Article
Irène Mykhailova · Mai 25, 2022 9m de lecture

Traitement des opérations sur les dates et les heures dans Caché

Voici quelques exemples de conversions et d'opérations dont vous pourriez avoir besoin, ainsi que des liens vers la documentation où vous pourrez en apprendre davantage.

Au moment où j'ai écrit ces lignes, l'heure d'été était en vigueur pour mon système Caché.

Comment Caché conserve l'heure et la date

Caché a un format d'heure simple, avec une plus grande gamme de dates reconnues par rapport à certaines autres technologies.

L'heure actuelle est conservée dans une variable spéciale $HOROLOG ($H) :

```
USER>WRITE $H

64146,54027

USER>
```

Le premier nombre entier est le nombre de jours écoulés depuis le 31 décembre 1840. Le second nombre entier est le nombre de secondes écoulées depuis minuit le jour actuel.

Vous pouvez également obtenir l'heure et la date actuelles avec $SYSTEM.SYS.Horolog().

Comment établir un horodatage

$HOROLOG comptabilise le temps avec une précision de l'ordre de la seconde. $ZTIMESTAMP a une forme similaire à $HOROLOG, mais il suit les fractions de seconde dans la partie temps et conserve le temps universel coordonné (UTC), plutôt que l'heure locale. La précision des fractions de seconde dépend de votre plate-forme.

Par conséquent, $ZTIMESTAMP fournit un horodatage qui est uniforme dans tous les fuseaux horaires. L'horodatage que vous voyez à un moment donné peut avoir une date et une heure différentes de votre heure locale actuelle. Dans cet exemple, mon heure locale est l'heure avancée de l'Est, soit quatre heures de moins que l'heure UTC.

```
WRITE !,"$ZTIMESTAMP: "_$ZTIMESTAMP_" $HOROLOG: "_$HOROLOG

$ZTIMESTAMP: 64183,53760.475 $HOROLOG: 64183,39360
```

La différence (sans compter les secondes fractionnées) est de 14400 secondes. Mon $HOROLOG est donc quatre heures "derrière" $ZTIMESTAMP.

Comment convertir le format interne en format d'affichage

Vous pouvez utiliser $ZDATETIME. La conversion de la date et de l'heure actuelles à partir du format interne peut être aussi simple que suit

```
WRITE !, "Avec le format de date et d'heure par défaut : ",$ZDATETIME($HOROLOG)

Avec le format de date et d'heure par défaut : 09/22/2016 10:56:00
```

Cela prend les paramètres par défaut des paramètres locaux Caché et NLS.

Les deuxième et troisième arguments (facultatifs) servent à spécifier le format de la date et le format de l'heure.

```
WRITE !, "With dformat 5 and tformat 5: ", $ZDATETIME($HOROLOG,5,5)

Avec dformat 5 et tformat 5: Sep 22, 2016T10:56:00-04:00
```

Le format horaire 7, par exemple, affiche l'heure en temps universel coordonné comme vous le voyez ici.

```
WRITE !, "With dformat 5 and tformat 7: ", $ZDATETIME($HOROLOG,5,7)

Avec dformat 5 et tformat 7: Sep 22, 2016T14:56:00Z
```

Outre les formats de date et d'heure, il existe de nombreux autres arguments facultatifs qui vous permettent de contrôler l'affichage. Par exemple, vous pouvez

- Spécifier les limites inférieure et supérieure des dates valides si elles sont également autorisées par les paramètres régionaux actuels.
- Contrôler si les années sont affichées avec deux ou quatre chiffres.
- Contrôler l'affichage des erreurs.

Comment convertir un format d'affichage en un format interne à Caché

Utilisez $ZDATETIMEH pour passer du format d'affichage au format interne, comme $HOROLOG. Le H à la fin est un rappel que vous finirez avec le format $HOROLOG. De nombreux formats d'entrée différents peuvent être utilisés.

```
SET display = "09/19/2016 05:05 PM"
  WRITE !, display_" is "_$ZDATETIMEH(display)_" in internal format"
  WRITE !, "Suffixes AM, PM, NOON, MIDNIGHT can be used"
  SET startdate = "12/31/1840 12:00 MIDNIGHT"
  WRITE !, startdate_" is "_$ZDATETIMEH(startdate)_" in internal format"


09/19/2016 05:05 PM est 64180,61500 en format interne

Les suffixes AM, PM, NOON, MIDNIGHT peuvent être utilisés

12/31/1840 12:00 MIDNIGHT est 0,0 en format interne
```

Comment convertir l'heure UTC en heure locale et vice versa au format interne

Vous pouvez utiliser $ZDATETIME et $ZDATETIMEH avec un spécificateur spécial de format de date (-3) pour le

deuxième argument.

La meilleure façon de convertir l'heure UTC en heure locale au format interne de Caché est d'utiliser la fonction $ZDATETIMEH(datetime, -3). Ici, le premier argument contient l'heure UTC au format interne.

```
SET utc1 = $ZTIMESTAMP
  SET loctime1 = $ZDATETIMEH(utc1, -3)
  WRITE !, "$ZTIMESTAMP returns a UTC time in internal format: ", utc1
  WRITE !, "$ZDATETIMEH( ts,-3) converts UTC to local time: ", loctime1
  WRITE !, "$ZDATETIME converts this to display formats: ", $ZDATETIME(utc1)
  WRITE !, "which is "_$ZDATETIME(loctime1)_" in local time"

$ZTIMESTAMP renvoie une heure UTC au format interne : 64183,53760.475

$ZDATETIMEH( ts,-3) convertit l'UTC en heure locale : 64183,39360.475

$ZDATETIME le convertit en format d'affichage : 09/22/2016 14:56:00

qui est 09/22/2016 10:56:00 en heure locale
```

Si vous avez besoin de passer de l'heure locale à UTC, toujours au format interne, utilisez $ZDATETIME(datetime, -3). Ici, le paramètre datetime contient l'heure reflétant votre fuseau horaire local au format interne.

```
SET loctime2 = $HOROLOG
  SET utc2 = $ZDATETIME(loctime2, -3)
  WRITE !, "$HOROLOG returns a local time in internal format: ", loctime2
  WRITE !, "$ZDATETIME(ts, -3) converts this to UTC: ", utc2
  WRITE !, "$ZDATETIME converts this to display formats:"
  WRITE !, "Local: ", $ZDATETIME(loctime2)
  WRITE !, "UTC: ", $ZDATETIME(utc2)

$HOROLOG renvoie une heure locale au format interne : 64183,39360

$ZDATETIME(ts, -3) le convertit en UTC : 64183,53760

$ZDATETIME le convertit en format d'affichage :

Local: 09/22/2016 10:56:00

UTC: 09/22/2016 14:56:00
```

Gardez ces points à l'esprit lorsque vous effectuez des conversions d'heure locale et UTC :

- Les conversions entre l'heure locale et l'UTC doivent utiliser les règles de fuseau horaire en vigueur pour la date et le lieu spécifiés. Caché dépend du système d'exploitation pour suivre ces changements au cours du temps. Si le système d'exploitation ne le fait pas correctement, les conversions ne seront pas correctes.
- Les conversions de dates et d'heures futures utilisent les règles actuelles gérées par le système d'exploitation. Cependant, les règles pour les années futures peuvent changer.

Comment déterminer le fuseau horaire du système

Vous pouvez obtenir le décalage du fuseau horaire actuel en examinant la valeur de $ZTIMEZONE ou de %SYSTEM.SYS.TimeZone(). La valeur par défaut est définie par le système d'exploitation.

```
WRITE !, "$ZTIMEZONE is set to "_$ZTIMEZONE
 WRITE !, "%SYSTEM.SYS.TimeZone() returns "_$System.SYS.TimeZone()

$ZTIMEZONE est réglé sur 300

%SYSTEM.SYS.TimeZone() renvoie 300
```

Vous ne devez pas modifier la valeur de $ZTIMEZONE. Si vous le faites, vous affecterez les résultats de IsDST(), $ZDATETIME, et $ZDATETIMEH, parmi de nombreux autres effets. L'heure pour le processus ne changera pas l'heure correctement pour l'heure d'été. La modification de $ZTIMEZONE n'est pas un moyen cohérent de changer le fuseau horaire utilisé par Caché.

À partir de la version 2016.1, Caché fournit la méthode $System.Process.TimeZone() qui vous permet de définir et de récupérer le fuseau horaire pour un processus spécifique en utilisant la variable d'environnement TZ. Elle renvoie -1 si TZ n'est pas défini.

```
WRITE !,$System.Process.TimeZone()
WRITE !, "Current Time: "_$ZDT($H)
WRITE !, "Set Central Time"
DO $System.Process.TimeZone("CST6CDT")
WRITE !, "New current time: "_$ZDT($H)
WRITE !, "Current Time Zone: "_$System.Process.TimeZone()


-1

L'heure actuelle : 10/03/2016 15:46:04

Réglage de l'heure centrale

Nouvelle heure actuelle : 10/03/2016 14:46:04

Le fuseau horaire actuel : CST6CDT
```

Comment déterminer si l'heure d'été est en vigueur

Utilisez $SYSTEM.Util.IsDST(). Ici aussi, Caché s'appuie sur le système d'exploitation pour appliquer les règles correctes permettant de déterminer si l'heure d'été est en vigueur.

```
SET dst = $System.Util.IsDST()
  IF (dst = 1) {WRITE !, "DST is in effect"}
  ELSEIF (dst = 0) { WRITE !, "DST is not in effect" }
  ELSE { WRITE !, "DST cannot be determined" }
```

Comment effectuer l'arithmétique des dates

Puisque le format interne de Caché maintient un compte des jours et un compte des secondes dans chaque jour, vous pouvez faire de l'arithmétique de date d'une manière directe. La fonction $PIECE vous permet de séparer les parties date et heure du format interne.

Voici une courte routine qui utilise $ZDATE et $ZDATEH pour déterminer le dernier jour de l'année dernière afin de pouvoir compter le jour de l'année d'aujourd'hui. Cette routine utilise les méthodes de la classe %SYS.NLS pour définir le format de date que nous voulons, obtenir le séparateur de date et rétablir les valeurs par défaut.

```
DATECALC ; Exemple d'arithmétique de date.
  W !, "Extracting date and time from $H using $PIECE"
  W !, "-----------------------------------------"
  set curtime = $H
  set today = $PIECE(curtime,",",1)
  set now = $PIECE(curtime,",",2)
  W !, "Curtime: "_curtime_" Today: "_today_" Now: "_now

  W !, "Counting the days of the year"
  W !, "----------------------------"
  ; set to US format
  SET rtn = ##class(%SYS.NLS.Format).SetFormatItem("DateFormat",1)
  set sep = ##class(%SYS.NLS.Format).GetFormatItem("DateSeparator")
  SET lastyear = ($PIECE($ZDATE($H),sep,3) - 1)
  SET start = $ZDATEH("12/31/"_lastyear)
  W !, "Today is day "_(today - start)_" of the year"
  ; put back the original date format
  SET rtn=##class(%SYS.NLS.Format).SetFormatItem("DateFormat",rtn)
```

Comment obtenir et définir d'autres paramètres NLS

Utilisez la classe %SYS.NLS.Format pour des paramètres tels que le format de la date, les dates maximum et minimum et d'autres paramètres. Les paramètres initiaux proviennent des paramètres régionaux actuels et les modifications que vous apportez à cette classe n'affectent que le processus actuel.

Heure et date en SQL

Caché fournit une variété de fonctions SQL pour travailler avec les dates et les heures. Celles-ci sont également disponibles en ObjectScript via la classe $System.SQL.

TODATE : Convertit une date au format CHAR ou VARCHAR2 en une date. Ceci est disponible en ObjectScript en utilisant $System.SQL.TODATE("string", "format")

DAYOFYEAR : Renvoie le jour de l'année pour une expression d'année donnée, qui peut être dans plusieurs formats, comme un entier de date de $HOROLOG.

DAYNAME : renvoie le nom du jour qui correspond à une date spécifiée.

```
W $ZDT($H)

10/12/2016 11:39:19


w $System.SQL.TODATE("2016-10-12","YYYY-MM-DD")

64203


W $System.SQL.DAYOFYEAR(+$H)

286


W $System.SQL.DAYNAME(+$H)
```

```
Wednesday
```

Il y a beaucoup plus d'informations dans la documentation de Cache' sur la façon d'utiliser ces fonctions (et bien d'autres). Référez-vous aux références de la section suivante.

Références

Liens vers la documentation en ligne d'InterSystems pour les éléments abordés dans le présent article.

$HOROLOG

$PIECE

SQL Functions reference

%SYS.NLS.Format

%SYSTEM.Process.TimeZone()

%SYSTEM.SQL

%SYSTEM.SYS.Horolog

%SYSTEM.Util.IsDST()

$ZDATE

$ZDATEH

$ZDATETIME

$ZDATETIMEH

#Administration du système #ObjectScript #Caché