12-31 23:00:00|2014-01-01 00:00:00	65
gcpvj|2013-12-31 14:00:00|2013-12-31 15:00:00	62
gcpvj|2013-12-31 08:00:00|2013-12-31 09:00:00	60
gcpvj|2013-12-31 15:00:00|2013-12-31 16:00:00	60

The demand was highest in geohash gcpvj, between 2013-12-31 09:00:00|2013-12-31 10:00:00.

Having this information you could align the number of taxi cabs with the demand. Or, you could plot the demand for a specific time frame, using mapping software. In the latter case, you can use MODELER's tb_centroid_latitude and stb_centroid_longitude function to create a field that gives the center of the geohash, so that you can plot the frequencies on the center of the geohash areas. The tb_centroid_latitude and stb_centroid_longitude function are available under General Functions in the Expression Builder.

Note: Another function is to_geohash, which returns the geohashed string corresponding to a specified latitude, longitude and density.

14. Close the **Preview** output window, and then close the **Sort** dialog box.

    Leave the stream open for the next task.

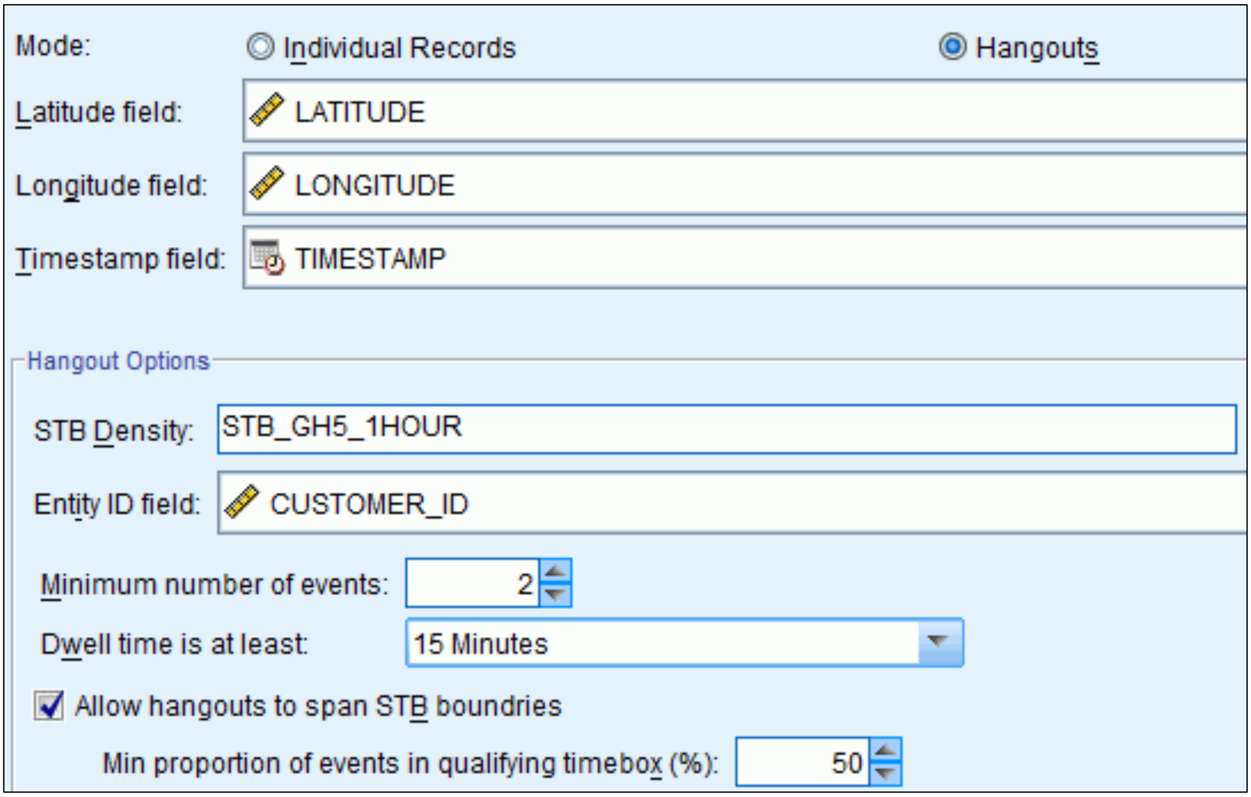
Task 6. Analyzing geospatial and time data, using the Hangouts mode.

In the previous task you have determined the demand by counting the number of customers in each space-time-box. A more sophisticated way to determine the demand is to use the Hangouts mode in the Space-Time-Boxes node. For example, if one pings at 12:55h in geohash gcpvj and another ping from the same person is at 13:10h in the same geohash, you cannot conclude from the previous analysis that this person hangs out in geohash gcpvj because of the two different time windows. By allowing space-time-boxes to overlap, as the Hangouts mode does, MODELER will qualify these events as a hangout (for geohash gcpvj, in the 13:00h - 14:00h time interval).

To demonstrate the Hangouts mode, you will use the same dataset to get the number of people hanging out in each of the space-time-boxes.

1. Add a **Space-Time-Boxes** node (Record Ops) downstream from the **Var. File** node.
2. Edit the **Space-Time-Boxes** node, and then:
   - for **Mode**, select **Hangouts**
   - for **Latitude field**, select **LATITUDE**
   - for **Longitude field**, select **LONGITUDE**
   - for **Timestamp** field, select **TIMESTAMP**
   - for **STB Density**, select **GH5 (approx 2.4 kilometers)**, and a **1 Hour** time interval
   - for **Entity ID field**, select **CUSTOMER_ID**
   - for minimum **Number of events**, select **2** (a customer should have at least 2 records in a certain space-time-box to be considered as a hangout for that space-time-box)
   - for **Dwell time**, select **15 Minutes** (a customer should at least be 15 minutes in a certain space-time-box to be considered as a hangout for that space-time-box)
   - ensure that the **Allow hangouts to span STB boundaries** option is enabled
   - ensure that the **Min proportion of events in qualifying timebox (%) value** is **50** (at least 50% of the events should be in the specific time frame; for example, suppose that a customer has events in the same geohash at 12:50h, 12:55h, 12:59h, 13:04h and 13:10h, then this will not qualify as a hangout for 13:00h - 14:00h because only two out of the five = 40% of the events is in the qualifying time box).

A section of the results appear as follows:



Mode: ○ Individual Records ● Hangouts

Latitude field: LATITUDE

Longitude field: LONGITUDE

Timestamp field: TIMESTAMP

Hangout Options

STB Density: STB_GH5_1HOUR

Entity ID field: CUSTOMER_ID

Minimum number of events: 2

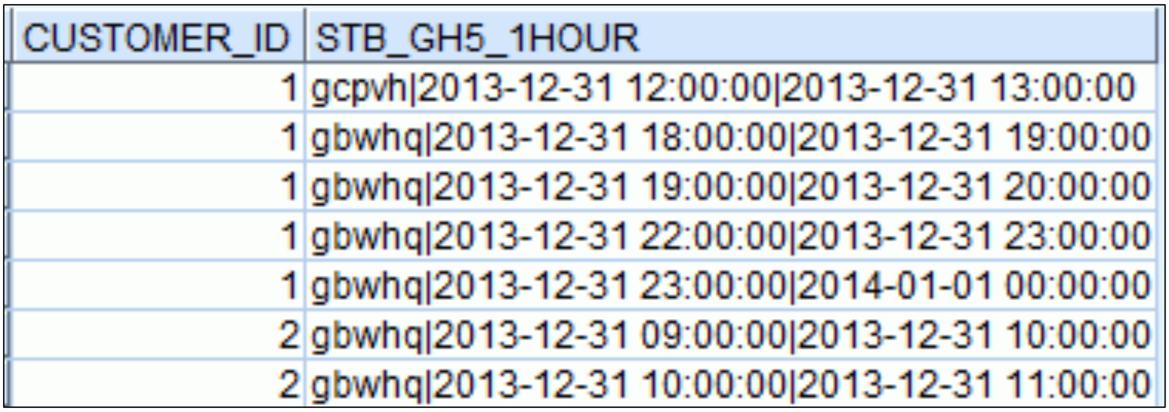
Dwell time is at least: 15 Minutes

☑ Allow hangouts to span STB boundries

Min proportion of events in qualifying timebox (%): 50

3. Click **Preview**.

   A section of the results appear as follows:



CUSTOMER_ID	STB_GH5_1HOUR
1	gcpvh|2013-12-31 12:00:00|2013-12-31 13:00:00
1	gbwhq|2013-12-31 18:00:00|2013-12-31 19:00:00
1	gbwhq|2013-12-31 19:00:00|2013-12-31 20:00:00
1	gbwhq|2013-12-31 22:00:00|2013-12-31 23:00:00
1	gbwhq|2013-12-31 23:00:00|2014-01-01 00:00:00
2	gbwhq|2013-12-31 09:00:00|2013-12-31 10:00:00
2	gbwhq|2013-12-31 10:00:00|2013-12-31 11:00:00

   The first customer especially hung out in geohash gbwhq in the evening hours.

4. Close the **Preview** output window, and then close the **Space-Time-Boxes** dialog box.

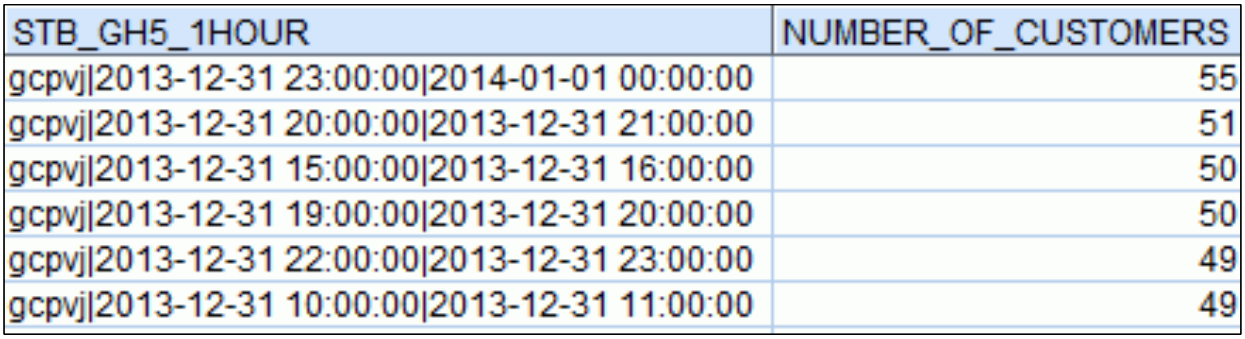
   You will now aggregate the records on the space-time-box field, to know how many people were hanging out in a specific space-time-box.

5. Add an **Aggregate** node downstream from the **Space-Time-Boxes** node.

6. Edit the **Aggregate** node, and then:
   - for **Key fields**, select **STB_GH5_1HOUR**
   - for **Include record count in field**, type **NUMBER_OF_CUSTOMERS**
   - close the **Aggregate** dialog box

   To know in which area and time interval the demand is highest, you will sort the data descending on NUMBER_OF_PERSONS.
7. Add a **Sort** node downstream from the **Aggregate** node.
8. Edit the **Sort** node, and then:
   - select **NUMBER_OF_CUSTOMERS**
   - set **Order** to **Descending (Order** column)
   - click **Preview**

   A section of the results appear as follows:



STB_GH5_1HOUR	NUMBER_OF_CUSTOMERS
gcpvj|2013-12-31 23:00:00|2014-01-01 00:00:00	55
gcpvj|2013-12-31 20:00:00|2013-12-31 21:00:00	51
gcpvj|2013-12-31 15:00:00|2013-12-31 16:00:00	50
gcpvj|2013-12-31 19:00:00|2013-12-31 20:00:00	50
gcpvj|2013-12-31 22:00:00|2013-12-31 23:00:00	49
gcpvj|2013-12-31 10:00:00|2013-12-31 11:00:00	49

   The busiest location and time is gcpvj, from 2013-12-31 23:00:00 to 2014-01-01 00:00:00. This was the second busiest space-time-box when individual records were examined.
9. Close the **Preview** output window, and then close the **Sort** dialog box.

This completes the demo for this module. You will find the solution results in **demo_working_with_sequence_data_completed.str**, located in the **03-Working_with_Sequence_Data\Solutions** sub folder.

Results:
You have applied cross-record functions to derive new fields and you have restructured your data so that you could do the required analyses. Also, you have used the Space-Time-Boxes node to determine the most popular place and time where entities are located or hang out

Apply Your Knowledge

Use the questions in this section to test your knowledge of the course material.

Question 1: Which of the following is the correct statement? Refer to the table below. Which value is computed for the first two records when you use the @MEAN (X, 3) function ?

A. Record 1 and record 2 both are assigned the undefined ($null$) value.

B. Record 1 is assigned the value 1; record 2 is assigned the value 2.

C. Record 1 is assigned the value 1; record 2 is assigned the value 1.5.

D. None of the above statements are correct.

X	**@MEAN (X, 3)**
1	?
2	?
3	2
4	3
5	4

Question 2: Which of the following is the correct statement? Refer to the table below. What was the condition used for the true value, for the derived flag field Z ?

A. CUSTOMER_ID = @OFFSET(CUSTOMER_ID, 1)

B. CUSTOMER_ID = @OFFSET(CUSTOMER_ID, -1)

C. CUSTOMER_ID = @INDEX (CUSTOMER_ID, 1)

D. PRODUCT = @OFFSET(PRODUCT, 1)

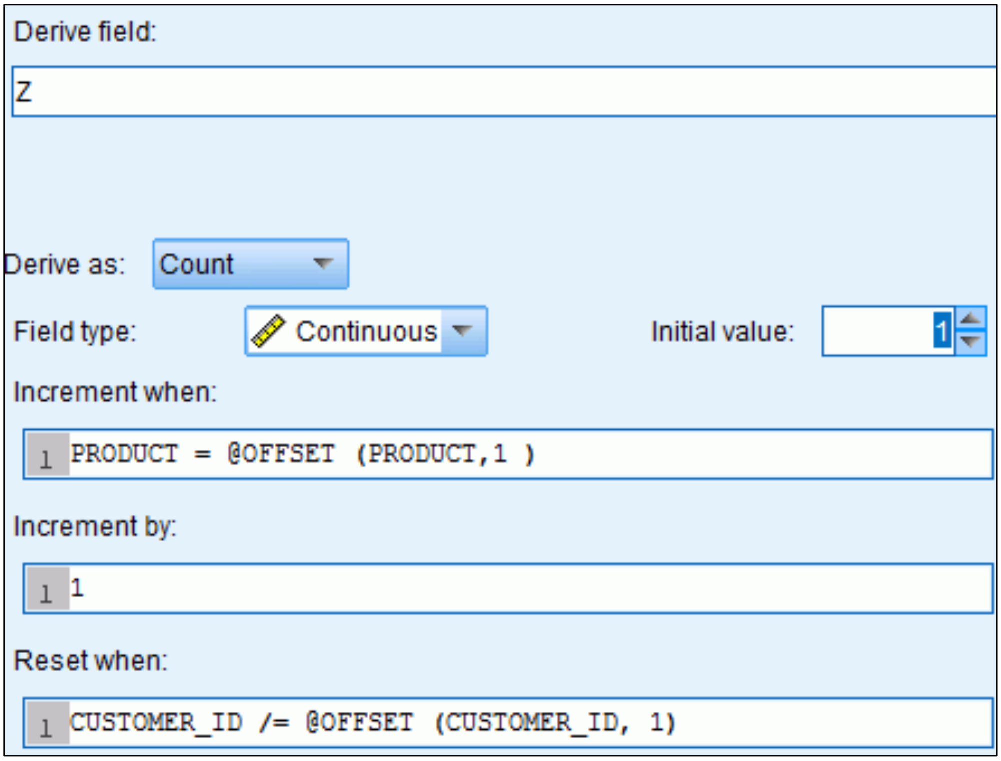
CUSTOMER_ID	**PRODUCT**	**Z**
1	A	F
2	A	F
2	C	T
3	B	F
3	C	T
3	F	T

Question 3: Refer to the data and Derive dialog box below. Is the following statement true or false? Record #3 (RECORD_ID = 3) has value 2 for Z.

Data

RECORD_ID	CUSTOMER_ID	PRODUCT
1	1	A
2	2	B
3	2	B
4	3	C
5	3	D
6	3	E

Derive dialog box for Z



Derive field:

Z

Derive as: Count

Field type: Continuous

Initial value: 1

Increment when:

```
PRODUCT = @OFFSET (PRODUCT,1 )
```

Increment by:

```
1
```

Reset when:

```
CUSTOMER_ID /= @OFFSET (CUSTOMER_ID, 1)
```

A. True

B. False

This material is meant for IBM Academic Initiative purposes.

Question 4: Is the following statement true or false? The Restructure node is located in the Record Ops palette.

A. True

B. False

Question 5: Is the following statement true or false? The Restructure node has the option to aggregate the data using a key field.

A. True

B. False

Question 6: Is the following statement true or false? A geohash relates to a time zone.

A. True

B. False

Question 7: Is the following statement true or false? The area defined by geohash wx4fbutzzbr is part of the area defined by geohash wx4fbutz.

A. True

B. False

Question 8: Is the following statement true or false? When you use the Hangouts mode in the Space-Time-Boxes node and you use a time interval of one day, you will get <u>more</u> records output from the Space-Time-Boxes node than when you use a time interval of 1 hour (all other settings equal).

A. True

B. False

Answers to questions:

Answer 1: C. The @MEAN (X, 3) function computes the mean over the previous two records and the current record. Thus, the result for the first record is the value itself, 1. The result for the second record is the mean computed over records 1 and 2, which equals 1.5.

Answer 2: A. The derived field is true when CUSTOMER_ID equals CUSTOMER_ID of the previous record, else it is F. This is what the expression CUSTOMER_ID = @OFFSET(CUSTOMER_ID, 1) flags.

Answer 3: A. True. The field Z initializes at 1 for each CUSTOMER_ID and increments by 1 when the customer's product is the same as the product of the previous record. The third record represents the same customer as the second record and the product is the same (product B), so the outcome equals 2.

Answer 4: B. False. The Restructure node is located in the Field Ops palette, not in the Record Ops palette.

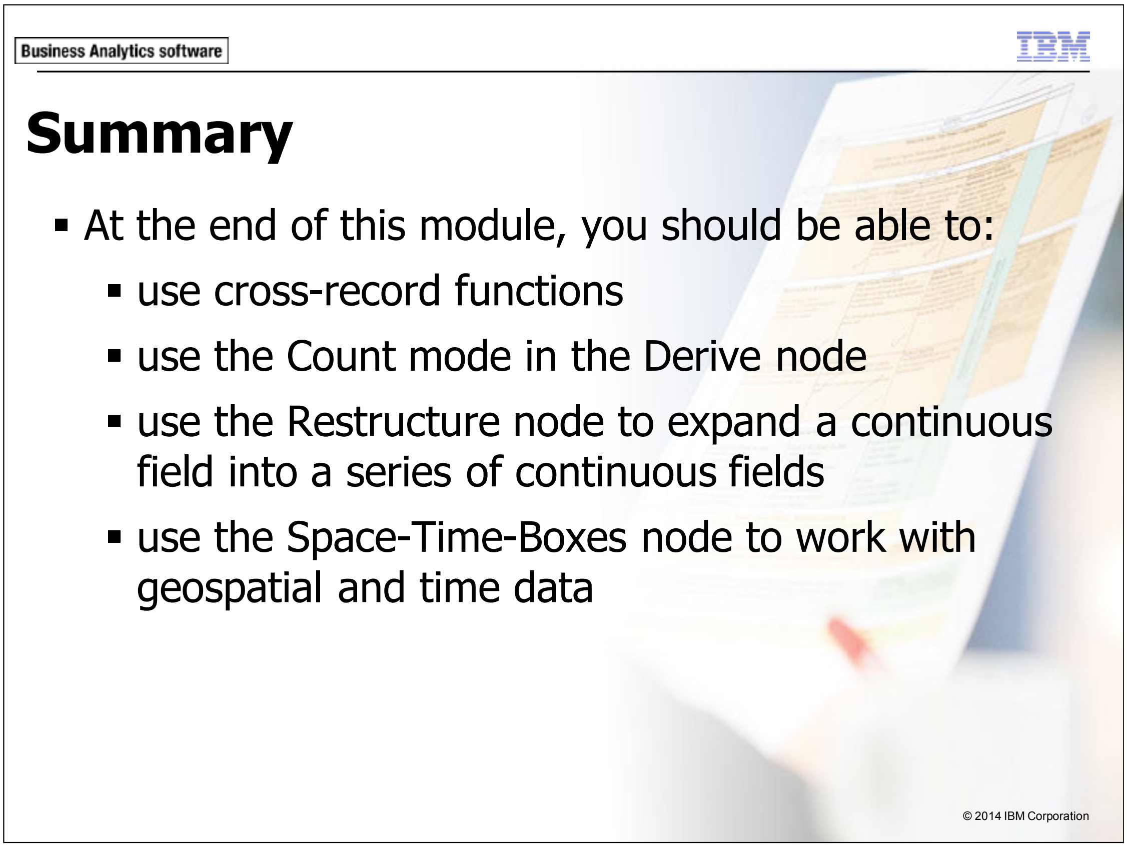
Answer 5: B. False. The Restructure node does not have an option to aggregate the data. Use the Aggregate node downstream from the Restructure node to aggregate the data.

Answer 6: B. False. A geohash relates to a spatial area.

Answer 7: A. True. Geohashes are hierarchical, so geohash wx4fbutz encloses geohash wx4fbutzzbr.

Answer 8: B. False. A time interval of one day will result in fewer records than a time-interval of one hour.

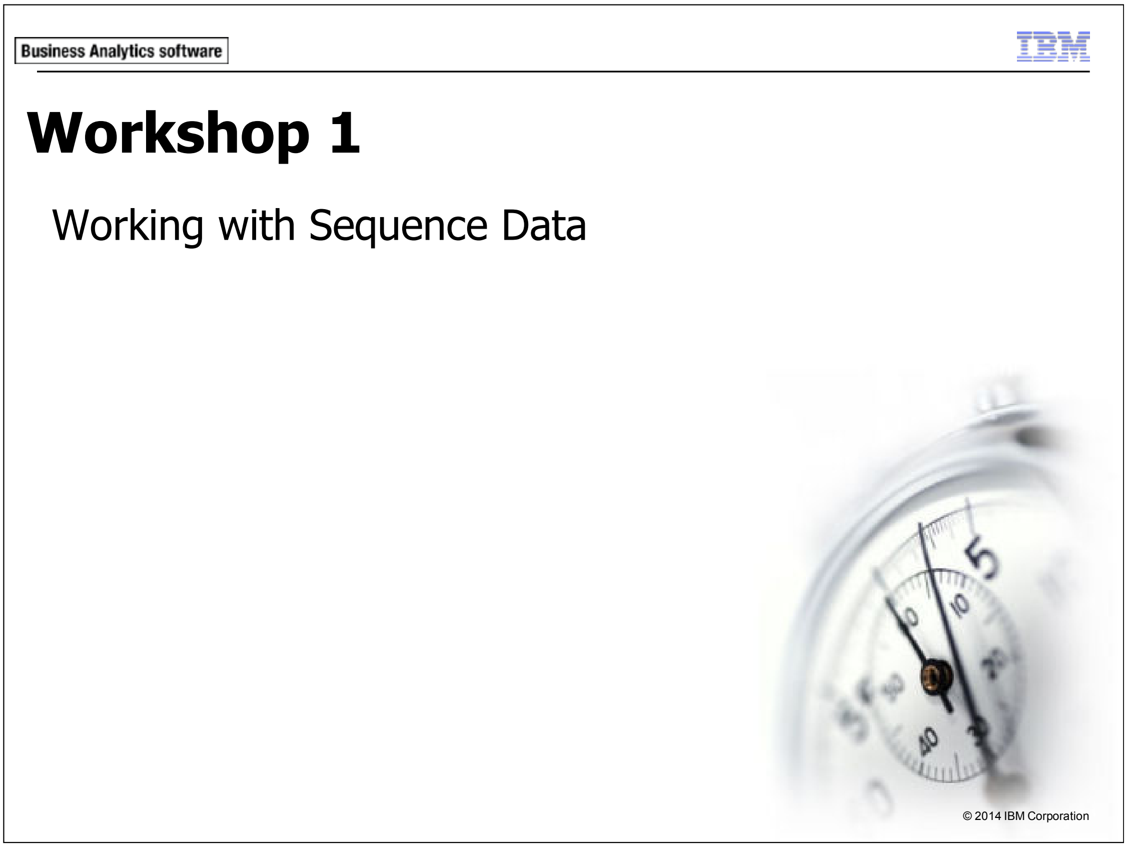
This material is meant for IBM Academic Initiative purposes.



Business Analytics software

Summary

- At the end of this module, you should be able to:
  - use cross-record functions
  - use the Count mode in the Derive node
  - use the Restructure node to expand a continuous field into a series of continuous fields
  - use the Space-Time-Boxes node to work with geospatial and time data



Business Analytics software

Workshop 1

Working with Sequence Data

© 2014 IBM Corporation

The following (synthetic) files are used in this workshop:

- **customers_and_holidays.dat**: A text file storing data on holiday destinations for 411 customers of a travel agency.
- **taxi_locations_and_times.txt**: A text file storing the location and time of taxi cabs on Dec-31-2013.

Both files are located in **C:\Train\0A055**.

Before you begin with the workshop, ensure that:

- You have started MODELER.
- You have set MODELER's working folder. (In MODELER, select **File\Set Directory**. When you are working in an IBM operated environment, browse to the folder **C:\Train\0A055** and then click **Set** to select it. When you are not working in an IBM operated environment, ask your trainer for the folder name.)

Workshop 1: Working with Sequence Data

In this workshop you will work on a dataset storing customers and their holidays. You will derive new fields to answer questions such as "What is the mean age of the customers?", "What was the most popular country?" and so forth.Import data from **customers_and_holidays.dat**.

- Derive a field named **RECORD_NUMBER** that stores the record number.
- Derive a field **CUSTOMER_NUMBER** running from 1 to the number of customers. Records having the same CUSTID are the same person, so must have the same value for **CUSTOMER_NUMBER** (because a customer can have multiple records, this field will differ from the RECORD_NUMBER field).

  Hint: Ensure that the records are sorted by CUSTID. Add a **Derive** node downstream from the **Sort** node, edit the **Derive** node and for **Derive as** select **Count**. Increment when CUSTID differs from that of the previous record, and ensure that the field is never reset.

  What is the value for CUSTOMER_NUMBER for CUSTID GS101012?
- Suppose that you want to examine how the money spent on leisure (SPENT_LEISURE) in one country correlates with the money spent on leisure in another country. Hereto, SPENT_LEISURE must be expanded in a series of fields, giving the money spent on leisure in each country.

  Restructure the dataset so that extra fields are created giving the SPENT_LEISURE for each country.

  Hints:

  - Cleanse the COUNTRY field, so that the country names are in upper case (use a **Filler** node, and the **lowertoupper** function to accomplish this).
  - Add a **Type** downstream from the **Filler** node, and instantiate the data so that MODELER knows the values for **COUNTRY**.
  - Add a **Restructure** node downstream from the **Type** node to expand SPENT_LEISURE.
- Create a dataset that has one record per **CUSTID** and the money that the customer has spent on leisure in each of the countries.

Note: Having this dataset you could explore the correlations between the fields. This is not pursued in this course.

In the following task you will continue with the taxi cab example. In the demo you analyzed pings from people who opted in on the taxi cab company's cell phone app. Apart from this customer dataset, there is a dataset that stores the location and time of taxi cabs on Dec-31-2013.

- Import data from **taxi_locations_and_times.txt**, and determine the top 3 space-time-boxes where taxis hang out (the three space-time-boxes with the highest number of taxi cabs). Use a 2.4 kilometers density for the geohash, and a 15 minutes time interval. Furthermore, a hangout qualifies as such if the dwell time is at least 5 minutes. (Use defaults for the other parameters).

  Note: The stream **taxi_availabity_stb.str** that ships with IBM SPSS Modeler 16 uses cell phone information and taxi times/locations to match the demand with the supply. Those interested are referred to this sample stream for more information on space-time-boxes.

For more information about where to work and the workshop results, refer to the Task and Results section that follows. If you need more information to complete a task, refer to earlier demos for detailed steps.

Workshop 1: Tasks and Results

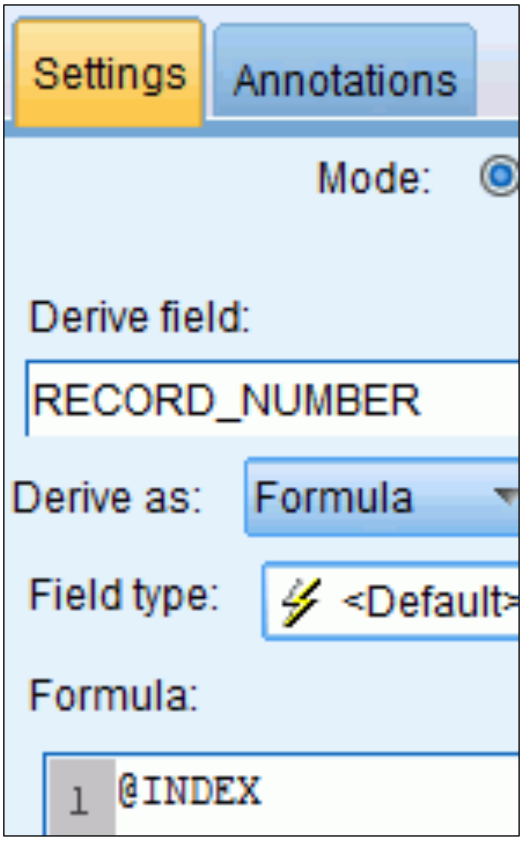
Task 1. Import the data.

- Use a **Var. File** node to import the data from **customers_and_holidays.dat**.

Task 2. Derive a record identifier.

- Add a **Derive** node downstream from the **Var. File** node.
- Edit the **Derive** node, and then:
  - for **Derive field**, type **RECORD_NUMBER**
  - for **Formula**, type **@INDEX**

A section of the results appear as follows:

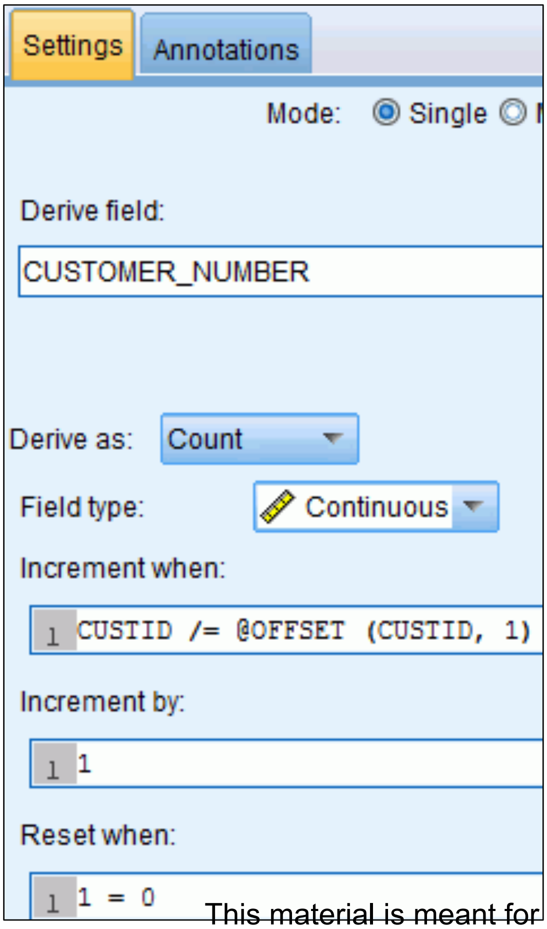


Derive field: RECORD_NUMBER
Derive as: Formula
Field type: <Default>
Formula:
```
1 @INDEX
```

Task 3. Derive a customer identifier.

- Add a **Sort** node downstream from the **Derive** node named **RECORD_NUMBER**, edit the **Sort** node, and then sort ascending on **CUSTID**.
- Add a **Derive** node downstream from the **Sort node**.
- Edit the **Derive** node, and then:
  - for **Derive field**, type **CUSTOMER_NUMBER**
  - for **Derive as**, select **Count**
  - for **Initial value**, type **1**
  - for **Increment when**, type **CUSTID /= @OFFSET (CUSTID, 1)**
  - for **Reset when**, type **1=0** (this condition will never be true, so the counter will never be reset)

A section of the results appear as follows:



Mode: ● Single ○ I

Derive field: CUSTOMER_NUMBER

Derive as: Count

Field type: Continuous

Increment when:

```
CUSTID /= @OFFSET (CUSTID, 1)
```

Increment by:

```
1
```

Reset when:

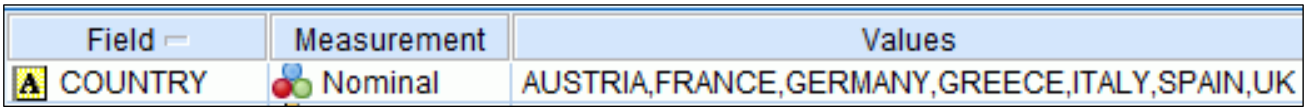
```
1 = 0
```

- Add a **Table** node downstream from the **Derive** node, run the **Table** node and locate customer **GS101012**. The CUSTOMER_NUMBER field is **15** for this customer.

Task 4. Restructure the dataset.

- Add a **Filler** node downstream from the **Derive** node named **CUSTOMER_NUMBER**.
- Edit the **Filler** node, and then:
    - for **Fill in fields**, select **COUNTRY**
    - for **Replace**, select **Always**
    - for **Replace with**, enter **lowertoupper (@FIELD)**
- Add a **Type** node downstream from the **Filler** node.
- Edit the **Type** node, and then click **Read Values** to instantiate the data (so MODELER knows the countries).

  A section of the results appear as follows:

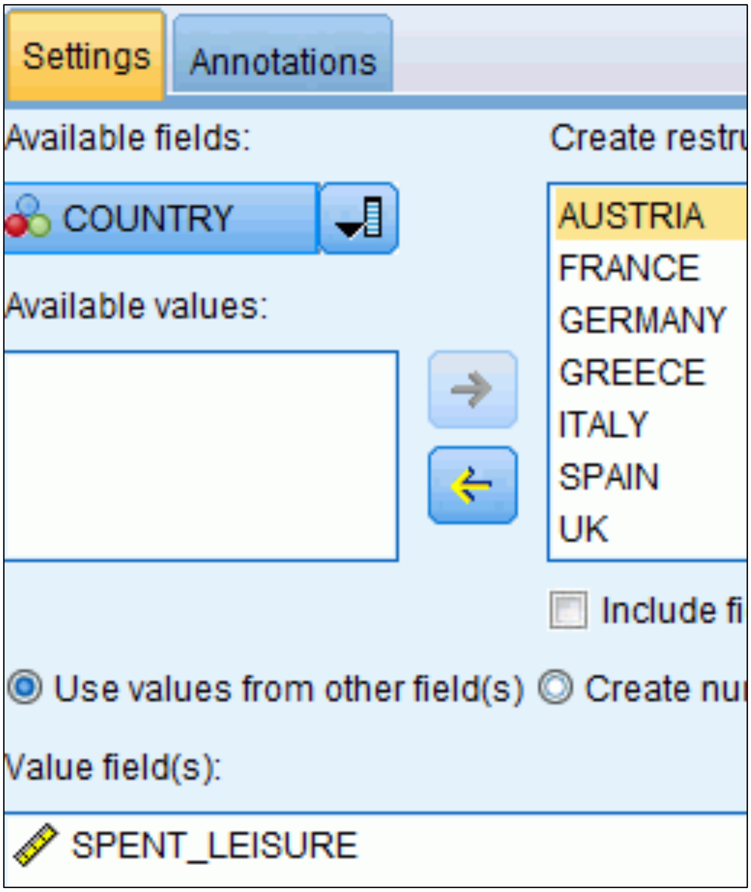


Field	Measurement	Values
COUNTRY	Nominal	AUSTRIA,FRANCE,GERMANY,GREECE,ITALY,SPAIN,UK

- Close the **Type** dialog box.
- Add a **Restructure** node (Field Ops) downstream from the **Type** node.
- Edit the **Restructure** node, and then:
    - for **Available fields**, select **COUNTRY**
    - under **Available values**, select **all values**, and move them to **Create restructured fields**
    - ensure the **Use values from other field(s)** option is selected
    - select **SPENT_LEISURE**

This material is meant for IBM Academic Initiative purposes.

A section of the results appear as follows:



Available fields:

COUNTRY

Create restru

AUSTRIA
FRANCE
GERMANY
GREECE
ITALY
SPAIN
UK

Available values:

□ Include fi

● Use values from other field(s) ○ Create nu

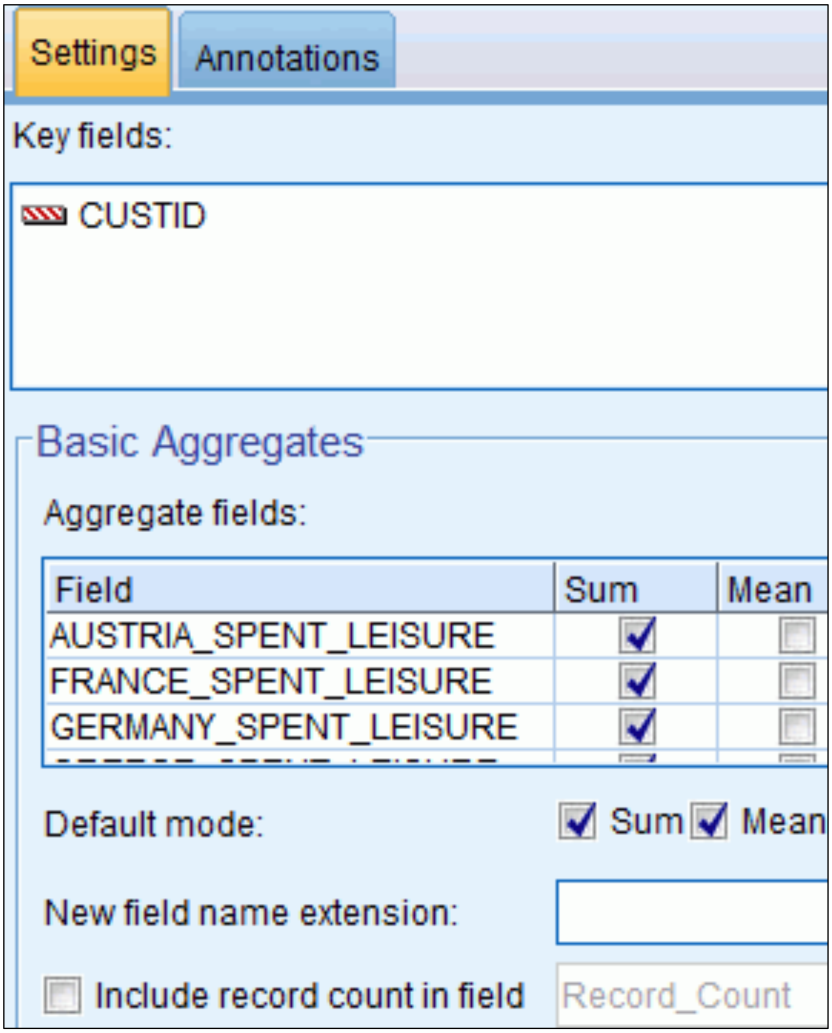
Value field(s):

SPENT_LEISURE

Task 5. Use Aggregate to have one record per customer.

- Add an **Aggregate** node downstream from the **Restructure** node.
- Edit the **Aggregate** node, and then:
  - for **Key fields**, select **CUSTID**
  - under **Aggregate fields**, select **COUNTRY_AUSTRIA_SPENT_LEISURE** to **COUNTRY_UK_SPENT_LEISURE**
  - select the **Sum** statistic
  - disable the **Include record count in field** option

A section of the results appear as follows:



Key fields:

CUSTID

Aggregate fields:

Field	Sum	Mean
AUSTRIA_SPENT_LEISURE	☑	□
FRANCE_SPENT_LEISURE	☑	□
GERMANY_SPENT_LEISURE	☑	□

Default mode: ☑ Sum ☑ Mean

New field name extension:

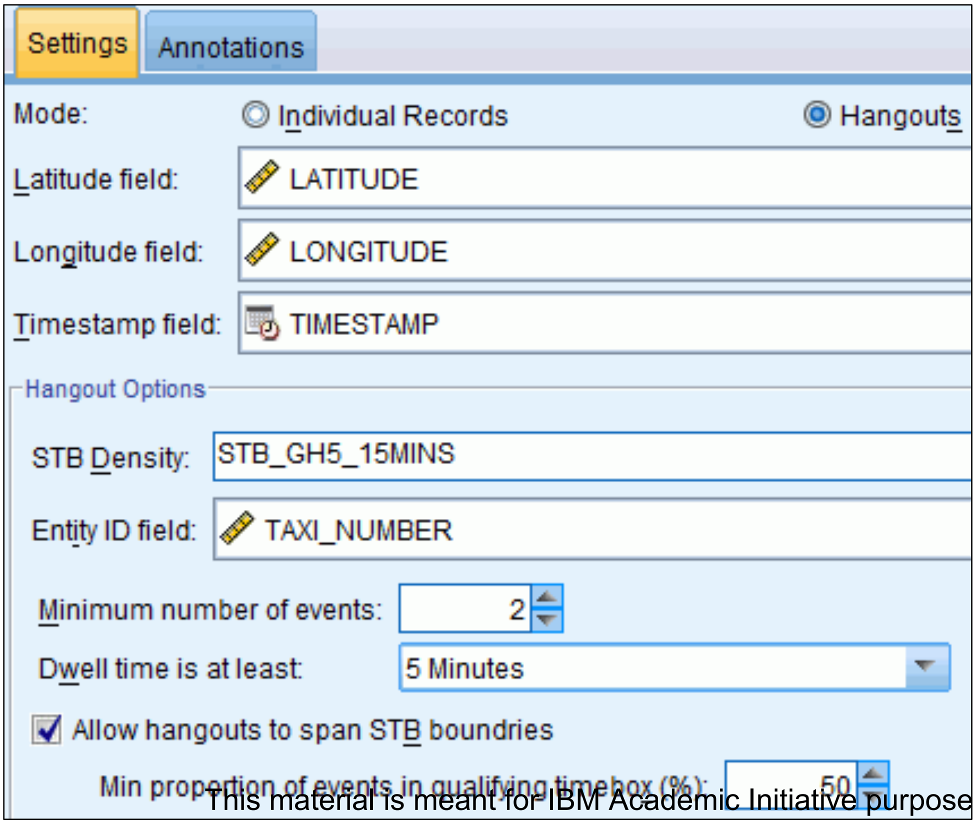
□ Include record count in field Record_Count

This material is meant for IBM Academic Initiative purposes.

Task 6. Analyze geospatial and time data.

- Use a **Var. File** node to import data from **taxi_locations_and_times.txt**.
- Add a **Space-Time-Boxes** node downstream from the **Var. File** node.
- Edit the **Space-Time-Boxes node**, and then:
  - for **Mode**, select **Hangouts**
  - for **Latitude field**, select **LATITUDE**
  - for **Longitude field**, select **LONGITUDE**
  - for **Timestamp** field, select **TIMESTAMP**
  - for **STB Density**, select **GH5 (approx 2.4 kilometers)** and a **15 Minutes** time interval
  - for **Entity ID field**, select **TAXI_NUMBER**
  - ensure that **Minimum number of events** is **2**
  - for **Dwell time**, select **5 Minutes**

A section of the results appear as follows:



Mode: ○ Individual Records ● Hangouts

Latitude field: LATITUDE

Longitude field: LONGITUDE

Timestamp field: TIMESTAMP

Hangout Options

STB Density: STB_GH5_15MINS

Entity ID field: TAXI_NUMBER

Minimum number of events: 2

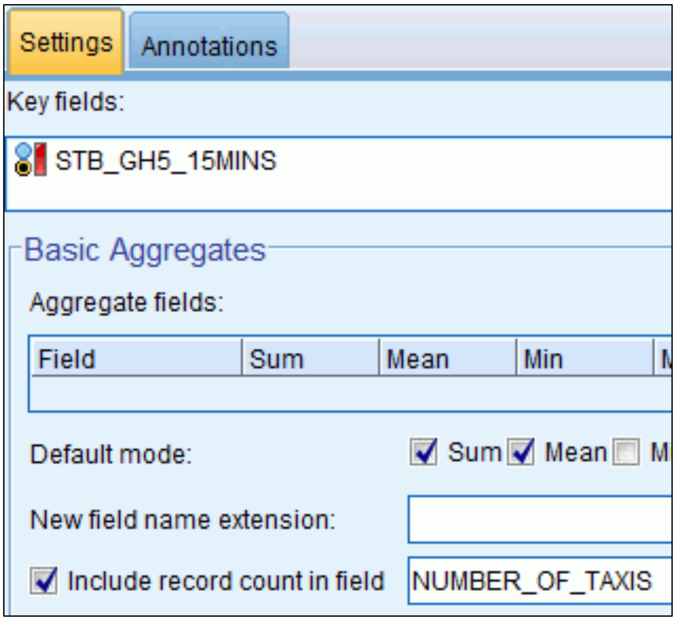
Dwell time is at least: 5 Minutes

☑ Allow hangouts to span STB boundries

Min proportion of events in qualifying timebox (%): 50

- Add an **Aggregate** node downstream from the **Space-Time-Boxes** node.
- Edit the **Aggregate** node, and then:
  - for **Key fields**, select **STB_GH5_15MINS**
  - ensure that the **Include record count in field** option is enabled, and type **NUMBER_OF_TAXIS**

A section of the results appear as follows:



Key fields: STB_GH5_15MINS

Default mode: ☑ Sum ☑ Mean □ M

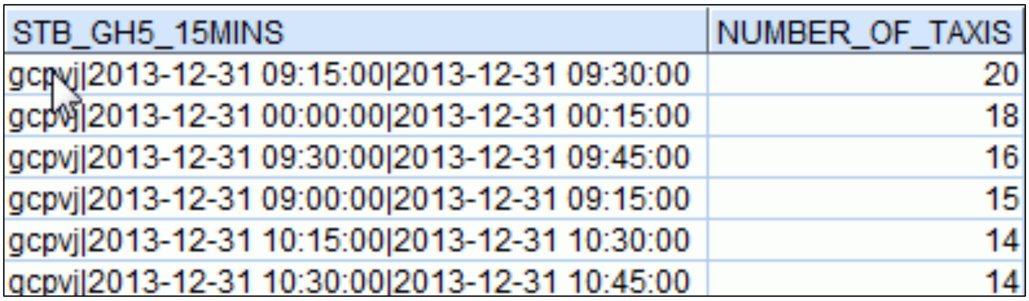
New field name extension:

☑ Include record count in field NUMBER_OF_TAXIS

To know in which area and time interval the availability is highest, you will sort the data descending on NUMBER_OF_TAXIS.

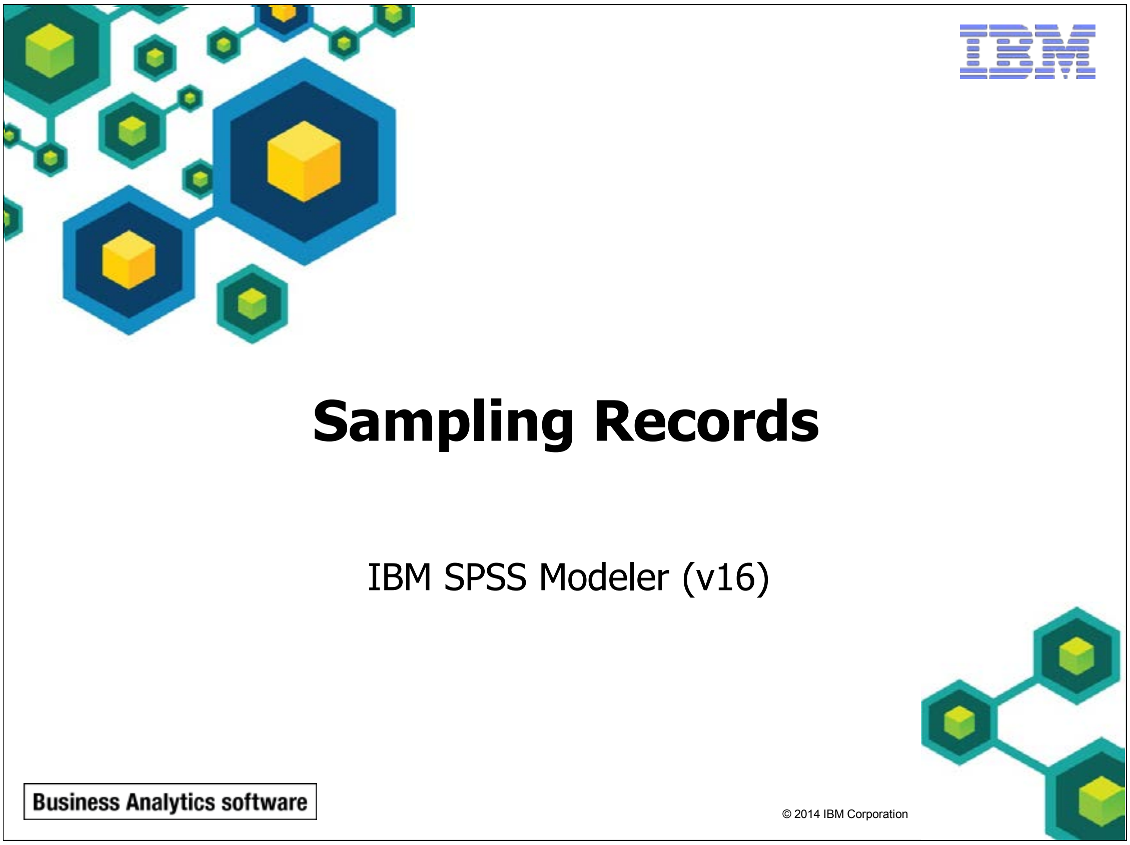
- Add a **Sort** node downstream from **Aggregate** node.
- Edit the **Sort** node, and then:
  - select **NUMBER_OF_TAXIS**
  - set **Order** to **Descending**
  - click **Preview**

A section of the results appear as follows:



STB_GH5_15MINS	NUMBER_OF_TAXIS
gcpvj|2013-12-31 09:15:00|2013-12-31 09:30:00	20
gcpvj|2013-12-31 00:00:00|2013-12-31 00:15:00	18
gcpvj|2013-12-31 09:30:00|2013-12-31 09:45:00	16
gcpvj|2013-12-31 09:00:00|2013-12-31 09:15:00	15
gcpvj|2013-12-31 10:15:00|2013-12-31 10:30:00	14
gcpvj|2013-12-31 10:30:00|2013-12-31 10:45:00	14

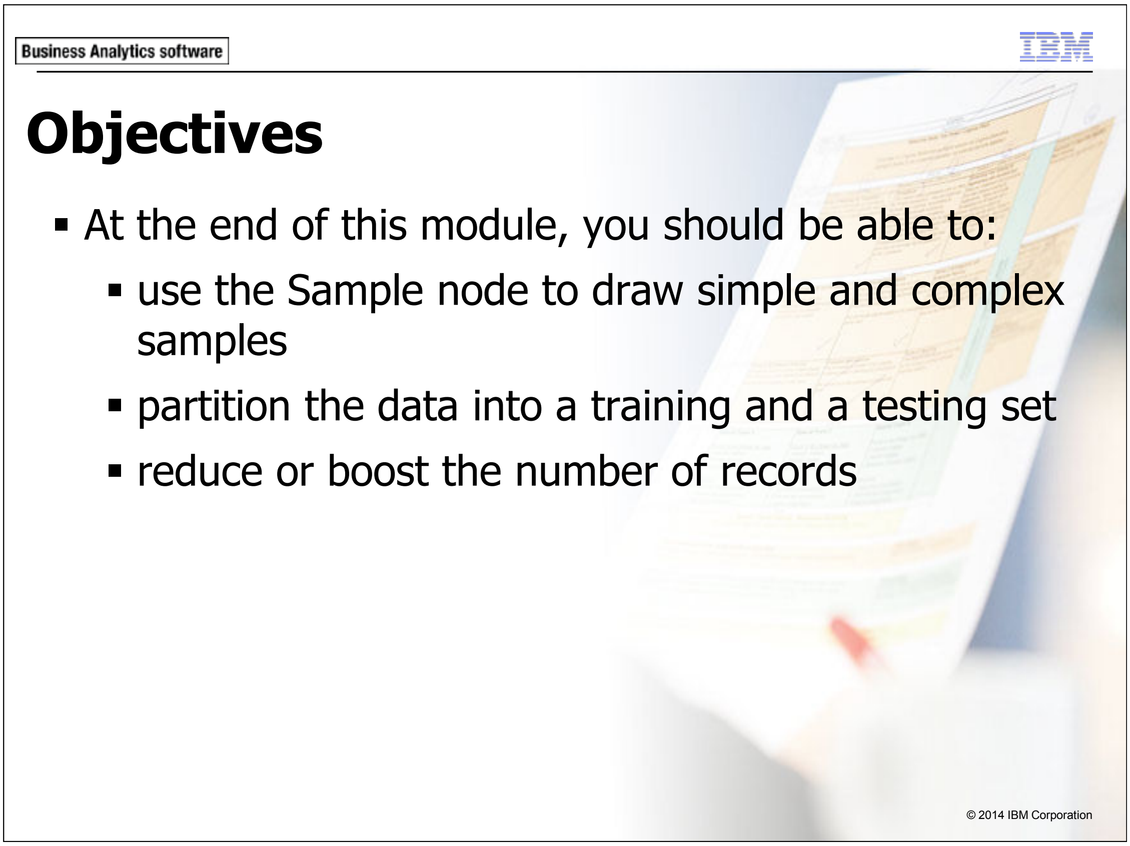
Note: The stream **workshop_working_with _sequence_data_completed.str**, located in the **03-Working_with_Sequence_Data\Solutions** sub folder, provides a solution to the workshop.



Sampling Records

IBM SPSS Modeler (v16)

Business Analytics software



Business Analytics software

IBM

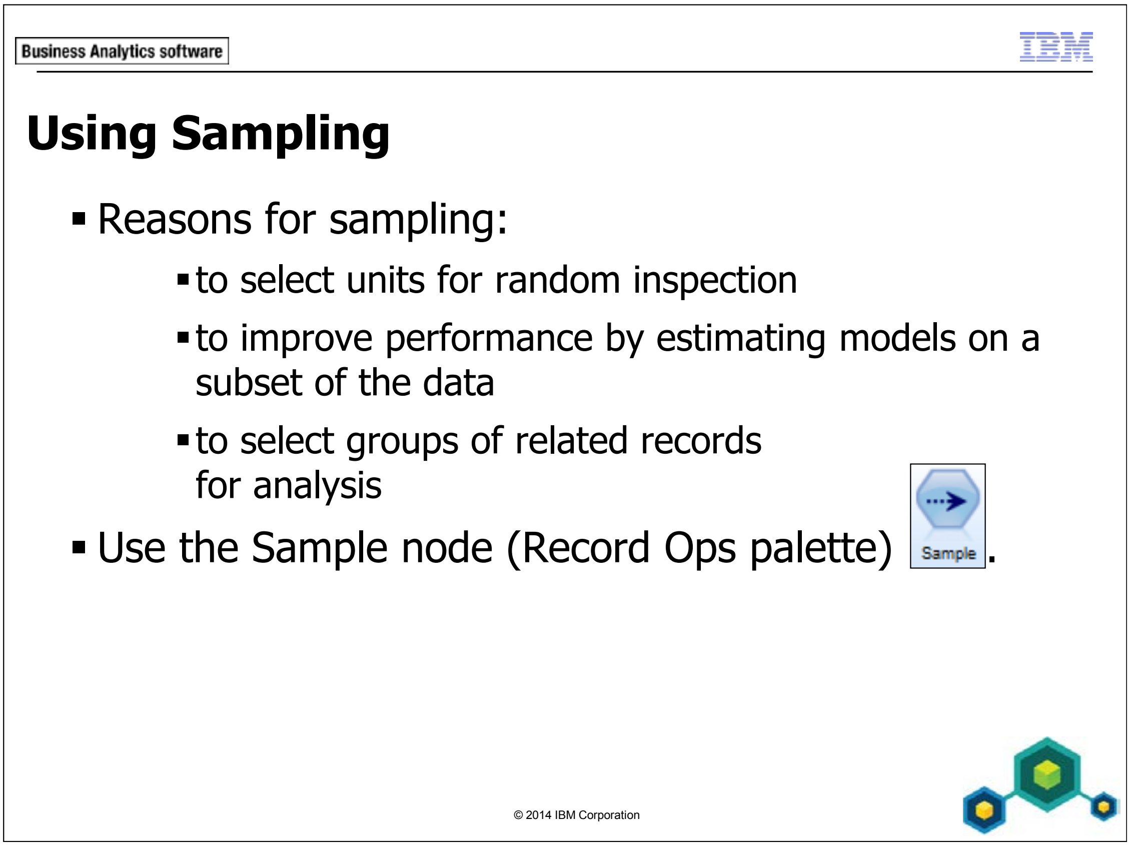
Objectives

- At the end of this module, you should be able to:
  - use the Sample node to draw simple and complex samples
  - partition the data into a training and a testing set
  - reduce or boost the number of records

© 2014 IBM Corporation

Before reviewing this module you should be familiar with:

- working with MODELER (streams, nodes, SuperNodes, palettes)
- importing data (Var. File node)
- defining measurement levels, roles, blanks, and instantiating data (Type node)
- selecting fields and records (Filter node, Select node)
- CLEM and the Expression Builder
- deriving fields (Derive node)
- running a model and examining the generated model nugget
- examining the data with graphs (Distribution node, Histogram node) and tables (Table node, Data Audit node, Statistics node, Matrix node, Means node)



Business Analytics software

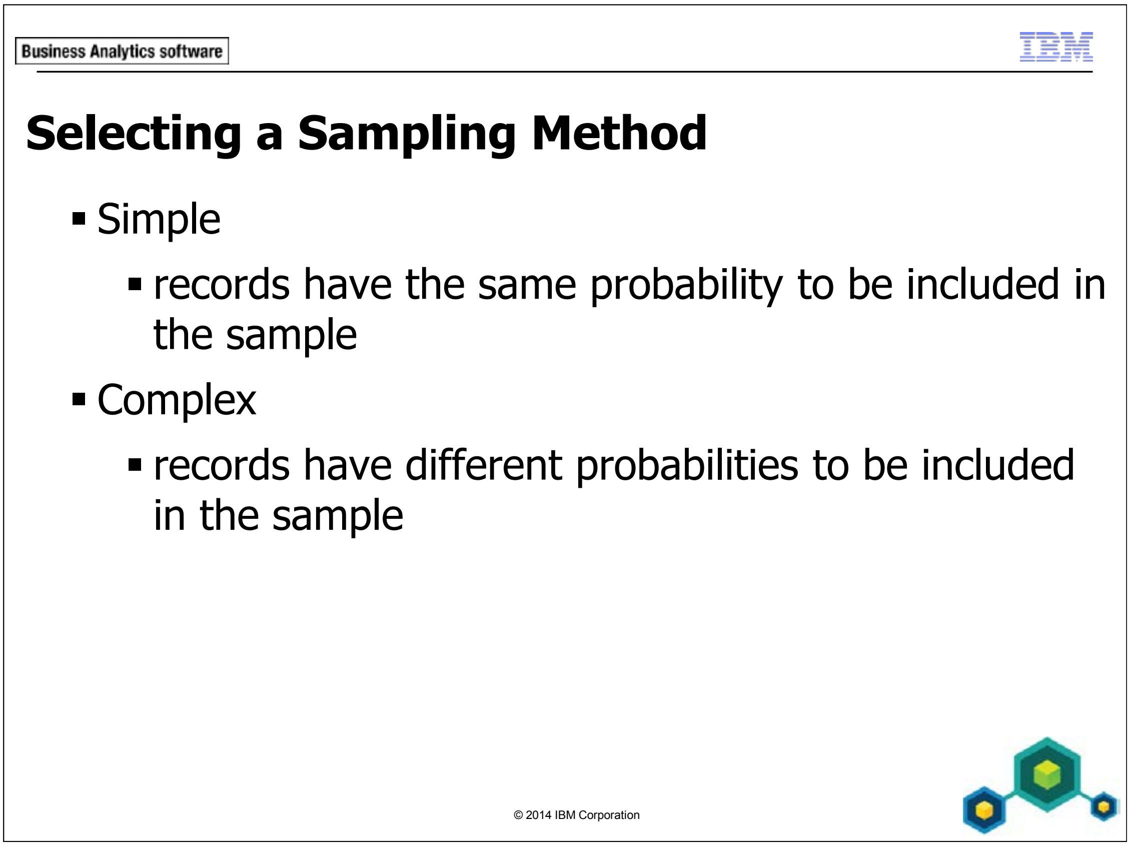
Using Sampling

- Reasons for sampling:
  - to select units for random inspection
  - to improve performance by estimating models on a subset of the data
  - to select groups of related records for analysis
- Use the Sample node (Record Ops palette) Sample.

Sampling can be used for several reasons:

- To identify units or cases for random inspection in the interest of quality assurance, fraud prevention, or security.
- To improve performance by estimating models on a subset of the data. Models estimated from a sample are often as accurate as those derived from the full dataset, and may be more so if the improved performance allows the user to experiment with different methods one might not otherwise have attempted.
- To select groups of related records or transactions for analysis, such as selecting all the items in an online shopping cart (or market basket), or all the properties in a specific neighborhood.

To draw samples, use the Sample node in the Record Ops palette.



Business Analytics software

IBM

Selecting a Sampling Method

- Simple
  - records have the same probability to be included in the sample
- Complex
  - records have different probabilities to be included in the sample

© 2014 IBM Corporation

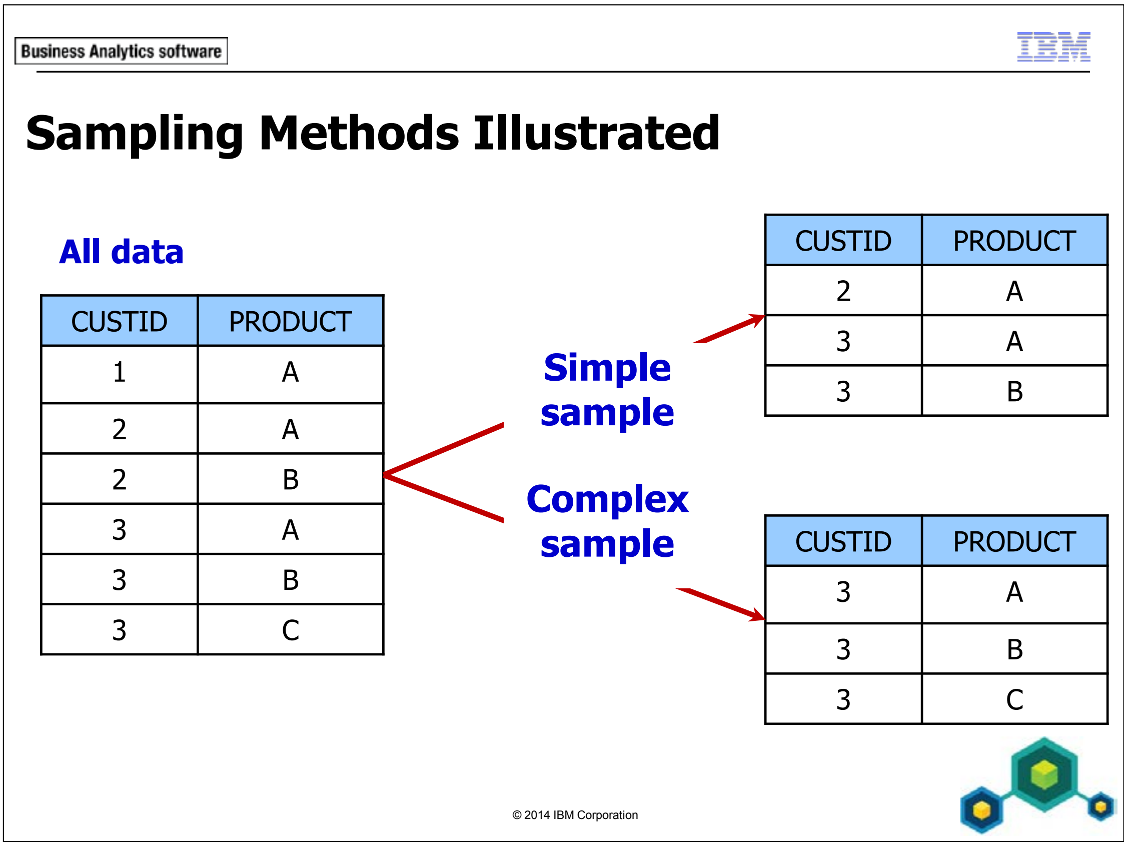
MODELER supports two sampling methods:

- Simple: Drawing a sample is achieved by a simple mechanism, such as selecting a sample where every record has the same probability of being included in the sample. Refer to the next slide for other simple sampling techniques.
- Complex: Records have a different probability of being included in the sample. For example, if there are small subgroups in the data but if you want to ensure that records are sampled from these subgroups, random sampling does not suffice (you would run the risk of not including any person from these subgroups).

The sampling method must be fit to the problem at hand. For example, if you have a database of similar customers (they all buy a specific product) and you simply want to reduce the file size for modeling, you can use one of the simple methods. But if you need to sample directly from a database with customers of different types, you may want to draw a complex sample.

.

This material is meant for IBM Academic Initiative purposes.



Business Analytics software

Sampling Methods Illustrated

All data

CUSTID	PRODUCT
1	A
2	A
2	B
3	A
3	B
3	C

Simple sample

CUSTID	PRODUCT
2	A
3	A
3	B

Complex sample

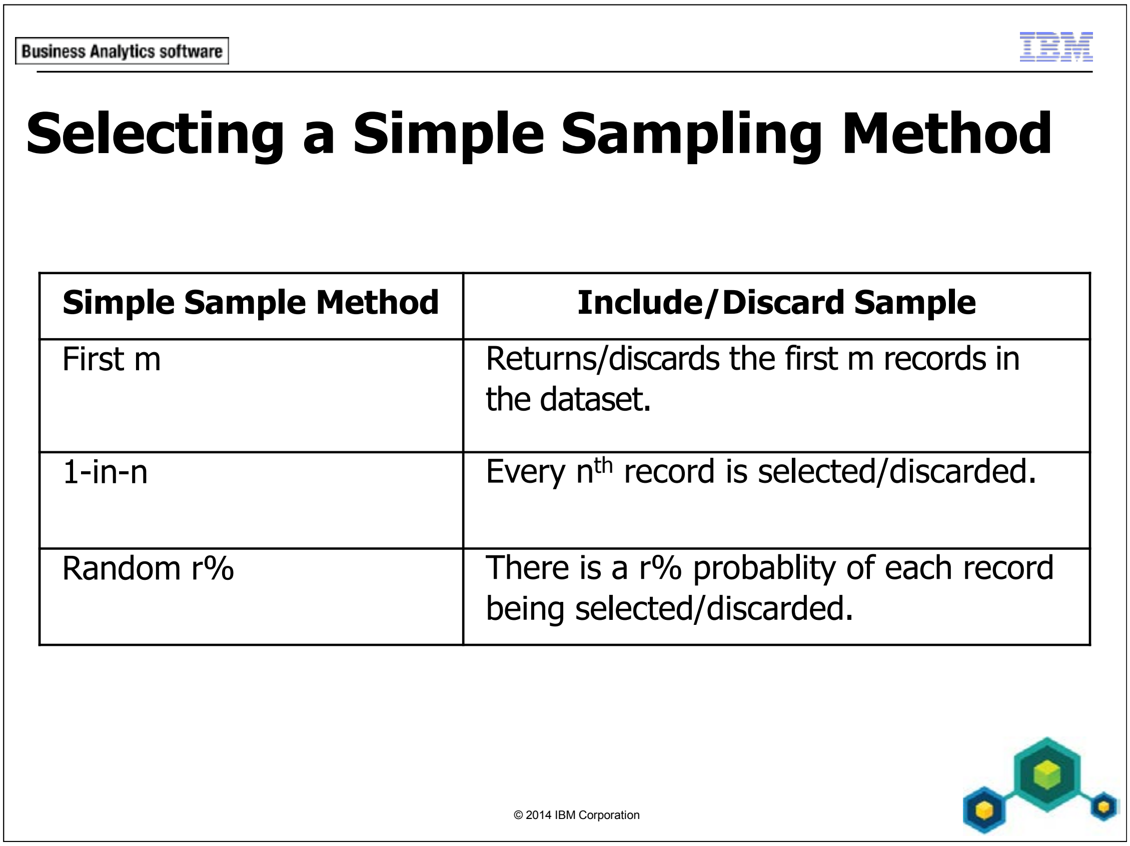
CUSTID	PRODUCT
3	A
3	B
3	C

Suppose that a dataset has 6 records, representing data from 3 customers, as depicted in the figure on the left.

A simple sample, with each record having the same probability of being included, may result in a dataset with three records, with data from two customers.

The complex sample did not draw a sample from records, but a sample from customers. Thus, customers have the same probability of being in the sample, not records. Having randomly selected customers, all records for the selected customers were included in the sample. In this example, a random selection of customers resulted in one customer being selected, CUSTID 3. All records of this customer were output to arrive at the complex sample.

In this example, a business question may be to find popular product combinations, an analysis known as market basket analysis. Imagine a dataset such as the one depicted here, but then with millions of records. An algorithm may have difficulties to find associations in a dataset of this size, and you may consider drawing a sample. Then, which sample would you draw? When you look for product combinations it is important to have all products for a certain customer in the sample, so a complex sample would be preferred over a simple sample.



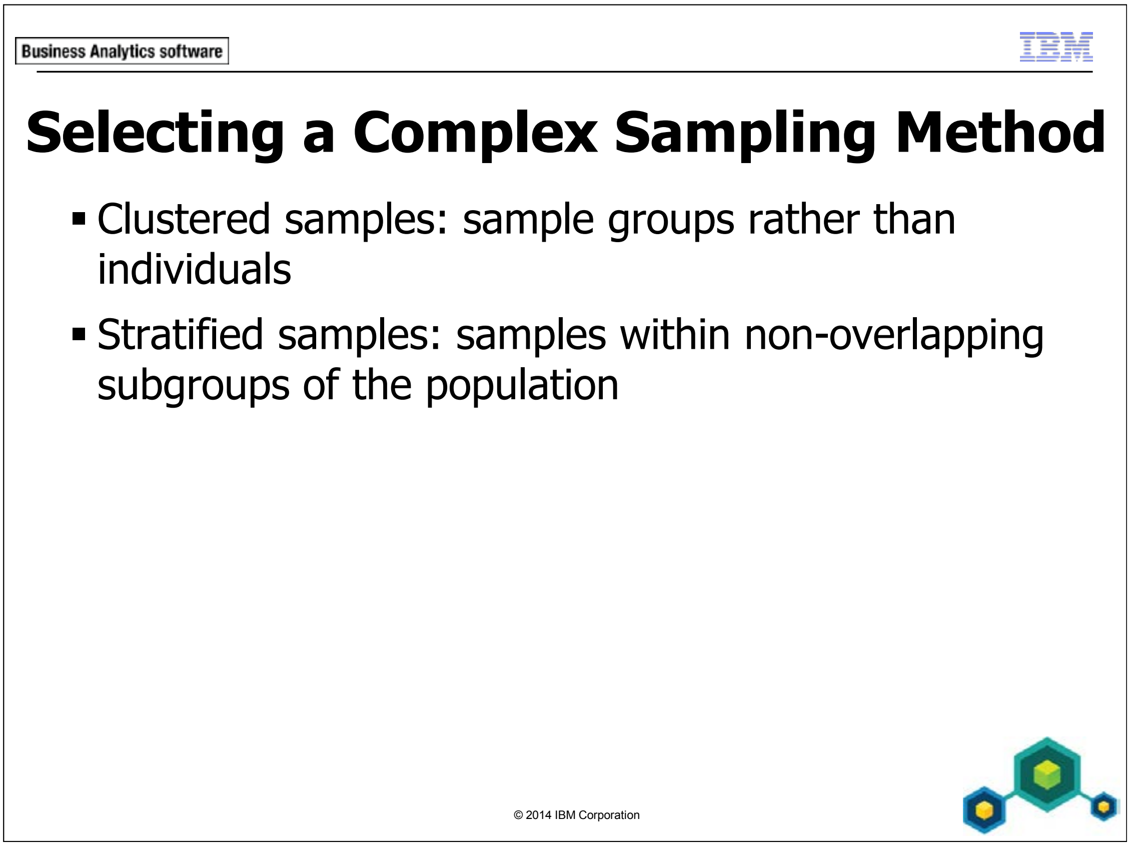
Business Analytics software

Selecting a Simple Sampling Method

Simple Sample Method	Include/Discard Sample
First m	Returns/discards the first m records in the dataset.
1-in-n	Every n^{th} record is selected/discarded.
Random r%	There is a r% probablity of each record being selected/discarded.

This table lists the methods of simple sampling that MODELER supports.

When a sample is drawn, the records in the sample can either be retained or discarded. The table describes each of the simple sample methods and their effect when used with the include sample or discard sample modes. For example, when a random sample of size r% is drawn in conjunction with the option to discard methods, r% of the records will be discarded, leaving 100%-r% in the sample.



Business Analytics software

IBM

Selecting a Complex Sampling Method

- Clustered samples: sample groups rather than individuals
- Stratified samples: samples within non-overlapping subgroups of the population

© 2014 IBM Corporation

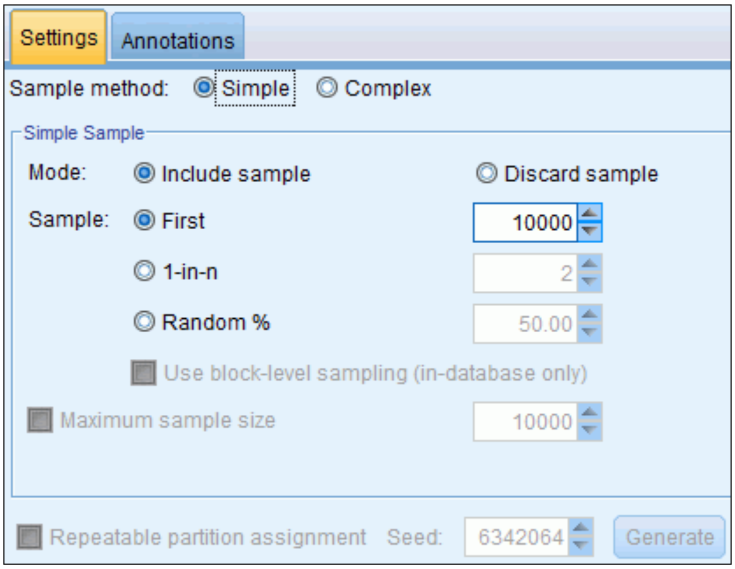
MODELER supports two complex sample methods:

- Clustered samples: Sample groups rather than individuals. Examples are:
  - Sample students by clustering by school. First randomly select schools, and then pick every student from each selected school. This will reduce costs in interviewing students, because you only have to visit a limited number of schools.
  - In market basket analysis (where each customer's transaction makes up a record) by clustering by customer. This ensures that all items from a customer are included in the sample.
- Stratified samples: Sample independently within subgroups, called strata in this context. For example, you can stratify by gender to ensure that men and women are sampled in equal proportions, or by region to ensure that each region within an urban population is represented. It is also possible to specify a different sample size for each stratum. For example, when you want to sample for a survey, you can give men a higher probability of being sampled than women when you think that the willingness to participate in the survey is lower for men than for women.

Business Analytics software

IBM

Exploring the Sample Dialog Box



Sample method: ● Simple ○ Complex

Simple Sample

Mode: ● Include sample ○ Discard sample

Sample: ● First 10000

○ 1-in-n 2

○ Random % 50.00

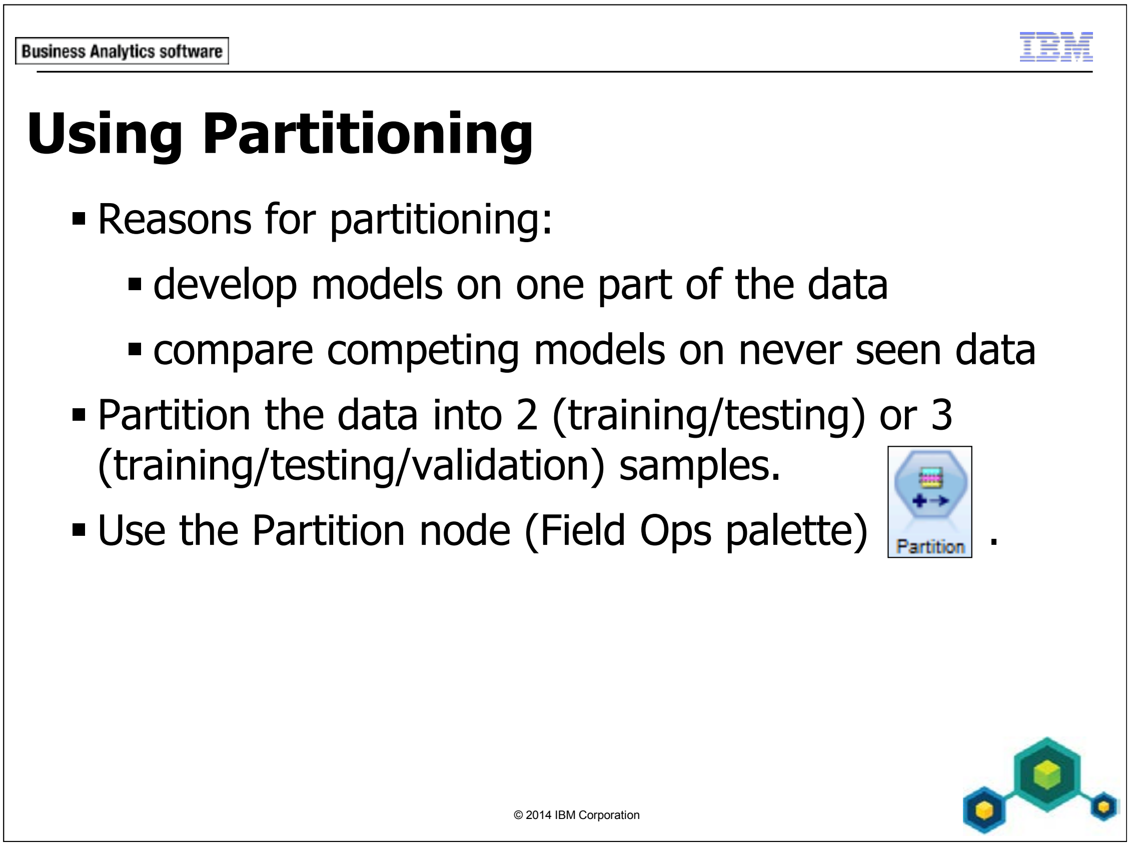
□ Use block-level sampling (in-database only)

□ Maximum sample size 10000

□ Repeatable partition assignment Seed: 6342064

This slide shows the Sample dialog box when the Simple sample method is selected. You can choose whether to pass records in the sample (Include sample) or discard them (Discard sample). You can then select the type of simple sampling and enter the appropriate value for sampling, for example the sample percentage that you want to have for a random sample. If you want to obtain the same sample every time, you can fix the value for the seed (the Repeatable partition assignment option).

When you select the Complex sample method, the dialog box will reflect the changes. In the Complex sample dialog box, you can specify at most one cluster field, and one or more stratification fields. Furthermore, you can draw the sample by specifying proportions or counts, and you can customize the sample size for each stratum.



Business Analytics software

IBM

Using Partitioning

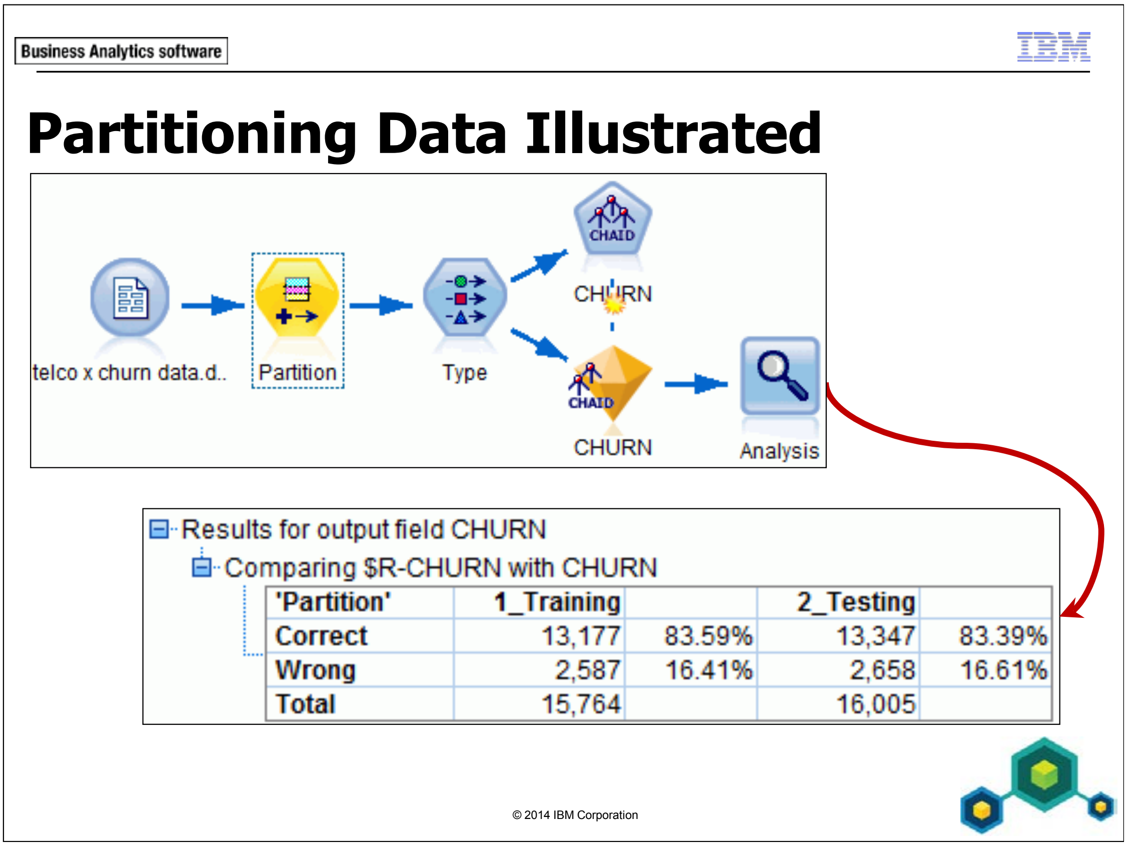
- Reasons for partitioning:
  - develop models on one part of the data
  - compare competing models on never seen data
- Partition the data into 2 (training/testing) or 3 (training/testing/validation) samples.
- Use the Partition node (Field Ops palette) .

© 2014 IBM Corporation

Partitioning the dataset means that you split the data randomly into a training dataset and a testing dataset. The model is built on the training dataset, and to establish the model’s the model's credibility/reliability model's, you want to see how the model performs on the testing set, which was used in model building. Or, you may develop competing models and to compare them you want to test them on never seen data.

You can subdivide the records into three samples instead of just two. The model will still be built on the training set and tested on the testing set. However, the testing set can then be used to further refine the model. For example, if you believe that the performance of the model needs improvement, the parameters can be changed, and the model is then rebuilt using the training sample, after which the performance on the testing set is examined. You may have to do this several times before the results are satisfactory. The validation sample, which unlike the training and testing sets played no role in developing the final model, is then used to assess the model the model’s performance against yet unseen data.

Use the Partition node to create the partitions. Rather than physically creating two datasets, the Partition node adds a new field that assigns a record to one of the partitions, retaining one dataset.



Partitioning Data Illustrated

Results for output field CHURN

Comparing $R-CHURN with CHURN

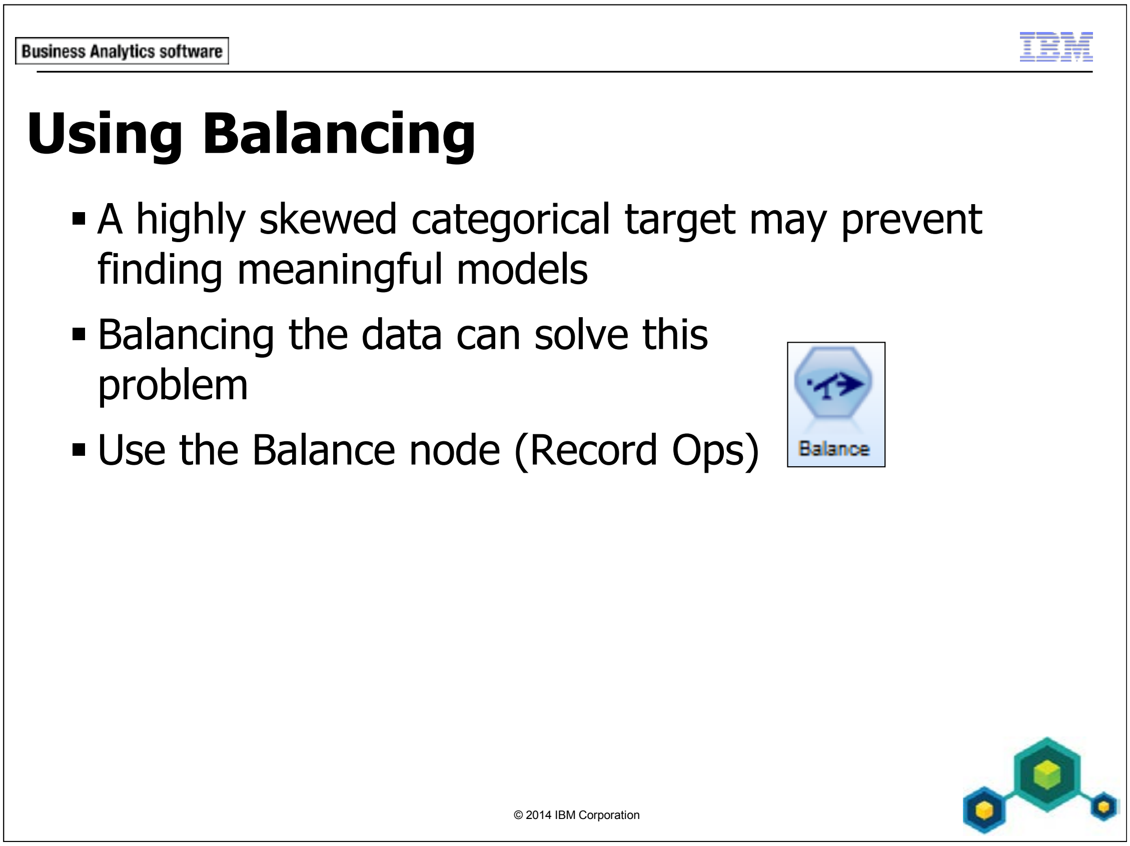
'Partition'	1_Training		2_Testing	
Correct	13,177	83.59%	13,347	83.39%
Wrong	2,587	16.41%	2,658	16.61%
Total	15,764		16,005	

To make use of the Partition node, insert the node upstream from the Type node that sets the roles for the fields in modeling.

When you compute performance measures, the results will be reported for all partitions, so models can be compared easily.

This slide presents an example of partitioning the data. A CHAID model is used to predict CHURN. The Partition node is found upstream from the Type node. When you run the CHAID model, only the records that are assigned to the training set are used to build the model. When you evaluate the CHAID model nugget with an Analysis node, the Analysis node will show the results for both the training set and the testing set.

When you would have tested multiple models, the Analysis node would have presented the results for all models, for both partitions, so that you can easily determine which best model performed best on the testing set.



Business Analytics software

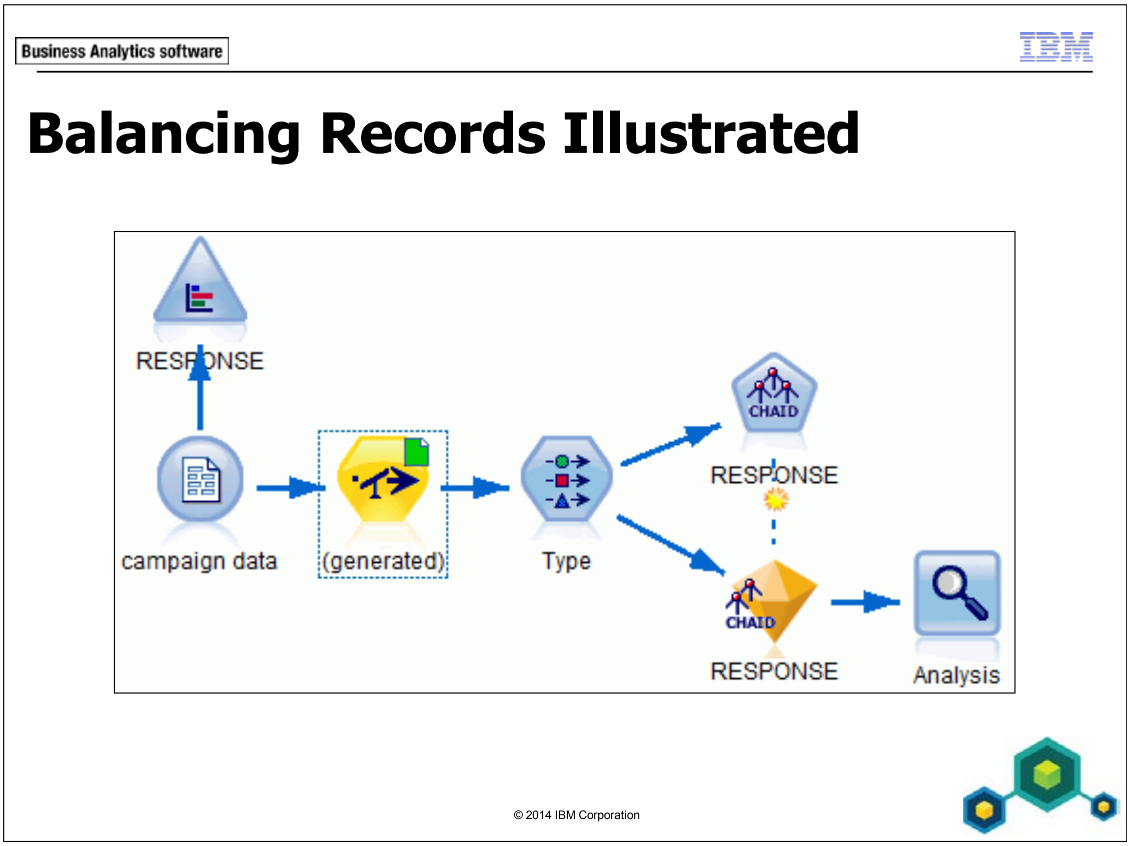
Using Balancing

- A highly skewed categorical target may prevent finding meaningful models
- Balancing the data can solve this problem
- Use the Balance node (Record Ops)

For modeling purposes, the data may have too few records in certain categories of the target field. For example, in predicting response to a mailing, in a dataset of 100,000 records, there may only be 500 responders, and 99,500 non-responders. Models may not be able to identify the responders when the distribution is that skewed. Balancing makes the distribution of a categorical field more equal, so better models can be built.

A balanced dataset is achieved by duplicating ("boosting") records in the lower-frequency categories or discarding ("reducing") records in the higher-frequency categories. In general reducing records in the higher-frequency categories is preferred over boosting records in the lower-frequency categories. The reason is that when you boost record you will run the risk of duplicating anomalies in the data. But when a dataset is very small, or when the number of records in specific categories is too few, boosting may be the only method that makes it possible to develop a reasonable model. Therefore, models developed with balanced data must be validated, and must be validated on a dataset that matches the population distribution of the target field.

To balance records, use the Balance node in the Record Ops palette. Alternatively, you can run a Distribution node on the field on which the data must be balanced, and generate the Balance node from the Distribution output window.



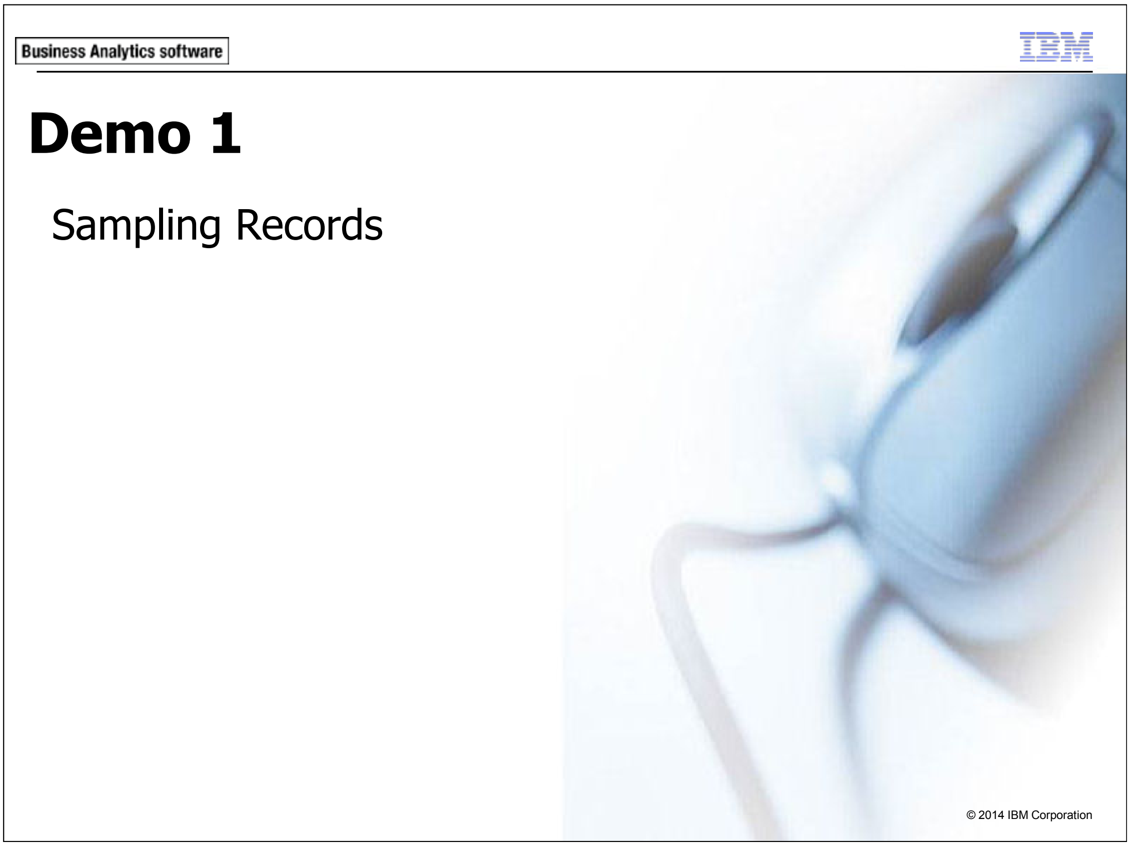
Business Analytics software

Balancing Records Illustrated

© 2014 IBM Corporation

In the stream depicted on this slide, a Balance node is placed upstream from the Type node that sets the roles for modeling.

The Balance node has no option to set the random and each time that records pass through the node other records will be output because of the random mechanism. If you want to replicate the results, cache the data at the Balance node.



Business Analytics software

Demo 1

Sampling Records

The following files are used to demonstrate how you can sample records:

- **property_assessment.dat**: A (synthetic) text file storing property values across a range of locations. The dataset contains 11,128 records with information on a property's county and township. There are 5 counties, and 49 townships. The file is located in **C:\Train\0A055**.
- **demo_sampling_records.str**: A MODELER stream file that imports the data, sets appropriate measurement levels and roles, and instantiates the data. The file is located the **04-Sampling_Records\Start** sub folder.

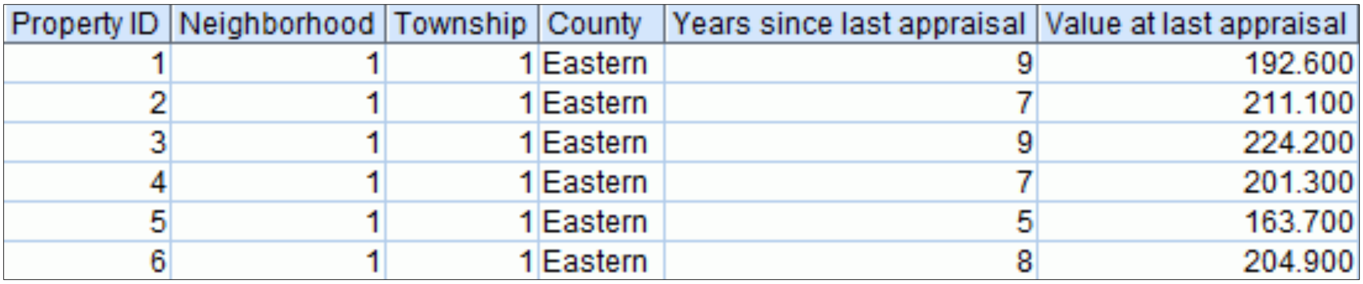
Before you begin with the demo, ensure that:

- You have started MODELER.
- You have set MODELER's working folder. (In MODELER, select **File\Set Directory**. When you are working in an IBM operated environment, browse to the folder **C:\Train\0A055** and then click **Set** to select it. When you are not working in an IBM operated environment, ask your trainer for the folder name.)
- You have opened **demo_sampling_records.str**.

Demo 1: Sampling Records

Purpose:
You will sample data using various techniques, and use partitioning to select the best predictive model.

This demo uses a dataset storing property values across a range of locations. The figure below shows the first records.



Property ID	Neighborhood	Township	County	Years since last appraisal	Value at last appraisal
1	1	1	Eastern	9	192.600
2	1	1	Eastern	7	211.100
3	1	1	Eastern	9	224.200
4	1	1	Eastern	7	201.300
5	1	1	Eastern	5	163.700
6	1	1	Eastern	8	204.900

Each property is one record in the dataset. The value at the last appraisal is the value in thousands. For example, 192.6 represents a value of 192,600.

Task 1. Draw a simple sample and a complex sample.

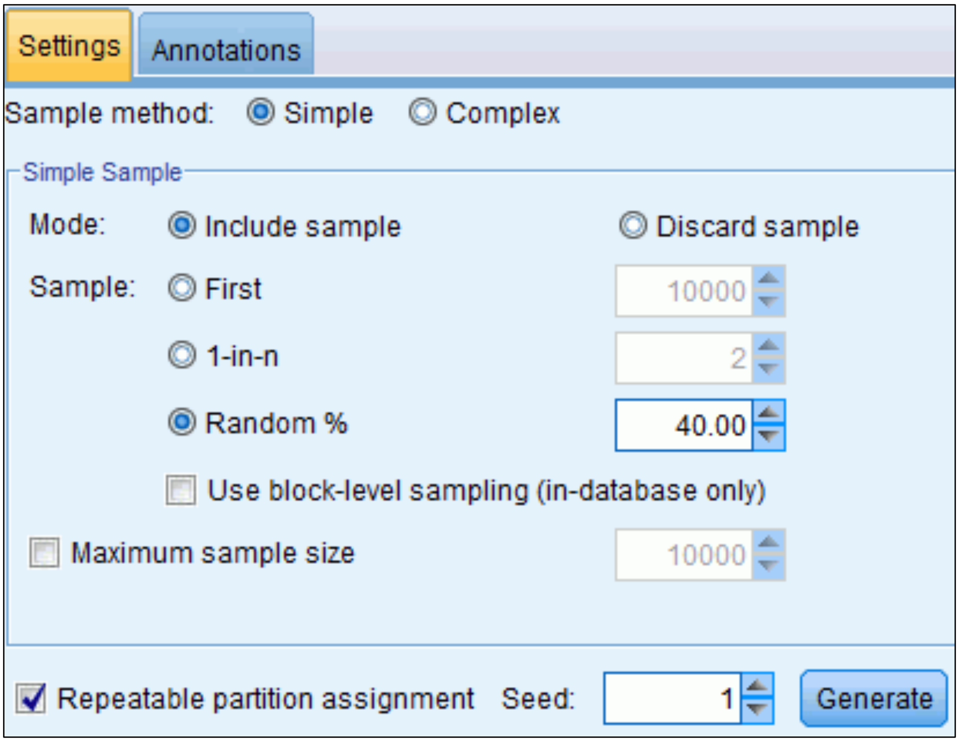
Properties must be visited for inspection, for example to investigate if the value at the last appraisal was appropriate. In order to cut costs in the project, not all properties will be visited, but only 40% of them. Thus, the task is to randomly select 40% of the properties. You will use simple sampling and complex sampling to select, approximately, 40% of the properties.

1. Scroll to the **comment** reading **Simple and complex sampling**.

   Note: Ensure that you have opened **demo_sampling_records.str**.

   You will sample 40% of the original dataset, using simple random sampling
2. Add a **Sample** node downstream from the **Type** node.
3. Edit the **Sample** node, and then:
   - for **Sample method**, ensure that **Simple** is selected
   - for **Sample**, select **Random %**, and set the value to **40**
   - enable the **Repeatable partition assignment** option, and for **Seed**, type **1**

A section of the results appear as follows:



Sample method: ● Simple ○ Complex

Simple Sample

Mode: ● Include sample ○ Discard sample

Sample: ○ First 10000

○ 1-in-n 2

● Random % 40.00

□ Use block-level sampling (in-database only)

□ Maximum sample size 10000

☑ Repeatable partition assignment Seed: 1

Setting the value of the seed will accomplish that every sample will be comprised of the same records.

4. Close the **Sample** dialog box.
5. Add a **Table** node downstream from the **Sample** node, and then run the **Table** node.

A section of the results appear as follows:

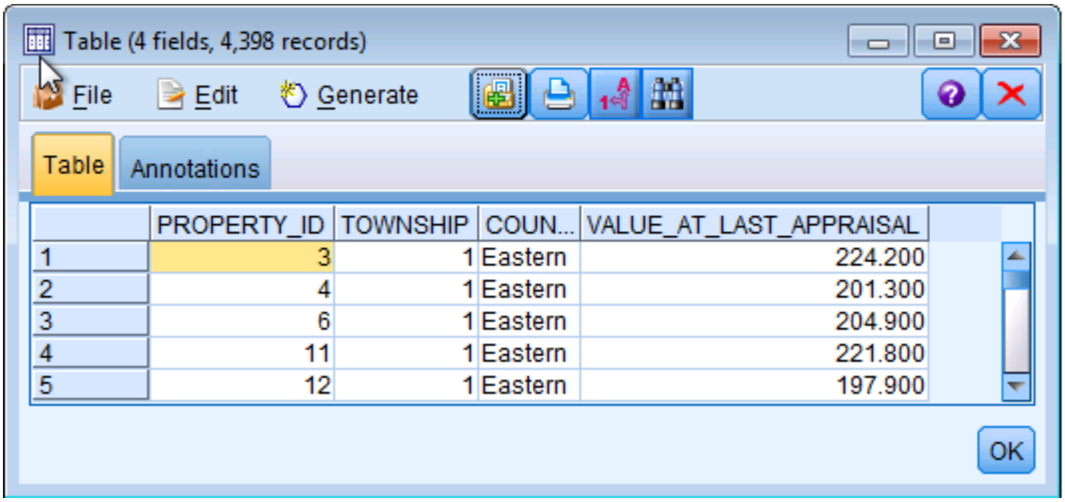


Table (4 fields, 4,398 records)

	PROPERTY_ID	TOWNSHIP	COUN...	VALUE_AT_LAST_APPRAISAL
1	3	1	Eastern	224.200
2	4	1	Eastern	201.300
3	6	1	Eastern	204.900
4	11	1	Eastern	221.800
5	12	1	Eastern	197.900

There were 4,398 records sampled, which is 39.5% of the full dataset with 11,128 records. Thus, approximately 40% of all records were selected.

6. Close the **Table** output window.

   You can use this sample and visit the selected properties, but you will run the risk that properties from some counties in the sample are under-represented, which might be a necessary condition for the project. Therefore, rather than drawing a simple random sample you will draw a stratified sample. Also, suppose that the properties to visit should come from a limited number of townships, in order to reduce costs for visiting the properties. Thus, clustering is needed to draw an independent sample of townships from within each county. Drawing a clustered sample means that some townships will be included and others will not, but for a township that is included all properties will be included in the sample.
7. Add a second **Sample** node downstream from the **Type** node.
8. Edit the **Sample** node, and then:
   - for **Sample method**, select **Complex**
   - click the **Cluster and Stratify** button
   - for **Clusters**, select **TOWNSHIP** (notice that only one field can be selected, so you can cluster on one field only)
   - for **Stratify** by, select **COUNTY**

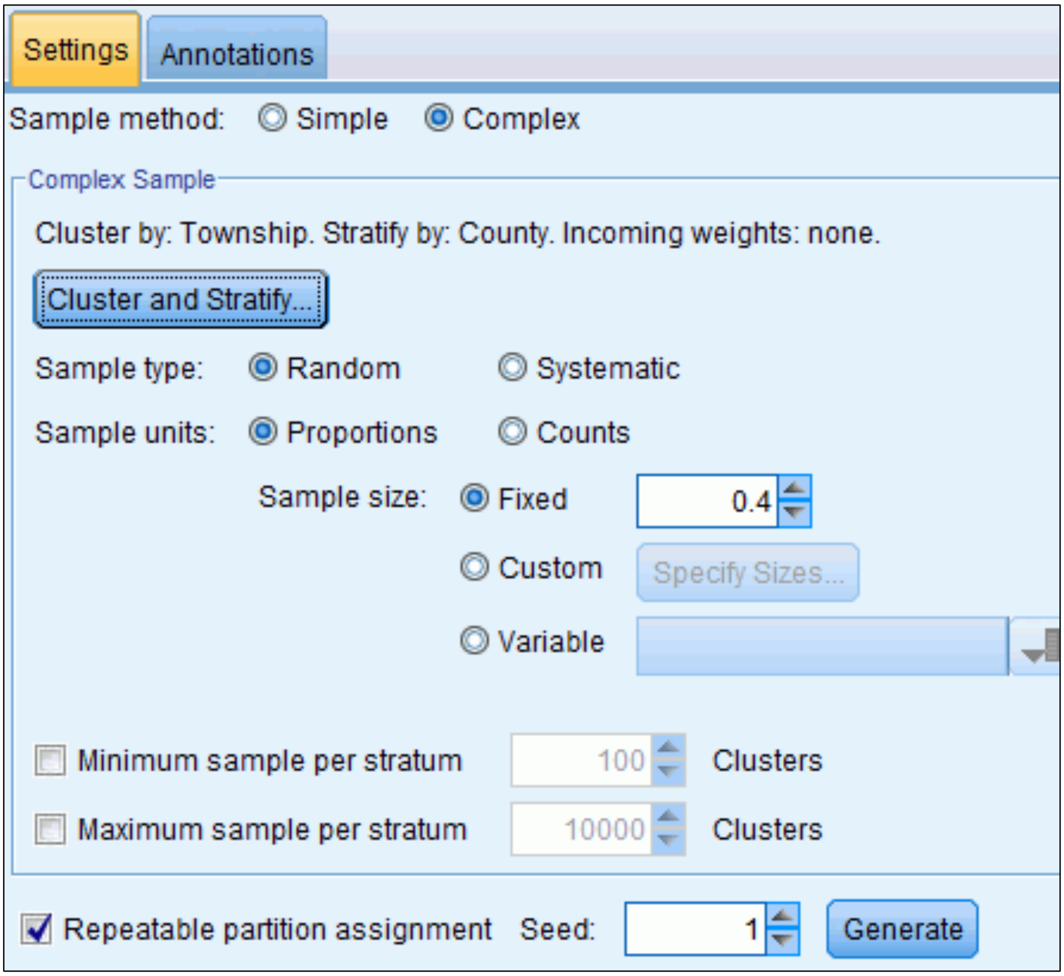
     Note: Multiple fields can be selected, so it is possible to stratify on multiple fields.
   - click **OK** to return to the main dialog box

   The sample size is set to 0.5 by default. This means that approximately half of the units are sampled within each stratum, units being the clusters (townships) here. For example, if a county has 8 townships, it is expected that 4 of them are selected. Within each selected township, all records belonging to that township will be selected because of the clustering.

   In this example you will draw a sample of 40% rather than of 50%, so you will change this parameter.

9. Continue editing the **Sample** node, and then:
   - for **Sample size**, option **Fixed**, enter **0.4**
   - enable the **Repeatable partition assignment** option, and for **Seed** enter value **1.**

   A section of the results appear as follows:



Sample method: ○ Simple ● Complex

Complex Sample

Cluster by: Township. Stratify by: County. Incoming weights: none.

Sample type: ● Random ○ Systematic

Sample units: ● Proportions ○ Counts

Sample size: ● Fixed 0.4

○ Custom

○ Variable

□ Minimum sample per stratum 100 Clusters

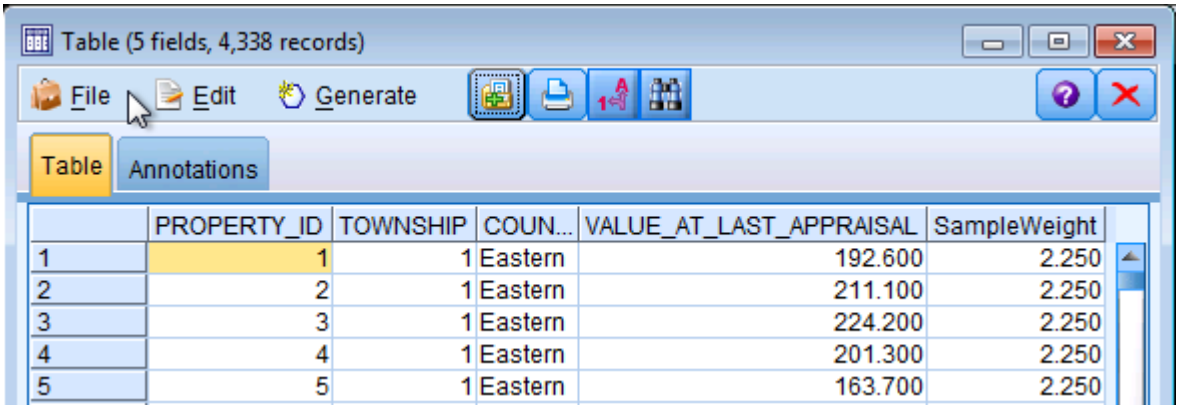
□ Maximum sample per stratum 10000 Clusters

☑ Repeatable partition assignment Seed: 1

10. Close the **Sample** dialog box.

11. Add a **Table** node downstream from the **Sample** node named **Complex**, and run the **Table** node.

    A section of the results appear as follows:



	PROPERTY_ID	TOWNSHIP	COUN...	VALUE_AT_LAST_APPRAISAL	SampleWeight
1	1	1	Eastern	192.600	2.250
2	2	1	Eastern	211.100	2.250
3	3	1	Eastern	224.200	2.250
4	4	1	Eastern	201.300	2.250
5	5	1	Eastern	163.700	2.250

    The SampleWeight field is automatically added to the data when you draw a complex sample. This field's values correspond, approximately, to the frequency that each sampled record represents in the original data. For example, the first record's sample weight is 2.25, which means that this record represents 2.25 records in the original data. When you sum the weights you would get, approximately, the number of records in the original dataset.

    It can be verified that 20 townships were drawn, which is (20/49) = 40.8% of the townships, close to the 40% that was requested. This results in 4,338 records drawn, which is (4,338/11,128) = 38.9% of all properties.

12. Close the **Table** output window.

    Leave the stream open for the next task.

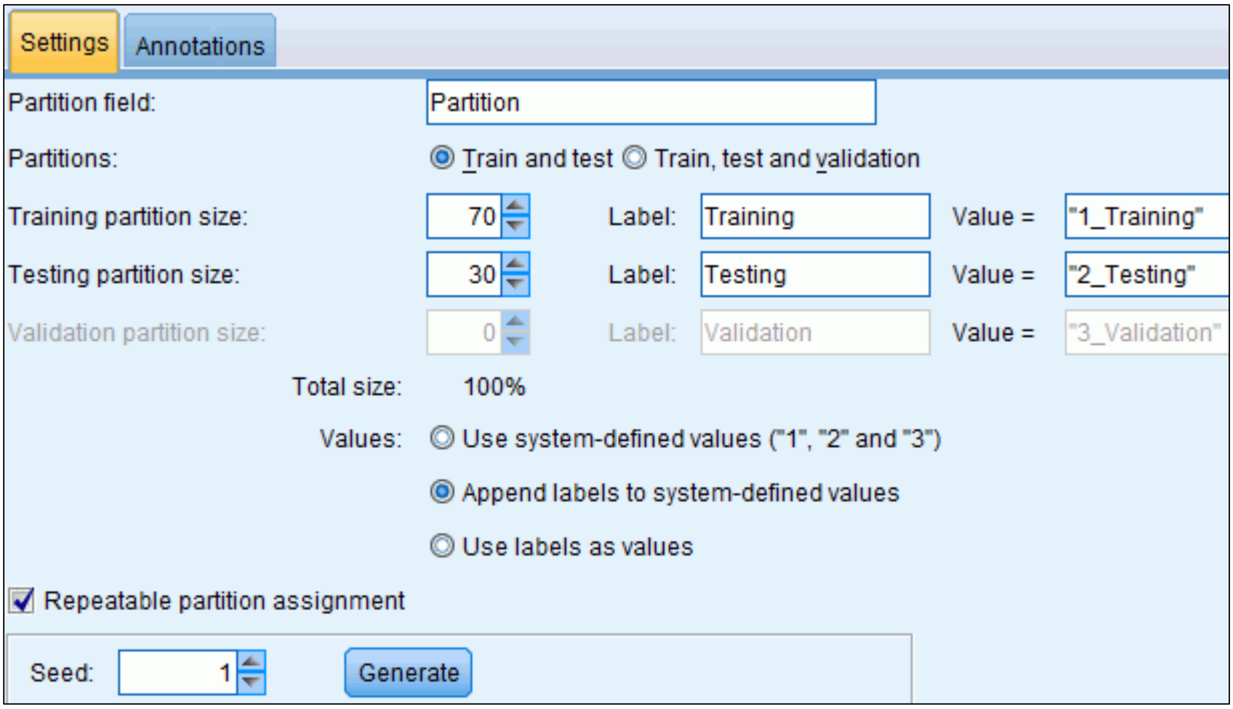
Task 2. Partition the data into a training set and a testing set.

You want to predict the property value using a number of competing models. You will build the models on 70% of the data, and test the models on 30% of the data.

You will build from the stream that you created in the previous task.

1. Scroll to the comment that reads **Partition data**.
2. Insert a **Partition** node (Field Ops) between the **Var. File** node and the **Type** node.
3. Edit the **Partition** node, and then:
    - for **Training partition size**, enter **70**
    - for **Testing partition size**, enter **30**
    - for **Repeatable partition assignment**, type **1**

A section of the results appear as follows:



Partition field: Partition

Partitions: ● Train and test ○ Train, test and validation

Training partition size: 70 Label: Training Value = "1_Training"

Testing partition size: 30 Label: Testing Value = "2_Testing"

Validation partition size: 0 Label: Validation Value = "3_Validation"

Total size: 100%

Values: ○ Use system-defined values ("1", "2" and "3")

● Append labels to system-defined values

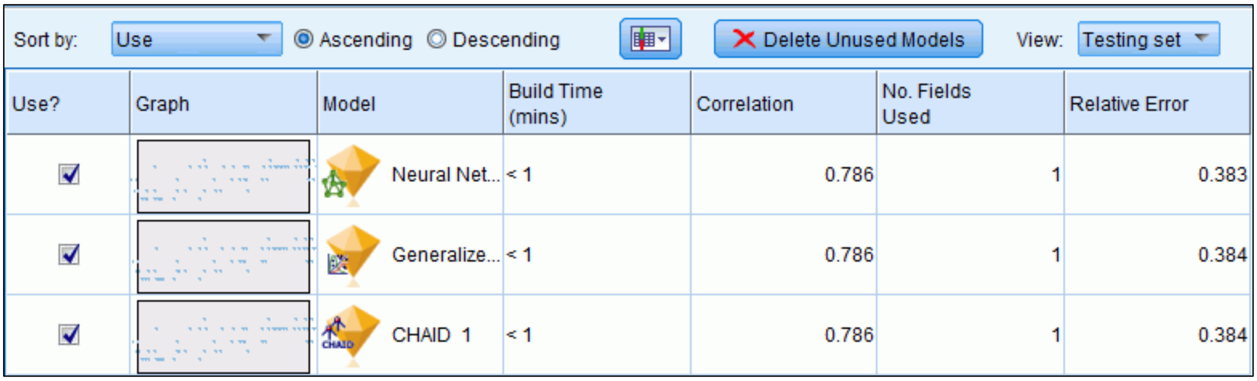
○ Use labels as values

☑ Repeatable partition assignment

Seed: 1

4. Close the **Partition** dialog box.
5. Run the **Auto Numeric** node named **VALUE_AT_LAST_APPRAISAL.**
6. Edit the **generated model nugget**.

A section of the results appear as follows:



Sort by: Use ● Ascending ○ Descending View: Testing set

Use?	Graph	Model	Build Time (mins)	Correlation	No. Fields Used	Relative Error
☑		Neural Net...	< 1	0.786	1	0.383
☑		Generalize...	< 1	0.786	1	0.384
☑		CHAID 1	< 1	0.786	1	0.384

The best three models are presented, based on the results on the testing set. All three models perform equally well in terms of correlation and relative error, viewing the testing set. Thus, it is a matter of taste which model you prefer.

Note: The results for the Neural Net model may differ from those presented here, because the Neural Net algorithm implements a random sample mechanism itself.

At this point you will not go into the modeling details. All in all, it was demonstrated how you can compare a number of competing models on never seen data, by partitioning the data into a training set and a testing set.

7. Close the **model nugget** window.

   Leave the stream open for the next task.

Task 3. Balance the data.

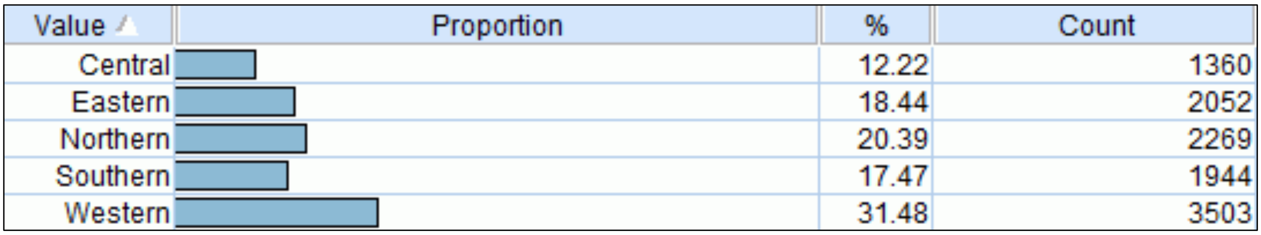
You will use the Balance node to get the same number of records in each county, by reducing the number of records in the larger categories.

Note: Balancing will in general be done on the target field. Although this demonstration does not follow this principle, it suffices to see the effect of balancing.

You will build from the stream that you created in the previous task.

1. Scroll to the third part of the stream, where the comment reads **Balancing records**.
2. Run the **Distribution** node for **COUNTY**.

   The results appear as follows.



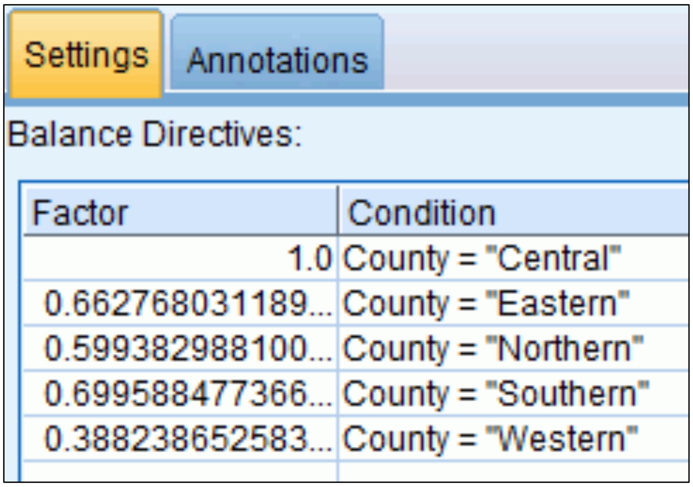
Value	Proportion	%	Count
Central		12.22	1360
Eastern		18.44	2052
Northern		20.39	2269
Southern		17.47	1944
Western		31.48	3503

   The Central category has the lowest frequency, so this category will have a weight of 1 when you reduce the number of records in the larger categories. The other categories will have a weight less than 1 and the Western category will have the smallest weight.

3. In the **Distribution** output window, select **Generate\Balance Node (reduce)**.
4. Close the **Distribution** output window.
5. Add the generated **Balance** node (located in the upper left corner on the stream canvas) downstream from the **Type** node.

6. Edit the **Balance** node.

   A section of the results appear as follows:



Factor	Condition
1.0	County = "Central"
0.662768031189...	County = "Eastern"
0.599382988100...	County = "Northern"
0.699588477366...	County = "Southern"
0.388238652583...	County = "Western"

   The weights are as expected.

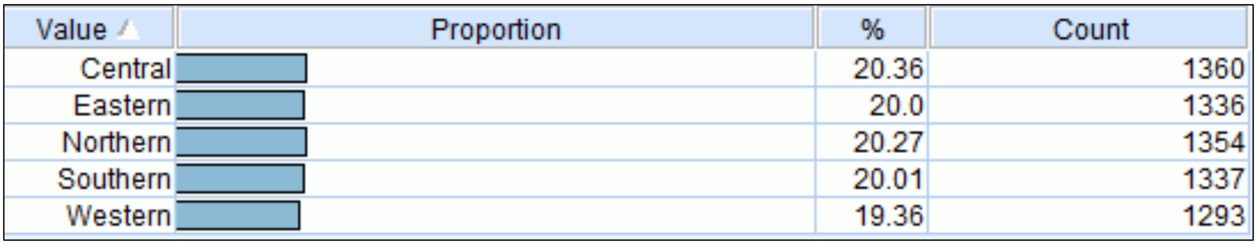
7. Close the **Balance** dialog box.

   To check the effect of balancing, you will run a Distribution node on the balanced data.

8. Add a **Distribution** node downstream from the **Balance** node.

9. Edit the **Distribution** node, select **COUNTY**, and then run the **Distribution** node.

   A section of the results appear as follows:



Value	Proportion	%	Count
Central		20.36	1360
Eastern		20.0	1336
Northern		20.27	1354
Southern		20.01	1337
Western		19.36	1293

   All categories have approximately the same number of records, as required.

   Note: Your results may be slightly different because the Balance node has no seed value for the random mechanism and so the results will differ from run to run.

   You may want to build a new model to predict the value. We leave that as an exercise. As noticed, balancing is usually done to balance the number of records in the categories of the target field. In this respect the demo is a-typical, but serves its purpose.

This completes the demo for this module. You will find the solution results in **demo_sampling_records_completed.str**, located in the **04-Sampling_Records\Solutions** sub folder.

Results:
You have sampled data using various techniques and you partitioned your data so that you could choose the predictive model that performed best on a testing dataset.

Apply Your Knowledge

Use the questions in this section to test your knowledge of the course material.

Question 1: Which of the following statements are correct? In MODELER, the Sample node, mode Simple, enables the user to:

A. Draw a random sample of 40%.

B. Include the first 1000 records, discard records 1001 thru 2000 and include records 2001 and further.

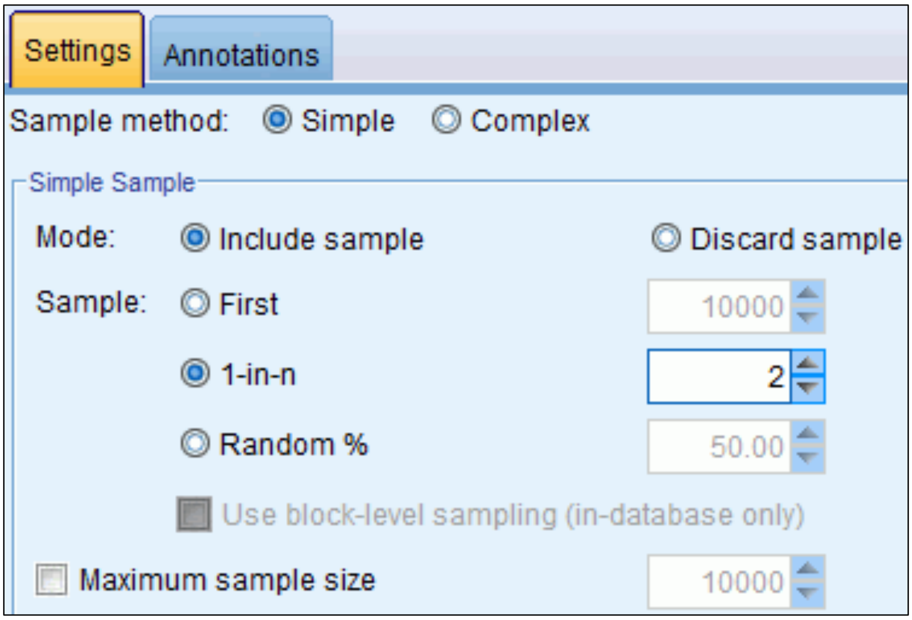
C. Replicate a sample.

D. Draw a random sample with a maximum sample size.

Question 2: Is the following statement true or false? Refer to the figure below. This sample will retain the records with an even record number (record #2, record #4, and so forth).

A. True

B. False



Sample method: ● Simple ○ Complex

Simple Sample

Mode: ● Include sample ○ Discard sample

Sample: ○ First 10000

● 1-in-n 2

○ Random % 50.00

☒ Use block-level sampling (in-database only)

□ Maximum sample size 10000

Question 3: Is the following statement true or false? Suppose that a dataset has 1,000 records, and a random sample of size 50% is drawn. Then exactly 500 records will be drawn.

A. True

B. False

Question 4: Which of the following is the correct statement? Suppose that you have a categorical field that defines certain groups and that you want to ensure that records from all these groups are represented in the sample. The type of sample you need is a:

A. Simple sample

B. Clustered sample

C. Stratified sample

Question 5: Is the following statement true or false? The Sample node enables you to draw a clustered sample within a stratified sample.

A. True

B. False

Question 6: Which of the following is the correct statement? Suppose you want to first draw a clustered sample from schools (field SCHOOL), and then draw a clustered sample from classes (field CLASS) within schools. What will you do?

A. Specify SCHOOL and CLASS in the Cluster by area in the Sample node.

B. Specify SCHOOL in the Cluster by area in a first Sample node, and then add a second Sample node downstream from the first sample node and specify CLASS in the Cluster by area in this second Sample node.

Question 7: Which of the following is the correct statement? When analyzing product associations in shopping carts (each product bought making up one record in the dataset) you want to work with sample data but you want to ensure that all products bought from a selected transaction are included in the sample. The type of sampling needed is:

A. Simple Sampling

B. Clustered sampling

C. Stratified sampling

Question 8: Is the following statement true or false? The reason for partitioning is to evaluate the model on data that is not used in the modeling process.

A. True

B. False

Question 9: Is the following statement true or false? The partition field has measurement level continuous.

A. True

B. False

Question 10: Is the following statement true or false? The objective for balancing records is to build better models.

A. True

B. False

Question 11: Which of the following is the correct statement? When you want to replicate the results of the balance node within a MODELER session, you need to:

A. Set the value of the random seed in the Balance node.

B. Cache the data on the Balance node.

C. Do not use the Balance node, but the Partition node to boost or reduce records.

D. None of the above statements are correct.

Answers to questions:

Answer 1: A, C, D. The Sample node enables you draw a random sample of 40%, to replicate a sample, and to limit the size of the sample. You cannot sample blocks, as alternative B suggests.

Answer 2: A. True. The selected sample method is 1-in-2, including records. This will sample the records #2, #4, #6, and so forth.

Answer 3: B. False. Approximately 50% will be drawn.

Answer 4: C. Want you want to ensure that records from all groups are in the sample, you need to draw a stratified sample, where the strata are defined by the categorical field.

Answer 5: A. True. The Complex sample mode allows you to specify a cluster field and a stratification field.

Answer 6: B. You can only specify one field as cluster field in the Sample dialog box. In order to cluster on two fields you can add two Sample nodes to your stream.

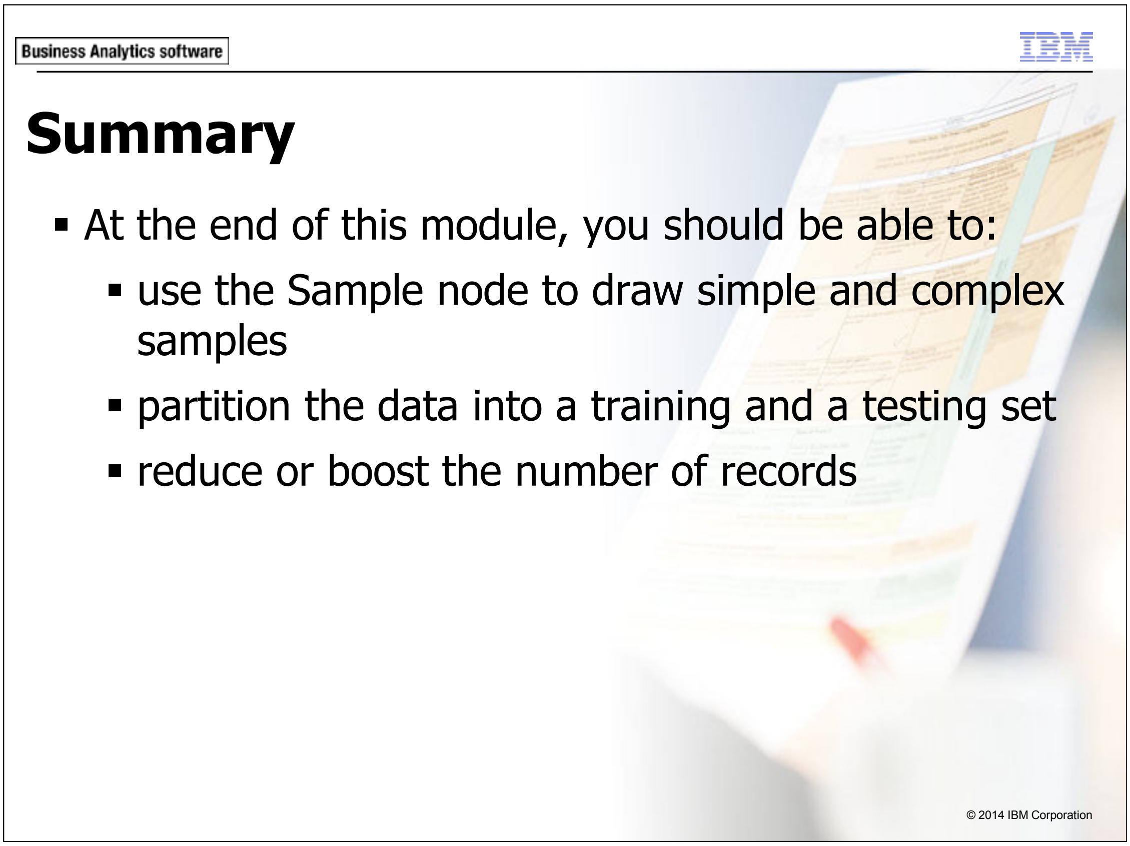
Answer 7: B. When you want to ensure that all records for a selected transactions are sample, you need a clustered sample.

Answer 8: A. Partitioning data enables you to evaluate the model on data that is not used when you built the model.

Answer 9: B. False, the partition field is a nominal field, not continuous.

Answer 10: A. True. Data are balanced to build better models.

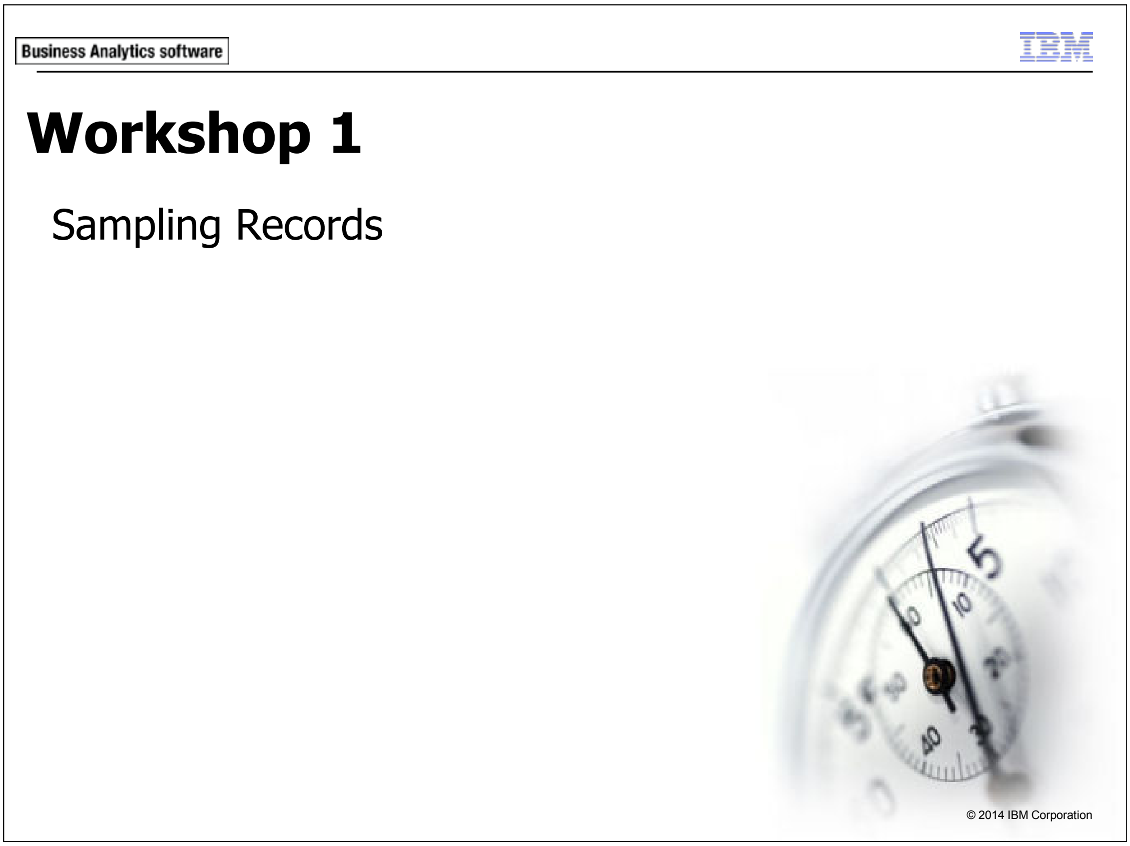
Answer 11: B. The Balance node does not have the option to set the seed. If you want to replicate results within the same session you should cache the data at the Balance node.



Business Analytics software

Summary

- At the end of this module, you should be able to:
  - use the Sample node to draw simple and complex samples
  - partition the data into a training and a testing set
  - reduce or boost the number of records



Business Analytics software

Workshop 1

Sampling Records

The following (synthetic) file is used in this workshop:

- **charity.txt:** A text file storing data on donations to charity. The file is located in **C:\Train\0A055**.

Before you begin with the workshop, ensure that:

- You have started MODELER.
- You have set MODELER's working folder. (In MODELER, select **File\Set Directory**. When you are working in an IBM operated environment, browse to the folder **C:\Train\0A055** and then click **Set** to select it. When you are not working in an IBM operated environment, ask your trainer for the folder name.)

Workshop 1:Sampling Records

You are working for a charity organization and you have to sample data for a satisfaction survey. Also, you have to build a model to predict response to the campaign

- Import the data from **charity.txt**, add a **Type** node downstream from the source node, set the measurement level for **CUSTOMER_ID** to **Typeless**, for **MOSAIC_BAND** to **Nominal**, instantiate the data, and then run a **Distribution** node on **RESPONSE_TO_CAMPAIGN**.

  What percentage responded to the campaign?

- Your organization wants to conduct a satisfaction survey on a 10% random sample. Sample 10% of the records, using seed value 1.

  How many records were sampled? Is this exactly 10% of all records?

- Instead of drawing a random sample, you want to draw a sample of 0.5% (a proportion of 0.005) of the non-responders, and 10% (a proportion of 0.1) of the responders. Draw this sample, by using a complex sample method. Use 1 as seed value.

  Run a **Distribution** node on **RESPONSE_TO_CAMPAIGN** to check your results.

  In the next tasks you will build a model, using a training and testing set.

- Edit the **Type** node downstream from the **Var. file** node, and then:

  - set the **Role** for **RESPONSE_TO_CAMPAIGN** to **Target**
  - set the **Role** for the other fields to **Input** (except for **CUSTOMER_ID**, which is **Typeless** and has **Role None**)

- Insert a **Partition** node between the **Var. File** and the **Type** node and partition the data into a **70%** training set and **30%** testing set; set the seed to **1**.

- Add an **Auto Classifier** node (Modeling palette, Automated item) downstream from the **Type** node, and then run the **Auto Classifier** node.

  Edit the **generated model nugget**. Which model has the highest **Overall Accuracy** on the testing set?

- Insert a **Balance** node between the **Partition** node and the **Type** node, and see to it that all responders pass through the **Balance** node, and 5% (a proportion of 0.05) of the non-responders. Rerun the **Auto Classifier** model.

  Did the overall accuracy on the testing set improve? What is your conclusion?

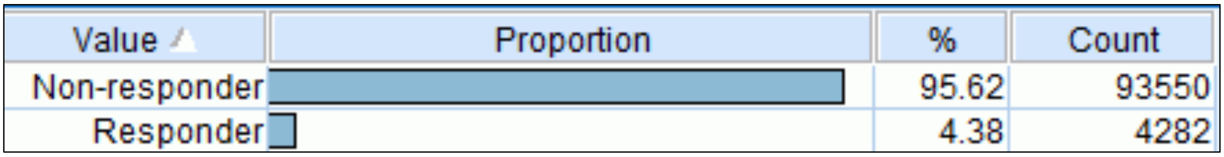
For more information about where to work and the workshop results, refer to the Task and Results section that follows. If you need more information to complete a task, refer to earlier demos for detailed steps.

Workshop 1: Tasks and Results

Task 1. Import the data, instantiate the data and examine the response.

- Use a **Var. File** node to import data from **charity.txt**.
- Add a **Type** node downstream from the **Var. File** node, edit the **Type** node, set the measurement level for **CUSTOMER_ID** to **Typeless**, for **MOSAIC_BAND** to **Nominal**, and then run click **Read Values** to instantiate the data.
- Add a **Distribution** node downstream from the **Type** node, edit the **Distribution** node, select **RESPONSE_TO_CAMPAIGN**, and then run the **Distribution** node.

A section of the results appear as follows:



Value	Proportion	%	Count
Non-responder		95.62	93550
Responder		4.38	4282

Only 4.38% responded to the campaign.

Task 2. Draw a random sample of 10%.

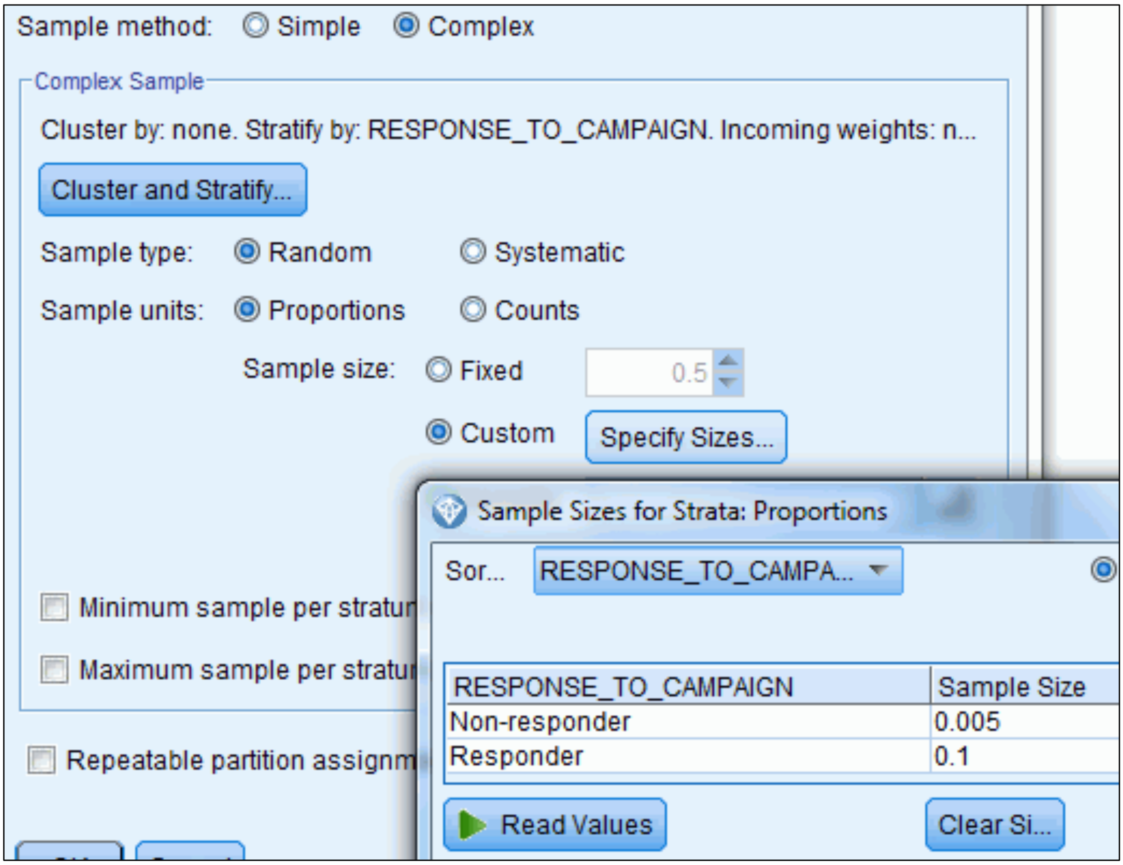
- Add a **Sample** node downstream from the **Type** node, select the **Random %** option and the value to **10**; for **Repeatable partition assignment**, set the value to **1**.
- Run a **Table** downstream from the **Sample** node.

The Table output shows that a sample of 10,010 records will be drawn, which is (10,010 / 97,832) * 100 = 10.2318% of all records, approximately 10%. The Random r% will never sample exactly r%, because each record will have a r/100 probability of being passed through the Sample node, making the outcome unpredictable for each record, and hence for the total number of records that will pass through the node.

Task 3. Draw a stratified sample.

- Add a **Sample** node downstream from the **Type** node.
- Edit the **Sample** node, and then:
    - for **Sample method**, select **Complex**
    - click **Cluster and Stratify,** and for **Stratify by** select **RESPONSE_TO_CAMPAIGN**.
    - for **Repeatable partition assignment**, type **1**
    - for **Sample size**, select **Custom**, and click **Specify Sizes**
    - click **Read Values** to populate the categories
    - for **Non-responder**, specify **0.005**, for **Responder** specify **0.1**

A section of the results appear as follows:



Sample method: ○ Simple ● Complex

Complex Sample

Cluster by: none. Stratify by: RESPONSE_TO_CAMPAIGN. Incoming weights: n...

Sample type: ● Random ○ Systematic

Sample units: ● Proportions ○ Counts

Sample size: ○ Fixed 0.5

● Custom

□ Minimum sample per stratu

□ Maximum sample per stratu

□ Repeatable partition assignm

Sample Sizes for Strata: Proportions

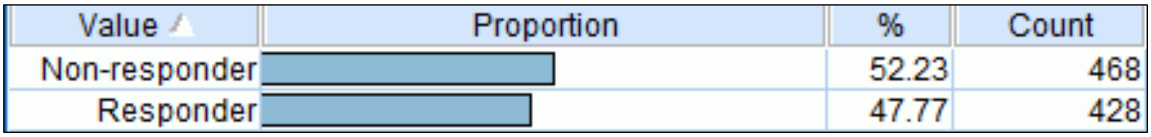
Sor... RESPONSE_TO_CAMPA...

RESPONSE_TO_CAMPAIGN	Sample Size
Non-responder	0.005
Responder	0.1

This material is meant for IBM Academic Initiative purposes.

- Add a **Distribution** node downstream from the **Sample** node, edit the **Distribution** node , select **RESPONSE_TO_CAMPAIGN**, and then run the **Distribution** node.

  A section of the results appear as follows:



Value	Proportion	%	Count
Non-responder		52.23	468
Responder		47.77	428

Task 4. Prepare for modeling by using a Type node.

- Edit the **Type** node downstream from the **Var. file** node, and then:
  - set the **Role** for **RESPONSE_TO_CAMPAIGN** to **Target**
  - set the **Role** for the other fields to **Input** (except for CUSTOMER_ID, which is Typeless)

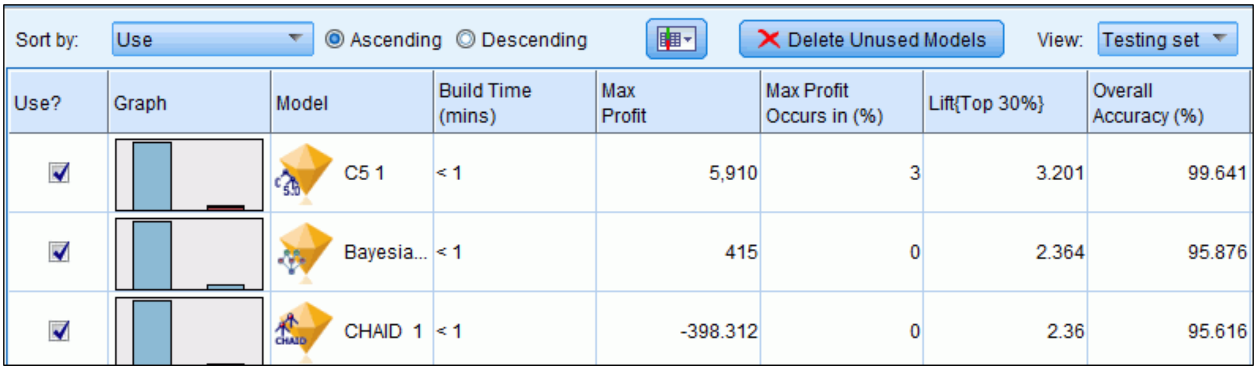
Task 5. Partition the data into a training set and testing set.

- Insert a **Partition** node between the **Var. File** and the **Type** node.
- Edit the **Partition** node, and then:
  - for **Training partition size**, type **70**
  - for **Testing partition size**, type **30**
  - for **Repeatable partition assignment**, type **1**

Task 6. Run models on the training set and select the best model on the testing set.

- Add an **Auto Classifier** node (Modeling palette, Automated item) downstream from the **Type** node, and then run the **Auto Classifier** node.
- Edit the **generated model nugget**.

  A section of the results appear as follows:



Sort by: Use ◉ Ascending ○ Descending | ✕ Delete Unused Models | View: Testing set

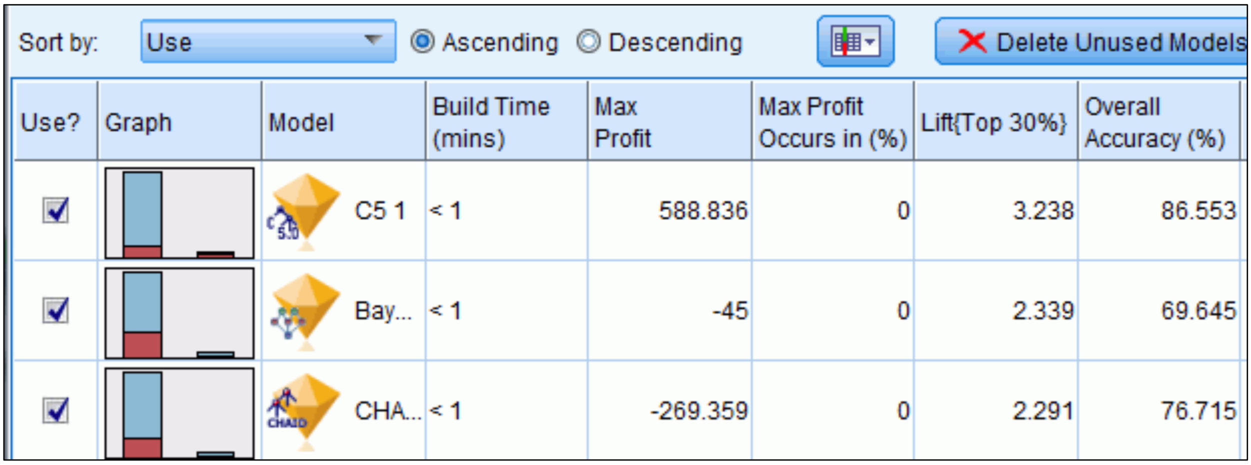
Use?	Graph	Model	Build Time (mins)	Max Profit	Max Profit Occurs in (%)	Lift{Top 30%}	Overall Accuracy (%)
☑		C5 1	< 1	5,910	3	3.201	99.641
☑		Bayesia...	< 1	415	0	2.364	95.876
☑		CHAID 1	< 1	-398.312	0	2.36	95.616

The C5.1 model has the highest overall accuracy on the testing set (99.641%).

Task 7. Use balancing to build models.

- Rerun the **Distribution** node downstream from the **Type** node.
- From the **Distribution** output window, select **Generate\Balance Node (reduce)**.
- Insert the **generated Balance** node between the **Partition** node and the **Type** node.
- Edit the **generated Balance node** and set the value for **Factor** to **0.05** for **non-responders**.
- Rerun the **Auto Classifier** node.
- Edit the **generated model nugget**.

A section of the results appear as follows:



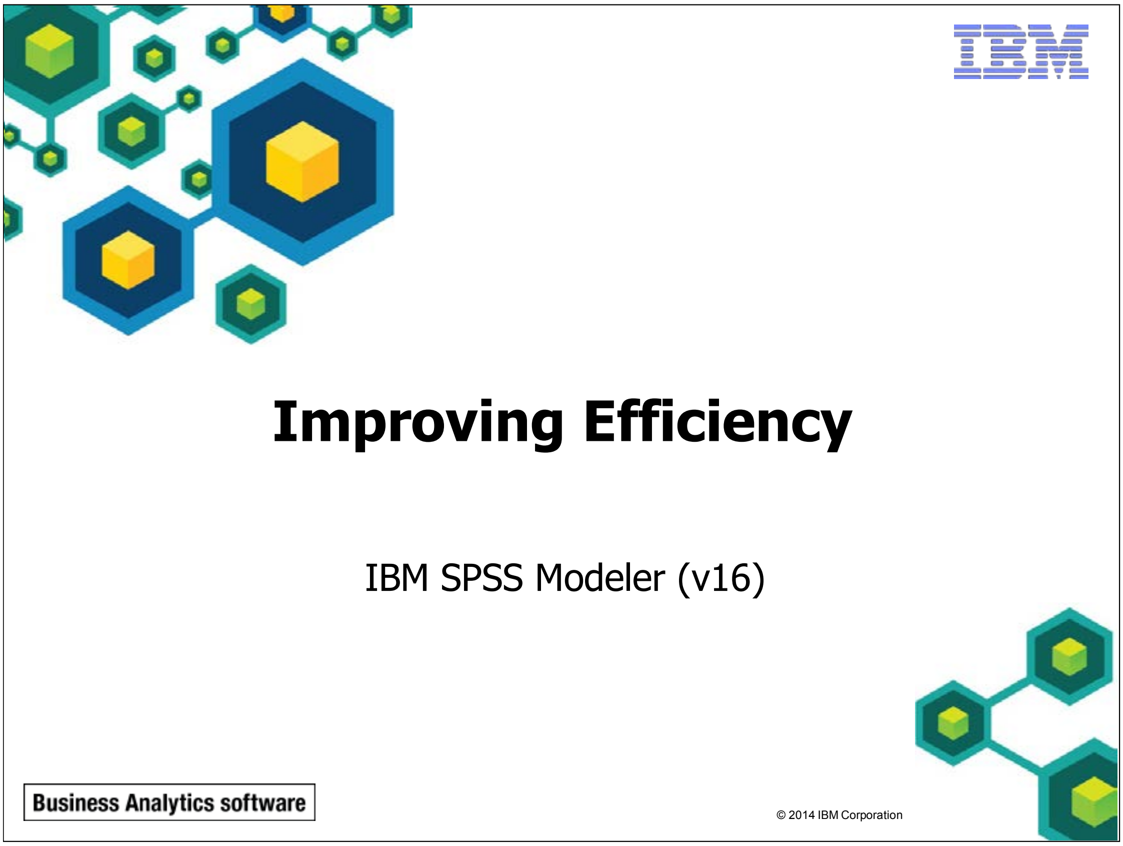
Sort by: Use ● Ascending ○ Descending ✕ Delete Unused Models

Use?	Graph	Model	Build Time (mins)	Max Profit	Max Profit Occurs in (%)	Lift{Top 30%}	Overall Accuracy (%)
☑		C5 1	< 1	588.836	0	3.238	86.553
☑		Bay...	< 1	-45	0	2.339	69.645
☑		CHA...	< 1	-269.359	0	2.291	76.715

Balancing did not improve the results, so there is no reason for balancing in this dataset to build better models to predict **RESPONSE_TO_CAMPAIGN**.

Note: Your results may differ from those presented here, because of the random selection of records in the Balance node.

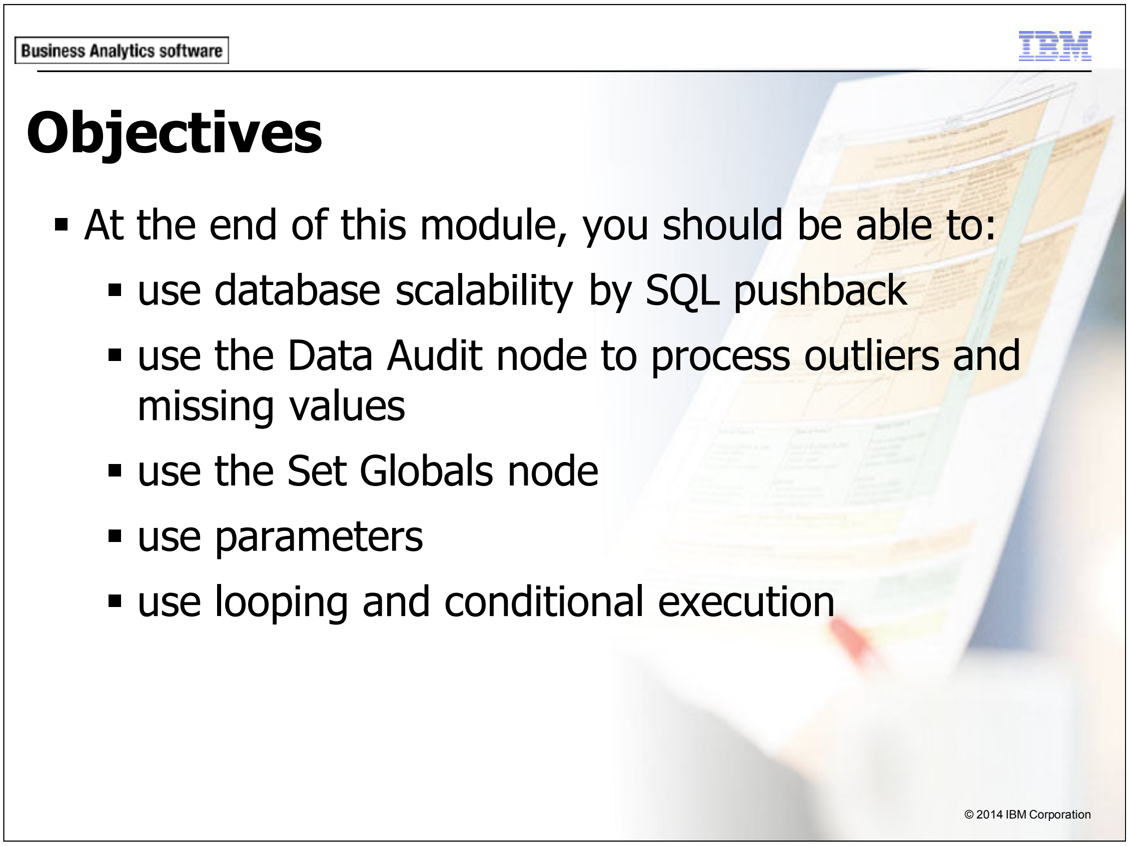
Note: The stream **workshop_sampling_records_completed.str**, located in the **04-Sampling_Records\Solutions** sub folder, provides a solution to the workshop.



Improving Efficiency

IBM SPSS Modeler (v16)

Business Analytics software



Business Analytics software

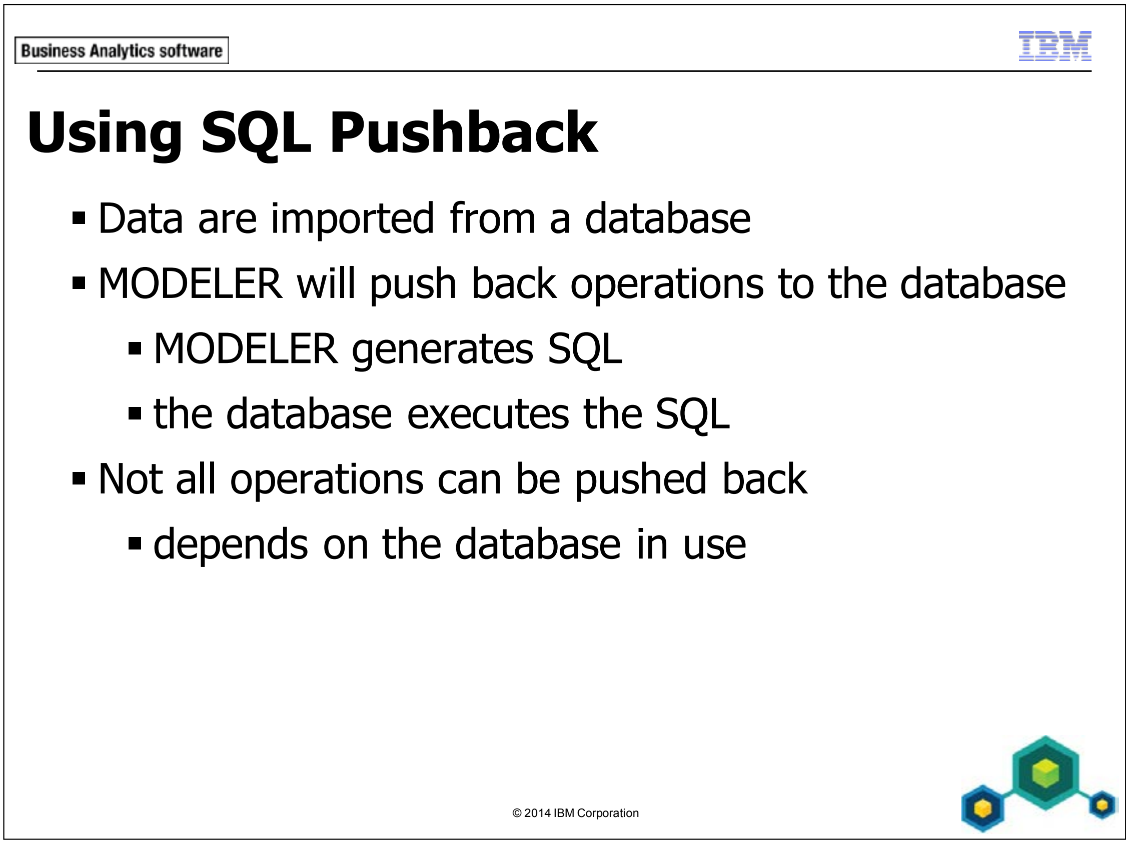
IBM

Objectives

- At the end of this module, you should be able to:
  - use database scalability by SQL pushback
  - use the Data Audit node to process outliers and missing values
  - use the Set Globals node
  - use parameters
  - use looping and conditional execution

Before reviewing this module you should be familiar with:

- working with MODELER (streams, nodes, SuperNodes, palettes)
- importing data (Var. File node)
- defining measurement levels, roles, blanks, and instantiating data (Type node)
- selecting fields and records (Filter node, Select node)
- CLEM and the Expression Builder
- deriving and filling fields (Derive node, Filler node)
- running a model and examining the generated model nugget
- examining the data with graphs (Distribution node, Histogram node) and tables (Table node, Data Audit node, Statistics node, Matrix node, Means node)



Business Analytics software

Using SQL Pushback

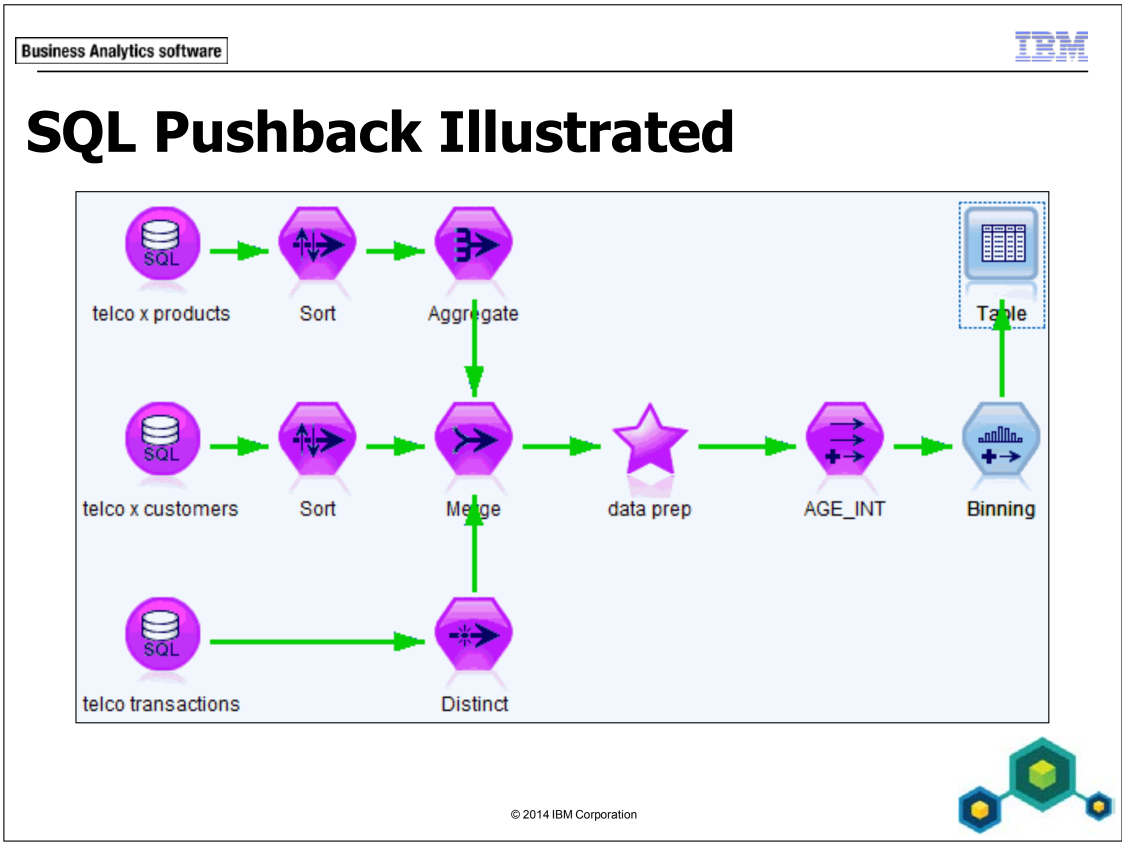
- Data are imported from a database
- MODELER will push back operations to the database
  - MODELER generates SQL
  - the database executes the SQL
- Not all operations can be pushed back
  - depends on the database in use

© 2014 IBM Corporation

Many data miners will access data directly from a database rather than work on extracts that have been written out to separate files. When you work with database sources, one of the most powerful capabilities of MODELER is the ability to perform data preparation operations directly in the database. By generating SQL code that is pushed back to the database for execution, many operations, such as deriving fields, merging, aggregating, sampling, and sorting can be performed in the database itself rather than that they are executed by MODELER. This functionality is called SQL pushback.

Not all operations can be pushed back to the database and it depends on the database that is in use which operations are supported. When MODELER encounters a node that cannot be compiled to SQL, MODELER will extract the data from the database and process the data from there.

When you import data from a database, you can make use of various database aggregate functions in the Derive node and Select node. For example, you can use these aggregation functions to compute a moving average, not using MODELER's @MEAN function, but using the appropriate database function to ensure SQL pushback.



Business Analytics software

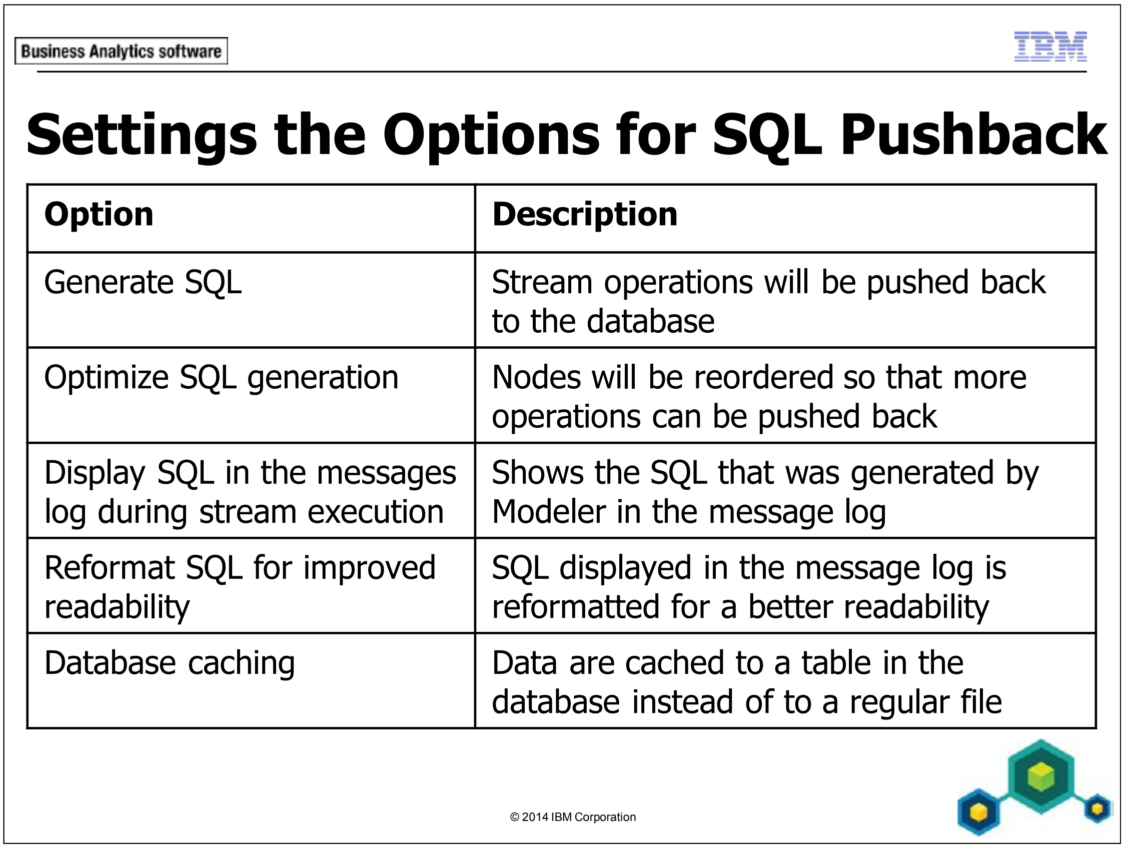
SQL Pushback Illustrated

© 2014 IBM Corporation

When a stream is executed and there is SQL pushback, the nodes whose functionality is being pushed back to the database will be highlighted in purple.

The stream depicted on this slide reads data from three database sources, sorts the data, merges the data, runs some data preparation nodes (encapsulated in the SuperNode), derives a field AGE_INT, runs a Binning node, and ends with a Table. When the stream is executed, all the nodes that can be pushed back to the database turn purple. Here, all nodes turn purple, except for the Binning node and the Table node. This means, for example, that MODELER outsourced creating the AGE_INT field to the database (by generating SQL code and having that code executed by the database). In comparison, binning a field was not supported by this database. Thus, MODELER has extracted the data from the database after the AGE_INT field was created in the database, and will proceed with the Binning operation, using MODELER's binning algorithms.

When this stream is executed using another database, it is not guaranteed that the same nodes turn purple. The operations that are pushed back are determined by the database in use.



Business Analytics software

Settings the Options for SQL Pushback

Option	Description
Generate SQL	Stream operations will be pushed back to the database
Optimize SQL generation	Nodes will be reordered so that more operations can be pushed back
Display SQL in the messages log during stream execution	Shows the SQL that was generated by Modeler in the message log
Reformat SQL for improved readability	SQL displayed in the message log is reformatted for a better readability
Database caching	Data are cached to a table in the database instead of to a regular file

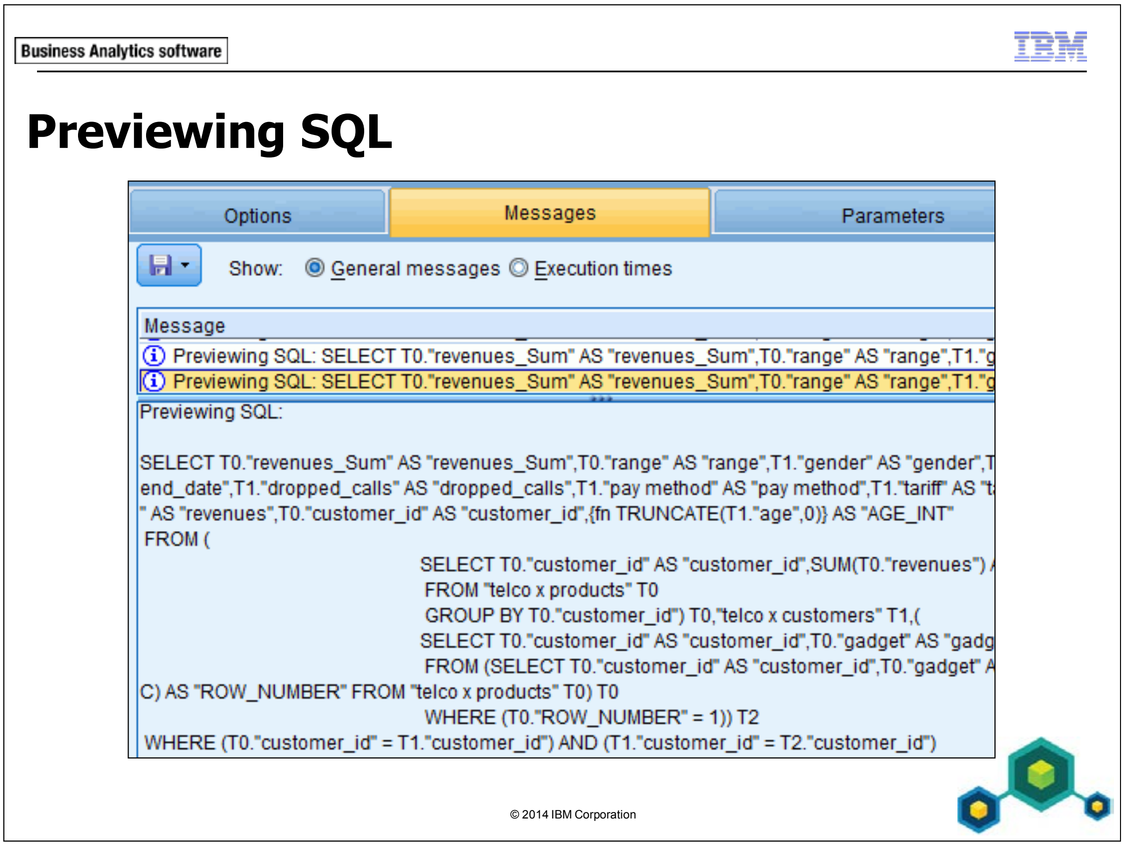
© 2014 IBM Corporation

Options for SQL Pushback can be set in the Options tab in the Stream Properties dialog box (accessed via Tools\Stream Properties\Options in MODELER's main menu; refer to the items Optimization and Logging and Status).

The Generate SQL option must be enabled to facilitate SQL pushback. With the Optimize SQL generation option enabled, MODELER will look for a node that cannot be compiled into SQL code and then look downstream from that node to check for nodes that can be rendered into SQL code. MODELER will then reorder the nodes, provided that it does not affect the results.

The SQL code that was generated can be displayed in the messages log, and MODELER can reformat the code to improve the readability. Also, you can choose whether the SQL code is displayed in the database's native SQL functions, or in standard ODBC functions of the form.

To further improve efficiency, you can cache the data on a midstream node, in which case MODELER will try to store the data in a temporary database table rather than in an IBM SPSS Statistics .sav file (the regular format for the cache).



Business Analytics software
IBM

Previewing SQL

Options
Messages
Parameters
Show: ● General messages ○ Execution times
Message

```
Previewing SQL: SELECT T0."revenues_Sum" AS "revenues_Sum",T0."range" AS "range",T1."g
Previewing SQL: SELECT T0."revenues_Sum" AS "revenues_Sum",T0."range" AS "range",T1."g
```

```
Previewing SQL:

SELECT T0."revenues_Sum" AS "revenues_Sum",T0."range" AS "range",T1."gender" AS "gender",T
end_date",T1."dropped_calls" AS "dropped_calls",T1."pay method" AS "pay method",T1."tariff" AS "t
" AS "revenues",T0."customer_id" AS "customer_id",{fn TRUNCATE(T1."age",0)} AS "AGE_INT"
 FROM (
                                        SELECT T0."customer_id" AS "customer_id",SUM(T0."revenues") A
                                         FROM "telco x products" T0
                                         GROUP BY T0."customer_id") T0,"telco x customers" T1,(
                                        SELECT T0."customer_id" AS "customer_id",T0."gadget" AS "gadg
                                         FROM (SELECT T0."customer_id" AS "customer_id",T0."gadget" A
C) AS "ROW_NUMBER" FROM "telco x products" T0) T0
                                         WHERE (T0."ROW_NUMBER" = 1)) T2
 WHERE (T0."customer_id" = T1."customer_id") AND (T1."customer_id" = T2."customer_id")
```

© 2014 IBM Corporation

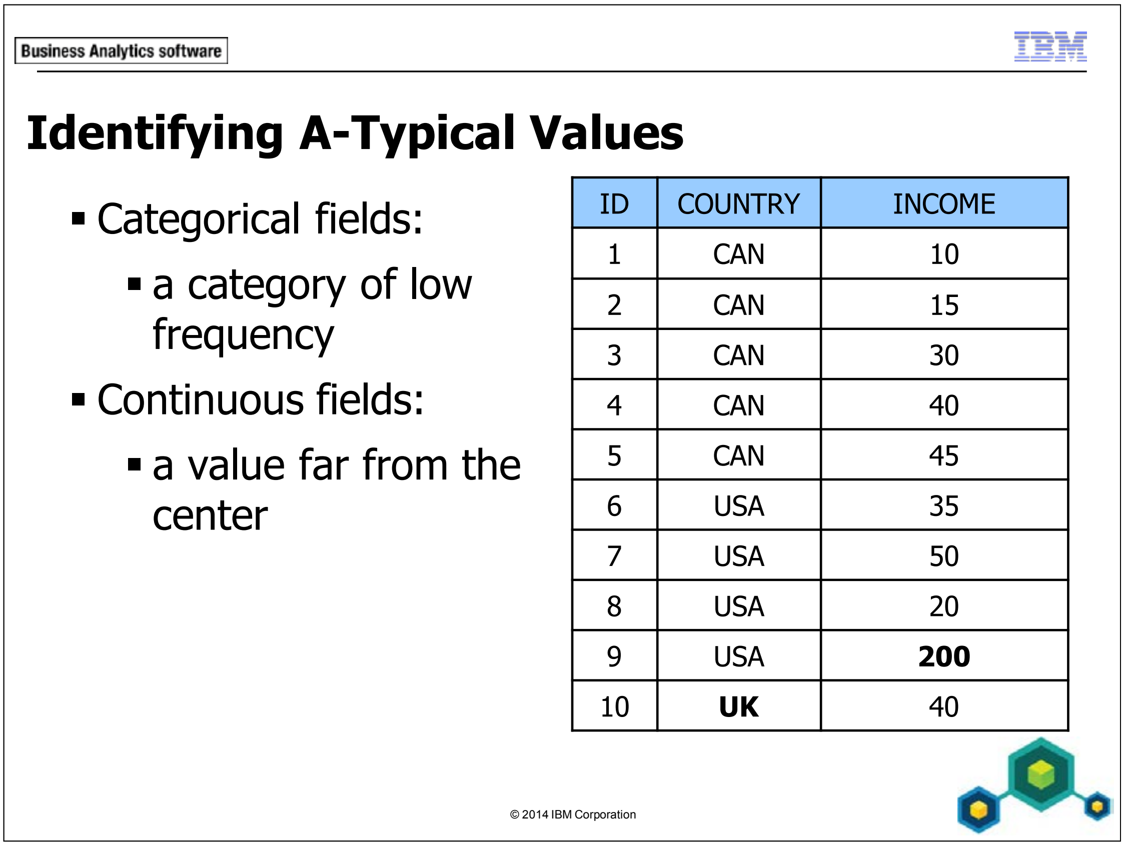
You can preview the SQL that will be generated in the messages log, by clicking the Preview run button in the Toolbar.

This slide shows a section of the messages log when the Preview run button was clicked. The SQL code that was generated by MODELER is in the lower part of the log.

You can copy and paste the SQL code to another application or to MODELER itself (in the Database node, you can enter an SQL query). In these situations it helps if the SQL code is reformatted for readability, as was done in the example here.

Next to previewing the SQL code, you can examine the SQL code in the messages log when you have executed a stream that implemented SQL pushback.

Note: SQL pushback is not demonstrated in this course, because it depends on the database in use. For the latest information on which databases are supported and tested for use with IBM SPSS Modeler 16, refer to the corporate Support site at http://www.ibm.com/support.



Business Analytics software

Identifying A-Typical Values

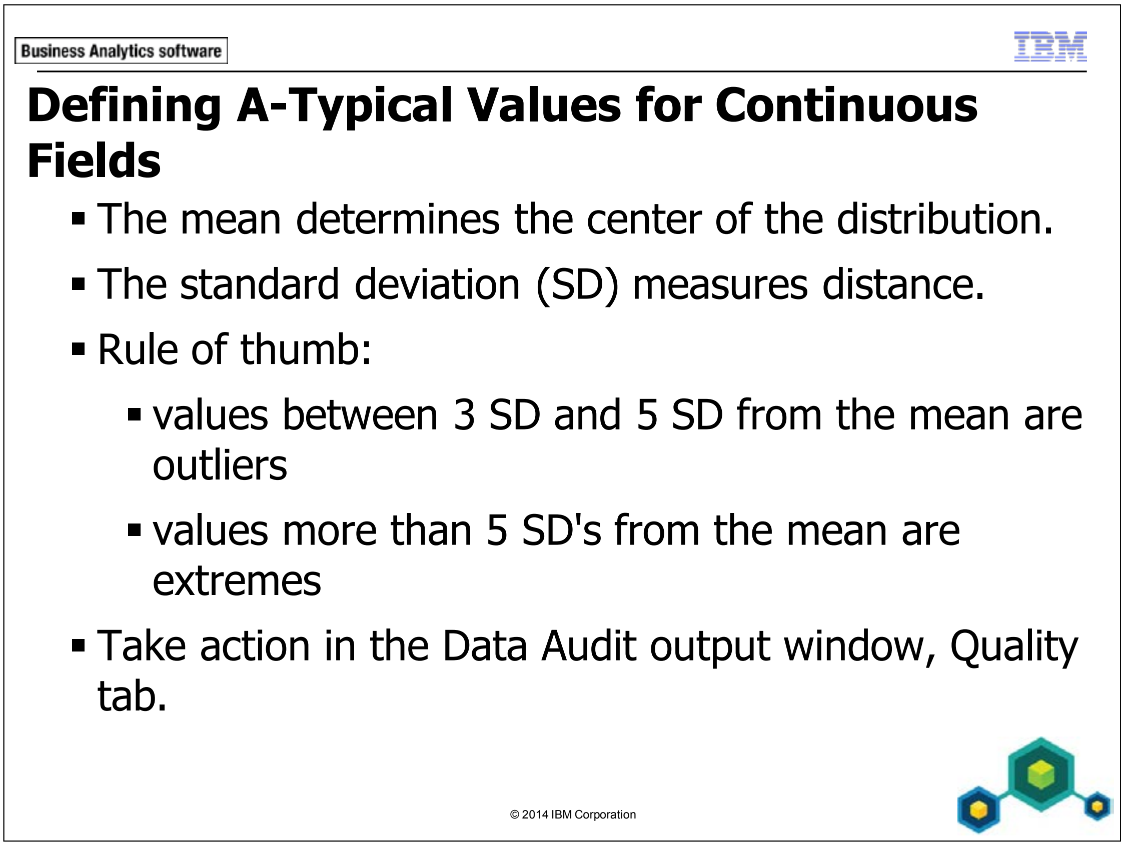
- Categorical fields:
  - a category of low frequency
- Continuous fields:
  - a value far from the center

ID	COUNTRY	INCOME
1	CAN	10
2	CAN	15
3	CAN	30
4	CAN	40
5	CAN	45
6	USA	35
7	USA	50
8	USA	20
9	USA	**200**
10	**UK**	40

In any data file there may be values on one or more fields that can be considered to be unusual, or a-typical. What a-typical values are depends on the measurement level of the fields. For a categorical field, a category with only a few records in it may be considered to be a-typical. For a continuous field, a value far of the center of the distribution is considered as a-typical. This slide gives an example for each measurement level.

A-typical values may make it more difficult to build models. For example classical statistical techniques like regression are sensitive to a-typical values and the smaller the number of records, the higher the impact these values. Also, a-typical outliers can affect data mining techniques such as neural networks or they simply can be errors in data coding. Some types of models are less affected by extreme data values, such as CHAID.

There are several angles to look at a-typical values. A-typical values can be defined by looking at a single field, or at several fields simultaneously. In the latter case you can use the Anomaly node. Refer to the *Clustering and Association Using IBM SPSS Modeler (v16)* course for a presentation of the Anomaly node. In this course a-typical values are defined by looking at a single field, especially a-typical values on a continuous field.



Business Analytics software

Defining A-Typical Values for Continuous Fields

- The mean determines the center of the distribution.
- The standard deviation (SD) measures distance.
- Rule of thumb:
  - values between 3 SD and 5 SD from the mean are outliers
  - values more than 5 SD's from the mean are extremes
- Take action in the Data Audit output window, Quality tab.

The center of a distribution is usually measured by the mean. How far away a value is from the mean is typically expressed in terms of standard deviations. For example, if a record has an income of 3,000, and the mean income and standard deviation are 2,000 and 500 respectively, then that income is (3,000 - 2,000)/500= 2 standard deviations above the mean income. An income of 1,500 is 1 standard deviation below the mean.

The mean and standard deviation have the important property that when a field's unit of measurement changes, then the mean and standard deviation change by the same factor. Consequently, the relative position of a score does not change when the unit of measurement changes. Scores computed in this way are called standardized scores or z-scores in the statistical literature.

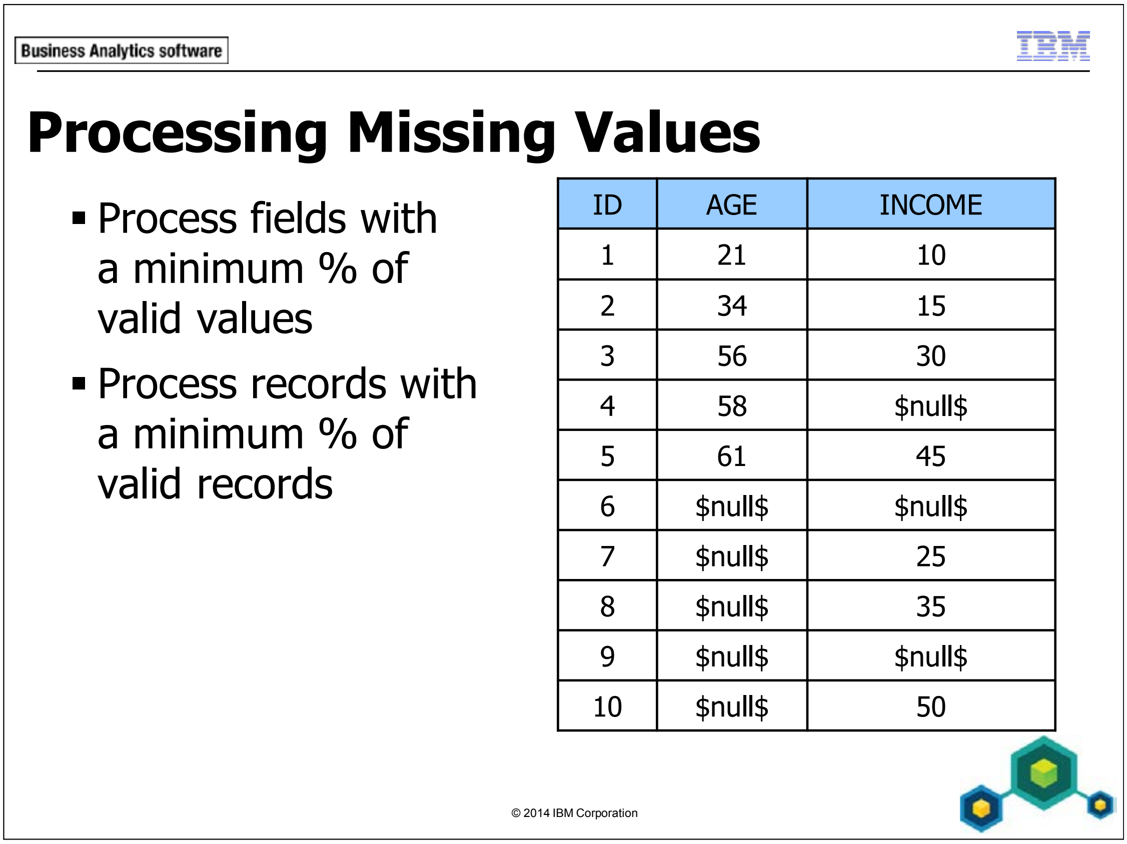
Typically, a threshold of 3 is used to mark outliers. An outlier is said to be extreme if it is more than 5 standard deviations away from the mean. The motivation for these thresholds comes from the normal distribution. Refer to text books on statistics for more information on mean, standard deviation, normal distribution and standardized scores.

You can deal with outliers and extremes by running a Data Audit node and then taking a specific action on the Quality tab in the Data Audit output window. You can choose to:

- Coerce outliers or extremes to the lower limit (mean - 3 * SD) or upper limit (mean - 3 * SD). The disadvantage of this approach is that all the values for these cases will be the same, causing a pile-up of records at one value.
- Discard records with outliers or extremes from the dataset. If their numbers are very small, this might be an acceptable strategy if one presumes that outliers and extremes values are of no special interest.
- Nullify outliers and extremes. Recode outliers and extremes to $null$; depending on the modeling technique in use, these records will be used for modeling.
- Coerce outliers, and either discard records with extreme values or nullify them.

You can try models with and without the modified fields to examine the effect of the action that was taken.

Note: The mean and the standard deviation themselves are sensitive to extreme values, especially in small datasets. For example, in a series of data of 1, 2, 3, 4, 5, 50, the mean (10.8) and standard deviation (19.2) will be inflated because of the value 50, and consequently the value 50 will not be marked as an outlier. Rather than using mean and standard deviation to measure center and spread, other statistics such as median and interquartile range are in use to define outliers and extremes. A discussion of these measures is beyond the scope for this course. Refer to the online Help for more information.



Business Analytics software

Processing Missing Values

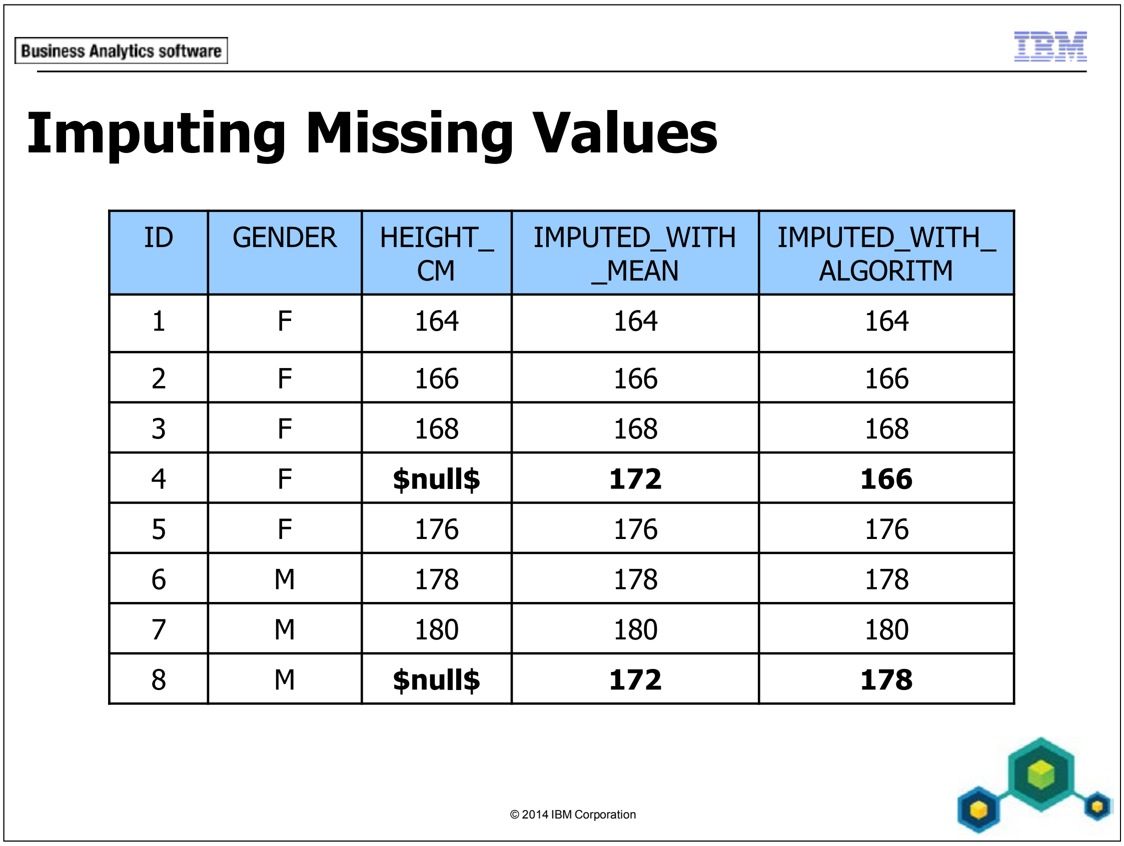
- Process fields with a minimum % of valid values
- Process records with a minimum % of valid records

ID	AGE	INCOME
1	21	10
2	34	15
3	56	30
4	58	$null$
5	61	45
6	$null$	$null$
7	$null$	25
8	$null$	35
9	$null$	$null$
10	$null$	50

The Data Audit output window not only enables you to process outliers and extremes, it also provides options to deal with missing values. When you have executed the Data Audit node you can generate a SuperNode from the Data Audit output that processes missing values. The Generate menu in the Data Audit output, Quality tab, enables you to filter fields that have a minimum percentage of valid values, or to select records that have a minimum percentage of valid values.

In the dataset depicted on this slide, when you specify that you want to retain only fields with at least 60% of valid values, you can generate a SuperNode that encapsulates a Filter node that will remove AGE from the dataset. Likewise, you can generate a SuperNode that contains a Select node that will retain records with, say, 50% valid values.

How you handle missing data when you are developing a model has some complications. For example, when you use only valid records to develop a model to predict an outcome, you cannot use the model successfully on new data unless all the fields have non-missing data, which is not very likely. Unless the number of records with missing data is negligible, following this scenario would prevent you from making predictions on an appreciable fraction of records in new data, which is not desirable.



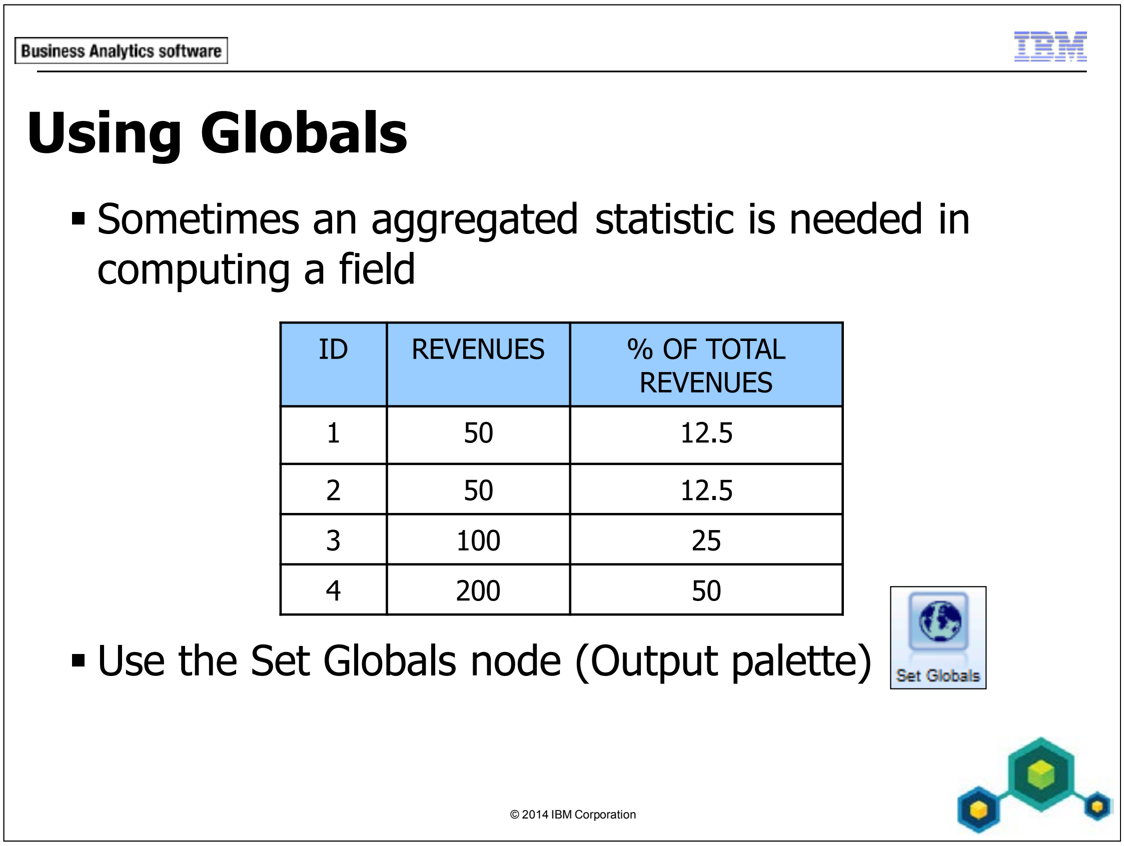
Business Analytics software

Imputing Missing Values

ID	GENDER	HEIGHT_ CM	IMPUTED_WITH _MEAN	IMPUTED_WITH_ ALGORITM
1	F	164	164	164
2	F	166	166	166
3	F	168	168	168
4	F	**$null$**	**172**	**166**
5	F	176	176	176
6	M	178	178	178
7	M	180	180	180
8	M	**$null$**	**172**	**178**

Rather than discarding fields or records you could decide to replace missing values with valid values, a technique which is called imputation. MODELER enables you to replace missing values with a fixed value such as the mean, with a value from a certain distribution such as the normal distribution, or, the most sophisticated alternative, you can use the C&RT algorithm to replace missing values.

This slide presents two examples of imputation: when HEIGHT is missing it can be replaced with the overall mean or you can use the C&RT algorithm to impute values, which will result in different values for men and women.



Business Analytics software

Using Globals

- Sometimes an aggregated statistic is needed in computing a field

ID	REVENUES	% OF TOTAL REVENUES
1	50	12.5
2	50	12.5
3	100	25
4	200	50

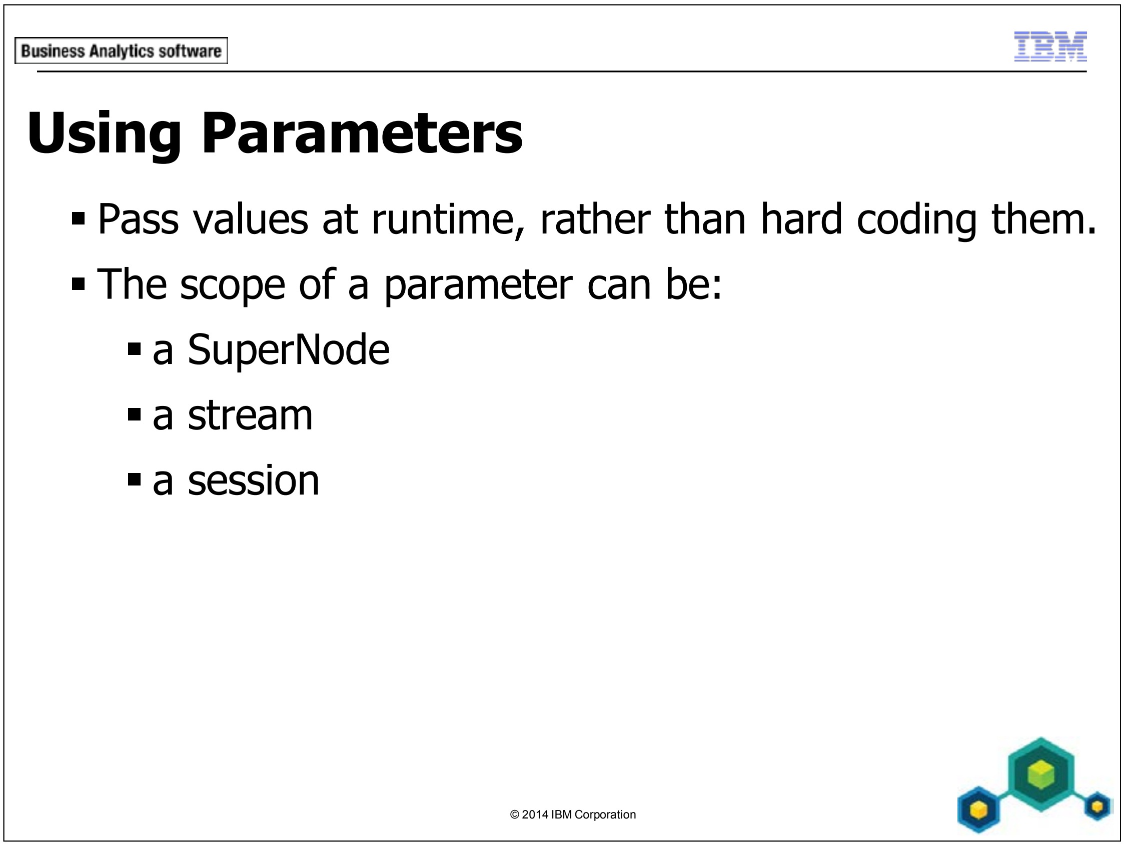
- Use the Set Globals node (Output palette)

Set Globals

Sometimes you need an aggregated statistic to compute a field. An example is depicted on this slide. The dataset stores customers and their revenues. Suppose that you need to derive a field that gives the share of the customer's revenues in the total revenues, as a percentage. In this example, the total revenues is 400, so for the first customer the result should be (50/400) * 100 = 12.5.

You need the sum of the revenues for this computation. Manually entering the sum is not an option. An alternative is to aggregate the data and to merge the aggregated dataset and the original dataset. This would add a new field to the data, with a value of 400 for each record. This means that the dataset is expanded needlessly, especially if there are a huge number of records.

A more efficient option is to use the Set Globals node. This node, located in the Output palette, calculates the mean, standard deviation, minimum, maximum and sum, and stores them in memory. When you have run the Set Globals node you can use the stored values in CLEM expressions.



Business Analytics software

Using Parameters

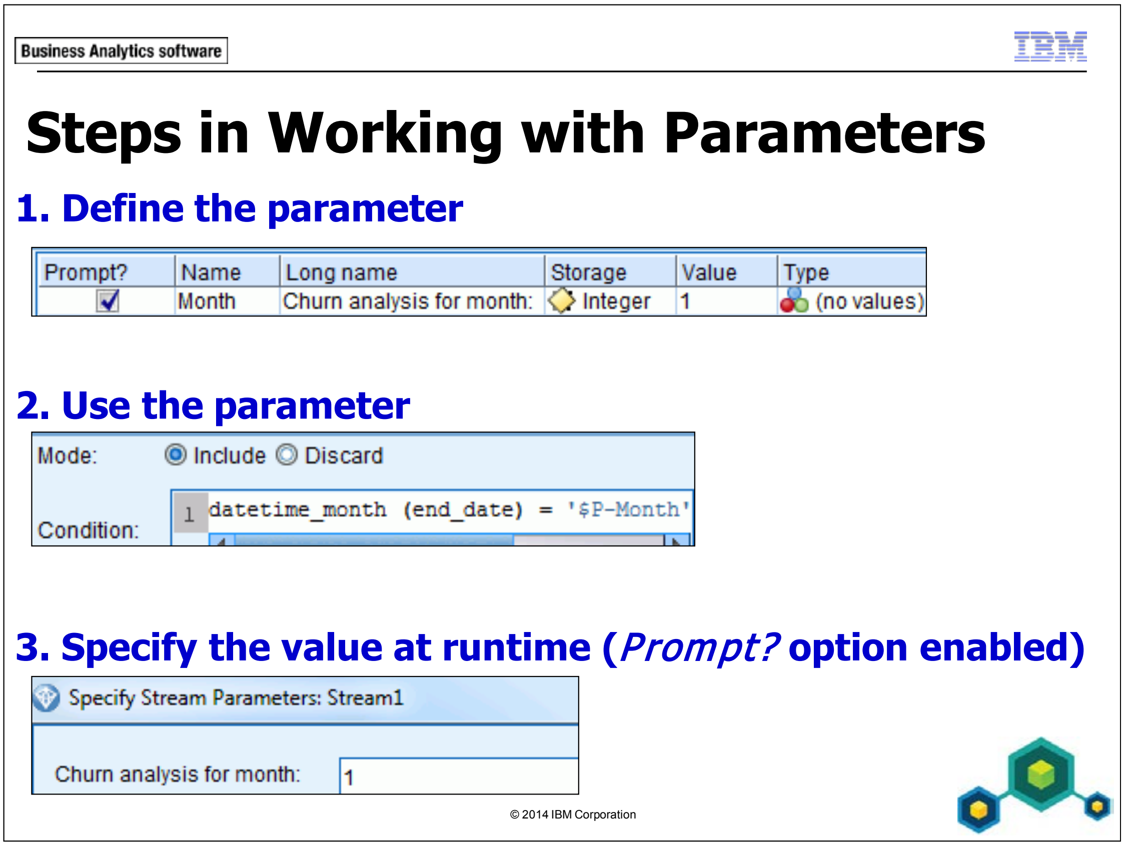
- Pass values at runtime, rather than hard coding them.
- The scope of a parameter can be:
  - a SuperNode
  - a stream
  - a session

© 2014 IBM Corporation

There are several situations where it is not efficient to enter the values themselves in CLEM expressions. Suppose, for example, that you have a stream in place that runs analyses for a certain group, with the name of the group hard coded in a Select node. When you want to run the analyses for another group, you have to edit the Select node and replace the name with the new name. It is more efficient that you will be prompted for the name of the group at run time.

Another situation is when you have several nodes using the same CLEM expression, with the same value(s) hard coded in it. If you need to replace these values you not only need to edit several nodes value(s), but you have to be very careful in replacing them all and not to forget one or more. Here, it is more efficient to use a stand-in name for the value which you have to replace only once, if desired on a prompt at run time.

In these situations you can use parameters that substitute hard coded values. You can specify parameters for SuperNodes, streams, and sessions. SuperNode parameters have the smallest scope and can only be used within a specific SuperNode, while session parameters have the largest scope and are available to all streams in the session.



Business Analytics software

Steps in Working with Parameters

1. Define the parameter

Prompt?	Name	Long name	Storage	Value	Type
☑	Month	Churn analysis for month:	Integer	1	(no values)

2. Use the parameter

Mode: ● Include ○ Discard

Condition:

```
datetime_month (end_date) = '$P-Month'
```

3. Specify the value at runtime (*Prompt?* option enabled)

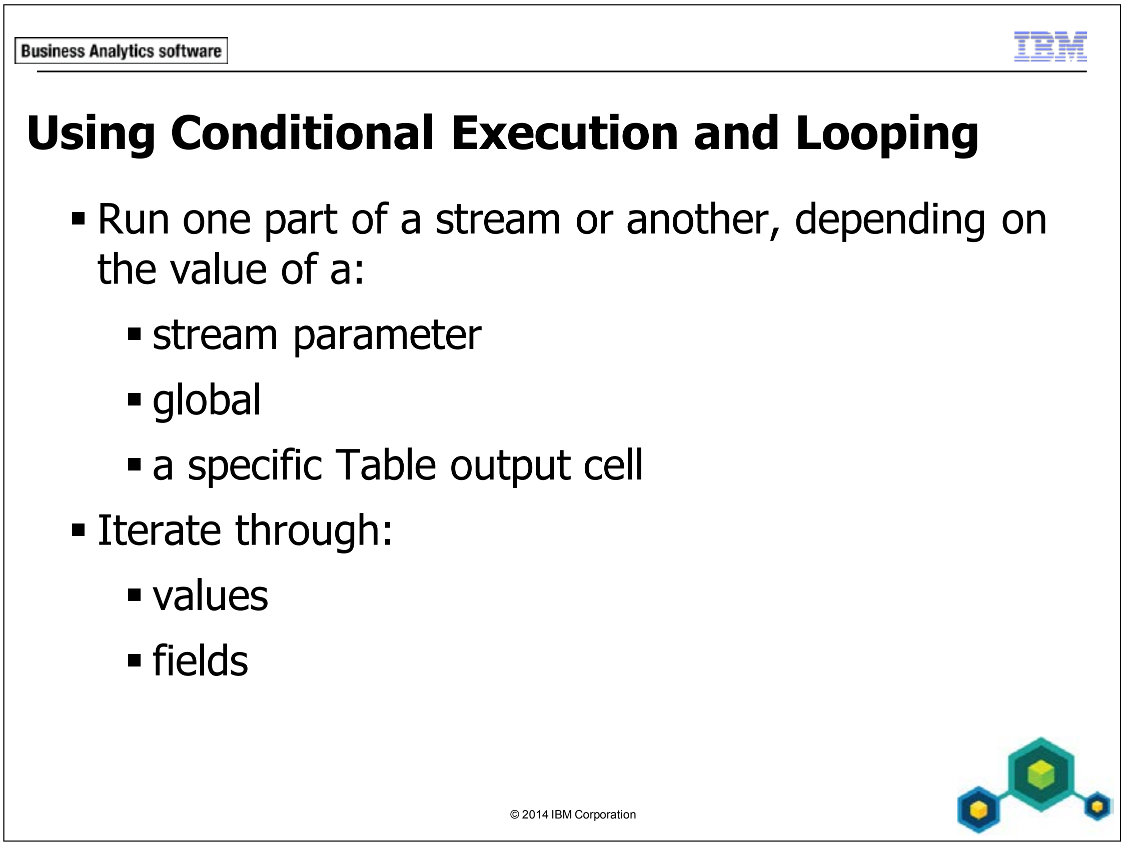
Specify Stream Parameters: Stream1

Churn analysis for month: 1

SuperNode parameters can be accessed in the SuperNode dialog box by clicking the Define Parameter button. Stream and session parameters are defined by selecting Tools\Stream Properties\Parameters and Tools\Set Session Parameters in MODELER's main menu, respectively. Whether it is a SuperNode, stream or session parameter, the way that a parameter is defined is the same in all cases:

- Enable or disable a prompt: When this option is enabled, a prompt will be presented to the user when the stream execution is run.
- Name: The parameter will be represented in CLEM expressions by $P-<pname>, where <pname> is the name specified here.
- Long name: When the Prompt option is enabled, the text specified here will be presented to the user in the prompt.
- Storage, Value, Type: Specify the storage, the default value, and measurement level.

When you have defined parameters, the next step is to replace the hard coded values with the parameter name. Finally, run the stream and specify the value at run time (when the Prompt option is enabled).



Business Analytics software

IBM

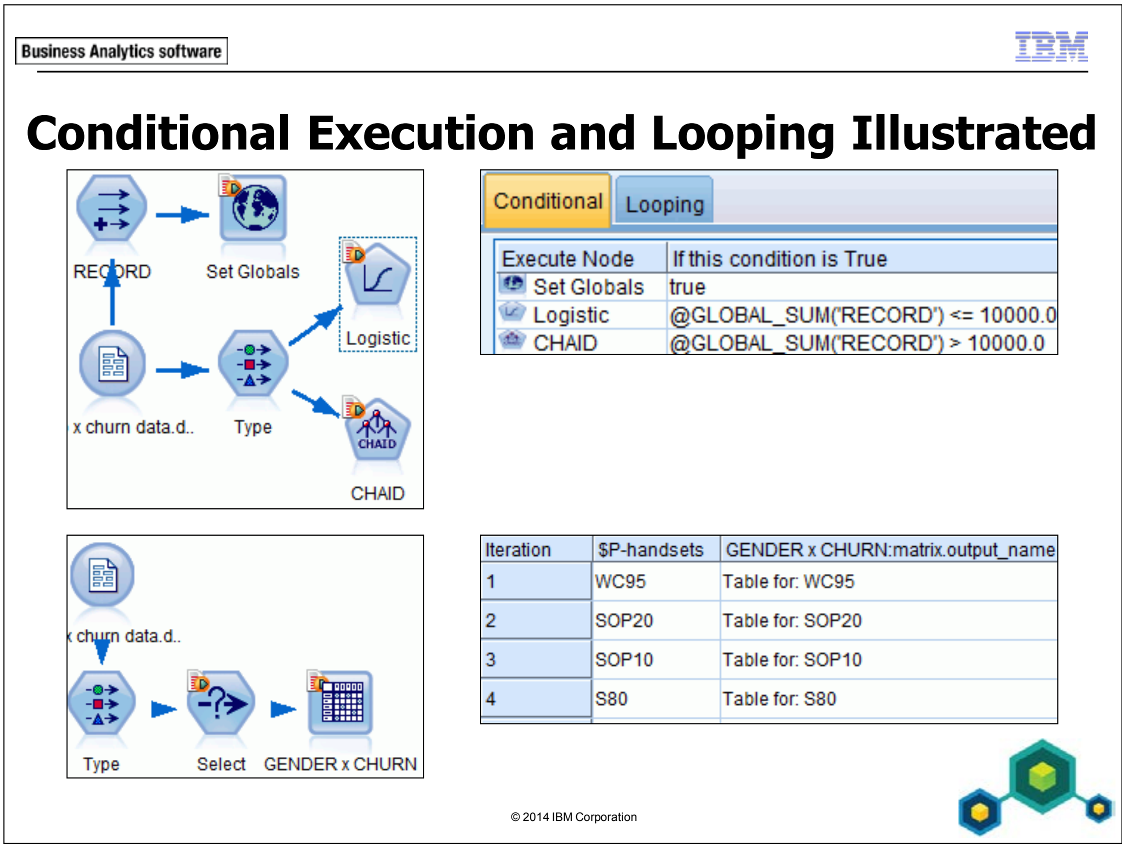
Using Conditional Execution and Looping

- Run one part of a stream or another, depending on the value of a:
  - stream parameter
  - global
  - a specific Table output cell
- Iterate through:
  - values
  - fields

© 2014 IBM Corporation

Suppose that you want to execute one branch of your stream when a certain condition is met and another branch when the condition is not met. MODELER provides you with the Conditional Execution feature, where you can base the condition on a stream parameter, a global, or a Table output cell.

MODELER also offers the option to automate stream execution by looping through values or fields.



Conditional Execution and Looping Illustrated

Conditional | Looping

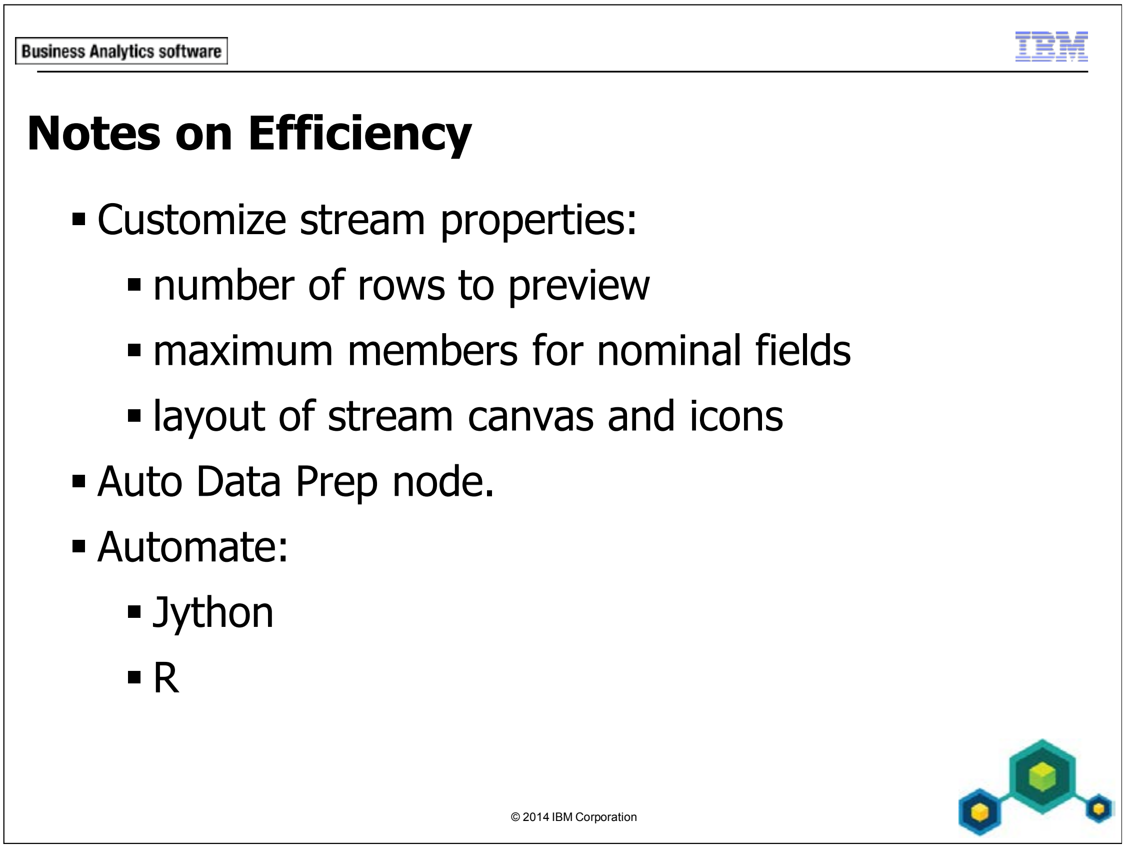
Execute Node	If this condition is True
Set Globals	true
Logistic	@GLOBAL_SUM('RECORD') <= 10000.0
CHAID	@GLOBAL_SUM('RECORD') > 10000.0

Iteration	$P-handsets	GENDER x CHURN:matrix.output_name
1	WC95	Table for: WC95
2	SOP20	Table for: SOP20
3	SOP10	Table for: SOP10
4	S80	Table for: S80

© 2014 IBM Corporation

An application of conditional execution is when you want to run one model or the other. Suppose, for example, that you want to run a Logistic model when the number of records is small and a CHAID model when the number of records is large. You can build a stream that stores the number of records in a global (using the Set Globals node), and depending on that number you run either a Logistics node or a CHAID node.

As an example of looping, suppose that you want to cross tabulate gender by churn in a Matrix node for each type of handset, which you select in a Select node. When there are, say, 12 handsets this means that you have to edit the Select node and run the Matrix node 12 times. It is more efficiently to build a loop through the handsets, each time selecting another handset and running the Matrix node.



Business Analytics software

Notes on Efficiency

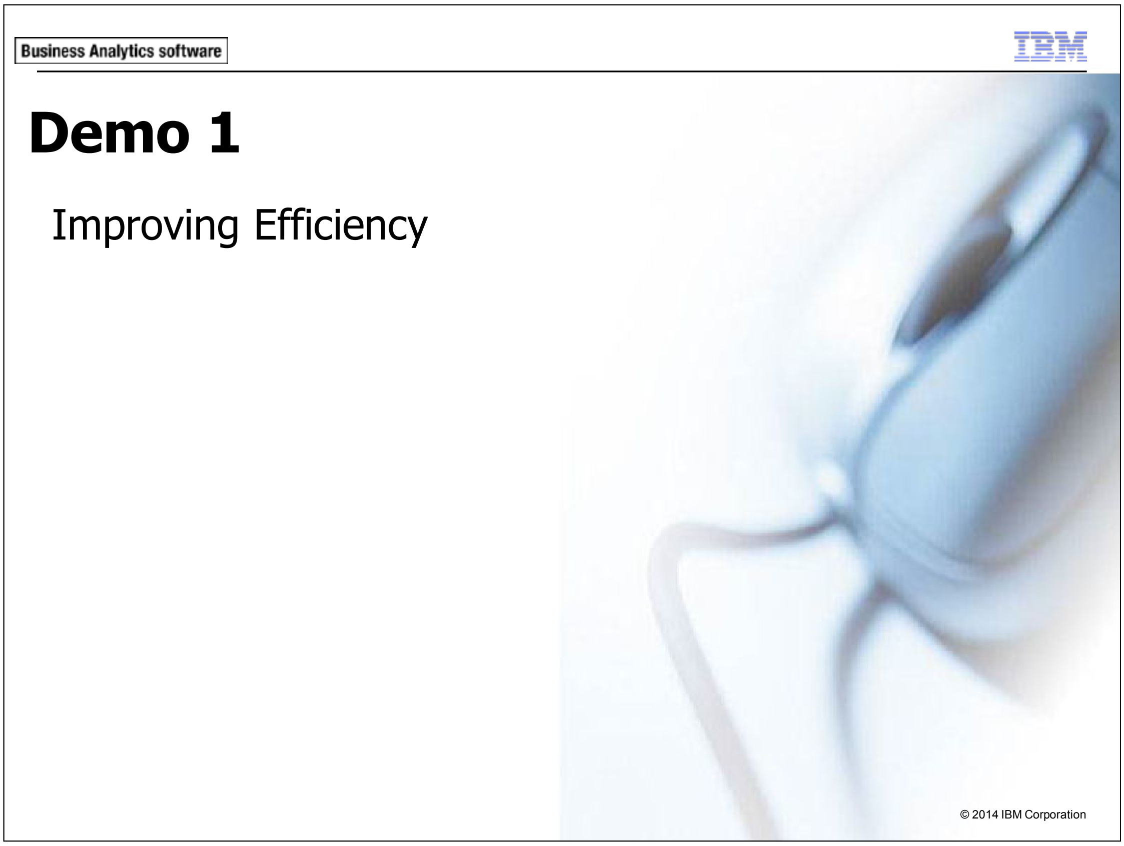
- Customize stream properties:
  - number of rows to preview
  - maximum members for nominal fields
  - layout of stream canvas and icons
- Auto Data Prep node.
- Automate:
  - Jython
  - R

There are more ways to optimize efficiency. The Tools\Stream Properties\Options dialog box enables you to fine tune:

- The number of rows that are displayed when you click the Preview button.
- The Maximum members for nominal fields. This option will automatically set a field's measurement level to typeless when the number of categories exceeds the value specified here. Checking this option will prevent streams from containing long lists of categories to be saved in memory, which speeds loading, saving and processing. Also, it is recommended to set the number to a smaller value, typically in the range from 50 to 100, when you want to use nominal field in modeling.

The Auto Data Prep node screens out fields that are problematic or not likely to be useful, derives new attributes when appropriate, and improves performance through various screening techniques. Refer to the Automated Data Mining with IBM SPSS Modeler course for more information.

The next step in efficiency would be to automate tasks using Jython and/or R. Jython is an implementation of Python, written in the Java language and running in any environment that supports a Java virtual machine (JVM). To complement MODELER, the R nodes enable expert R users to input their own R script to carry out data processing, model building and model scoring. Refer to online Help for more information on Jython and R.



Business Analytics software

Demo 1

Improving Efficiency

© 2014 IBM Corporation

The following files are used to demonstrate how you can work efficiently with MODELER:

- **telco x churn data.dat**: A (synthetic) text file storing data on customers of a (fictitious) telecommunications firm. The file is located in **C:\Train\0A055**.
- **demo_improving_efficiency.str**: A MODELER stream file that imports the dataset, sets measurement levels, declares blanks, instantiates the data, and serves as the starting point for the demo. The file is located in the **05-Improving_Efficiency\Start** sub folder.

Before you begin with the demo, ensure that:

- You have started MODELER.
- You have set MODELER's working folder. (In MODELER, select **File\Set Directory**. When you are working in an IBM operated environment, browse to the folder **C:\Train\0A055** and then click **Set** to select it. When you are not working in an IBM operated environment, ask your trainer for the folder name.)
- You have opened **demo_improving_efficiency.str**.

Demo 1: Improving Efficiency

Purpose:
You work for a telecommunications firm and you will check your data for outliers and extremes, and take appropriate action. Also, you will compute standardized scores so that you can use your own cut-off values for outliers and extremes. Finally, you want to explore the relationship between GENDER and CHURN for each region, which you will automate by using parameters and looping.

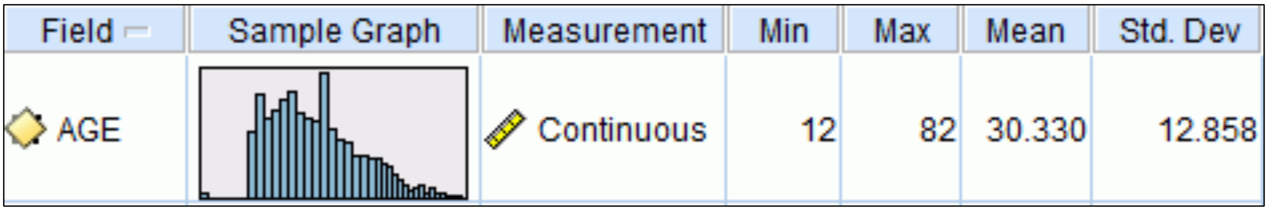
Task 1. Use the Data Audit node to process outliers, extremes and missing values.

1. Scroll to the branch where the **comment** reads **Outliers and missing values**.

   Note: Ensure that you have opened **demo_improving_efficiency.str**.

2. Add a **Data Audit** node downstream from the **Type** node, and then run the **Data Audit** node.

   A section of the results appear as follows:



Field	Sample Graph	Measurement	Min	Max	Mean	Std. Dev
AGE		Continuous	12	82	30.330	12.858

The mean for the AGE field is 30.330, with standard deviation 12.858. Applying the rule that values less than mean - 3 * standard deviation and values greater than mean - 3 * standard deviation are outliers, there are outliers in the right tail of the distribution (values above 30.330 + 3 * 12.858 ~= 69).

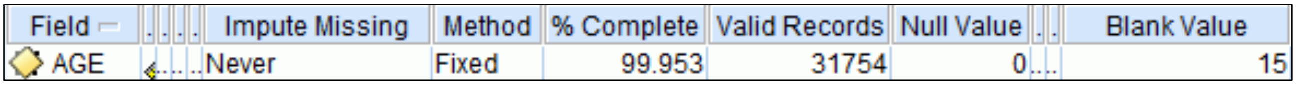
The question is: do you want to consider persons older than of 69 as outliers? An age higher than 69 is certainly not a-typical, or odd. Deciding which action to take depends on the business question. Maybe the group of retired people is not targeted for any campaign. So from this perspective retired people should be removed from the dataset, to begin with. Or, suppose the company has special customer loyalty programs in place for the group of 65+ people. In that scenario, it is useful to monitor the behavior of this group.

This material is meant for IBM Academic Initiative purposes.

All in all, outliers will not be replaced for AGE. Next, you will examine missing values for this field.

3. Click the **Quality** tab in the **Data Audit** output window, and then scroll to the last columns in the **Data Audit** output window.

   A section of the results appear as follows:



Field	.	.	.	Impute Missing	Method	% Complete	Valid Records	Null Value	.	.	Blank Value
AGE	.	.	.	Never	Fixed	99.953	31754	0	.	.	15

   The AGE field has 15 blank values. You will replace these blank values, using the C&RT algorithm.

   Note: A discussion of C&RT is beyond the scope of this course. Please refer to the *Classifying Customers Using IBM SPSS Modeler (v16)* course for more information.

4. Click on the **AGE** field name, and then:

   - for **Impute Missing**, select **Blank Values**
   - for **Method** select **Algorithm**, select the **Age** field name again. (losses focus)
   - from the main menu, select **Generate\Missing Values SuperNode**
   - in the **Missing Values SuperNode** dialog box that pops up, select **Selected fields only** and then close the **Missing Values SuperNode** dialog box

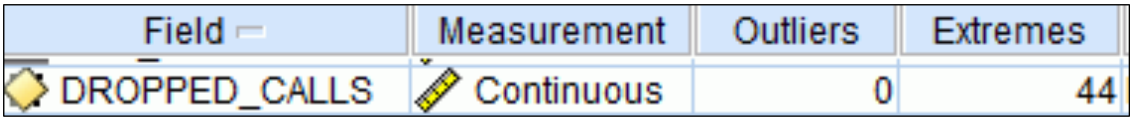
   A SuperNode named Missing Value Imputation has been generated and is placed in the upper left corner on the stream canvas. You will add this SuperNode to the stream later in this task.

   You will examine outliers and extremes for DROPPED_CALLS. The mean for this field was 3.190, the standard deviation is 4.195. Thus, outliers are values above 3.190 + 3 * 4.195= 15.775, extremes are values above 3.190 + 5 * 4.195 = 24.165. Given the maximum value (45) there is at least one extreme value. The Quality tab gives you the full details on outliers and extremes.

This material is meant for IBM Academic Initiative purposes.

5. Run the **Data Audit** node again, and click the **Quality** tab.
6. Scroll to **DROPPED_CALLS** in the **Data Audit** output window, **Quality** tab.

   A section of the results appear as follows:



Field	Measurement	Outliers	Extremes
DROPPED_CALLS	Continuous	0	44

   There are no outliers, but 44 extremes. You will coerce these values to the upper limit, which is the mean + 3 * standard deviation.

7. Click on the **DROPPED_CALLS** field name, and then:
   - for **Action**, select **Coerce,** select the **DROPPED_CALLS** field name again. (losses focus)
   - from the main menu, select **Generate\Outlier & Extreme SuperNode**
   - in the **Outlier SuperNode** dialog box that displays, select **Selected fields only**
   - click **OK** to close the **Outlier SuperNode** dialog box
   - close the **Data Audit** output window

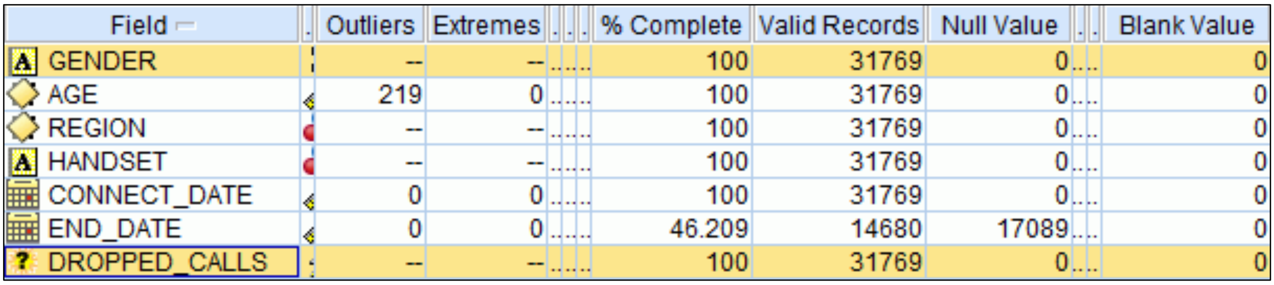
   A SuperNode named Outlier and Extreme has been generated and has been placed in the upper left corner on the stream canvas.

   You will add the two generated SuperNodes to the stream and rerun the Data Audit node to examine how replacing blanks and coercing extremes affect the statistics.

8. Insert the **SuperNode** named **Outlier and Extreme** between the **Type** node and the **Data Audit** node.
9. Insert the **SuperNode** named **Missing Value Imputation** between the **Outlier and Extreme** super node and the **Data Audit** node.
10. Run the **Data Audit** node.

11. Click the **Quality** tab in the **Data Audit** output window.

    A section of the results appear as follows:



Field	.	Outliers	Extremes	.	.	% Complete	Valid Records	Null Value	.	Blank Value
A GENDER		--	--	.	.	100	31769	0	.	0
AGE		219	0	.	.	100	31769	0	.	0
REGION		--	--	.	.	100	31769	0	.	0
A HANDSET		--	--	.	.	100	31769	0	.	0
CONNECT_DATE		0	0	.	.	100	31769	0	.	0
END_DATE		0	0	.	.	46.209	14680	17089	.	0
? DROPPED_CALLS		--	--	.	.	100	31769	0	.	0

    The AGE field has no longer blanks, which is the effect of replacing the values in the Missing Value SuperNode.

    The number of outliers and extremes is not reported for DROPPED_CALLS, because the field's storage has to be re-instantiated.

12. Close the **Data Audit** output window.

    Leave the stream open for the next task.

Task 2. Compute standardized scores using globals.

To have more control over outliers and extremes you will compute the number of standard deviations that a value is above or below the mean, known as standardized scores or z-scores. Having a field with standardized scores enables you to select outliers and/or extremes with any cut-off value that you want, or to sort records on the standardized scores, and so forth.

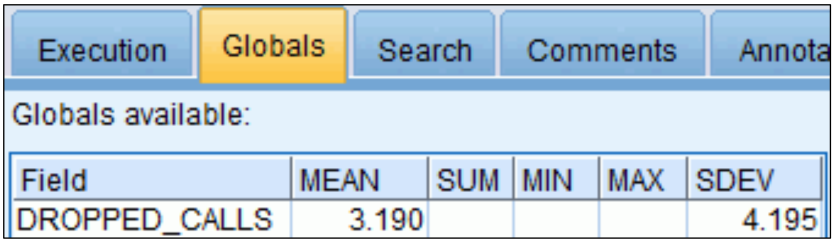
A standardized score is computed as (value - mean) / standard deviation. You will use the Set Globals node to obtain the mean and standard deviation, and then compute standardized scores.

You will build from the stream that you created in the previous task.

1. Scroll to the branch where the **comment** reads **Using globals**.
2. Add the **Set Globals** node (Output palette) downstream from the **Type** node.

3. Edit the **Set Globals** node, and then:
   - for **Field**, select **DROPPED_CALLS**
   - enable the **Mean** and **SDev** option, and disable the other options
   - enable the **Display preview of globals created after execution** option (you can then examine the statistics immediately when the Set Globals node is executed)
   - click **Run**

   A section of the results appear as follows:



Globals available:

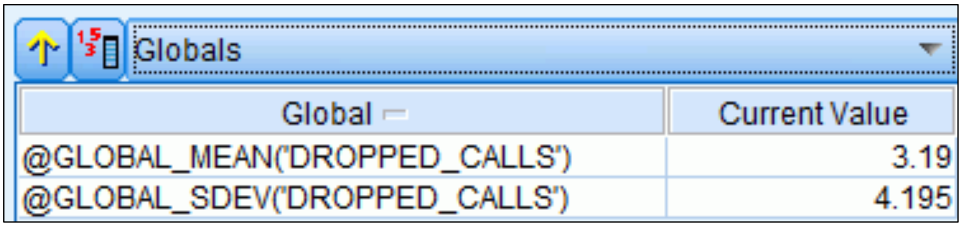
Field	MEAN	SUM	MIN	MAX	SDEV
DROPPED_CALLS	3.190				4.195

   The mean and standard deviation are in agreement with the output from the Data Audit node for this field in the previous task.

   You will now use the mean and standard deviation in a Derive node to compute standardized scores, or z-scores, for DROPPED_CALLS.

4. Close the **Set Globals** window.
5. Add a **Derive** node downstream for the **Type** node.
6. Edit the **Derive** node, and then:
   - for **Derive field**, type **Z_SCORES_DROPPED_CALLS**
   - click the **Launch expression builder** button
   - in the **Field category** drop down list, select **Globals**

   A section of the results appear as follows:



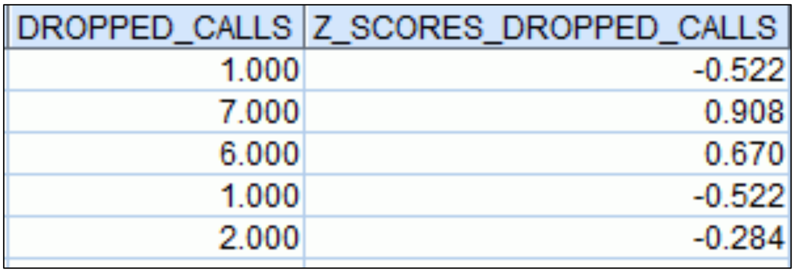
Global	Current Value
@GLOBAL_MEAN('DROPPED_CALLS')	3.19
@GLOBAL_SDEV('DROPPED_CALLS')	4.195

   The Globals field is populated with the values.

7. Create the expression **(DROPPED_CALLS - @GLOBAL_MEAN ('DROPPED_CALLS')) / @GLOBAL_SDEV('DROPPED_CALLS')**

8. Close the **Expression** builder.
9. Click **Preview**, and then move **Z_SCORES_DROPPED_CALLS** next to **DROPPED_CALLS** in the **Preview** output window.

   A section of the results appear as follows:



DROPPED_CALLS	Z_SCORES_DROPPED_CALLS
1.000	-0.522
7.000	0.908
6.000	0.670
1.000	-0.522
2.000	-0.284

   You could now select or discard the records using your own cut-off value. For example, if you want to select records that have a value for DROPPED_CALLS within 4 standard deviations from the mean, the expression would be: abs (Z_SCORES_DROPPED_CALLS)< 4.

   Note: the abs function takes the absolute value.

   Leave the stream open for the next task.

Task 3. Using parameters.

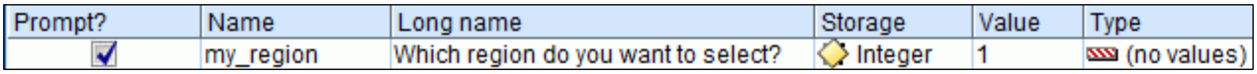
In this part you will replace a hard coded selection of records with a parameterized selection.

You will build from the stream that you created in the previous task.

1. Scroll to the branch where the **comment** reads **Using parameters**.

   The Select node selects records from a certain region (region 1 at this moment), and a Matrix node is run downstream from the Select node. Rather than hard coding the region, you want to be prompted for the region that you want to run the Matrix node for. You will accomplish this by defining a parameter.
2. Select **Tools\Stream Properties\Parameters** from the main menu, and then:
   - enable the **Prompt?** option
   - for **Name**, type **my_region**
   - for **Long name**, type **Which region do you want to select?**
   - for **storage**, select **Integer** (since REGION is integer valued, the parameter values will be integers also)
   - for **Value**, type **1** (this is the default value)

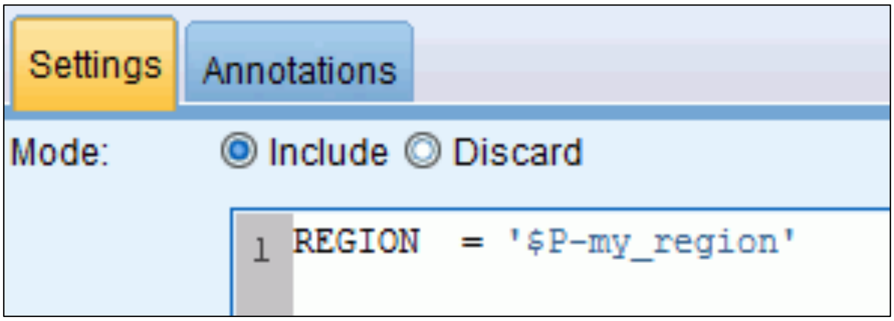
A section of the results appear as follows:



Prompt?	Name	Long name	Storage	Value	Type
☑	my_region	Which region do you want to select?	Integer	1	(no values)

3. Close the **Stream Properties** dialog box.
4. Edit the **select region** node, and then:
   - delete the value **1** in the condition
   - click the **Expression Builder** button, select the **Parameters** category, select **'$P-my_region'** from there, and then return to the main dialog box

A section of the results appear as follows:

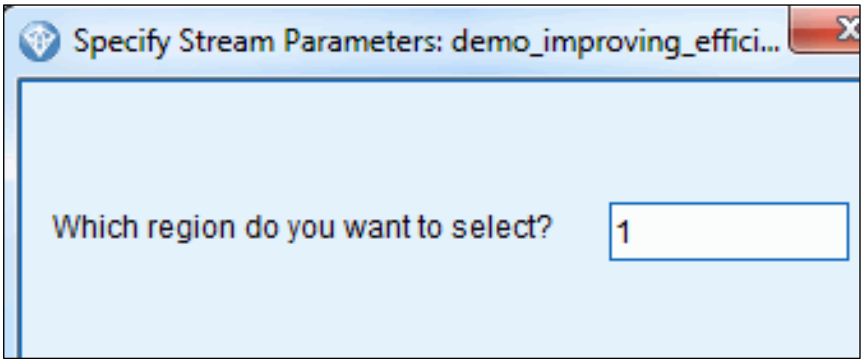


Mode: ● Include ○ Discard

```
REGION = '$P-my_region'
```

5. Close the **Select** dialog box.
6. Run the **Matrix** node.

A section of the results appear as follows:



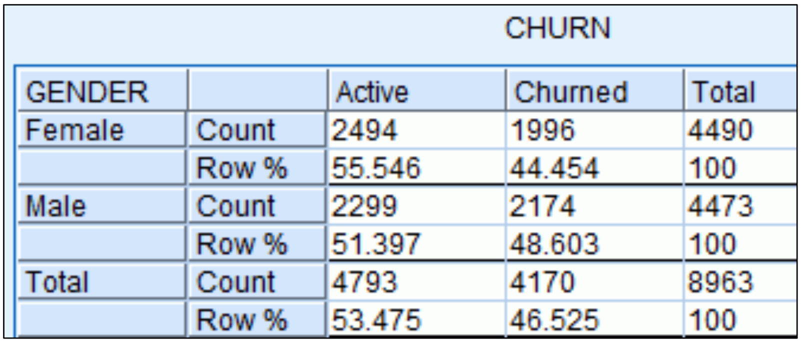
Specify Stream Parameters: demo_improving_effici...

Which region do you want to select? 1

You will run the analysis for region 4.

7. Replace **1** with **4**, and then click **OK**.

   A section of the results appear as follows:



		CHURN		
GENDER		Active	Churned	Total
Female	Count	2494	1996	4490
	Row %	55.546	44.454	100
Male	Count	2299	2174	4473
	Row %	51.397	48.603	100
Total	Count	4793	4170	8963
	Row %	53.475	46.525	100

   These are the results for region 4.

8. Close the **Matrix** output window.

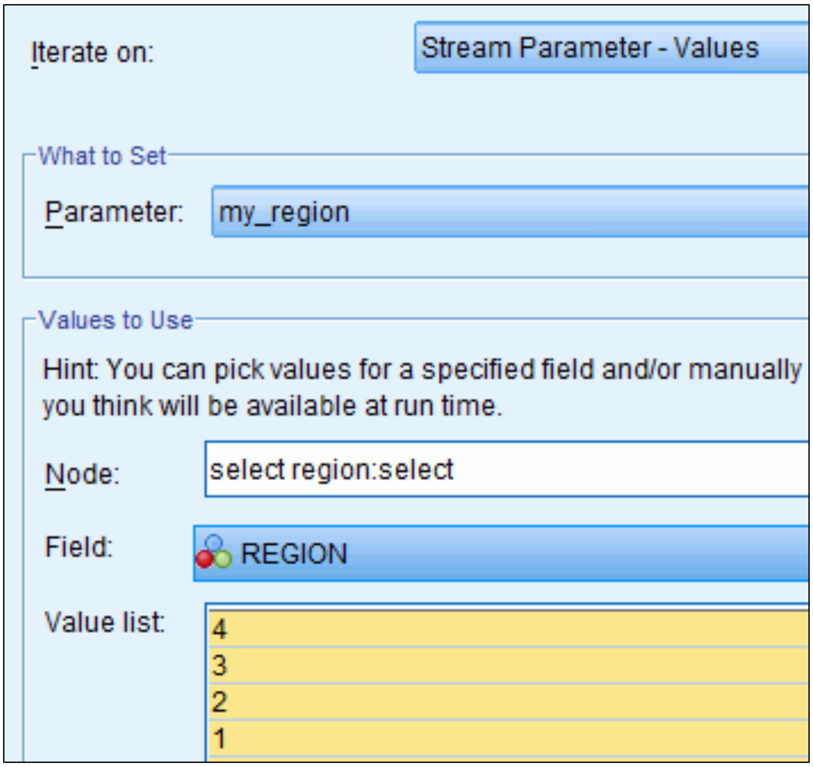
Task 4. Creating a loop through values.

In the previous task you have created a parameter to enable you to select the region at the prompt. In this task you will loop through the regions, and run the Matrix node for each region. In order to loop through the values of REGION you must have a parameter in place, that serves as the placeholder for the values. You already have a parameter in place, as per the previous task. In this case, however, you do not want to have a prompt, but just loop through the values.

You will build from the stream that you created in the previous task.

1. Select **Tools\Stream Properties\Parameters** from the main menu, disable the **Prompt?** option, and then close the **Stream Properties** window.
2. Right-click the s**elect region** node, choose **Looping/Conditional Execution** from the context menu, select **Define Iteration Key (Values)**, and then:
   - for **Iterate on**, select **Stream Parameter - Values**
   - for **Parameter**, select **my_region**
   - for **Node**, choose the **select region** node
   - for field, select **REGION**
   - for **Values**, click the **Specify values** button, and select values **1, 2, 3** and **4**

A section of the results appear as follows:



Iterate on: Stream Parameter - Values

What to Set

Parameter: my_region

Values to Use

Hint: You can pick values for a specified field and/or manually
you think will be available at run time.

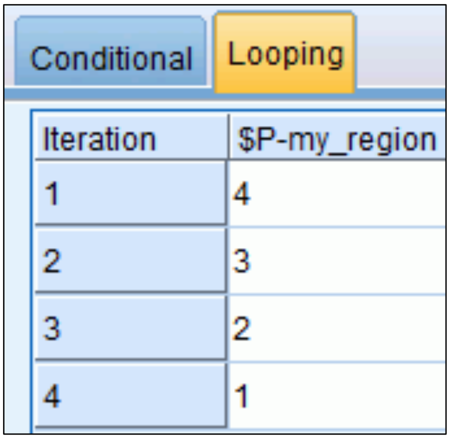
Node: select region:select

Field: REGION

Value list:
4
3
2
1

3. Close the **Define Iteration Key** dialog box.

A section of the results appear as follows:



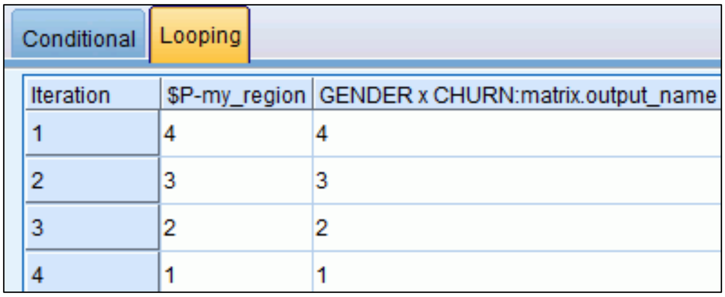
Iteration	$P-my_region
1	4
2	3
3	2
4	1

A loop has been created which iterates through the values 1 to 4 of the parameter $P-my_region.

Next to defining the iteration key you will generate a Matrix table for each value of the loop. Thus, you will generate 4 Matrix tables. To accomplish this you will assign a unique Matrix output name to each table, reflecting the region selected (if you do not use a unique output name, the Matrix table that is produced will be overwritten in the next iteration).

4. Click the **Add Variable** button, and then:
   - for **Change**, select **Node Property**
   - for **Node**, select **GENDER x CHURN**
   - for **Property**, select **output_name**
   - click **OK** to close the **Add Iteration Variable** dialog box

   A section of the results appear as follows:



Iteration	$P-my_region	GENDER x CHURN:matrix.output_name
1	4	4
2	3	3
3	2	2
4	1	1

5. Click **OK** to close the dialog box.

   The Select node and Matrix node show a little icon indicating that looping has been set on these nodes.

6. Click the **Run the current stream** ▶ button in the main menu.

   The Outputs tab will contain the four tables (you will have some additional output, because all terminal nodes are run).

This completes the demo for this module. You will find the solution results in **demo_improving_efficiency_completed.str**, located in the **05-Improving_Efficience\Solutions** sub folder.

Results:
You have checked your data for outliers and extremes, and you have taken appropriate action. You also have computed standardized scores so that you could use your own cut-off values to process outliers and extremes. Finally, you have used parameters and looping to automate tasks.

Apply Your Knowledge

Use the questions in this section to test your knowledge of the course material.

Question 1: Which of the following statements are correct?

A. The standard deviation cannot be computed for a string field.

B. If all records have the same value for the (continuous) field X, the standard deviation for X equals 0.

C. The standard deviation for a field X will not change if we add 5 to each score of X.

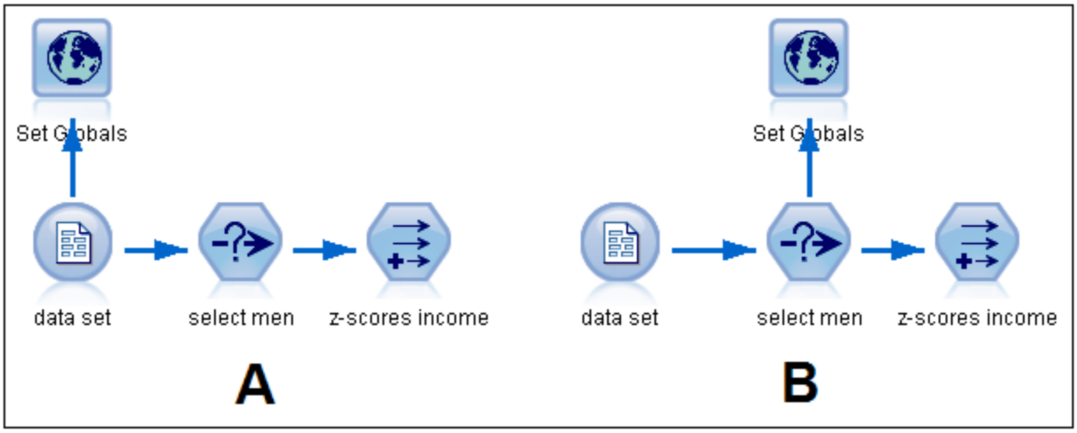
D. If two fields X and Y have the same unit of measurement (say, inches), and X has a standard deviation of 10, and Y of 100, the histogram for Y will show more spread than the histogram for X.

Question 2: Is the following statement true or false? The median is one of the global statistics that can be requested.

A. True

B. False

Question 3: See the figures A and B below. Needed are standardized scores for income, for men only. Thus, mean income and standard deviation for income should be computed only for men. Stream ______ (A/B) accomplishes this.



This material is meant for IBM Academic Initiative purposes.

Question 4: Which of the following is the correct statement? SQL pushback is relevant for:

A. Text files

B. Databases

C. IBM SPSS Statistics (*.sav) files

D. XML files

Question 5: Is the following statement true or false? The operations that can be pushed back to the database depend on the database in use.

A. True

B. False

Question 6: Is the following statement true or false? A parameter that you define for a particular SuperNode will automatically be available for all SuperNodes in your session.

A. True

B. False

Question 7: Suppose that you have a field named HANDSET in your dataset with 24 categories and that you want to export your data to 24 data files, one data file for each handset. Which automation method will you use?

A. Conditional execution

B. Looping

C. Auto Cluster

Question 8: Is the following statement true or false? You can run a Distribution graph for each of your (categorical) fields in your dataset with a single Distribution node by setting a loop on the node.

A. True

B. False

Answers to questions:

Answer 1: A, B, C, D. Computing the standard deviation is only relevant for numeric fields. When all the scores are the same, there is no spread and the standard deviation will be 0. The spread will not change when you add 5 to the scores, so the standard deviation will be the same. When two fields X and Y have the same unit of measurement, the field with the larger standard deviation will show more spread.

Answer 2: B. False. The median cannot be requested as a global variable. If you want the median you can use an Aggregate node and merge the aggregated dataset and the original dataset.

Answer 3: B. The mean needs to be computed for men only, so downstream from the node where men are selected.

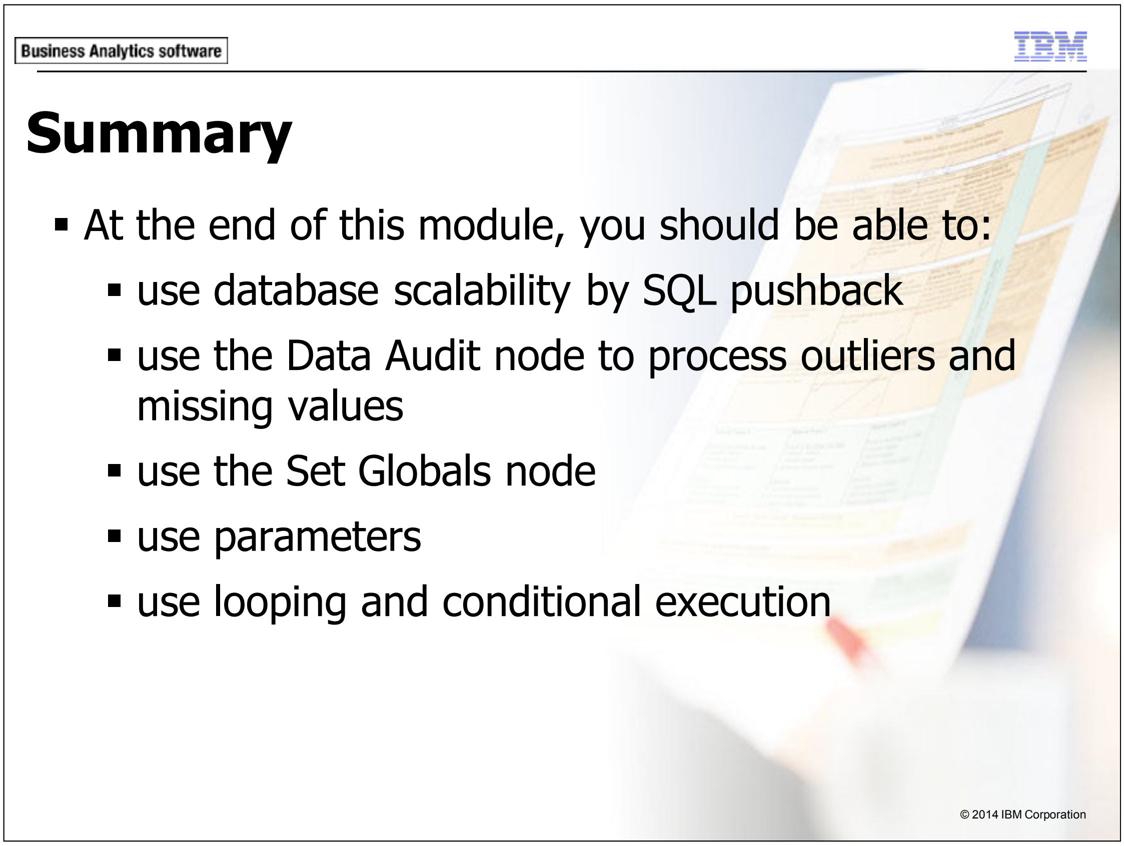
Answer 4: B. SQL pushback is only relevant when you import data from a database.

Answer 5: A. True. It is possible that an operation can be pushed back to database A, but not to database B.

Answer 6: B. False. A SuperNode parameter is available only in the SuperNode where you defined the parameter.

Answer 7: B. You will loop through the categories of HANDSET.

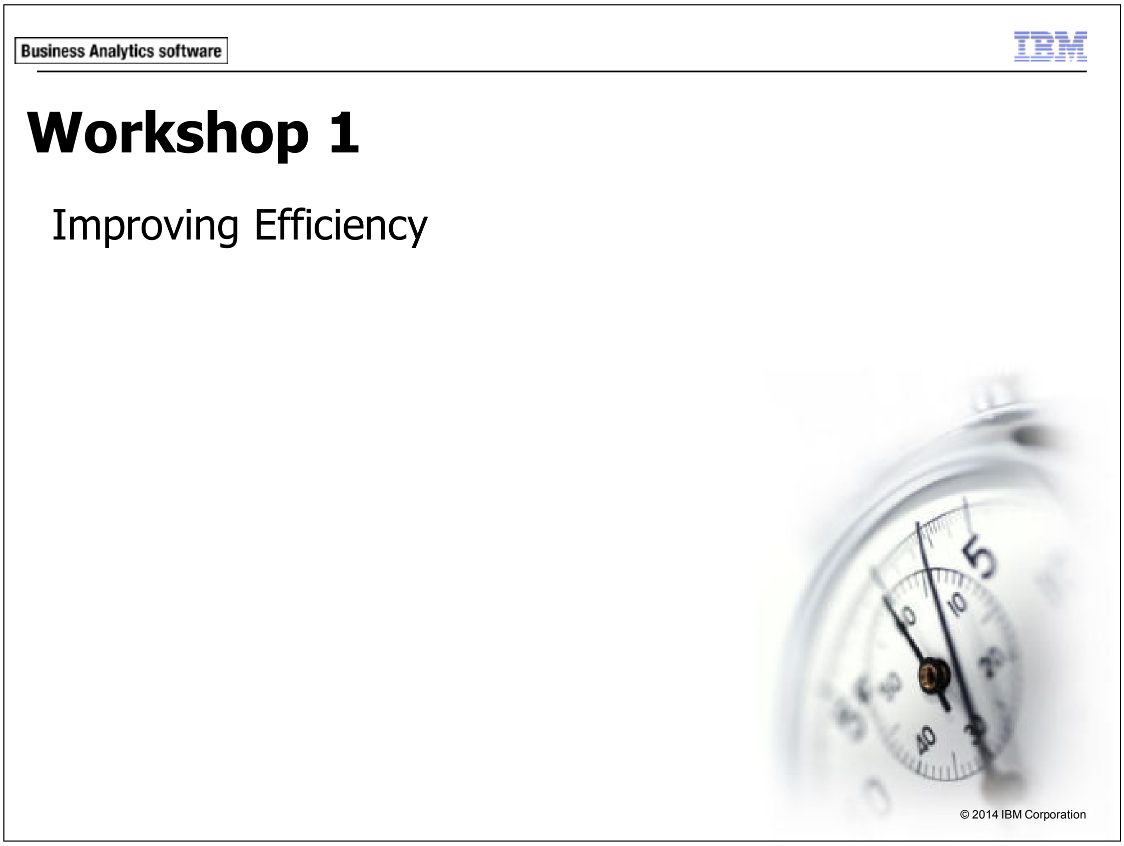
Answer 8: A. True. You can build a loop to go through fields in a Distribution node.



Business Analytics software

Summary

- At the end of this module, you should be able to:
  - use database scalability by SQL pushback
  - use the Data Audit node to process outliers and missing values
  - use the Set Globals node
  - use parameters
  - use looping and conditional execution



Business Analytics software

Workshop 1

Improving Efficiency

The following (synthetic) file is used in this workshop:

- **customers_and_holidays.dat**: A text file containing (synthetic) data on 411 customers of a travel agency. The file is located in **C:\Train\0A055**.

Before you begin with the workshop, ensure that:

- You have started MODELER.
- You have set MODELER's working folder. (In MODELER, select **File\Set Directory**. When you are working in an IBM operated environment, browse to the folder **C:\Train\0A055** and then click **Set** to select it. When you are not working in an IBM operated environment, ask your trainer for the folder name.)

Workshop 1: Improving Efficiency

In this learning activity you will process outliers, extremes and missing values, using the Data Audit node. You will use the Set Globals node to replace missing values, and you will be introduced to automation by using parameters and looping.

- Import the data from **customers_and_holidays.dat** (a text file), add a **Type** node downstream from the source node, declare **-999** as blank for **DISTANCE_TO_BEACH**, and instantiate the data.

- Run a **Data Audit** node downstream from the **Type** node.

  What is the minimum, maximum, mean and standard deviation for **DISTANCE_TO_BEACH**? Do you expect outliers or extremes?

  Check your expectations by examining the number of outliers and extremes for **DISTANCE_TO_BEACH** on the **Quality** tab of the **Data Audit** output.

- Coerce outliers and nullify extremes for **DISTANCE_TO_BEACH**, instantiate the data (since the storage type for **DISTANCE_TO_BEACH** changed after filling this field), and then run a **Data Audit** node to examine the effect of coercing outliers and nullifying extremes.

  Are there still outliers and extremes after you coerced outliers and nullified extremes?

- Replace undefined ($null$) values for **DISTANCE_TO_BEACH** with the **mean DISTANCE_TO_BEACH**, by using the **Set Globals** node (amongst others).

  Note: This can also be accomplished by generating a Missing Value SuperNode from the Quality tab in the Data Audit output. However, there is a subtle difference because the Missing value SuperNode will have the mean hard coded, while the global stores a value which can updated easily (by re-executing the Set Globals node).

  In the following tasks you will use parameters and looping. At this point, however, the data need some cleaning.

- Add a **Filler** node downstream from the **first Type** node, and always replace the values for **COUNTRY** to uppercase (use the lowertoupper function).

- Select all customers that went to **SPAIN**, and request a cross tabulation (Matrix node) of **GENDER** in the row by **SATISFIED** in the column for customers that went to Spain. Request row percentages on the **Appearance** tab in the Matrix node.

  What is the percentage of men that is satisfied? And what is the percentage of women that is satisfied?

- Define a parameter defined named **my_country**, long name **What was your country destination?**, default **SPAIN**, and use this parameter so that you are prompted for the country name when you run the **Matrix** node of **GENDER** by **SATISFIED**.

  Test whether the parameterization is working correctly.

- Create a loop so that the **Matrix** node is run for **GENDER**, **REGION**, **HOLCODE**, **POOL** and **ACCOM** by **SATISFIED**.

  Hint: create a loop on the Matrix node, with a loop thru fields, in particular a loop through the row fields in the Matrix node.

  - right-click the **Matrix** node, and select **Looping/Conditional Execution\Define Iteration Key (Fields)**
  - for **Iterate on**, ensure that **Node Property - Fields** is selected
  - for **Node**, ensure that **GENDER x SATISFIED:matrix** is selected
  - for **Property**, select **row**
  - in the area **Fields to use**, select **GENDER**, **REGION**, **HOLCODE**, **POOL**, and **ACCOM**
  - close the dialog boxes and then click the **Run the current stream** ▶ button

For more information about where to work and the workshop results, refer to the Task and Results section that follows. If you need more information to complete a task, refer to earlier demos for detailed steps.

Workshop 1: Tasks and Results

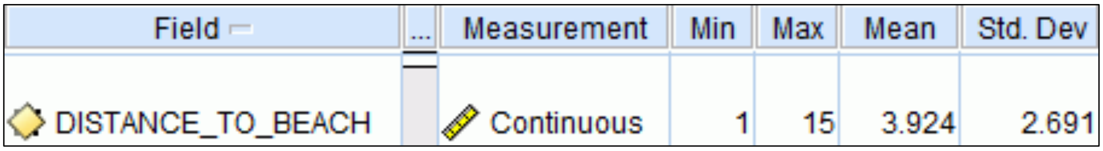
Task 1. Import and instantiate the data.

- Use a **Var.File** node to import the file **customers_and_holidays.dat**.
- Add a **Type** node downstream from the **Var. File** node, declare **-999** as blank for **DISTANCE_TO_BEACH**, and then click **Read Values** to instantiate the data.

Task 2. Examine DISTANCE_TO_BEACH.

- Add a **Data Audit** node downstream from the **Type** node, and then run the **Data Audit** node.

A section of the results appear as follows:

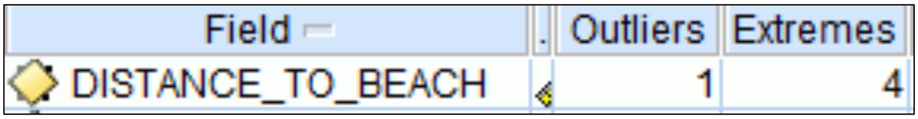


Field	...	Measurement	Min	Max	Mean	Std. Dev
DISTANCE_TO_BEACH		Continuous	1	15	3.924	2.691

The scores range from 1 to 15. The mean and standard deviation are 3.92 and 2.69, respectively. It is expected that there are no outliers or extremes in the left tail of the distribution. It is expected that there are outliers (values above 3.92 + 3 * 2.69 ~= 12) and no extremes (values above 3.92 + 5 * 2.69 ~= 17.3) in the right tail of the distribution.

- Click the **Quality** tab in the **Data Audit** output window.

A section of the results appear as follows:



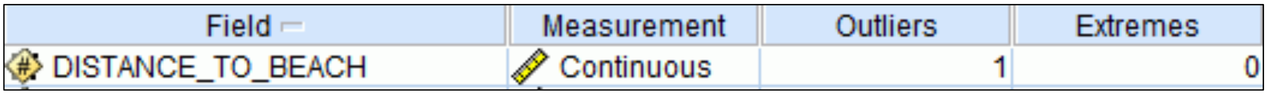
Field	.	Outliers	Extremes
DISTANCE_TO_BEACH		1	4

One outlier is reported, which must be the score of 15. Contrary to the expectations there are 4 extremes. Apparently the blank value -999, which has a frequency of 4, is reported as an extreme value.

Task 3. Processing outliers and extremes for DISTANCE_TO_BEACH.

- Set action to **Coerce outliers/nullify extremes** for **DISTANCE_TO_BEACH**.
- Ensure that **DISTANCE_TO_BEACH** is selected (click on the field name so that it is highlighted).
- Select **Generate\Outlier & Extreme SuperNode**, and then choose **For selected fields only**.
- Close the **Data Audit** output window.
- Add the **generated SuperNode** downstream from the **Type** node.
- Add a **Type** node downstream from the generated **SuperNode**, and then click **Read Values** to instantiate the data.
- Add a **Data Audit** node downstream from the second **Type** node, and then run the **Data Audit** node.
- Select the **Quality** tab.

A section of the results appear as follows:



Field	Measurement	Outliers	Extremes
DISTANCE_TO_BEACH	Continuous	1	0

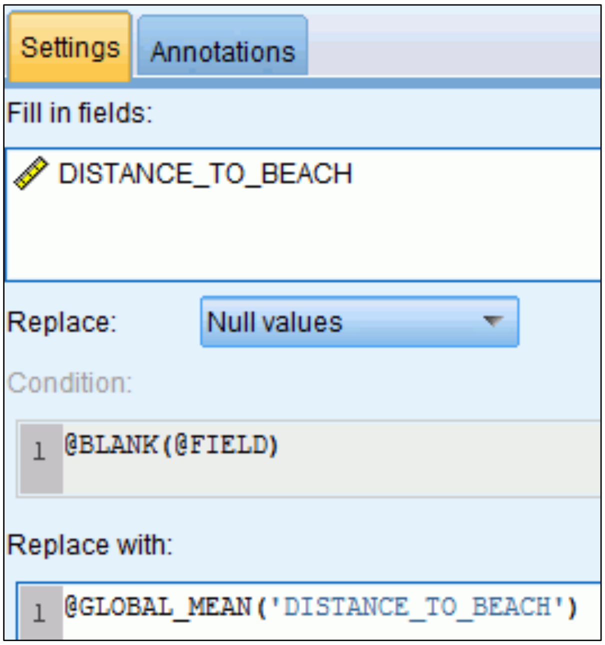
There is one outlier, which is, when you examine the Audit tab, just above mean + 3 * SD, so there is no reason to take further action.

Task 4. Use globals to replace undefined values with the mean.

- Add a **Set Globals** node downstream from the **Type** node (the **Type** node that you last added to the stream).
- Edit the **Set Globals** node, select **DISTANCE_TO_BEACH**, select **MEAN** (disable the other statistics), and then run the **Set Globals** node.
- Add a **Filler** node downstream from the **Type** node (the **Type** node that you last added to the stream).

- Edit the **Filler** node, select **DISTANCE_TO_BEACH**, set **Replace:** to **Null values**, and **Replace with** to **GLOBAL_MEAN('DISTANCE_TO_BEACH')** (use the Expression Builder to create the expression)

  A section of the results appear as follows:



Fill in fields:

DISTANCE_TO_BEACH

Replace: Null values

Condition:

```
@BLANK(@FIELD)
```

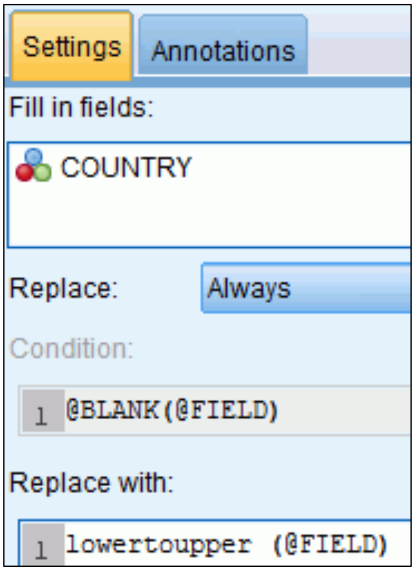
Replace with:

```
@GLOBAL_MEAN('DISTANCE_TO_BEACH')
```

Task 5. Cleanse COUNTRY.

- Add a **Filler** node downstream from the **Filler** node, select **COUNTRY**, set **Replace:** to **Always**, and for **Replace with** enter **lowertoupper (@FIELD)**.

  A section of the results appear as follows:



Fill in fields:

COUNTRY

Replace: Always

Condition:

```
1 @BLANK(@FIELD)
```

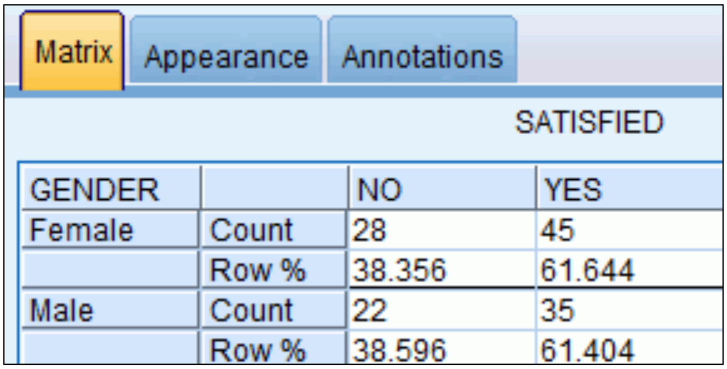
Replace with:

```
1 lowertoupper (@FIELD)
```

Task 6. Cross tabulate GENDER by SATISFIED, for customers with destination SPAIN.

- Add a **Select** node downstream from the **Filler** node (the **Filler** node that you last added to the stream).
- Edit the **Select** node, and enter the condition **COUNTRY = "SPAIN"**.
- Add a **Matrix** node downstream from the Select node, and then:
  - select **GENDER** in the row
  - select **SATISFIED** in the column
  - click the **Appearance** tab and enable the **Percentage of row** option
  - run the **Matrix** node

A section of the results appear as follows:

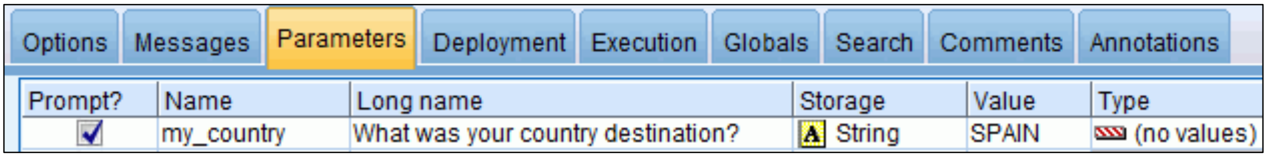


		SATISFIED	
GENDER		NO	YES
Female	Count	28	45
	Row %	38.356	61.644
Male	Count	22	35
	Row %	38.596	61.404

There is almost no difference in satisfaction between men and women who went to Spain.

Task 7. Define a parameter to select the destination.

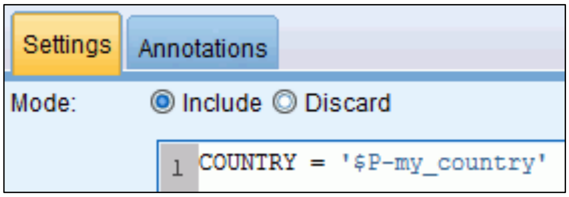
- Select **Tools\Stream Properties\Parameters** from the main menu and specify as shown in the figure below.



Prompt?	Name	Long name	Storage	Value	Type
☑	my_country	What was your country destination?	A String	SPAIN	▨ (no values)

- Edit the **Select** node, and then replace **"SPAIN"** with **'$P-my_country'**.

A section of the results appear as follows:



Mode: ● Include ○ Discard

```
COUNTRY = '$P-my_country'
```

- Run the **Matrix** node. You will be prompted for the country. Keep the country that is suggested, SPAIN (the default), and then click **OK** to continue.

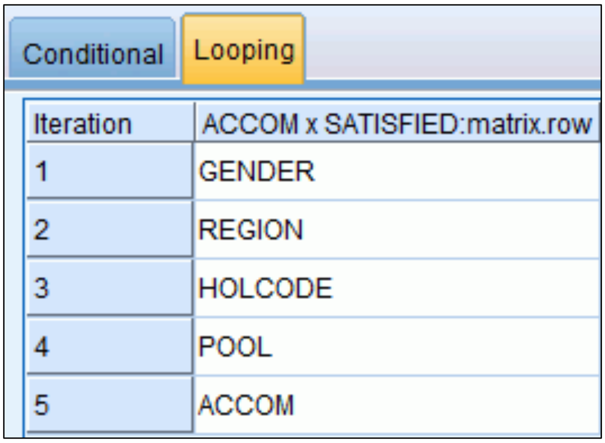
The Matrix node output is the same as in the previous task, so the parameter is in place.

This material is meant for IBM Academic Initiative purposes.

Task 8. Create a loop through the row fields in the Matrix node.

- Right-click the **Matrix** node, and select **Looping/Conditional Execution\Define Iteration Key (Fields)**, and then:
    - for **Iterate on**, ensure that **Node Property - Fields** is selected
    - for **Node**, ensure that **GENDER x SATISFIED** is selected
    - for **Property**, select **row**
    - in the area **Fields to use**, select **GENDER**, **REGION**, **HOLCODE**, **POOL**, and **ACCOM**
    - close the **Define Iteration Key** dialog box

    A section of the results appear as follows:



Iteration	ACCOM x SATISFIED:matrix.row
1	GENDER
2	REGION
3	HOLCODE
4	POOL
5	ACCOM

- Close the **Stream Properties** dialog box, and then click the **Run the current stream** ▶ button.
- Click **OK** to the Stream Parameter dialog.

    The Matrix node is executed 5 times, generating a cross tabulation for each of the row fields (more output is produced because of the various Data Audit nodes).

Note: The stream **workshop_improving_efficiency _completed.str**, located in the **05-Improving_Efficiency\Solutions** sub folder, provides a solution to the workshop exercises.

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.