



Another new datatype are Intervals, these are a flexible ways of storing differences between two time and date values.

Two ranges are possible for intervals ... Year to Month and Day to Second, these are mutually exclusive and therefore cannot be defined together, if that sort of precision is required then two individual columns will need to be defined.

```
create table test_interval  
(col1 interval year to month  
,col2 interval day to second);
```



The first part of the definition datatype is termed the leading field and the second the trailing field ..

The precision of both Year and Day leading fields can be specified

```
create table test_interval  
(col1 interval year(2) to month  
,col2 interval day(6) to second);
```



However only the 'second' trailing field can have a precision amount ...

```
create table test_interval  
(col1 interval year(2) to month  
,col2 interval day(6) to second(2));
```

The default value for leading fields is 2, for second trailing fields is 6.



A Year to Month interval effectively requires two values representing the Year and the Month difference ... this is inserted in the following format ...

```
insert into <table> values ('aa-bb');
```

Where aa represents the year and bb the month, note the hyphen separating the two values

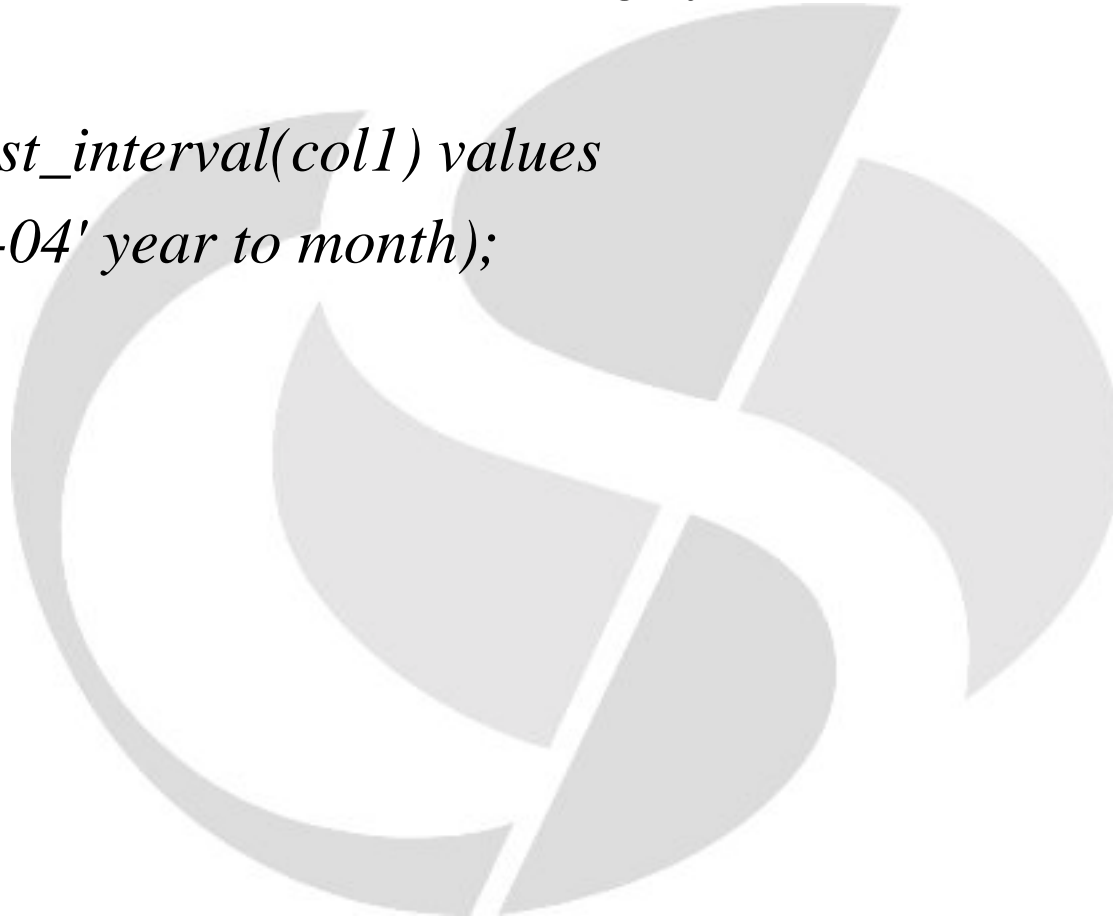
To specify an interval of six years, ten months the following is used

```
insert into test_interval values ('06-10');
```



The alternative is to use the following syntax

```
insert into test_interval(col1) values  
(interval '03-04' year to month);
```





A Day to Second interval requires sufficient values for the Day, Hour, Minute and Second portions of the datetime

```
insert into <table> values ('aaa bb:cc:dd.ee');
```

Where aaa are the days, bb the hours, cc the minutes, dd the seconds and ee the fractions of a second, note the space between days and hours, colons between hours, minutes and seconds and the period between seconds and fractions of a second



To specify an interval of 1 day, 12 hours, 10 minutes and 3 seconds the following is used

```
insert into test_interval(col2) values ('1 12:10:03');
```

Alternatively ...

```
insert into test_interval(col2) values  
(interval '1 12:10:03' day to second);
```

can be used.



Intervals can be used in a variety of ways ... in the following example both columns in the test\_interval table are applied to sysdate ..

*insert into test\_interval values ('06-10', '10 12:17:11') - Todays date was 19-May-2003*

*select to\_char(sysdate + col1, 'DD-MON-RRRR HH24:MI:SS')  
,to\_char(sysdate + col2, 'DD-MON-RRRR HH24:MI:SS') from test\_interval;*

*TO\_CHAR(SYSDATE+COL1*

*TO\_CHAR(SYSDATE+COL2*

*-----*

*-----*

*19-MAR-2010 11:01:56*

*29-MAY-2003 23:19:07*



Negative values can also be assigned to the leading field ...

```
insert into test_interval values ('-06-10', '-10 12:17:11');
```

```
TO_CHAR(SYSDATE+COL1
```

```
TO_CHAR(SYSDATE+COL2
```

-----

-----

```
19-MAR-2010 11:11:19
```

```
29-MAY-2003 23:28:30
```

```
19-JUL-1996 11:11:19
```

```
08-MAY-2003 22:54:08
```



Four new conversion functions have been created for use with intervals, these convert varchar2 and number datatypes to either Year to Month (YM) or Day to Second (DS)

For converting varchar2 use either to\_ymininterval or to\_dsinterval

```
select * from test_interval where col1 = to_ymininterval('06-10')
```

```
select * from test_interval where col2 = to_dsinterval('100 12:11:20')
```

# 9i Date Changes - Using Intervals



Converting number values to interval requires the Developer to specify the subtype the number represents ... Year, month etc ...

Use `numtodsinterval` or `numtoyminterval` ...

```
insert into test_interval values  
(numtoyminterval(12,'year')  
, numtodsinterval(16,'day'))
```

All other subtype values are set to zero

```
COL1  
-----  
+12-00
```

```
COL2  
-----  
+000016 00:00:00.00
```

# 9i Date Changes - Using Intervals



This method can also be used when updating intervals ..

```
update test_interval  
set col1 = col1 + numtoyminterval(05, 'month')  
where col1 = '12-00';
```

*COL1*

*COL2*

-----  
*+12-05*

-----  
*+000016 00:00:00.00*

# 9i Date Changes - Using Intervals



Interestingly, although Interval month can only contain values from 0 to 12 if a larger amount is specified when using numtoyminterval then the year amount is added to (this applies to numtodsinterval as well)

```
update test_interval  
set col1 = col1 + numtoyminterval(15, 'month')  
where col1 = '12-05';
```

<i>COL1</i>	<i>COL2</i>
+13-08	+000016 00:00:00.00