Oracle Dates and Times

Overview

Oracle supports both date and time, albeit differently from the SQL2 standard. Rather than using two separate entities, date and time, Oracle only uses one, `DATE`. The `DATE` type is stored in a special internal format that includes not just the month, day, and year, but also
the hour, minute, and second.

The `DATE` type is used in the same way as other built-in types such as `INT`. For example, the following SQL statement creates a relation with an attribute of type `DATE`:

```
create table x(a int, b date);
```

`DATE` Format

When a `DATE` value is displayed, Oracle must first convert that value from the special internal format to a printable string. The conversion is done by a function `TO_CHAR`, according to a `DATE` *format*. Oracle's default format for `DATE` is "`DD-MON-YY`". Therefore, when you issue the query

```
select b from x;
```

you will see something like:

```
B
---------
01-APR-98
```

Whenever a `DATE` value is displayed, Oracle will call `TO_CHAR` automatically with the default `DATE` format. However, you may override the default behavior by calling `TO_CHAR` explicitly with your own `DATE` format. For example,

```
SELECT TO_CHAR(b, 'YYYY/MM/DD') AS b
FROM x;
```

returns the result:

```
B
---------------------------------------------------------------------------
1998/04/01
```

The general usage of `TO_CHAR` is:

```
TO_CHAR(<date>, '<format>')
```

where the `<format>` string can be formed from over 40 options. Some of the more popular ones include:

, for example.

`MM`	Numeric month (*e.g.*, `07`)
`MON`	Abbreviated month name (*e.g.*, `JUL`)
`MONTH`	Full month name (*e.g.*, `JULY`)
`DD`	Day of month (*e.g.*, `24`)
`DY`	Abbreviated name of day (*e.g.*, `FRI`)
`YYYY`	4-digit year (*e.g.*, `1998`)
`YY`	Last 2 digits of the year (*e.g.*, `98`)
`RR`	Like `YY`, but the two digits are ``rounded" to a year in the range 1950 to 2049. Thus, `06` is considered `2006` instead of `1906`
`AM` (or `PM`)	Meridian indicator
`HH`	Hour of day (`1-12`)
`HH24`	Hour of day (`0-23`)
`MI`	Minute (`0-59`)
`SS`	Second (`0-59`)

You have just learned how to output a `DATE` value using `TO_CHAR`. Now what about inputting a `DATE` value? This is done through a function called `TO_DATE`, which converts a string to a `DATE` value, again according to the `DATE` format. Normally, you do not have to call `TO_DATE` explicitly: Whenever Oracle expects a `DATE` value, it will automatically convert your input string using `TO_DATE` according to the default `DATE` format "`DD-MON-YY`". For example, to insert a tuple with a `DATE` attribute, you can simply type:

```
insert into x values(99, '31-may-98');
```

Alternatively, you may use `TO_DATE` explicitly:

```
insert into x
values(99, to_date('1998/05/31:12:00:00AM', 'yyyy/mm/dd:hh:mi:ssam'));
```

The general usage of `TO_DATE` is:

```
TO_DATE(<string>, '<format>')
```

where the `<format>` string has the same options as in `TO_CHAR`.

Finally, you can change the default `DATE` format of Oracle from "`DD-MON-YY`" to something you like by issuing the following command in `sqlplus`:

```
alter session set NLS_DATE_FORMAT='<my_format>';
```

The change is only valid for the current `sqlplus` session.

The Current Time

The built-in function `SYSDATE` returns a `DATE` value containing the current date and time on your system. For example,

```
select to_char(sysdate, 'Dy DD-Mon-YYYY HH24:MI:SS') as "Current Time"
from dual;
```

returns

```
Current Time
---------------------------------------------------------------------------
Tue 21-Apr-1998 21:18:27
```

which is the time when I was preparing this document `:-)` Two interesting things to note here:

- You can use double quotes to make names case sensitive (by default, SQL is case insensitive), or to force spaces into names. Oracle will treat everything inside the double quotes literally as a single name. In this example, if `"Current Time"` is not quoted, it would have been interpreted as *two* case insensitive names `CURRENT` and `TIME`, which would actually cause a syntax error.
- `DUAL` is built-in relation in Oracle which serves as a dummy relation to put in the `FROM` clause when nothing else is appropriate. For example, try "`select 1+2 from dual;`".

Another name for the built-in function `SYSDATE` is `CURRENT_DATE`. Be aware of these special names to avoid name conflicts.

Operations on `DATE`

You can compare `DATE` values using the standard comparison operators such as `=`, `!=`, `>`, *etc.*

You can subtract two `DATE` values, and the result is a `FLOAT` which is the number of days between the two `DATE` values. In general, the result may contain a fraction because `DATE` also has a time component. For obvious reasons, adding, multiplying, and dividing two `DATE` values are not allowed.

You can add and subtract constants to and from a `DATE` value, and these numbers will be interpreted as numbers of days. For example, `SYSDATE+1` will be tomorrow. You cannot multiply or divide `DATE` values.

With the help of `TO_CHAR`, string operations can be used on `DATE` values as well. For example, `to_char(<date>, 'DD-MON-YY') like '%JUN%'` evaluates to true if `<date>` is in June.

This document was written originally by Kristian Widjaja for Prof. Jeff Ullman's CS145 class in Autumn, 1997; revised by Jun Yang for Prof. Jennifer Widom's CS145 class in Spring, 1998; further revisions by Prof. Ullman in Autumn, 1998.