MultiBase Cosmos

Notes to version 5.5

BASE100

Índice

1. Implementations

- The following functions have been implemented in the CTSQL DBMS:

    - DECODE
    - NVL
    - ABS
    - TRUNC
    - SIGN
    - TO_CHAR
    - TO_DATE

- A number of features have been implemented in the API TTCHARTDLL in order to customize the color of the lines on the ordinate and abscissa (ChartSetXAxisColorLabel, ChartSetYAxisColorLine), customize the text color of the labels of the X axis (ChartSetAxisColorLabel), showing label elements as tooltips (ChartShowElementLabelAsTooltip), indicate the maximum and minimum values of the scale of values (ChartSetViewDataRange) and indicate the decimal scale of these labels in the ordinate (CharSetYAxisLabelPrecision).

- The following methods have been implemented in the Date class for the date formatting:

    - TOCHAR
    - FROMCHAR

- A new environment variable has been implemented (FETCHLASTBEFOREEDITNEW) in order to fetch the last row of the grid in the lines table in a header-lines form after the EditNew method of the FormTable class has been executed and the value of the QUERYBUFFERING environment variable is TRUE. Thus, the apparent loss of lines can be avoided.

- It has been implemented the ability of changing the value of the Background property (background color) and Foreground property (foreground color) in a Multiline EditField.

- A new environment variable has been implemented (EDITDISABLEDCTRTEXTCOLOR) in order to change the background color in all the disabled Text controls that are not owned by a FormTable.

2. Improvements

2.1 ODBC

- The ODBC 32-bits installer has been changed.

- Modifications have been done to improve compatibility with LibreOffice.

2.2 Monitor

- A new Check control has been added "Auto Refresh" to allow changing between Automatic and Manual refresh of the Monitor's data.

- A new functionality has been added to the Monitor application that allows to show the tables and rows locks.

2.3 Cosmos

- The communications buffer size has been changed in Cosmos applications (cosrun, csql) in order to allow a faster data-reading in a client-server communication in tables with a row-size greater than 1024 bytes .

- The method Last of the FormTable class has been optimized in buffered connections to a database.

- The Cosmos installer allows to install the 32-bit ODBC driver and the 64-bit ODBC driver.

- The Font used in a Drop List control of an editable List control will be the same as defined in the list control.

- The AutomaticSort property in a Grid control can be changed with the method SetProperty of the Control class.

- if the number of digits are not specified in grouped Lists (ShowMultiColumnGroupDlg and ShowListAsMultiColumnGroup) , it will be 3 by default.

- Editable List Control. If the column's type is Date type and a calendar has been unfolded, when a value is selected the ListAcceptEdit event will be launched.

- A new data-view has been implemented in grouped Lists. This data-view allows to choice if the last-group's totals will be shown in the same row as the group when the no detail data-view is selected. The methods ShowMultiColumnGroupDlg and ShowListAsMultiColumnGroup have been modified.

  The ShowMultiColumnGroupDlg shows a new Check control with the "Compact" text in the bar. If the control is checked and the "Detail" check-control is unchecked, the aggregates of the last aggregates column will be shown in the same row as the group.

  To enable this option with the ShowListAsMultiColumnGroup, the value must be COMPACTMODE:YES/COMPACTMODE:TRUE or COMPACTMODE:NO/COMPACTMODE:FALSE.

- New environment variable (HIDECOLUMNIFNODETAIL). This environment variable allows to hide the columns in the grouped Lists (ShowMultiColumnGroupDlg and ShowListAsMultiColumnGroup methods) that do not belong to group's break or do not belong to aggregates column when the "Detail" check-control is unchecked.

- New messages have been added in Cosmos Windows to show to the user that the process is running.

    - CSQL. The program shows the new message "Running …" when the user clicks in the "Run" icon.

    - CSQL and SelectWindow method. If the user exports the resultset to csv, xls, unl, ods, pdf, etc., the message "Exporting list, please wait …" is shown in the window.

    - Preview method. If the user exports the report to xls, csv, etc., a progress bar is shown in a window.

- In the processes shown in the previous section and in the ExportToExcel method of the SimpleControl's class, the Cosmos runtime will refresh the window automatically, minimizing the number of times that the "Not responding" message is shown.

- From this version, the data export with ExportToExcel, ExportToHTML, ExportToODS and ExportToPdf in the Grid-controls will be complete regardless of whether the data is stored in the buffer control (environment variable QUERYBUFFERING = TRUE).

- Environment variables DEACTIVATEDPKCOLOR and DEACTIVATEDTEXTCOLOR allow a value in RGB(r,g,b) format. Also supported as an integer value.

- Method AllowColumnHeaderFilter. The load process of unique registers has been improved.

2.4   Ctsql

The new CTSQL functions (ABS, INITCAP, SUBSTR, CONCAT, LOWER, UPPER, LTRIM, RTRIM, DECODE, NVL, SIGN, TO_DATE and TO_CHAR) can be used in the "insert" statement.

2.5   APIS

2.5.1   BMAPSDLL

Loading maps and resources optimization.

2.5.2   TTCHARDLL

The DLL is optimized for use on tablet-pc.

BASE100

3. Environment variables

FETCHLASTBEFOREEDITNEW

If it's value is TRUE, the cursor will be positioned in the last record of the table when the EditNew method of the FormTable class is executed.

This environment variable should only be used when the QUERYBUFFERING environment variable value is TRUE.

The behavior will be the same than when the QUERYBUFFERING environment variable value is FALSE.

The allowed values are TRUE and FALSE.

The value of this environment variable can be changed in runtime. Also, the value of the environment variable can be defined in the Environment section of the cosmos.ini file and Environment section of the project configuration file.

EDITDISABLEDCTRTEXTCOLOR

This environment variable allows to define the text color of the controls of type Edit Field, Drop Edit and Drop List that are disabled (property Disabled= TRUE).

This environment variable can be defined in the Environment section of the project configuration file and the Environment section of cosmos.ini, also with the SetOption method of the Module class or Form class, and the PutEnv method of the Module class.

If this variable is defined in the project configuration file or in cosmos.ini, it's value must be an integer number result from the combination of the red, green and blue colors, or the RGB string with each color separated by comma.

Syntax:

```
EDITDISABLEDCTRTEXTCOLOR=65280

EDITDISABLEDCTRTEXTCOLOR=RGB(0,255,0)
```

HIDECOLUMNIFNODETAIL

This environment variable allows you to specify if the columns that don't belong to a break-group or aggregates will be hidden or shown.

The allowed values are: TRUE o YES and FALSE o NO.

The default value is FALSE.

This environment variable can be defined in the "Environment Global" of the cosmos.ini file or the project's configuration file. This variable can also be defined with the PutEnv method of the Module class.

GWSENDLABELINDESCRIBE

This variable is only available in a connection to Oracle or Informix with Gworacle or Gwinformix.

When the SqlDescribe method of the SqlServer Class is executed, this environment variable tells the runtime that it should return de column alias in expression type columns.

If this environment variable is not defined, the string returned is "(expression)".

This environment variable can be defined with the PutEnv method of the SqlServer class.

MINIMIZEFORMCASCADE

This environment variable tells the runtime that if a window is minimized the parent windows will be minimized .

This environment variable can be defined in the cosmos.ini file or the project's configuration file.

DBCONNBLOCKSIZE

This environment variable tells the runtime the connection buffer size in bytes in a connection with CTSQL, Gworacle or Gwinformix.

4. TTCHARTDLL API functions

- **ChartShowElementLabelAsTooltip**. This function allows to show a tooltip with the element's label when the mouse pointer is over a element.

  Syntax:

  ```
  ChartShowElementLabelAsTooltip(hWindows as integer, showAsTooltip as booe-
  lan) return boolean
  ```

  Parameters:

  hWindow   Window handle of chart control.

  showAsTooltip   Boolean parameter. If it's value is TRUE, the tooltip is shown when the mouse pointer is over the element. If it's value is FALSE, the label is shown over the element.

  Returns: TRUE if the function is executed successfully.

  The tooltip's text corresponds with the text inticated in the elementLabel paramenter in the NewChartElement function. The tooltip's title corresponds with the value of the label in the abscissa axis.

  The tooltip's title can be changed by defining it in the parameter elementLabel in the NewChartElement function. The text and the title will be separated by a pipe character ("|").

- **ChartSetXAxisColorLabel.** This function allows to define the text color of the X-axis labels.

  Syntax:

  ```
  ChartSetXAxisColorLabel(hWindows as integer, textColor as integer) return
  boolean
  ```

  Parameters:

  hWindow   Window handle of chart control.

  textColor   Text color.

  Returns: TRUE if the function is executed successfully.

- **ChartSetXAxisColorLine.** This function allows to define the color of the X-axis lines.

  Syntax:

  ```
  ChartSetXAxisColorLine (hWindows as integer, lineColor as integer) return
  boolean
  ```

  Parameters:

  hWindow   Window handle of chart control.

  lineColor   Line color.

  Returns: TRUE if the function is executed successfully.

- **ChartSetYAxisColorLine.** This function allows to define the color of the Y-axis lines.

    Syntax:

    ```
    ChartSetYAxisColorLine (hWindows as integer, lineColor as integer) return
    boolean
    ```

    Parameters:

hWindow	Window handle of chart control.
lineColor	Line color.

    Returns: TRUE if the function is executed successfully.

- **ChartSetViewDataRange.** This function allows to set de minimum and maximum display values of the scale of values in the ordinate axis.

    Syntax:

    ```
    ChartSetViewDataRange (hWindows as integer, mainYAxis as boolean, enable as
    boolean, minValue as decimal, maxValue as decimal) return boolean
    ```

    Parameters:

hWindows	Window handle of chart control.
mainYAxis	This parameter indicates whether the scale values are displayed in the main axis of ordinates or in the secondary axis of ordinates. TRUE=main. FALSE=secondary.
minValue	Lowest value.
maxValue	Highest value.

    Returns: TRUE if the function is executed successfully.

    This function has a number of limitations that are detailed below:

    - The minValue and the maxValue parameters can't be negative numbers.

    - The minValue parameter must be lower tan maxValue parameter.

    - If any of the values in the graph is negative, the library will draw the graph regardless of the values given as maximum and minimum.

- **ChartSetYAxisLabelScale.** This function allows to indicate the numeric scale (number of decimal values) of the values in the labels of the ordinates axis.

    Syntax:

    ```
    ChartSetYAxisLabelPrecision (hWindows as integer, scale as integer, main-
    YAxis as boolean) return boolean
    ```

    Parameters:

hWindows	Window handle of chart control.
scale	Integer number that indicates the numeric scale.

mainYAxis   This parameter indicates whether the scale will be displayed in de main or secondary ordinates axis. If it's value is TRUE, the values are from the main axis. If the value is FALSE, the values are from the secondary axis.

Returns: TRUE if the function is executed successfully.

This function should not be used if the labels are shown in abbreviated mode (ChartSetShortYAxisLabel function).

5. SimpleControl class methods

- **SetListColorByLevel.** This method allows to set the background color in the different levels of the grouped Lists to make an easier display.

  Syntax:

  ```
  SetListColorByLevel(levelNumber as Smallint ,levelBackColor as Integer)
  ```

  Parameters:

  levelNumber Break level where the background color will be set. The first level is the level number 1.

  levelBackColor Integer number in RGB format that indicates the background color of the break level specified in the first parameter.

  NOTE: The list will not be displayed with the colors defined by this method until the method EnableListColorByLevel is not executed.

- **EnableListColorByLevel.** This method indicates to the runtime that the colors defined with the SetListColorByLevel method will be displayed.

  Syntax:

  ```
  EnableListColorByLevel (setColor as Boolean)
  ```

  Parameters:

  setColor Boolean. Indicates whether the color by level display will be turned on or turned off.

  If the parameter value is TRUE, the aggregate's labels set in the ShowMultiColumnGroupDlg and ShowListAsMultiColumnGroup methods will be shown in the break-group column, not in the aggregate's column. In addition, a new blank line will be added at the end of the aggregates, thereby facilitating the location of the breaking group.

Detail	Compact	ColorByLevel	Aggregate's label function	Blank line between groups
Yes	No	No	No	No
Yes	No	Yes	Yes	Yes
No	No	No	No	No
No	No	Yes	Yes	Yes
No	Yes	No	No	No
No	Yes	Yes	Yes	Yes

  NOTE: the call to this method must be done before the call to the ShowMultiColumnGroupDlg y ShowListAsMultiColumnGroup methods.

- **SetGridColumnDisabled.** This method allows to change the disabled property in the Grid controls.

  Syntax:

  ```
  SetGridColumnDisabled (index as integer, disabled as boolean)
  ```

BASE100

Parameters:

index	Column id.
disabled	Boolean. This parameter indicates if the column will be disabled or not.

The available values are: TRUE and FALSE:
TRUE. Disables the column.
FALSE. Enables the column.

- **SetTextDireccion.** This method allows to set the text orientation in a TEXT control.

    Syntax:

    ```
    SetTextDireccion(align as Smallint)
    ```

    Parameters:

    align   Integer value that sets the text direction. The available values are:

    1. Horizontal orientation (default orientation).
    2. Bottom vertical orientation.
    3. Top vertical orientation.

- **SetTextHorinzontalAlign.** This method allows to set the horizontal text align (left, center, right) in the TEXT controls.

    Syntax:

    ```
    SetTextHorizontalAlign(align as Smallint)
    ```

    Parameters:

    align   Integer value that sets the text align. The available values are:

    1. Left aligned text.
    2. Centered text.
    3. Right aligned text.

- **SetTextVerticalAlign.** This method allows to set the vertical text align in the multiline TEXT controls.

    Syntax:

    ```
    SetTextVerticalAlign(align as Smallint)
    ```

    Parameters:

    align   Integer value that sets the text align. The available values are:

    1. The text will be aligned at the top of the control.
    2. The text will occupy all the vertical space of the control (vertical centered alignment) This is the default alignment.

6. DATE class methods

- **ToChar**. This method returns a formatted string as from a date type object with a format template received as parameter.

  Syntax:

  ```
  ToChar(format as Char ,NLS as Char default NULL) return Char
  ```

  Parameters:

format	Format string (see Annex I).
NLS	Optional parameter that specifies the conversion language (see Annex II).

  Returns: Formatted string.

  For example:

  ```
  lDtDate =  '28/02/2014';
  lDtDate.ToChar('YYYY-DD-MONTH');
  ```

  Returns: 2014-28-February.

- **FromChar**. This method assigns a formatted string-date to a date type object.

  Syntax:

  ```
  FromChar(date as Char ,format as Char ,NLS as Char default NULL) return Date
  ```

  Parameters:

date	String-date.
format	String date-format for the first parameter (see Annex III).
NLS	Optional parameter that specifies the language of the day and month for the first parameter (see Annex II).

  Returns: Date object with the value specified in the first parameter.

  For example:

  ```
  DtDate.FromChar('2009-28-FEBRUARY', 'YYYY-DD-MONTH')
  ```

  In this example the value "28th February 2009" will be assigned to tha DtDate object.

7. CTSQL Functions

7.1 DECODE

This function compares the parameter *expr* with every one of the search values one by one. If *expr* is equal to a search value, returns the corresponding value.

This function works like a "if xxx then xxx" or a "switch-case" into the select statement.

Syntax:

```
DECODE(expression, search, result [, search, result]... [, default ])
```

Parameters:

expression	Expression to compare.
search	Value to compare with the first parameter.
result	Return value if expression and search are equal.

For example:

```
SELECT customer,
DECODE(province,  1, 'Álava',
                  8, 'Barcelona',
                  28, 'Madrid',
                   'Another province') result
FROM customers;
```

In this query, for every returned register, CTSQL compares the value of the province field with 1, 8 and 28. If the value is equal than some of these values, returns the corresponding value (1: Álava, 8: Barcelona y 28: Madrid). If the value is not equal than none of these values, it will return "Another province".

7.2 NVL

This function compares expr1 with NULL. If expr1 is NULL, then will return expr2.

This function works like a "if expr1 == NULL then expr2, else expr1".

Syntax:

```
NVL(expr1,expr2)
```

Parameters:

expr1	Value to compare with NULL.
expr1	Value that returns if expr1 is NULL.

For example:

```
SELECT customer,
NVL(description, "customer without description")
FROM customers
```

In this query, for every returned record the description column is compared with NULL. If so, the "customer without description" string will be returned. If the column value is not null, then returns the column value.

7.3 ABS

This function returns the absolute value of a numeric expression in a query.

Syntax:

```
ABS(expr)
```

Parameters:

expr   Numeric value to return the absolute value.

For example:

```
Select  abs(income - outgoings) balance from movements
```

This query fetch every record of the movements table, and returns the absolute value of the difference between income and outgoings columns.

7.4 TRUNC

7.4.1 Date

This function returns a date truncated to the specified unit of measure.

Syntax:

```
TRUNC(date, format)
```

Parameters:

date   Date value to truncate.

format   Date's format (see Annex IV).

For example:

```
select
date(today) today_column,
trunc(date(today) ,"year") first_day_january,
trunc(date(today), "iy") first_day_year_iso,
trunc(date(today), "w") first_day_week,
trunc(date(today), "iw") first_day_week_iso,
trunc(date(today), "day") day_monday
from systables where rowid = 1
```

If today is January 20, returns:

today	First_day_january	First_day_year_iso	First_day_week	First_day_week_iso	Day_monday
20/01/2014	01/01/2014	30/12/2013	15/01/2014	20/01/2014	20/01/2014

BASE100

7.4.2   Numbers

This function truncates the decimal number of the first parameter to the decimals that are specified in the second parameter.

Syntax:

```
TRUNC(number, decimals)
```

Parameters:

number   Decimal number to truncate.

decimals   Number of decimals to truncate de first parameter.

If the second parameter is a negative number, then it will truncate to the left of the decimal point with 0 on these places (-1 units, -2 tens, -3 hundreds, …).

For example:

TRUNC(125.815, 0)	125
TRUNC(125.815, 1)	125.8
TRUNC(125.815, 2)	125.81
TRUNC(125.815, 3)	125.815
TRUNC(-125.815, 2)	-125.81
TRUNC(125.815, -1)	120
TRUNC(125.815, -2)	100
TRUNC(125.815, -3)	0
TRUNC(45633,-3)	45000
TRUNC(45633,-2)	45600

7.5   SIGN

This functions returns -1 if the value of the parameter is a negative number, 1 if the parameter is a positive number and 0 if it's value is 0.

Syntax:

```
SIGN(num)
```

Parameters:

Num   Number.

```
SELECT
decode(sign( det.diario - tope.diario ), -1, det.diario, 0, det.diario,
tope.diario ) AS diario
decode( sign( det.semanal - tope.semanal ), -1, det.semanal, 0, de-
t.semanal, tope.semanal ) AS semanal
decode( sign( det.mensual - tope.mensual ), -1, det.mensual, 0, de-
t.mensual, tope.mensual ) AS mensual
FROM detalle det, tope;
```

7.6 TO_DATE

This function converts a character string into a DATE type value from an input format.

Syntax:

```
To_date(string[, format [, nlsparams ]])
```

Parameters:

string	Character string that represents a date.
format	String date-format for the first parameter (see Annex III).
nlsparams	Optional parameter that specifies the language of the day and month for the first parameter (see Annex II).

```
Select TO_DATE('31/10/2009','DD/MM/YYYY') FROM
RESULT: 31/10/2009
Select TO_DATE('2009-31-JAN,'YYYY-DD-MON') FROM
RESULT: 31/01/2009
```

7.7 TO_CHAR

This functions converts numeric and date values into a formatted string.

7.7.1 Date

This function returns a character string from a date type column with a format template received as parameter.

Syntax:

```
TO_CHAR(fecha, [formato [, nlsparams] ])
```

Parameters:

date	Date-type value to convert into string.
format	String format format (see Annex I).
nlsparams	Optional parameter that specifies the conversion language (see Annex II).

```
SELECT
TO_CHAR(date(today), "dd-fmMonth-yyyy", "NLS_DATE_LANGUAGE=spanish") FROM
….
If today is 20/01/2013, returns "20-Enero-2013".
SELECT
TO_CHAR(date(today), "dd-FMmm-yyyy") FROM ….
If today is 20/01/2013, returns "20-1-2013".
Returns 1 instead of 01 because the FM operator is present.
```

7.7.2 Numbers (smallint, integer, serial, money, decimal)

This function converts a numeric value into a character string with a format specified in the second parameter.

Syntax:

```
TO_CHAR(numero, [formato [, nlsparams] ])
```

BASE100

Parameters:

number	Numeric value to format.
format	This parameter specifies the format of the returned character string (see Annex V).
nlsparams	Optional parameter that allows to specify different modifiers like the decimal separator, the thousands separator and the currency identifier.
	The available values are NLS_NUMERIC_CHARACTERS, NLS_CURRENCY, NLS_ISO_CURRENCY. You may indicate one or more variables in the same statement. The values can be quoted, unquoted or single quoted (see Annex VI).

For example:

```
SELECT TO_CHAR(-10000,'L99G999G999G999D99PR',
'NLS_NUMERIC_CHARACTERS=''.,''
   NLS_CURRENCY = ''AusDollars''
     ')….
```

Returns: <AusDollars10,000.00>.

The value is shown between the < and > characters because is a negative number and the PR operator is specified.

Returns "AusDollars" because the L operator is specified, and the NLS_CURRENCY variable has this value.

```
SELECT TO_CHAR(-10000,'C99G999G999G999D99pr',
   'NLS_NUMERIC_CHARACTERS = '',.''
   NLS_ISO_CURRENCY = ''united kingdom'' ')
```

Returns: <GBP10.000,00>.

The string "GBP" is the international ISO currency of the United Kingdom (NLS_ISO_CURRENCY value is United Kingdom). The "C" operator is replaced by the "GBP" string.

The number of characters in the returned string will be the number of characters of the format string.

If the number of digits in the integer part of the number is greater than the number of digits in the integer part of the format, it will return a character string with the "#" character.

If the number of digits in the integer part of the number is less than the number of digits in the integer part of the format, it will return the number converted and filled with white spaces at left of the string.

If the number of digits in the decimal part of the number is greater than the number of digits in the decimal part of the format, it will return a string with the number rounded to the number of decimals specified in the format.

If the number of digits in the decimal part of the number is less than the number of digits in the decimal part of the format, it will return a string with the same number of decimals specified in the format.

```
SELECT
TO_CHAR(1234.567, "999g999d999") FROM ….
```

If the DBMONEY environment variable is not defined or is defined with the "," (comma) value, it will return:

" 1.234,567"
[blank][blank]1.234,567

If the DBMONEY environment variable is defined with the "." (dot) value, it will return:

" 1,234.567"
[blank][blank]1,234.567.

If the NLS_NUMERIC_CHARACTERS variable is defined in the third parameter, the decimal separator and thousand separator will be the defined in this parameter. The NLS_NUMERIC_CHARACTERS variable has precedence over the environment variable DBMONEY.

8. Fixes

8.1 Runtime

- Cosrun. If the last column of a strings list control was edited, when you made an horizontal scroll the width of the edit field was not the same of the width of the column.

- Sometimes, when the SelectWindow method was executed and the user clicked the header of the column in order to sort by this column, the result was incorrect. This problem happens when the database connection is buffered.

- Cosrun. When the user tried to filter an SQL list with the ShowListFilterBar method, if the SQL statement had a constant in the columns list, the runtime didn't build the SQL phrase properly and the list didn't show any record.

- Cosrun. In a grouped List with the ShowMultiColumnGroupDlg method, the performance was affected if it had a big amount of rows and the cursor was at the end of the list.

- Cosrun. By grouping and sorting a list, if it had many records and the process took too long, the mouse pointer disappeared and no one knew if the process would have finished or not. Now the clock pointer is shown during the process.

- Cosrun. Negative values are allowed in the editable lists.

- Cosrun. Editable lists. If the user accepted or cancelled the edition process in an editable list with drop-edit control, the ListAcceptEdit and ListCancelEdit events were not launched.

- Cosrun. If the descriptive text in the label of the GroupListColumns method of the SimpleControl class was very long, there was an error in the runtime that aborted the process.

- Cosrun. The method Using of the Date class with the parameter 5 didn't return the correct month name in Windows 8.1.

- Cosrun. Editable list with drop-edit control. When the drop-edit control was unfolded and the mouse pointer was positioned in another cell, the focus was in the drop-edit control and didn't pass to the new cell.

- Cosrun. Error in the Excel export in the grouped list with the ShowListAsMultiColumnGroup method without detail lines and AlternateBackColor turned on.

- Cosrun. Error in export to Excel from the Preview Method, in only-data mode and with a great amount of pages, when the XLS document was opened Excel returned an error.

- Cosrun. The method TotalizeColumn returned an error if the syntax of the third parameter was not correct.

- Cosrun. An error was fixed when a Form was closed when a list control had not finished yet the loading process.

- Cosrun. The "by column" list were not properly exported to Excel.

- Cosrun. The export to Excel was not performed properly if the list had "computed columns" and "decimal columns".

From this version of Cosmos it is not mandatory to use the NumericColumnsFormattedToChar method to export to Excel a list with computed columns.

- Cosrun. The multiline text controls with "vertical-right" or "vertical-left" property were not properly shown if they had more than one line.

- Cosrun. AllowColumnHeaderFilter method. If the list control was Sql-type and the environment variable QUERYBUFFERING had the value TRUE, when the header list was unfolded not all the records were shown.

8.2   DBMS

- CTSQL. Concat function. If the first parameter was a decimal-type or money-type, didn't eliminate the blank spaces.

- CTSQL. The DateTime-type columns were not being sorted properly. This error was from version 3.6 0.27 to version 3.6 0.33, both included. This error affects the Cosmos versions from 5.0 to 5.4.

- CTSQL. If there was an index with the A and D fields and an index with the A,B and C fields, in the "order by A,B,C" the engine choose the first index instead of the second, that was more optimal. It has been fixed for the cases where there is more than one index than can be chosen for sorting. The engine will chose the index with more coincident fields.

- TTTOOLS. TTTOOLS application. The database Views were displayed as items to repair or check . When you tried to repair, the app showed the message: "Error in DB name and table".

- TTTOOLS. When the user clicked the "update statistics" button, the list of tables was not displayed. It was mandatory to reopen the database to display the list of tables.

8.3   ODBC

- ODBC. Connection from OpenOffice or LibreOffice. When a query was launched, not all the records were displayed.

- ODBC. Connection from OpenOffice or LibreOffice. If a table had a date type fields in the PK, the scroll was not made properly. If the user tried to modify the data from LibreOffice Base, an error was displayed.

8.4   Gateway Oracle

- GWORACLE. The outer statements were not properly translated if it had constant fields in the where clause.

- GWORACLE. The outer statements were not properly translated if it had alias in the name of the tables.

- GWORACLE. When the SqlDescribe method was launched with an expression column with CTSQL, it returns the alias. However, the same statement with Oracle or Informix returned "(expression)". To have the same behavior than CTSQL, the GWSENDLABELINDESCRIBE environment variable must be defined with TRUE or YES value.

8.5   API TTCHARTDLL

- TTChartDll. If the ChartSetTile function included a title with the Ç letter or the euro symbol, an error was returned.

BASE100

ANNEX I

Available values in the format parameter from the ToChar method of the Date class and the TO_CHAR function of CTSQL:

AD	AD indicator without periods.
BC	BC indicator without periods.
A.D.	AD indicator with periods.
B.C.	BC indicador with periods.
SCC	Century.
CC	Century.
DDD	Day of year.
DD	Day of month.
DAY	Name of week day in 9 chars with blanks at right.
	The language of the week day is the selected in the regional configuration ("locale") of the operating system, unless another language is selected in the third parameter with the NLS_DATE_LANGUAGE variable.
DL	Date value in long format.
	For example: "25/12/2013" returns "Wednesday 25 december 2013".
DS	Date value in short format. The function will return the date in the format specified in the DBDATE environment variable, by default: "dd/mm/yyyy".
DY	Abbreviated name of day.
IW	Week of year (1-52 or 1-53) based on the ISO[1] standard.
IYYY	4-digit year based on the ISO[1] standard.
IYY	Last 3 digits of ISO[1] year.
IY	Last 2 digits of ISO[1] year.
I	Last digit of ISO[1] year.
MM	Month (01-12; January = 01).
MONTH	Name of month in 10 chars, padded with blanks at right.
	The language of the month is the selected in the regional configuration ("locale") of the operating system, unless another language is selected in the third parameter with the NLS_DATE_LANGUAGE variable.
MON	Abbreviated name of month.
	The language of the week day is the selected in the regional configuration ("locale") of the operating system, unless another language is selected in the third parameter with the NLS_DATE_LANGUAGE variable.
Q	Quarter of year (1, 2, 3, 4; January - March = 1).
RM	Roman numeral month (I-XII; January = I).

---

[1] According to ISO 8601, the week always start on Monday and end on Sunday. The first week of the year will be the one that contains the first Thursday of the year. Thus, it may be the case there are years with 52 weeks and years with 53 weeks, and the 29, 30 and 31 December, a year belong to the first week of the following year, or the days 1, 2 and January 3 belong to the last week of the previous year.

WW	Week of year (1-53), where the week 1 starts on the first day of the year and continues to the seventh day of the year.
W	Week of month (1-5), where the week 1 starts on the first day of the month and ends on the seventh day of the month.
SYYYY	4-digit year; The S key indicate that the years before the year 1 will have a minus sign ("-").
YYYY	4-digit year.
Y,YYY	Year with comma in this position.
YYY	Last 3 digits of year.
YY	Last 2 digits of year.
YEAR	Spelled year (in spanish).
Y	Last digit of year.
D	Day of week (1-7).
FM	Returns a value with no zeros at left. Valid for the next operators: MM, DD, IW, DDD, WW, YYY, IYY[2], YY, IY, RR.
FX	Requieres exact matching between the carácter data and the format model. CTSQL accepts this operator but it is not used.
RRRR	Rounded year. Accepts an input of 2 or 4 digits. If it is a 2 digits year, returns the same value as RR. If it is a 4 digit year, returns the same than YYYY.
RR	Last 2 digits of the year. If the year is < 50 then belongs to 21th century. If the year is >= 50, then belongs to 20th century.
SP	*Spelled number*. Suffix that, located at right of an operator, returns it's value as spelled number. It is possible to locate next to the operators: DD, MM, YYYY, IYYY, SYYYY, CC, SCC, DDD, WW, W, IW, IYY[1], YYY, YY, IY[2], Q, RR, Y,YYY.

The operators can be lowercase or uppercase. Si they are in uppercase, in the case of displaying a text (month name, week day, day numeral, etc.), it will displayed in uppercase. If the first letter of the operator is in uppercase, the text will be displayed in uppercase.

---

[2] See note in previous page.

Annex II

NLS_DATE_LANGUAGE	This variable indicates the language in the week days and months.
	The language name will be in English. The quotes are optional.
	For example:
	NLS_DATE_LANGUAGE=spanish
	NLS_DATE_LANGUAGE=''spanish''

Annex III

Available values in the format parameter from the FromChar method of the Date class and the TO_DATE function of CTSQL:

D	Day of week. It is accepted and ignored.
DD	Day of month.
DDD	Day of year.
DAY	Day of week in long format. It is accepted and ignored.
DY	Abbreviated name of day (3 characters). It is accepted and ignored.
W	Week of month (2 characters). It is accepted and ignored.
WW	Week of year. It is accepted and ignored.
MM	Month (2 characters).
MON	Abbreviated name of month (3 characters).
MONTH	Name of month (n characters).
RM	Roman numeral month.
Q	Quarter of year (1 character). Accepted and ignored.
YYY	Last 3 digits of year.
YYYY	4-digit year.
YY	Last 2 digits of year.
RR	Last 2 digits of year. If the year is < 50 then belongs to 21th century. If the year is >= 50, then belongs to 20th century.
RRRR	Rounded year in 4 digits. Accepts an input of 2 or 4 digits. If it is a 2 digits year, returns the same value as RR. If it is a 4 digit year, returns the same than YYYY.
Y	Last digit of year.

Annex IV

These are the date formats allowed by the format parameter in the TRUNC function of CTSQL.

Format	Value
SYYYY, YYYY, YEAR, SYEAR, YYY, YY, Y	Returns the January 1 of the year in the date of the first parameter.
IYYY, IY, I	Returns the first day of the ISO[3] year in the date of the first parameter. I.e. the Monday prior to the January 4 in the date of the first parameter
MONTH, MON, MM, RM	Returns the first day of the month (day 1) in the date of the first parameter.
WW	Rounds to the same day of the week than the first day of the year in the date of the first parameter. For example, if the first day of the year is a Wednesday, this format will return the previous Wednesday in the date of the first parameter.
W	Returns the day of the week that coincides with the first day of the month in the date of the first parameter. If the first day of the month is a Tuesday and the date in the second parameter is a Thursday, returns the date corresponding to the Tuesday in this week. If the date in the second parameter is a Monday, the date returned will be the Tuesday of the previous week.
IW	ISO[3] week. Returns the date of the week day equal than the first day of the week in the ISO[3] year. I.e. the previous Monday to the date in the second parameter.
DDD, DD	Returns the same date than the second parameter.
DAY, DY, D	Returns the Monday of the week that the date belongs. I Works the same than IW.

---

[3] See the note in page 22.

Annex V

Allowed values in the format parameter of the TO_CHAR function for numeric values in CTSQL:

Format	Example	Description
"," (comma)	9,999	Returns a comma in the specified position. It will correspond with the thousands separator. You can include more than one comma in a number format. Restrictions: A comma can't be positioned at right of a decimal separator in the number format.
"." (dot)	99.99	Returns a dot in the specified position. It will correspond with de decimal separator. Restrictions: A decimal dot can't be positioned at right of another decimal dot.
$	$9999	Returns de dollar sign in the specified position.
0 (cero)	0999 9990	Fills with zeros at left (integer part) or right (decimal part) until the position where the 0 character is found.
9	9999	Returns the value with the specified number of digits and with a blank space in the first position if the number is positive, or a minus sign if the number is negative. The initial zeros will be blanks, except if the "0" modifier is in the format.
B	B9999 B.9999	If the number is 0, returns a blanks string, If the integer part is 0, returns a blank string in the integer's place.
C	C999	Returns the ISO currency name in the specified position. This name will be the corresponding in the regional configuration ("locale") of the OS. The currency symbol can be changed by assigning a value to the NLS_ISO_CURRENCY variable.
D	99D99	Returns the decimal character in the specified position. If the DBMONEY variable is not define, it will be the "," sign. If the DBMONEY variable is defined, it will be the DBMONEY value. If the NLS_NUMERIC_CHARACTERS is defined in the third parameter, it will correspond with the value indicated in this variable, overriding this variable on DBMONEY.
FM	FM90.9	Returns the value without blanks at left.
G	9G999	Returns the thousands separator character in the specified position. If the DBMONEY variable is not defined, it will be the dot sign ".". If the DBMONEY variable is defined, it will be the opposed value: If DBMONEY=, (comma), the G will be "." (dot). If DBMONEY). (dot), the G will be "," (comma). If the NLS_NUMERIC_CHARACTERS is defined in the third parameter, it will correspond with the value indicated in this variable, overriding this variable on DBMONEY.. A group separator can't be positioned at right of a decimal separator.
L	L999	Returns the currency symbol in the specified position. The default value is determined by the regional configuration ("locale") of the OS. The value can be modified by defining the NLS_CURRENCY variable in the third parameter.
MI	9999MI	If the number is negative, returns the number with a "-" sign at the end of the number If the number is positive, returns the number with a blank at the end of the number. The MI format modifier can appear only in the last position of a number format.

BASE100

Format	Example	Description
PR	9999PR	If the number is negative, returns the number between angle brackets (<) and (>). If the number is positive, returns the number with a blank at first and a blank at the end. The PR format modifier can appear only in the last position of a number format.
RN	RN	Returns the number in roman numbers. The value can be between 1 y 3999.
S	S9999 9999S	If the number is negative and the "S" modifier is at first of the format, returns the value with a leading minus sign (-). If the number is positive and the "S" modifier is at first of the format, returns the value with a leading plus sign (+). If the number is negative and the "S" modifier is at end of the format, returns the value with a trailing minus sign (-). If the number is positive and the "S" modifier is at end of the format, returns the value with a trailing plus sign (+).
X	XXXX xxxx	Returns the hexadecimal representation of the integer part of the number. The allowed values are 0 and positive numbers.

Annex VI

Allowed values of the nlsparams parameter (Native Language Support) in the TO_CHAR function for numeric values:

NLS_NUMERIC_CHARACTERS	This value specifies the decimal character and the thousands separator character that will be used by the "G" and "D" modifiers in the format numeric string.
	The value must be a two characters string where the first character indicates the decimal separator ("D") and the second character indicates the thousands separator ("G").
	If the variable is not specified, the DBMONEY values will be used.
NLS_CURRENCY	This value specifies a character string (free text) that will be used like local currency when the "L" operator is used in the format string in the second parameter.
	If this variable is not defined, when the "L" is specified, the local currency symbol used will be the corresponding in the regional configuration ("locale") in the OS.
NLS_ISO_CURRENCY	If the "C" modifier is used in the format string, the string will be replaced with the international currency name corresponding with the regional configuration ("locale") in the OS.
	If you want to use a different name for the currency as the specified in the regional configuration, you must define the NLS_ISO_CURRENCY variable with the name of the country (in English) whose currency want to use.