Oracle Cheatsheet

Contents

[Oracle Cheatsheet 1](#_Toc443547295)

[Case Statements 3](#_Toc443547296)

[Dates and Times 3](#_Toc443547297)

[Convert a string to a Timestamp 3](#_Toc443547298)

[Convert a timestamp to ISO 8601 3](#_Toc443547299)

[Convert from UTC to a different timezone 3](#_Toc443547300)

[Timezones 3](#_Toc443547301)

[Difference between 2 Timestamps in minutes 5](#_Toc443547302)

[Difference between 2 Timestamps in seconds 5](#_Toc443547303)

[Format Elements for Dates 5](#_Toc443547304)

[Timezones 7](#_Toc443547305)

[DDL (Data Description Language) 7](#_Toc443547306)

[Examine indexes on tables in the database 7](#_Toc443547307)

[Functions 8](#_Toc443547308)

[Creating a Function 8](#_Toc443547309)

[Geodesy 9](#_Toc443547310)

[Create an SDO Geometry Point 9](#_Toc443547311)

[Distance between two points 9](#_Toc443547312)

[Extract latitude and longitude from an SDO.Geometry point 9](#_Toc443547313)

[Extract latitude and longitude from an SDO.Geometry polygon 9](#_Toc443547314)

[JSON 10](#_Toc443547315)

[Parsing JSON 10](#_Toc443547316)

[Lead / Lag Queries 10](#_Toc443547317)

[Math 11](#_Toc443547318)

[Average 11](#_Toc443547319)

[Half-round even 11](#_Toc443547320)

[Median 11](#_Toc443547321)

[Rounding 11](#_Toc443547322)

[Numbers 12](#_Toc443547323)

[Converting a number to a character string 12](#_Toc443547324)

[Converting ASCII-coded hex to a number 12](#_Toc443547325)

[Right pad with zeros 12](#_Toc443547326)

[Suppress leading spaces for numbers converted to a string 13](#_Toc443547327)

[Regular Expressions 14](#_Toc443547328)

[Replace unwanted characters 14](#_Toc443547329)

[Constrain a character field to numeric values 14](#_Toc443547330)

[Specify a schema 14](#_Toc443547331)

[Statistics 14](#_Toc443547332)

[Average 14](#_Toc443547333)

[Standard Deviation 14](#_Toc443547334)

[Standard Deviation of a Sample 14](#_Toc443547335)

[Strings 15](#_Toc443547336)

[Finding a position of a sub-string 15](#_Toc443547337)

# Case Statements

SELECT table\_name,

CASE owner

WHEN 'SYS' THEN 'The owner is SYS'

WHEN 'SYSTEM' THEN 'The owner is SYSTEM'

ELSE 'The owner is another value'

END

FROM all\_tables;

SELECT table\_name,

CASE

WHEN owner='SYS' THEN 'The owner is SYS'

WHEN owner='SYSTEM' THEN 'The owner is SYSTEM'

ELSE 'The owner is another value'

END

FROM all\_tables;

# Dates and Times

## Convert a string to a Timestamp

TO\_TIMESTAMP('10-SEP-0214:10:10.123000','DD-MON-RRHH24:MI:SS.FF')

-----------------------------------------------------------------------

10-SEP-02 02.10.10.123000000 PM

## Convert a timestamp to ISO 8601

select to\_char(orig\_dep\_dt\_loc\_tz, 'YYYY-MM-DD"T"HH24:MI:SSTZH:TZM')

2015-10-14T20:54:00-05:00

## Convert from UTC to a different timezone

select from\_tz(to\_timestamp('15-OCT-15 01.54.00.000000000 AM',

'dd-mon-yy HH.MI.SS.FF9 AM'),

'UTC')

at time zone 'America/Mexico\_City' from dual;

### Timezones

<https://docs.oracle.com/cd/B13866_04/webconf.904/b10877/timezone.htm>

## Difference between 2 Timestamps in minutes

SELECT (extract(DAY FROM time2-time1)\*24\*60)+

(extract(HOUR FROM time2-time1)\*60)+

(extract(MINUTE FROM time2-time1))

## Difference between 2 Timestamps in seconds

SELECT (extract(DAY FROM time2-time1)\*24\*60\*60)+

(extract(HOUR FROM time2-time1)\*60\*60)+

(extract(MINUTE FROM time2-time1)\*60)+

extract(SECOND FROM time2-time1)

## Format Elements for Dates

***Table 2-15 Datetime Format Elements***

| **Element** | **Specify in TO\_\* datetime functions?** | **Description** |
| --- | --- | --- |
| -  /  ,  .  ;  :  "text" | Yes | Punctuation and quoted text is reproduced in the result. |
| AD  A.D. | Yes | AD indicator with or without periods. |
| AM  A.M. | Yes | Meridian indicator with or without periods. |
| BC  B.C. | Yes | BC indicator with or without periods. |
| CC  SCC | No | Century.   * If the last 2 digits of a 4-digit year are between 01 and 99 (inclusive), then the century is one greater than the first 2 digits of that year. * If the last 2 digits of a 4-digit year are 00, then the century is the same as the first 2 digits of that year.   For example, 2002 returns 21; 2000 returns 20. |
| D | Yes | Day of week (1-7). |
| DAY | Yes | Name of day, padded with blanks to display width of the widest name of day in the date language used for this element. |
| DD | Yes | Day of month (1-31). |
| DDD | Yes | Day of year (1-366). |
| DL | Yes | Returns a value in the long date format, which is an extension of Oracle Database's **DATE** format (the current value of the **NLS\_DATE\_FORMAT** parameter). Makes the appearance of the date components (day name, month number, and so forth) depend on the **NLS\_TERRITORY** and **NLS\_LANGUAGE** parameters. For example, in the **AMERICAN\_AMERICA** locale, this is equivalent to specifying the format **'fmDay,** **Month** **dd,** **yyyy'**. In the **GERMAN\_GERMANY** locale, it is equivalent to specifying the format '**fmDay, dd.** **Month yyyy**'.  **Restriction:** You can specify this format only with the **TS** element, separated by white space. |
| DS | Yes | Returns a value in the short date format. Makes the appearance of the date components (day name, month number, and so forth) depend on the **NLS\_TERRITORY** and **NLS\_LANGUAGE** parameters. For example, in the **AMERICAN\_AMERICA** locale, this is equivalent to specifying the format '**MM/DD/RRRR**'. In the **ENGLISH\_UNITED\_KINGDOM** locale, it is equivalent to specifying the format '**DD/MM/RRRR**'.  **Restriction:** You can specify this format only with the **TS** element, separated by white space. |
| DY | Yes | Abbreviated name of day. |
| E | No | Abbreviated era name (Japanese Imperial, ROC Official, and Thai Buddha calendars). |
| EE | No | Full era name (Japanese Imperial, ROC Official, and Thai Buddha calendars). |
| FF [1..9] | Yes | Fractional seconds; no radix character is printed (use the X format element to add the radix character). Use the numbers 1 to 9 after FF to specify the number of digits in the fractional second portion of the datetime value returned. If you do not specify a digit, then Oracle Database uses the precision specified for the datetime datatype or the datatype's default precision.  **Examples:** **'HH:MI:SS.FF'**  **SELECT TO\_CHAR(SYSTIMESTAMP, 'SS.FF3') from dual;** |
| FM | Yes | Returns a value with no leading or trailing blanks.  **See** **Also**: Additional discussion on this [format model modifier](http://docs.oracle.com/cd/B19306_01/server.102/b14200/sql_elements004.htm#SQLRF00216) in the [*Oracle Database SQL Reference*](http://docs.oracle.com/cd/B19306_01/server.102/b14200/toc.htm) |
| FX | Yes | Requires exact matching between the character data and the format model.  **See** **Also**: Additional discussion on this [format model modifier](http://docs.oracle.com/cd/B19306_01/server.102/b14200/sql_elements004.htm#SQLRF00216) in the [*Oracle Database SQL Reference*](http://docs.oracle.com/cd/B19306_01/server.102/b14200/toc.htm) |
| HH | Yes | Hour of day (1-12). |
| HH12 | No | Hour of day (1-12). |
| HH24 | Yes | Hour of day (0-23). |
| IW | No | Week of year (1-52 or 1-53) based on the ISO standard. |
| IYY  IY  I | No | Last 3, 2, or 1 digit(s) of ISO year. |
| IYYY | No | 4-digit year based on the ISO standard. |
| J | Yes | Julian day; the number of days since January 1, 4712 BC. Number specified with J must be integers. |
| MI | Yes | Minute (0-59). |
| MM | Yes | Month (01-12; January = 01). |
| MON | Yes | Abbreviated name of month. |
| MONTH | Yes | Name of month, padded with blanks to display width of the widest name of month in the date language used for this element. |
| PM  P.M. | No | Meridian indicator with or without periods. |
| Q | No | Quarter of year (1, 2, 3, 4; January - March = 1). |
| RM | Yes | Roman numeral month (I-XII; January = I). |
| RR | Yes | Lets you store 20th century dates in the 21st century using only two digits.  **See** **Also:** Additional discussion on [**RR** datetime format element](http://docs.oracle.com/cd/B19306_01/server.102/b14200/sql_elements004.htm#SQLRF00215) in the [*Oracle Database SQL Reference*](http://docs.oracle.com/cd/B19306_01/server.102/b14200/toc.htm) |
| RRRR | Yes | Round year. Accepts either 4-digit or 2-digit input. If 2-digit, provides the same return as RR. If you do not want this functionality, then enter the 4-digit year. |
| SS | Yes | Second (0-59). |
| SSSSS | Yes | Seconds past midnight (0-86399). |
| TS | **Yes** | Returns a value in the short time format. Makes the appearance of the time components (hour, minutes, and so forth) depend on the **NLS\_TERRITORY** and **NLS\_LANGUAGE** initialization parameters.  **Restriction:** You can specify this format only with the **DL** or **DS** element, separated by white space. |
| TZD | Yes | Daylight savings information. The TZD value is an abbreviated time zone string with daylight savings information. It must correspond with the region specified in TZR.  **Example:** **PST** (for US/Pacific standard time); **PDT** (for US/Pacific daylight time). |
| TZH | Yes | Time zone hour. (See **TZM** format element.)  **Example:** **'HH:MI:SS.FFTZH:TZM'**. |
| TZM | Yes | Time zone minute. (See **TZH** format element.)  **Example:** **'HH:MI:SS.FFTZH:TZM'**. |
| TZR | Yes | Time zone region information. The value must be one of the time zone regions supported in the database.  **Example:** US/Pacific |
| WW | No | Week of year (1-53) where week 1 starts on the first day of the year and continues to the seventh day of the year. |
| W | No | Week of month (1-5) where week 1 starts on the first day of the month and ends on the seventh. |
| X | Yes | Local radix character.  **Example:** **'HH:MI:SSXFF'**. |
| Y,YYY | Yes | Year with comma in this position. |
| YEAR  SYEAR | No | Year, spelled out; **S** prefixes BC dates with a minus sign (-). |
| YYYY  SYYYY | Yes | 4-digit year; **S** prefixes BC dates with a minus sign. |
| YYY  YY  Y | Yes | Last 3, 2, or 1 digit(s) of year. |

## Timezones

<https://docs.oracle.com/cd/B13866_04/webconf.904/b10877/timezone.htm>

# DDL (Data Description Language)

## Examine indexes on tables in the database

SELECT user\_tables.table\_name, user\_indexes.index\_name

FROM user\_tables JOIN user\_indexes on user\_indexes.table\_name = user\_tables.table\_name

ORDER by user\_tables.table\_name,user\_indexes.index\_name;

# Functions

## Creating a Function

CREATE OR REPLACE FUNCTION TEST\_HELLO

(

PARAM1 IN VARCHAR2

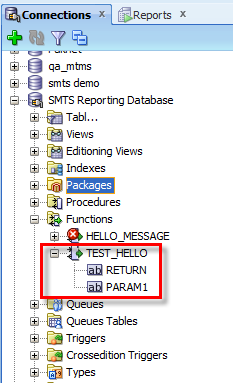
) RETURN VARCHAR2 AS

BEGIN

RETURN 'Hello' || PARAM1;

END TEST\_HELLO;

This creates a function which will exist in the database’s Functions node:



The function can be invoked as follows:

select TEST\_HELLO(msg)

from sgs\_africarpt.feed\_raw;

# Geodesy

## Create an SDO Geometry Point

select SDO\_GEOMETRY(2001, 8307, SDO\_POINT\_TYPE(36.86, -1.29431, NULL), NULL, NULL) from dual;

Here, ‘8307’ indicates geodetic (earth surface).

Distance between a point and a polygon

SDO\_GEOM.SDO\_DISTANCE(SDO\_GEOMETRY(2001, 8307, SDO\_POINT\_TYPE(feed\_lon, feed\_lat, NULL), NULL, NULL),

dest\_FENCE.polygon, 0.005, 'unit=KM') as km\_remaining

## Distance between two points

SELECT SDO\_GEOM.SDO\_DISTANCE(SDO\_GEOMETRY(2001, 8307, SDO\_POINT\_TYPE(36, -1, NULL), NULL, NULL),

SDO\_GEOMETRY(2001, 8307, SDO\_POINT\_TYPE(37, 0, NULL), NULL, NULL), 0.005, 'unit=KM')

FROM dual;



Obviously, units are in KM.

2001 indicates a point

8307 indicates WGS84

**The arguments are longitude first, latitude second**

## Extract latitude and longitude from an SDO.Geometry point

select JLL.LOC.sdo\_point.x as longitude, jll.LOC.sdo\_point.y as latitude

from sgs\_africarpt.journey\_loc\_log JLL;

## Extract latitude and longitude from an SDO.Geometry polygon

select SDO\_GEOM.SDO\_CENTROID(polygon, 0.05).SDO\_POINT.x as longitude,

SDO\_GEOM.SDO\_CENTROID(polygon, 0.05).SDO\_POINT.y as latitude

from sgs\_africarpt.fence

where NAME = 'NAIROBI';

## Extract latitude and longitude from an ST\_GEOM object

round(to\_binary\_float(SDO\_GEOM.SDO\_CENTROID(ANALYTICS.SDO\_GEOM\_FROM\_STGEOM(orig\_polygon), 0.01).SDO\_POINT.x \* 100)) / 100 as orig\_lon,

round(to\_binary\_float(SDO\_GEOM.SDO\_CENTROID(ANALYTICS.SDO\_GEOM\_FROM\_STGEOM(orig\_polygon), 0.01).SDO\_POINT.y \* 100)) / 100 as orig\_lat,

This requires the following function:

create or replace function sdo\_geom\_from\_stgeom(polygon st\_geometry)

return sdo\_geometry is vfence st\_geometry;

begin

if polygon is null then

return null;

else

return polygon.get\_sdo\_geom;

end if;

end;

# 

# JSON

## Parsing JSON

select

json\_ext.pp(json(json\_request), 'algorithmData.currentLocationTime') currentLocationTime,

json\_ext.pp(json(json\_request), 'algorithmData.currentLocation.gps.latitude') latitude,

json\_ext.pp(json(json\_request), 'algorithmData.currentLocation.gps.longitude') longitude,

json\_printer.pretty\_print\_any(json\_ext.get\_json\_value(json(json\_request), 'algorithmData')) formatted\_json\_algorithmData

from PG\_TRKNG\_RPT.JOURNEY\_ETA\_HISTORY

where rownum < 10

# Lead / Lag Queries

select journey\_checkpoint\_id,

journey\_id,

checkpoint\_id,

planned\_arrival\_dt,

seq, actual\_arrival\_dt,

eta\_dt,

departure\_dt,

lag(departure\_dt) over (order by seq) as prev\_departure\_dt

from (select journey\_checkpoint\_id,

journey\_id,

checkpoint\_id,

planned\_arrival\_dt,

seq, actual\_arrival\_dt,

eta\_dt,

departure\_dt

from car\_rpt.journey\_checkpoints jc

where journey\_id = 351330

order by seq

)

order by seq;

# Math

## Average

SELECT AVG(salary) "Average" FROM employees;

Average

--------

6425

## Half-round even

select round(to\_binary\_float(1.035 \* 100)) / 100

from dual;

## Median

select median(abs(hrs\_diff))

from …;

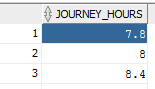
## Rounding

SELECT round(((extract(DAY FROM j.destination\_arrival\_dt - j.ORIGIN\_DEPARTURE\_DT)\*24\*60)+

(extract(HOUR FROM j.destination\_arrival\_dt-j.ORIGIN\_DEPARTURE\_DT)\*60)+

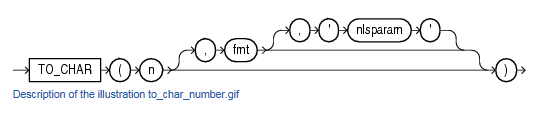
(extract(MINUTE FROM j.destination\_arrival\_dt-j.ORIGIN\_DEPARTURE\_DT))) / 60.0, 1) as journey\_hours

“1” is the number of decimal points.



# Numbers

## Converting a number to a character string



## Converting ASCII-coded hex to a number

select to\_number('4379E350', 'xxxxxxxx') from dual;

Answer: 1132061520

## Right pad with zeros

select to\_char(12.2, 'FM999.90')

from dual;

## Suppress leading spaces for numbers converted to a string

select to\_char(12.23, 'FM999.99'),

to\_char(12.23, '999.99')

from dual;



Use FM format modifier to suppress leading spaces

# Regular Expressions

## Replace unwanted characters

select regexp\_replace('+1.2-3 a', '[^(0-9).-/+]+', '') from dual;

+1.23

Replaces all non-numeric characters with an empty string

## Constrain a character field to numeric values

where regexp\_like(feed\_master.lat\_dms, '^[0123456789]{7}[NS]{1}$')

# Specify a schema

alter session set current\_schema=SGS\_AFRICARPT

# Statistics

## Average

select cmp\_name,

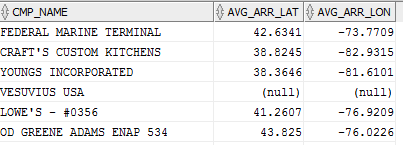
round(**avg**(stops.ariv\_lat), 4) as avg\_arr\_lat,

round(**avg**(stops.ariv\_long), 4) as avg\_arr\_lon

from analytics.mb\_stops stops

group by cmp\_name

;



## Standard Deviation

## Standard Deviation of a Sample

SELECT STDDEV(salary) "Deviation"

FROM employees;

Deviation

----------

3909.36575

# Strings

## Finding a position of a sub-string

INSTR( string, substring [, start\_position [, th\_appearance ] ] )

select instr('test', 's') from dual;

3