Summary of Exploring and Validating Data

Exploring Data Using CAS Actions

- Return rows from a table

```
table.fetch /
    table={caslib="caslib-name",
           name="table-name",
           where="where-expression",
           vars={column-names}
           ...},
    fetchVars={"column-name-1",..., "column-name-n"},
    from=first-row-number,
    to=last-row-number,
    index=TRUE | FALSE,
    sortBy={{name="column-name"<, order="DESCENDING">}}
;
```

- Show column information:

```
table.columninfo /
    table={castable},
    <, additional parameters>;
```

- Show the number of rows in a Cloud Analytic Services table:

```
simple.numrows /
    table={castable};
```

- Compute the distinct and missing values in a CAS table:

```
simple.distinct /
    table={castable},
    inputs={column-names},
    casout={casouttable}
    <, additional parameters>;
```

- Generates a frequency distribution for one or more columns:

```
simple.freq /
    table={castable},
    inputs={column-names/em>},
    casout={casouttable}
    <, additional parameters>;
```

- Store the result of a CAS action:

```
    action-set.action <result=variable> / ...;
```

Working with Result Tables

- Result table operators

Action	What it does
table-variable-name[*row-selection,column-selection*]	Selects rows and columns from a result table object.
result-table[{*row-1, row-2, ...*}]	Selects specific rows.
result-table[*row-lower-bound*:*row-upper-bound*]	Selects rows based on lower and upper bounds, inclusive.
result-table[,*column-positions*]	Selects columns by position.
result-table[*row-positions*,{"*column-1*","*column-2*",...}]	Selects specific rows and columns.
result-table[,{"*column-name-1*", "*column-name-2*", ...}]	Selects all rows and specific columns.
result-table.where(*filter-expression*)	Subsets the rows in a result table according to the expression and returns a new result table.
result-table.compute <{'*variable-name*', '*label*', '*format*'},> *expression*)	Adds a temporary computed column to a result table and returns a new result table. The values are computed according to an expression.

- Result Table Properties

Action	What it does
***result-table*.nrows;**	Contains the number of rows in the table.
***result-table*.ncols;**	Contains the number of columns in the table.
***result-table*.attrs;**	Contains action-specific attributes for the table. This property is not added by all actions.
***result-table*.name;**	Contains the name of the table. In most cases, it is the same as the dictionary key that is used to access the table from the results.
***result-table*.title;**	Contains the title of the table.

- Save a Result Table

Action	What it does
SAVERESULT *variable-name*	Save as a

<**DATAOUT**=<libref.>*data-set-name* | **LIB**=*libref*> <**REPLACE** | **NOREPLACE**>;	SAS data set.
SAVERESULT *variable-name* <**CSV**="*file-name*" | **FILE**=*output-file*> <**REPLACE** | **NOREPLACE**>;	Save as a CSV file.
SAVERESULT *variable-name* <**CASLIB**=*caslib-reference-name* <**CASOUT**="*table-name*"> <**REPLACE** | **NOREPLACE**>;	Save as a CAS table.

Validating Data with CAS Actions

- Eliminate rows that have duplicate or unique group-by variable values:

```
deduplication.deduplicate /
   table={caslib="caslib-name",
          name="table-name",
          groupBy={"column-name" <, ...>},
   }
   casOut={caslib="caslib-name", name="table-name"},
    <duplicateOut={caslib="caslib-name", name="table-name"},>
   noDuplicateKeys=TRUE | FALSE;
```

- Copy one table to another:

```
table.copytable /
   table={caslib="caslib-name",
          name="table-name",
          computedVars={
               {name="variable-name", format="string", label="string"...},
               <{column-metadata-n}>
           computedVarsProgram="statement;
                                <more statements;>",
   },
   casout={caslib="caslib", name="table-name"};
```

- Embed text in the program and assign it to a given variable:

```
SOURCE variable;
      <text>
ENDSOURCE;
```