Dates, Times and Time Zones

Last Modified on 07/06/2019 12:50 am AEST

Date and Time Data Types

ReadiNow supports three related data types:

- Date only
- Time only
- Date and Time

The date-time data type is time zone aware. The date-only and time-only data types are not time zone aware.

Date only type

A date-only field holds a year, month and day. Date only fields are not time zone aware, which is to say that if a record has a particular date stored in a date-only field, then all ReadiNow users will see the same year, month and day for that record, irrespective of what time zone was active when the record was created, or what time zone they are in when viewing the record.

Calculations and sub calculations can also return date-only values, and they are similarly (mostly) time zone unaware. For example, a calculation may refer to a date-only field on a record.

A constant date-only value can be added to a calculation using the ISO format: #yyyy-mm-dd#. For example:

```
[Review date] > #2015-10-21#
```

The one manner in which date-only data type is aware is when using the getdate() function. This will return a today's date, according to the current user's time zone, as date-only value.

```
getdate()
```

Time only type

A time-only field works in a similar manner, holding a hour, minute, and second. Note that the second value is recorded, even if it is often not visibly presented. This can cause some confusion with time calculations that perform 'equals' comparisons.

Time only fields and calculations are similarly not time zone aware, and all users will see the same hour, minute and second, regardless of their local time zone. (Time zone adjustments are not possible without a date in order to determine daylight savings offsets).

A constant time-only value can be added to a calculation using the ISO format: #hh:mm# or #hh:mm:ss#, where the hour is expressed in 24 hour time.

```
[Start time] < #17:15#
```

Time-only is similarly time-zone aware when the gettime() function is used. This function will return the current time, according to the current user's time zone, as a time-only value.

```
gettime()
```

Date time type

A date-time field represents a specific moment in history. It has the appearance of holding a date and a time. But it is time-zone aware, and its value will always be presented using the local time zone of each user. It does not, however, store a timezone in itself. Time zones are discussed below.

A constant date-time value can be added to a calculation using the following formats, all equivalently giving a result of 31st December 2019, 1:30pm and 59 seconds.

```
#2019-12-31T13:30:59#   -- ISO format
```

```
#2019-12-31 13:30:59#   -- 31st December 2019, 1:30:59pm
```

```
#2019-12-31 1:30 PM#    -- 31st December 2019, 1:30:59pm
```

The getdatetime function always returns the current moment in time as a date-time value, which is then always presented in the user's local time in the same manner as any other date-time value.

```
getdatetime()
```

Time zones and the date-time type

When a date-time field value is entered into a record through a form, or some other manner, the date and time numbers entered are understood to be in the local time zone region of the current user. When the record is then later used on a form or report, the values are then converted back to the user's local time zone region.

However, a different user viewing the same record from a different time-zone region will see different values - the same moment in time, but represented according to their time-zone region.

The general principle, however, is that any one user (operating within one time zone) will always see a consistent view of date-time record data, throughout the year, in their own local time zone.

Internally stored as UTC

The ReadiNow platform internally converts any date-time values entered to UTC (GMT) date-times for storage and processing. When the record is then later used on a form or report, the values are then converted back to the user's local time zone region.

Note that time-zone region, rather than time-zone offset, is used because different regions have different historical timings for the start and end of daylight saving. When a date-time value is being converted to local time, the date in that date-time value is used to determine whether daylight time is used (not whether the present time is currently in daylight saving). This similarly ensures a consistent view of the data year around.

In this manner:

- a user who enters a date-time value into a record, will see the same date-time value for that record throughout the platform. (assuming they do not change the time-zone of their computer).
- and will continue to see the same value presented throughout the year, even as their local offset adjusts for daylight saving.
- a different user who views the date-time record from a different time zone, will see the same moment in time - but converted to their own time zone.

For example:

- if a date-time value of 10am on 6/6/2019 is set on a record by a user in Sydney (GMT+10)
- then a user in California (GMT-7) viewing the same record will see a value of: 5pm on 5/6/2019
- six months later, the Sydney user will still see that record showing 10am on 6/6/2019.

Determining the time zone

All calculations are run in the context of a time zone.

Web Browsers

For calculations that are performed in the web user interface, the time-zone region information provided by the user's operating system and browser is used to determine the time-zone offsets.

Default Time Zone Region and Scheduled Workflows

For calculations that are performed in a workflow that is run on a schedule, or similarly triggered by a background operation, the time zone region is based on the Time Zone setting, located under Administration / Settings / General Settings.

Note that the general settings page shows an hours offset. However, it is the region, not the present offset, that is recorded as the setting, for the reasons above.

Date and Time Calculations

General principles

The following general principles guide the treatment of time zones for date-time values (but not date-only, nor time-only) in calculations:

1. whenever a date-time value gets created from individual parts, those parts are understood to be in the local time zone.
2. whenever a date-time value gets dissolved into its parts, or the parts shown, those parts will represent the local time zone.
3. whenever a function could be impacted by the time zone, that function is performed in local time.

Examples

1. Principle 1 - the following are all understood to be 9:30am on 30th June 2019,*in local time:*
    - #2019-06-30T09:30:00#
    - datetimefromparts(2019, 6, 30, 9, 30, 0)
    - convert(datetime, '2015-07-27 09:30:00')
2. Principle 2 examples:
    1. hour(getdatetime()) returns the current hour as a number, according to local time
    2. 'The time is ' + getdatetime() creates a string that includes the current date time, according to local time.
    3. datename(month, getdatetime()) returns the name of the current month, according to local time
3. Principle 3 example:
    1. dateadd(month, 1, getdatetime()) the result of adding one month can be slightly different depending on the time zone.

Impact of time zone on various date and time functions

getdate	returns the current date, according to the local time zone, as a date-only value.
gettime	returns the current time, according to the local time zone, as a time-only value.
getdatetime	returns the current moment, as a date-time value.
year	gets the year number of a date-only or date-time value (for the latter, according to the local time zone)
month	gets the month number from 1 to 12 of a date-only or date-time value (for the latter, according to the local time zone)
day	gets the day number from 1 to 31 of a date-only or date-time value (for the latter, according to the local time zone)

hour	gets the hour number from 0 to 23 of a time-only or date-time value (for the latter, according to the local time zone)
minute	gets the minute number from 0 to 59 of a time-only or date-time value (for the latter, according to the local time zone)
second	gets the second number from 0 to 59 of a time-only or date-time value (for the latter, according to the local time zone)
quarter	gets the quarter number from 1 to 4 of a date-only or date-time value (for the latter, according to the local time zone)
dayofyear	gets the day of the year from 1 to 366 of a date-only or date-time value (for the latter, according to the local time zone)
week	gets the week number of the year from 1 to 53 of a date-only or date-time value (for the latter, according to the local time zone)
weekday	gets the day of the week as a number from Sunday=1 to Saturday=7 of a date-only or date-time value (for the latter, according to the local time zone)
datefromparts	composes a date-only value from a year, month, and day, number, without regard to time-zone.
timefromparts	composes a time-only value from a hour, minute, and second number, without regard to time-zone.
datetimefromparts	composes a date-time value from year, month, day, hour, minute, and second numbers, interpreting those numbers

	with regard to the current time zone.
dateadd	add (or subtracts) a specified number of units to a date-only or date-time value. The former is performed without regard to time-zone. The latter is performed in the context of the current time zone.
datediff	calculates the difference between two date-only values, or two date-time values. The former is performed without regard to time-zone. The latter is performed in the context of the current time zone.
datename	returns a part of a date-only or date-time value as text (such as the month name). The former is performed without regard to time-zone. The latter is performed in the context of the current time zone.

Impact of time zone on conversion functions

A date-time can be implicitly converted to:

- a date-only
- a time-only
- text

In each case, the date-time is converted to local time, and the local components of the date and/or time are used to make up the new value.

A date-only can be implicitly converted to:

- a date-time - in which case, the date-time is constructed by taking midnight, according to the local time zone, of the date-only value and using that as the date-time value.
- text - in which case the time zone takes has no effect.

A time-only can be implicitly converted to text, and similarly has no effect.

A text string can be explicitly converted to:

- a date-time - in which case the text is processed as though it is representing a local date-time.
- a date-only - in which case, the numerical values of year, month, date are used directly, irrespective of time zone
- a date-only - in which case, the numerical values of hour, minute, second are used directly, irrespective of time zone