# **Oracle Vs Sql Server**

DDL COMMANDS

| Ser | Concept | ORACLE | SQL Server |
|---|---|---|---|
| 1 | Create Database/ Schema | We can create schema(user) only<br>Grant DBA touserName identified by (password); | Wecan create multiple databases in SQL Server<br>1. createDatabase Library<br>2. Go to Object Explorer<br>=>Right Click on Databases<br>=>Select New Database<br>=>Give Database Name<br>=>Select Location for Database files (.MDF and .LDF) |
| 1 | Create user/Schema | Create User username identified by password; | Create schema schema_name; |
| 2 | Create Table | 1. Createtable Book_Issue<br>(<br>BI_ID number(2)primarykey,<br>B_ID number(2)<br>references Book(B_ID),<br>C_ID number(2)<br>references Customer(C_ID),<br>Issue_dt timestamp(0),<br>Return_dt timestamp(0),<br>Remarks varchar(250)<br>);<br>2. Go to Connection in ORACLE SQL Developer<br>=>Expand Schemas<br>=>Right Click on Tables<br>=>Select New Table<br>=>Give Table Name<br>=>Tick on Primary Key Button if it is Primary Key field.<br>Type Column Name, Data Type, Size Not Null, Default and Comment<br>=>Click on + button to add columns<br>=>Save Table | 1. Createtable Book_Issue<br>(<br>BI_ID intprimarykeyidentity (1,1),<br>B_ID intforeignkey (B_ID)references Book(B_ID),<br>C_ID intforeignkey(C_ID)references Customer(C_ID),<br>Issue_dt Datetime,<br>Return_dt Datetime,<br>Remarks varchar(250)<br>)<br>2. Go to Object Explorer in SQL SERVER<br>=>Expand Databases<br>=>Expand Database Objects<br>=>Right Click on Tables<br>=>Select New Table<br>=>Type Column Name, Data Type and Allow Null<br>=>Save Table |
| 3 | Add Column | ALTER TABLE Customer<br>ADD dob DATE,<br>ADD address VARCHAR (250); | ALTERTABLE Customer<br>ADD dob DATE,<br>Address VARCHAR(200) |
| 4 | Drop Column | ALTER TABLE Customer drop column address ; | ALTERTABLE Cust<br>DROPCOLUMN dob,<br>address |
| 5 | Rename Table | RENAME BOOK TO BOOK_AUTHOR; | SP_RENAME Cust, Customer |
| 6 | Rename Column | Alter table customer rename column dob to birth_date; | SP_RENAME'Customer.dob','Birth_date' |
| 7 | Change Field Type | ALTER TABLE Book MODIFY B_Name varchar2(50); | ALTERTABLE CUST<br>ALTERCOLUMN DOB DATE |
| 8 | Truncate Command | Truncate table emp; | Truncate table emp; |
| 9 | Drop Command | Drop table emp; | Drop table emp; |

DRL OR DQL Commands

| Ser | Concept | ORACLE | SQL Server |
|---|---|---|---|
| 12 | Display all the EMPLOYEES information. | SELECT * FROM EMP; | SELECT * FROM EMP |
| 13 | Display DEPTNO, DNAME of all the departments | SELECT DEPTNO,DNAME FROM DEPT; | SELECT DEPTNO,DNAME FROM DEPT |
| 14 | Using Column Alias | SELECT EMPNO AS ECODE, ENAME AS EMPNAME FROM EMP | SELECT EMPNO AS ECODE, ENAME AS EMPNAME FROM EMP |
| 15 | Using Literals in a select statement | SELECT ENAME \|\| ' WORKING AS ' \|\| JOB RESULT FROM EMP | SELECT ENAME + ' WORKING AS ' + JOB RESULT FROM EMP<br><br>**Result : SMITH WORKING AS CLERK** |
| 16 | Mathematical expression in select statement | SELECT 10 + 20 FROM DUAL;<br>SELECT 10 * 20FROM DUAL;<br>SELECT 5/2 FROM DUAL;<br>(Will give result with decimals)<br><br>SELECT 10 + 5 * 6 M1, (10+5)* 6 M2 FROM DUAL; | SELECT 10 + 20-- {FOR ADDITION}<br>SELECT 10 * 20-- {FOR MULTIPLICATION}<br>SELECT 5/2 -- WILL TREAT AS INTEGER (Will removes the decimals)<br>SELECT 5/2.0 -Will give result with decimals<br>SELECT 10 + 5 * 6 M1, (10+5)* 6 M2 |
| 17 | Display all those EMPLOYEES who are working as **CLERKS**. | SELECT ENAME, JOB FROM EMP WHERE JOB = 'CLERK'; | SELECT ENAME, JOB FROM EMP WHERE JOB = 'CLERK' |
| 18 | Display all those EMPLOYEES who are working at Department Number 10 | SELECT * FROM DEPT WHERE DEPTNO = 10;<br>or<br>SELECT * FROM DEPT WHERE DEPTNO = '10'; | SELECT * FROM DEPT WHERE DEPTNO = 10<br>or<br>SELECT * FROM DEPT WHERE DEPTNO = '10' |
| 19 | Display ENAME, JOB of those EMPLOYES who are not working as MANAGER | SELECT ENAME, JOB FROM EMP WHERE JOB != 'MANAGER'; | SELECT ENAME, JOB FROM EMP WHERE JOB != 'MANAGER' |
| 20 | Display ENAME, JOB, and DEPTNO of those EMPLOYEES who are working as CLERK at DEPT NO 20. | SELECT * FROM EMP WHERE JOB = 'CLERK' AND DEPTNO = 20; | SELECT * FROM EMP WHERE JOB = 'CLERK' AND DEPTNO = 20 |
| 21 | Display ENAME, JOB of those EMPLOYEES who are working as CLERK, ANALYST. | SELECT ENAME, JOB FROM EMP WHERE JOB = 'CLERK' OR JOB = 'ANALYST';<br><br>SELECT ENAME, JOB FROM EMP WHERE JOB IN ('CLERK','ANALYST'); | SELECT ENAME, JOB FROM EMP WHERE JOB = 'CLERK' OR JOB = 'ANALYST'<br><br>SELECT ENAME, JOB FROM EMP WHERE JOB IN ('CLERK','ANALYST') |
| 22 | Display EMPNO, ENAME, DEPTNO of those EMPLOYEES who are working at 10, 30 Departments. | SELECT EMPNO, ENAME, DEPTNO FROM EMP WHERE DEPT IN (10,30) | SELECT EMPNO, ENAME, DEPTNO FROM EMP WHERE DEPT IN (10,30) |
| 23 | Display top 3 employees based on salary | Use Dense_rank() | SELECT * FROM EMP WHERE SAL IN ( SELECT DISTINCT TOP 3 SAL FROM EMP ORDER BY SAL DESC)<br>UPDATE TOP(3) FROM EMP SET SAL = SAL + 1000<br>DELETE TOP (3) FROM EMP |

| Ser | Concept | ORACLE | SQL Server |
|---|---|---|---|
| 20 | Display ENAME, JOB, and DEPTNO of those EMPLOYEES who are working as CLERK at DEPT NO 20. | SELECT * FROM EMP WHERE JOB = 'CLERK' AND DEPTNO = 20; | SELECT * FROM EMP WHERE JOB = 'CLERK' AND DEPTNO = 20 |
| 21 | Display ENAME, JOB of those EMPLOYEES who are working as CLERK, ANALYST. | SELECT ENAME, JOB FROM EMP WHERE JOB = 'CLERK' OR JOB = 'ANALYST';<br>SELECT ENAME, JOB FROM EMP WHERE JOB IN ('CLERK','ANALYST'); | SELECT ENAME, JOB FROM EMP WHERE JOB = 'CLERK' OR JOB = 'ANALYST'<br>SELECT ENAME, JOB FROM EMP WHERE JOB IN ('CLERK','ANALYST') |
| 22 | Display EMPNO, ENAME, DEPTNO of those EMPLOYEES who are working at 10, 30 Departments. | SELECT EMPNO, ENAME, DEPTNO FROM EMP WHERE DEPT IN (10,30) | SELECT EMPNO, ENAME, DEPTNO FROM EMP WHERE DEPT IN (10,30) |
| 23 | Display ENAME, SAL of EMPLOYEE SAL Greater than or equal to 2000 & SAL less than or equal to 3000. | SELECT ENAME, SAL FROM EMP WHERE SAL >= 2000 AND SAL <=3000;<br>SELECT ENAME, SAL FROM EMP WHERE SAL BETWEEN 2000 AND 3000; | SELECT ENAME, SAL FROM EMP WHERE SAL >= 2000 AND SAL <=3000<br>SELECT ENAME, SAL FROM EMP WHERE SAL BETWEEN 2000 AND 3000 |
| 24 | Display ENAME, JOB, DEPTNO of those Employees who are not working as CLERK, ANALYST. | SELECT ENAME, JOB, DEPTNO FROM EMP WHERE JOB NOT IN ('CLERK','ANALYST'); | SELECT ENAME, JOB, DEPTNO FROM EMP WHERE JOB NOT IN ('CLERK','ANALYST') |
| 25 | Display all those Employees whose salary is not in a range of 1000 and 3000. | SELECT * FROM EMP WHERE SAL NOT BETWEEN 1000 AND 3000; | SELECT * FROM EMP WHERE SAL NOT BETWEEN 1000 AND 3000 |
| 26 | Display all the Employees who are working as CLERK, ANALYST and SAL is greater than 1000. | SELECT * FROM EMP WHERE JOB = 'CLERK' OR 'ANALYST' AND SAL >1000;<br>(or)<br>SELECT * FROM EMP WHERE JOB IN ('CLERK','ANALYST') AND SAL >1000; | SELECT * FROM EMP WHERE JOB = 'CLERK' OR 'ANALYST' AND SAL >1000<br>(or)<br>SELECT * FROM EMP WHERE JOB IN ('CLERK','ANALYST') AND SAL >1000 |
| 27 | Display ENAME, SAL, ANNUAL SAL of those Employees whose annual salary is in range of 18000and 36000. | SELECT ENAME, SAL, SAL*12 AS ANNUALSAL FROM EMP WHERE SAL*12 BETWEEN 18000 AND 36000; | SELECT ENAME, SAL, SAL*12 AS ANNUALSAL FROM EMP WHERE SAL*12 BETWEEN 18000 AND 36000 |
| 28 | Display ENAME, JOB, SAL, COMM and DEPTNO who are working as SALESMAN at DEPTNO 30 andthere commission is > half of their salary. | SELECT * FROM EMP WHERE JOB = 'SALESMAN' AND DEPTNO = 30 AND COMM >SAL/2; | SELECT * FROM EMP WHERE JOB = 'SALESMAN' AND DEPTNO = 30 AND COMM >SAL/2 |
| 29 | Display all those Employees who are not earning commission. | SELECT * FROM EMP WHERE COMM IS NULL; | SELECT * FROM EMP WHERE COMM IS NULL |
| 30 | Display all those Employees whose are earning commission | SELECT * FROM EMP WHERE COMM IS NOT NULL;<br>SELECT * FROM EMP WHERE NOT COMM IS NULL; | SELECT * FROM EMP WHERE COMM IS NOT NULL<br>SELECT * FROM EMP WHERE NOT COMM IS NULL |

## DML Commands

| | | |
|---|---|---|
| To insert all columns | Insert into emp values(1101,’Arun’,’21-Jul-2018’) | Insert into emp Values(1101,’Arun’,’2018-07-21’) |
| To Insert Specific Columns | Insert into emp(empno,ename) Values(1101,’Ajay’) | Insert into emp(empno,ename) values(1101,’Ajay’),(1102,’Arun’),(1103,’Akil’) |
| To insert data from one table to another table which is already created | Insert into emp_tgt<br>Select * from source | Insert into emp_tgt<br>Select * from emp_src |
| To create the table and populate records from another table | Create table emp_new<br>As select * from scott.emp; | Select * into emp_new from emp; |
| To Update the columns | Update emp set sal=sal+1000<br>Where empno=1010 | Same as Oracle |
| To delete the records | Delete from emp<br>Where sal>1000; | Same as Oracle |
| TCL | Commit(To make change permanent)<br>Rollback(To undo change)<br>Savepoint a;(Create a marker)<br>Rollback to savepoint a; | Begin tran<br>Commit;<br>Rollback tran s;<br>Save tran F; |

## DCL Commands

| | | |
|---|---|---|
| Grant | Same | GRANT SELECT ON EMP TO AHMED<br>GRANT SELECT, DELETE ON DEPT TO AHMED<br>GRANT ALL ON PROD1 TO KHAN<br>GRANT SELECT ON SALGRADE TO AHMED WITH GRANT OPTION<br>GRANT SELECT ON EMP2 TO PUBLIC |
| Revoke | Same | REVOKE SELECT ON EMP FROM AHMED<br>REVOKE SELECT ON SALGRADE FROM AHMED CASCADE |
| Deny | No | DENY PRIVELLAGES/ALL ON OBJECT NAME TO USER1….. PUBLIC |

## Scalar Datatypes

| | | |
|---|---|---|
| Integer Data | Number | INT<br>SMALLINT<br>BIGINT<br>TINYINT<br>DECIMAL<br>NUMERIC<br>MONEY<br>SMALLMONEY |
| REAL Numbers | | FLOAT<br>REAL |

| | | |
|---|---|---|
| Date And Time | Date | DATE<br>DATETIME<br>SMALLDATETIME |
| STRING Datatype | Varchar<br>Varchar2 | CHAR<br>NCHAR<br>VARCHAR<br>NVARCHAR |
| Unstructured DATA | | BINARY<br>VARBINARY<br>IMAGE<br>XML |

DATA DICTIONARY VIEWS(METADATA)

| | | |
|---|---|---|
| To display list of User Defined Tables | SELECT*FROM USER_TABLES;<br>SELECT*FROM USER_OBJECTS WHERE OBJECT_TYPE='TABLE'; | SELECT NAME FROMSYS.TABLES<br>SELECT NAME FROMSYS.SYSOBJECTSWHERE XTYPE ='U'<br>Select * from information_schema.tables; |
| To display list of System Tables | | SELECT NAME FROMSYS.SYSOBJECTSWHERE XTYPE ='S' |
| To Display Column Information | Select column_name from User_Tab_Cols Where table_name='EMP' | select column_name from information_schema.columns where table_name='EMP' |
| To display list of User Defined Views | Select name from user_views | SELECT NAME FROM SYS.SYSOBJECTS WHERE XTYPE = 'V'<br>SELECT NAME FROM SYS.VIEWS<br>SP_HELPTEXT<View_name><br>SP_HELP<View_name><br>SP_DEPENDS<Table_name>/<View_name><br>SP_REFRESHVIEW<View_name><br>Select * from information_schema.views |
| To display STRUCTURE or DEFINITION or METADATA of a Table | Desc< TABLE_NAME > | SP_HELP < TABLE_NAME ><br>SP_HELP'EMP' |
| To Display List of Synonyms | Select * from User_synonyms | Select name,base_object_name from sys.synonyms<br>Sp_help<Synonym_name> |
| To display list of schema | | SP_help <schema_name.emp> |
| To display list of Indexes | Select * from User_Indexes | SP_HELPINDEX<Table_name><br>SELECT NAME FROM SYS.INDEXES;<br>select * from sys.indexes where object_id=(select object_id from sys.objects where name='EMP'); |
| Information about tables | All_tables or Dba_tables | select * from information_schema.tables |
| Information about constraints | User_constraints | Select * from information_schema.table_constraints |

**Constraints**

| | | |
|---|---|---|
| Domain Integrity<br>Entity Integrity<br>Referential Integrity | Not null<br>Check<br>Default<br>Primary key<br>Unique<br>Foreign Key | Not null<br>Check<br>Default<br>Primary key<br>Unique<br>Foreign Key |
| | Same | CREATE TABLE EMP1( ENO INT NOT NULL, ENAME VARCHAR(10) NOT NULL,<br>JOB VARCHAR(15) CHECK (JOB IN ('CLERK','MANAGER','OPERATIONS')),<br>SAL INT CONSTRAINT SAL_CHK CHECK (SAL BETWEEN 15000 AND 20000),<br>DOJ SMALLDATETIME DEFAULT GETDATE(),<br>DNO INT CONSTRAINT DF_DNO DEFAULT 20) |
| | Same | **COLUMN LEVEL**<br>CREATE TABLE STUDENT (RNO INT UNIQUE, SN VARCHAR(10) CONSTRAINT SN_UQ UNIQUE)<br>**TABLE / ENTITY LEVEL**<br>CREATE TABLE STUDENT (RNO INT , SN VARCHAR(10),UNIQUE(RNO) CONSTRAINT SN_UQ UNIQUE(SN)) |
| Composite Unique Key | | CREATE TABLE STUDENT( RNO INT, SNAME VARCHAR(10), CLASS INT, UNIQUE (RNO,CLASS)) |
| Composite Primary Key | | CREATE TABLE STUDENT ( RNO INT, SNAME VARCHAR(10), CLASS INT, PRIMARY KEY (RNO, CLASS)) |
| CREATING MULTIPLE CONSTRAINTS ON A SINGLE COLUMN | | CREATE TABLE EMPLOYEE (<br>ENO INT PRIMARY KEY,<br>ENAME VARCHAR(15) NOT NULL UNIQUE,<br>SAL INT NOT NULL UNIQUE CHECK (SAL BETWEEN 15000 AND 20000)) |

**MATHEMATICAL/NUMBERFUNCTIONS**

| Ser | ORACLE | SQL Server |
|---|---|---|
| 1 | Same as in SQL Server | ABS (NUMBER) – returns unsigned value of a given number (Number – Argument or Parameter.<br>SELECT ABS(-890), ABS(17) |
| 2 | Same as in SQL Server | SQRT (NUMBER) – Returns square root of a given positive number. |
| 3 | Same as in SQL Server | SQUARE (NUMBER) – Returns square of a given value |
| 4 | Same as in SQL Server | POWER ( NUMBER(Base), NUMBER(Exponent)) – It will find the power of a given number.<br>SELECT POWER (2,5) |
| 5 | Same as in SQL Server | SIGN ( NUMBER )<br>Returns 1 if a number is positive<br>Returns -1 if a number is negative<br>Returns 0 if a number is zero |
| 6 | No Concept in ORACLE | PI() – Returns PI value |
| 7 | No Concept in ORACLE | SIN (NUMBER) – By default this function will take input given in radians, hence radians should be |

| | | |
|---|---|---|
| | | converted to degrees by a standard formula is PI ()/180<br>SELECT SIN (30* PI()/180) |
| 8 | Same as in SQL Server | ROUND ( NUMBER, NUMBER , [NUMBER])<br>Argument1, Argument2, Argument3 (optional for TRUNCATE)<br>By default Argument3 is ZERO<br>ROUND(1234.5678,2,1) --- 1234.5600<br>ROUND(5/2.0,0) --- 3.00<br>ROUND(5/2.0,0,1) --- 2 |
| 9 | CEIL(NUMBER) | CEILING(NUMBER) – This function will increment a given number to its nearest integer. Based on any digit in decimal points is greater than zero.<br>CEILING (123.000) ---- 123<br>CEILING (123.010) ---- 124<br>CEILING (123.456) ---- 124 |
| 10 | Same as in SQL Server | FLOOR (NUMBER) – It decreases a nearest integer<br>FLOOR (-123.456) ---- {-124} |
| 11 | SELECT MOD(25,5)FROM DUAL;<br>---- will get the remainder | SELECT 25%5 |
| 12 | SELECT TRUNC(5.92,0)FROM DUAL; --5<br>SELECT TRUNC(5.92,1)FROM DUAL;--5.9<br>SELECT TRUNC(5.92,2)FROM DUAL;--5.92 | No Concept in SQL Server |

**STRING FUNCTIONS**

| Ser | ORACLE | SQL Server |
|---|---|---|
| 1 | Select UPPER('oracle') from dual | Select UPPER('oracle') |
| 2 | Select lower('oracle') from dual | Select lower('oracle') |
| 3 | Select initcap('oracle') from dual | No Concept |
| 4 | LENGTH(TEXT);<br><br>SELECT ENAME, LENGTH(ENAME) FROM EMP WHERE LENGTH(ENAME) >5; | LEN(TEXT)<br><br>SELECT ENAME, LEN(ENAME) FROM EMP WHERE LEN(ENAME) >5 |
| 5 | SELECT SUBSTR('COMPUTER',1,3) FROMDUAL; | LEFT('COMPUTER',3) ---- COM |
| 6 | SELECT SUBSTR('COMPUTER',-3) FROM DUAL; | RIGHT('COMPUTER',3)-----TER |
| 7 | SUBSTR('COMPUTER',4,3); | SUBSTRING ('COMPUTER',4,3)—PUT |
| 8 | LTRIM(STRING); | LTRIM(STRING) |
| 9 | RTRIM(STRING);<br>SELECT rtrim('100%','%') FROM DUAL;<br>Result : 100 | RTRIM(STRING) |
| 10 | No concept in ORACLE | REPLICATE(STRING, NUMBER)<br><br>REPLICATE('COMM',2) |
| 11 | REVERSE(STRING);<br><br>SELECT REVERSE('COMM') | REVERSE(STRING)<br><br>SELECT REVERSE('COMM')--- MMOC |
| 12 | Same as SQL Server | REPLACE (STRING1, STRING2, STRING3)<br><br>REPLACE ('COMPUTER','UT','IR') |
| 13 | No concept in ORACLE | STUFF(STRING1, NUMBER1, NUMBER2, STRING2)<br><br>STUFF('COMPUTER',5,2,'IL') -- COMPILER |
| 14 | INSTR ('COMPUTER','R',1,1) | CHARINDEX(STRING1, STRING2, [NUMBER])<br><br>SELECT CHARINDEX('R','COMPUTER',7) |
| 15 | SELECT LPAD('100',6,'0')FROM DUAL; | No concept in SQL SERVER |

| | | |
|---|---|---|
| | Result : 000100 | |
| 16 | SELECT RPAD('COMPUTER',10,'A') FROM DUAL;<br><br>Result : COMPUTERAA | No concept in SQL SERVER |
| 17 | SELECT TRANSLATE(COMPUTER','PT','NQ') FROM DUAL;<br><br>Result : COMNUQER | No concept in SQL SERVER |

## DATE FUNCTIONS

| Ser | ORACLE | SQL Server |
|---|---|---|
| 1 | SELECT SYSDATE FROM DUAL;<br>--It returns the Current Sys Date 20-Apr-18 | GETDATE()<br>--It returns current system DATE & TIME of the Server |
| 2 | SELECT TO_CHAR(SYSDATE,'DD')FROM DUAL;--23<br>SELECT TO_CHAR(SYSDATE,'MM')FROM DUAL;--04<br>SELECT TO_CHAR(SYSDATE,'YYYY')FROM DUAL;--2018 | SELECTDAY(GETDATE())--------20<br>SELECTMONTH(GETDATE())------ 4<br>SELECTYEAR(GETDATE())-----2018 |
| 3 | SELECT SYSDATE+10 FROM DUAL;<br>SELECT ADD_MONTHS(SYSDATE,10)FROM DUAL;<br>SELECT TO_CHAR(SYSDATE,'YYYY')+10 FROM DUAL; | SELECT DATEADD(DAY,10,GETDATE())<br>SELECT DATEADD(DD,10,GETDATE())<br>SELECT DATEADD(MONTH,10,GETDATE())<br>SELECT DATEADD(MM,10,GETDATE())<br>SELECT DATEADD(YEAR,10,GETDATE())<br>SELECT DATEADD(YY,10,GETDATE()) |
| 4 | SELECT TO_DATE('10-FEB-2018')-TO_DATE('10-FEB-17')FROM DUAL;<br>SELECT MONTHS_BETWEEN('10-FEB-2018','10-FEB-17')FROM DUAL;<br>SELECT TO_CHAR(TO_DATE(SYSDATE),'YYYY')-TO_CHAR(TO_DATE('10-FEB-17'),'YYYY')FROM DUAL; | SELECTDATEDIFF(DAY,'2016-04-20',GETDATE())<br>SELECTDATEDIFF(MONTH,'2016-04-20',GETDATE())<br>SELECTDATEDIFF(YEAR,'2016-04-20',GETDATE()) |
| 5 | SELECT EXTRACT(DAYFROM SYSDATE)FROM DUAL; --23<br>SELECT EXTRACT(MONTHFROM SYSDATE)FROM DUAL; --04<br>SELECT EXTRACT(YEARFROM SYSDATE)FROM DUAL; --2018 | DATEPART(DD,GETDATE())<br>--can use MM:YY:HH:MI:SS:DW instead of DD<br>SELECTDATEPART(DD,GETDATE())--23<br>SELECTDATEPART(MM,GETDATE())--04<br>SELECTDATEPART(YY,GETDATE())--2018<br>SELECTDATEPART(DW,GETDATE())--2<br>SELECTDATEPART(DY,GETDATE())--113(Day of Year) |
| 6 | SELECT TO_CHAR(SYSDATE,'YEAR')FROM DUAL;<br>--TWENTY EIGHTEEN<br>SELECT TO_CHAR(SYSDATE,'Month')FROM DUAL;--April<br>SELECT TO_CHAR(SYSDATE,'MONTH')FROM DUAL;--APRIL | DATENAME(DATEPART,DATE)<br>--In this function month name, day name will be extracted other date parts providing same output<br>SELECTDATENAME(MM,GETDATE())--April<br>SELECTDATENAME(DW,GETDATE()) -Monday |
| 7 | SELECT LAST_DAY(SYSDATE)FROM DUAL; --LAST DATE OF MONTH | SELECT EOMONTH(GETDATE())--LAST DATE OF MONTH |
| 8 | SELECT NEXT_DAY(SYSDATE,'MONDAY')FROM DUAL; --NEXT MONDAY'S DATE | NO CONCEPT IN SQL SERVER |
| 9 | SELECT TRUNC(SYSDATE,'YY')FROM DUAL;<br>SELECT TRUNC(SYSDATE,'MM')FROM DUAL;<br>SELECT TRUNC(SYSDATE,'DY')FROM DUAL;<br>SELECT TRUNC(SYSDATE,'Q')FROM | SELECTDATEADD(YEAR, DATEDIFF(YEAR,0,GETDATE()),0)<br>--to get 1st Day of the Current Year |

| | | |
|---|---|---|
| | DUAL; | |

## DATACONVERSIONS

| Ser | ORACLE | SQL Server |
|---|---|---|
| 1 | SAME<br><br>SELECT 5/CAST(2 ASFLOAT)FROM DUAL; | CAST(SOURCE_DATA AS TARGET_DATA_TYPE)<br><br>SELECT 5/ CAST(2 AS FLOAT) |
| 2 | SELECT TO_CHAR(6)FROM DUAL;<br><br>SELECT TO_NUMBER('6')FROM DUAL;<br><br>SELECT TO_DATE('12','DD')FROM DUAL; -- 12-APR-2018<br><br>SELECT TO_DATE('12','YYYY')FROM DUAL; -- 01-APR-2012<br><br>SELECT TO_CHAR(SYSDATE,'MM')FROM DUAL; --04<br><br>SELECT TO_CHAR(SYSDATE,'MONTH')FROM DUAL; --APRIL | CONVERT(TARGET_DATA_TYPE, SOURCE_DATA [NUMBER])<br>SELECT 5/ CONVERT(FLOAT,2)<br><br>SELECT CONVERT(VARCHAR(30), GETDATE(),0)<br>-- Apr 24 2018 1:51PM<br>SELECT CONVERT(VARCHAR(30), GETDATE(),8)<br>-- 13:48:53<br><br>**CONVERTION WOULDN'T WORK IF WE GIVE DATEPART AS INPUT AS IT WORKS IN ORACLE**<br><br>SELECTCONVERT(VARCHAR(30),GETDATE(),103)<br>-- 24/04/2018<br>SELECTCONVERT(VARCHAR(30),GETDATE(),105)<br>-- 24-04-2018<br>SELECTCONVERT(VARCHAR(30),GETDATE(),106)<br>--24 Apr 2018 |

## RANKING OR WINDOW FUNCTIONS

| Ser | ORACLE | SQL Server |
|---|---|---|
| 1 | Same as in SQL Server | ROW_NUMBER()<br>-- Provides unique row number for each row |
| | | RANK() --It calculates ranks with gaps |
| | | DENSE_RANK() -- It calculates ranks with out gaps |
| | | NTILE(INT) -- Provides the data in to groups / blocks. It will group the data based on the user specified number and number of rows available in a table. |

## OTHER FUNCTIONS

| Ser | ORACLE | SQL Server |
|---|---|---|
| 1 | NVL(EXPRESSION1, EXPRESSION2)<br><br>SELECT ENAME, SAL, COMM, NVL(COMM,0) RS FROM EMP;<br><br>SELECT ENAME, SAL, COMM, NVL(ENAME,'UNKNOWN') UN FROM EMP;<br><br>SELECT ENAME ,SAL, COMM, NVL(CAST(COMM ASVARCHAR(4)),'NC') RS FROM EMP;<br><br>SELECT ENAME,SAL,COMM,SAL+ NVL(COMM,0) RS FROM EMP; | ISNULL(EXPRESSION1, EXPRESSION2)<br><br>SELECT ENAME, SAL, COMM, ISNULL(COMM,0) RS FROM EMP<br><br>SELECT ENAME, SAL, COMM, ISNULL(ENAME,'UNKNOWN') UN FROM EMP<br><br>SELECT ENAME ,SAL, COMM, ISNULL ( CAST(COMM AS VARCHAR(4),'NC') RS FROM EMP<br><br>SELECT ENAME,SAL,COMM,SAL+ISNULL(COMM,0) RS FROM EMP |
| 2 | Same as in SQL Server | COALESCE(EXPRESSION1,EXPRESSION2……EXPRESSIONN) |

| Ser | ORACLE | SQL Server |
|---|---|---|
| | | NULLIF( EXPRESSION1, EXPRESSION2) |

| | | |
|---|---|---|
| 3 | Same as in SQL Server | -- It is used to compare two expressions of any datatype. If equal it returns NULL, if not equal returns value of Expression1<br><br>SELECT NULLIF(10,10) -----NULL<br>SELECT NULLIF(10,90) -----10<br>SELECT NULLIF(‘X’,’Y’)--- X |
| 4 | Same as in SQL Server<br>and we have one more option like<br><br>SELECT SAL, DECODE(SAL,<br>5000,'A',4000,'B','C')FROM EMP; | CASE EXPRESSION WHEN CONDITION1 THEN RESULT1<br>[ WHEN CONDITION2 THEN RESULT2 ]<br>ELSE DEFAULT_RESULT END [ALIAS_NAME]<br><br>SELECT ENAME, DEPTNO, SAL,<br>CASE DEPTNO<br>WHEN 10 THEN SAL* 20/100<br>WHEN 20 THEN SAL * 18/100<br>WHEN 30 THEN SAL*15/100<br>END INCR  FROM EMP |

## MULTIROW FUNCTIONS OR GROUP FUNCTIONS

| Ser | ORACLE | SQL Server |
|---|---|---|
| 1 | Same as in SQL Server<br>except COUNT_BIG .<br><br>There is no concept like<br>COUNT_BIG in ORACLE | SUM (EXPRESSION) - finds the sum of values in given expression.<br>AVG (EXPRESSION) - first finds the sum and then divide with number of values in theexpression.<br>MAX (EXPRESSION) - finds the maximum value in the given expression.<br>MIN (EXPRESSION) - finds the minimum value in the given expression.<br>COUNT (EXPRESSION) --- returns number of values in a expression including duplicates.<br>COUNT (DISTINCT (EXPRESSION)) - returns number of values in an expression excludingduplicates.<br>COUNT (*) - returns number of rows.<br>COUNT_BIG (EXPRESSION) - returns number of values.<br>GROUPING (EXPRESSION) - returns zero or one.<br><br>Examples :-<br>SELECT SUM(COMM) R1, AVG(COMM) R2,<br>COUNT(COMM)R3 FROM EMP<br>SELECT SUM(SAL), MAX(SAL)FROM EMP WHERE<br><br>DEPTNO=30<br><br>SELECT COUNT(SAL) FROM EMP<br><br>SELECT COUNT(DISTINCT(SAL))FROM EMP |
| **GROUP BY:** | | |
| 2 | Same as in SQL Server | SELECT DEPTNO, JOB, SUM(SAL), COUNT(*) FROM EMP<br>GROUP BY JOB , DEPTNO<br><br>SELECT DEPTNO, SUM(SAL), COUNT(*) FROM EMP<br>WHERE DEPTNO = 20 GROUP BY DEPTNO |

| | | |
|---|---|---|
| **ROLLUP:** | | |
| 3 | SELECT DEPTNO,SUM(SAL),COUNT(*)FROM EMP GROUPBY**ROLLUP(DEPTNO)**<br><br>SELECT DEPTNO, JOB,SUM(SAL),COUNT(*)FROM EMP GROUPBYROLLUP (DEPTNO, JOB)<br><br>SELECT DEPTNO, JOB,SUM(SAL),COUNT(*),GROUPING(DEPTNO) gpdpt,GROUPING(JOB) gpjob FROM EMP GROUPBYROLLUP(DEPTNO, JOB)<br><br>SELECT DEPTNO ,CASEGROUPING(JOB) WHEN 0 THEN JOB<br>WHEN 1 THEN'ALL JOBS' END<br>DEP,SUM(SAL),COUNT(*)<br>FROM EMP GROUPBYROLLUP (DEPTNO, JOB) | SELECT DEPTNO,SUM(SAL),COUNT(*)FROM EMP GROUPBY**DEPTNO WITHROLLUP**<br><br>SELECT DEPTNO, JOB,SUM(SAL),COUNT(*)FROM EMP GROUPBY DEPTNO, JOB WITHROLLUP<br><br>SELECT DEPTNO, JOB,SUM(SAL),COUNT(*),GROUPING(DEPTNO) gpdpt,GROUPING(JOB) gpjob FROM EMP GROUPBY DEPTNO, JOB WITHROLLUP<br><br>SELECT DEPTNO ,<br>CASEGROUPING(JOB)<br>WHEN 0 THEN JOB<br>WHEN 1 THEN'ALL JOBS'--Replaces NULLS With Value<br>END DEP,SUM(SAL),COUNT(*)<br>FROM EMP GROUPBY DEPTNO, JOB WITHROLLUP |
| **CUBE:** | | |
| 4 | SELECT DEPTNO, JOB,SUM(SAL),COUNT(*)FROM EMP GROUPBY**CUBE(DEPTNO, JOB)**;<br><br>SELECT DISTINCT JOB,<br>SUM (CASE DEPTNO WHEN 10 THEN SAL END) "D10",<br>SUM (CASE DEPTNO WHEN 20 THEN SAL END) "D20",<br>SUM (CASE DEPTNO WHEN 30 THEN SAL END) "D30",<br>SUM (SAL) "TOTAL SAL"<br>FROM EMP GROUP BY JOB | SELECT DEPTNO, JOB, SUM(SAL), COUNT(*) FROM EMP<br>GROUP BY **DEPTNO, JOB WITHCUBE**<br><br>SELECTDISTINCT JOB,<br>SUM(CASE DEPTNO WHEN 10 THEN SAL END)'D10',<br>SUM(CASE DEPTNO WHEN 20 THEN SAL END)'D20',<br>SUM(CASE DEPTNO WHEN 30 THEN SAL END)'D30',<br>SUM(SAL)'TOTAL SAL'<br>FROM EMP GROUPBY JOB |
| **HAVING CLAUSE:** | | |
| 5 | Same as in SQL Server | SELECT DEPTNO,SUM(SAL) FROM EMP GROUP BY DEPTNO<br>HAVING DEPTNO = 20 AND SUM(SAL) >9000 |
| **SOME/ANY:** | | |
| 6 | Same as in SQL Server | This Operator will allow a Sub Query to written multiple rows even though in OUTER Query<br>Condition is made by using relational operators.<br>It works like "OR" LOGICAL OPERATOR.<br>It Can be used with all RELATIONAL OPERATORS [ > ANY , < ANY, = ANY, !=ANY , >= ANY, <= ANY ]<br><br>SELECT * FROM EMP WHERE SAL < ANY (SELECT DISTINCT SAL FROM EMP<br>WHERE DEPTNO = 20) |
| **ALL :** | | |
| 8 | Same as in SQL Server | This Operator also allows a sub query to written muliple rows even though in Outer Querycondition is made using relational operators.<br>It works like "AND" LOGICAL OPERATOR.<br>It can also be used with ALL Relational Operators.<br><br>SELECT * FROM EMP WHERE SAL < ALL (SELECT DISTINCT SAL FROM EMP |

| | | |
|---|---|---|
| | | WHERE DEPTNO =20) |
| | | |
| **EXISTS / NOT EXISTS OPERATOR** | | |
| 9 | Same as in SQL Server | **EXISTS:**<br>It Returns Boolean value ie.. True or False<br>If Condition at inner query is satisfied than it will written True else written with False<br>Q) Display DEPTNO, DNAME of those Department's where atleast one Employe is working.<br><br>SELECT DEPTNO, DNAME FROM DEPT DWHERE EXISTS ( SELECT 1 FROM EMP WHERE DEPTNO = D.DEPTNO ) |
| 10 | Same as in SQL Server | **NOT EXISTS:**<br>This operator also writtens Boolean Value i.e. TRUE or FALSE<br>If Condition at INNER Query is FALSE then it returns TRUE<br>Q) Display DEPTNO, DNAME of those Department's where no Employee is working.<br><br>SELECT DEPTNO, DNAME FROM DEPT DWHERE NOT EXISTS (SELECT 1 FROM EMP WHERE DEPTNO = D.DEPTNO)<br>OR<br>SELECT DEPTNO,DNAME FROM DEPTWHERE DEPTNO = (( SELECT DEPTNO FROM DEPT )<br>EXCEPT (SELECT DISTINCT DEPTNO FROM EMP)) |

Database Objects
Synonyms

| | | |
|---|---|---|
| Synonyms | Create synonym syn_emp<br>For emp; | Same as Oracle |
| List of synonyms in database | Select * from user_synonyms | Select name,base_object_name from sys.synonyms |
| Details of synonym | Desc syn_emp | Sp_help syn_emp |
| Dropping synonyms | Drop synonym syn_emp | Same |
| Data Dictionary | Desc User_synonyms | |

Indexes

| | | |
|---|---|---|
| Indexes | Clustered<br>Nonclustered<br>Unique<br>Non Unique<br>Composite<br>Function based Index | Clustered<br>Nonclustered<br>Unique |
| Unique Index | Same | CREATE UNIQUE INDEX INDX4 ON DEPT(DEPTNO) |
| To create non clustered index | Create index inx_emp On employees(empno);(By default) | CREATE NONCLUSTERED INDEX INDX2 ON EMP(ENAME) |
| Dropping Index | Drop index inx_emp | Same |
| To view information | Desc user_indexes | Sp_helpindex <tablename> |
| Primary Key column by default | Unique Non Clustered index | Unique Clustered index |
| Create Clustered index | | Create clustered index inx_emp On emp(empno) |

| | | |
|---|---|---|
| Disabling Indexes | | Alter index inx_emp on emp disable; |
| Data Dictionary | Desc user_indexes; | SELECT *<br>FROM sys.indexes<br>WHERE object_id =<br>OBJECT_ID('schema.MyTableName') |

Views

| | | |
|---|---|---|
| Creating Views | CREATE VIEW < VIEW_NAME ><br>AS<br>SELECT QUERY<br>[ WITH CHECK OPTION ] | CREATE VIEW < VIEW_NAME ><br>[ WITH ENCRYPTION ]/ [ WITH SCHEMA BINDING ]<br>AS<br>SELECT QUERY<br>[ WITH CHECK OPTION ] |
| To refresh the view | | SP_REFRESHVIEW < VIEW_NAME > |
| Creating Views | Same | CREATE VIEW V1<br>AS<br>SELECT * FROM EMP |
| Check Option View | Same | CREATE VIEW V4<br>AS<br>SELECT EMPNO, ENAME FROM EMP<br>WHERE DEPTNO = 10 WITH CHECK OPTION |
| Read Only View | CREATE VIEW V4<br>AS<br>SELECT EMPNO, ENAME FROM EMP<br>WHERE DEPTNO = 10 WITH READ ONLY | |
| Information about view | Desc User_views | SP_HELP V1 |
| Query stored in the View | | SP_HELPTEXT < VIEW_NAME > |
| Display list of Views dependent on table | | SP_DEPENDS < TABLE_NAME > / < VIEW_NAME > |
| Encrypted View | No Concept | CREATE VIEW V9 WITH ENCRYPTION<br>AS<br>SELECT * FROM EMP<br><br>create view vw_emp_encry with encryption<br>as select empno,ename,sal from emp |
| Schema binding View | No Concept | CREATE VIEW V10 WITH SCHEMA BINDING<br>AS<br>SELECT * FROM EMP<br><br>--Error & also cannot drop base tables<br><br>create view vw_emp_schema with schemabinding<br>as select empno,ename,sal from dbo.emp |
| Materialized Or Indexed View Syntax | Materialized View | Indexed View<br><br>Creating unique clustered index on views<br><br>create unique clustered index inx_emp1 on vw_emp_schema(empno); |

| | | |
|---|---|---|
| Information about materialized view/Indexed Views | User_mviews | select<br>OBJECT_SCHEMA_NAME(object_id) as [SchemaName],<br>OBJECT_NAME(object_id) as [ViewName],<br>Name as IndexName<br>from sys.indexes<br>where object_id in<br>(<br>select object_id<br>from sys.views<br>)<br><br>The inner join version<br><br>select<br>OBJECT_SCHEMA_NAME(si.object_id) as [SchemaName],<br>OBJECT_NAME(si.object_id) as [ViewName],<br>si.Name as IndexName<br>from sys.indexes AS si<br>inner join sys.views AS sv<br>ON si.object_id = sv.object_id |

Sequences Or Identity

| | | |
|---|---|---|
| Creating sequence | Create sequence seq_emp<br>Start with 1<br>Increment by 1<br>Minvalue 1<br>Maxvalue 20<br>Cycle<br>Cache 5;<br><br>To get nextvalue<br><br>Select seq_emp.nextval from dual<br><br>To get current value<br><br>Select seq_emp.currval from dual | CREATE SEQUENCE Test.DecSeq<br>AS int<br>START WITH 125<br>INCREMENT BY 25<br>MINVALUE 100<br>MAXVALUE 200<br>CYCLE<br>CACHE 3<br><br>To get nextvalue<br><br>SELECT NEXT VALUE FOR Test.DecSeq;<br><br>To get current value<br><br>SELECT current_value<br>FROM sys.sequences<br>WHERE name = 'Seq_emp' ; |
| Altering Sequences | alter sequence seq_emp<br>increment by 5<br>cycle;<br><br>Start with value cannot be altered. | ALTER SEQUENCE Test. TestSeq<br>START WITH 100<br>INCREMENT BY 50<br>MINVALUE 50<br>MAXVALUE 200<br>NO CYCLE<br>NO CACHE |
| Dropping Sequences | Drop sequence seq_emp; | Drop sequence seq_emp; |
| Last Value stored into IDENTITY COLUMN | | SELECT @@IDENTITY OR SELECT SCOPE_IDENTITY() OR SELECT IDENT_CURRENT( < TABLE_NAME > |

| | | |
|---|---|---|
| | | ) |
| Information about sequences | User_sequences | select *<br>from sys.sequences<br>where object_id =<br>object_id('seq_test')<br><br>SYS.Sequences Or<br>SYS.IDENTITY_COLUMNS |
| start value of the IDENTITY COLUMN | INSERT Test.Orders (OrderID, Name, Qty)<br>VALUES<br>(Seq_name.nextval,’Tire’,2);<br><br>Seq_name.nextval(Nextvalue)<br>Seq_name.currval(Current Value) | INSERT Test.Orders (OrderID, Name, Qty)<br>VALUES (NEXT VALUE FOR Test.CountBy1, 'Tire', 2) ;<br><br>IDENT_SEED ( < TABLE_NAME > )<br>SELECT IDENT_SEED ('PRODUCT')<br><br>SET IDENTITY_INSERT < TABLE_NAME > ON / OFF [ DEFAULT IS OFF ] |
| incremented value of the IDENTITY COLUMN | | IDENT_INCR ( < TABLE_NAME > )<br>SELECT IDENT_INCR ('PRODUCT') |
| To restart the sequence of Numbers for IDENTITY COLUMN | Cycle | DBCC CHECKIDENT ( <TABLE_NAME>, RESEED , VALUE )<br>DBCC CHECKIDENT('PRODUCT',RESEED,3) |

With Clause Or CTE

| | | |
|---|---|---|
| Recursive Query | WITH cte(n)<br>AS<br>(SELECT 1 as n from dual<br>UNION ALL<br>SELECT n + 1 FROM cte WHERE n<10)<br>SELECT n FROM cte | with cte<br>as<br>(select 1 as n<br>union all<br>select n=n+1 from cte where n<10)<br>select n from cte |
| | | |

Pseudocolumns

| | | |
|---|---|---|
| Rownum | Select * from emp<br>Where rownum<=3; | Row_number() |
| Rowid(To delete duplicate records) | Delete from emp<br>Where rowid not in(select max(rowid) from tab<br>Group by col1,col2) | delete from emp<br>where %%physloc%% not in (select max(%%physloc%%) from emp group by empno); |
| Level | | Recursive CTE |
| User | user | User |
| Currval | | |
| Nextval | | |

Level

| | | |
|---|---|---|
| To display numbers from 1 to 10 | Select level from dual<br>Connect by level<=10 | |

| | | |
|---|---|---|
| | with cte(n)<br>as<br>(select 1 as n from dual<br>union all<br>select n+1 from cte where n<12)<br>select n from cte | with cte<br>as<br>(select 1 as n<br>union all<br>select n=n+1 from cte where n<12)<br>select n from cte |
| To display month names | with cte(n)<br>as<br>(select 1 as n from dual<br>union all<br>select n+1 from cte where n<12)<br>select to_char(add_months(trunc(sysdate,'yy'),n-1),'month') from cte<br><br>select to_char(to_date(level,'mm'),'month') from dual<br>connect by level<=12; | with cte<br>as<br>(select 1 as n<br>union all<br>select n=n+1 from cte where n<12)<br>select DateName(m,dateadd(m,n-1,0))<br>from cte |
| To display Day names | select to_char(trunc(sysdate,'dy')+level-1,'day') from dual<br>connect by level<=7<br><br>with cte(n) as<br>(select 1 as n from dual<br>union all<br>Select n+1 from cte where n<7 )<br>select to_char(trunc(sysdate,'dy')+n-1,'day') from cte | with cte as<br>(select 1 as n<br>union all<br>Select n=n+1 from cte where n<7 )<br>Select datename(dw,dateadd(d,n-1,Datename(yy,getdate()) +'-01-01'))<br>from cte |
| To display multiples of 5 | select level<br>from dual<br>where mod(level,5)=0<br>connect by level<=50<br><br>with cte(n) as<br>(select 1 as n from dual<br>union all<br>Select n+1 from cte where n<10 )<br>Select 5*n from cte | with cte as<br>(select 1 as n<br>union all<br>Select n=n+1 from cte where n<10 )<br>Select 5*n from cte |
| To display ‘Aroha’ vertically | with cte(n)<br>as<br>(select 1 as n from dual<br>union all<br>select n+1 from cte where n<length('AROHA'))<br>select substr('AROHA',n,1) from cte | with cte<br>as<br>(select 1 as n<br>union all<br>select n=n+1 from cte where n<len('AROHA'))<br>select substring('AROHA',n,1) from cte |
| To display AROHA | select substr('aroha',1,level)<br>from dual<br>connect by level<=length('aroha') | with cte<br>as<br>(select 1 as n<br>union all<br>select n=n+1 from cte where |

| | | |
|---|---|---|
| | with cte(n)<br>as<br>(select 1 as n from dual<br>union all<br>select n+1 from cte where n<length('AROHA'))<br>select substr ('AROHA',1,n) from cte | n<len('AROHA'))<br>select substring ('AROHA',1,n) from cte |
| To display December dates | with cte(n) as<br>(select 1 as n from dual<br>union all<br>Select n+1 from cte where n<31 )<br>Select (add_months(trunc(sysdate,'yy'),11))+n-1 from cte<br><br>select to_date('01-dec-18')+level-1 from dual<br>connect by level<=to_char(last_day('01-dec-18'),'dd'); | with cte as<br>(select 1 as n<br>union all<br>Select n=n+1 from cte where n<31 )<br>Select dateadd(dd,n-1,dateadd(m,-1,Datename(yy,getdate()) +'-01-01'))<br>from cte |
| Delete duplicate records | Only select and no delete works here. | WITH DuplicateCTE(col1,col2,Row_num)<br>AS<br>(<br>SELECT col1,col2,<br>ROW_NUMBER()OVER(PARTITION BY col1,col2 ORDER BY col1,col2)AS Row_num<br>FROM dup<br>)<br>DELETE FROM DuplicateCTE WHERE Row_num>1 |

Date Queries

| | | |
|---|---|---|
| First day of the Current year | Trunc(sysdate,'yy') | select dateadd(dd,1-datepart(dy,getdate()),getdate()) |
| Last day of Current Year | Add_months(trunc(sysdate,'yy'),12)-1 | select dateadd(mm,12,dateadd(d,1-datepart(dy,getdate()),getdate()))-1 |
| First day of the Current Month | Trunc(sysdate,'mm') | select dateadd(dd, 1-datepart(dd,getdate()),getdate()) |
| Last day of the Current Month | Add_months(Trunc(sysdate,'mm'),1)-1 | select eomonth(getdate()) |
| First day of the Current Week | Trunc(sysdate,'dy') | select dateadd(d,1-datepart(dw,getdate()),getdate()) |
| Year day number | To_char(sysdate,'ddd') | select datepart(dy,getdate()) Or Y |
| Month day number | To_char(sysdate,'dd') | select datepart(dd,getdate()) Or d |
| Week day number | To_char(sysdate,'d') | select datepart(w,getdate()) Or dw Or weekday |
| Year Week Number | To_char(sysdate,'ww') | select datepart(wk,getdate()) Or week or ww |

| | | |
|---|---|---|
| Week no of Month | To_char(sysdate,’w’) | select datepart(wk,getdate())-datepart(wk, dateadd(dd, 1-datepart(dd,getdate()), getdate()))+1 |
| Quarter no of current date | To_char(sysdate,’q’) | Select datepart(q,getdate()) |
| First day of current quarter | Trunc(sysdate,’q’) | select dateadd(qq,datediff(qq,0,getdate()),0) |
| First day of the next quarter | Add_months(Trunc(sysdate,’q’),3) | SELECT dateadd(qq, datediff(qq, 0, getdate()) + 1, 0) |
| First day of the previous quarter | Add_months(Trunc(sysdate,’q’),-3) | SELECT dateadd(qq, datediff(qq, 0, getdate()) - 1, 0) |

PLSQL/T-SQL

Data Dictionary for Subprograms:

| | | |
|---|---|---|
| Data Dictionary for Stored Procedures | User_procedures | Sp_Stored_procedures |
| Data Dictionary for Stored functions | User_procedures | |
| Dropping a procedure | Drop procedure <procedure_name> | DROP PROC / PROCEDURE < PROCEDURE_NAME > |
| To display list of Stored procedures | select object_name from user_procedures where object_type='PROCEDURE'; | SELECT NAME FROM SYSOBJECTS WHERE XTYPE='P' |
| Information about a Stored procedure | Select * from user_objects where object_type=’PROCEDURE’ | SP_HELP 'PROCEDURE_NAME' |
| Text for Stored procedure | Select text from user_source where name=’SP_P1’ | SP_HELPTEXT 'PROCEDURE_NAME' |
| Text for Stored Functions | Select text from user_source where name=’FN_EMP’ | SP_HELPTEXT < FUNCTION_NAME > |
| Dropping a function | Drop function <function_name> | DROP FUNCTION <FUNCTION_NAME > |
| To display list of user created scalar functions | select object_name from user_procedures where object_type='FUNCTION'; | SELECT NAME FROM SYSOBJECTS WHERE XTYPE='FN' |
| Information about user created function | Select * from user_objects where object_type=’FUNCTION’ | SP_HELP < FUNCTION_NAME > |

Cursors

| | | |
|---|---|---|
| Creating Cursors | Declare<br>cursor c1 is select * from emp<br>Begin<br>Open c1<br>Loop<br>Fetch c1 into variable | Declare<br>C1 cursor for select * from emp<br>Open c1<br>Fetch next from c1 into variables<br>While @@fetch_status <>-1<br>Begin |

| | | |
|---|---|---|
| | Exit when c1%notfound<br>Dbms_output.put_line(variable);<br>End loop;<br>Close c;<br>End; | Print variables<br>Fetch next from c1 into variables<br>End<br>Close c1<br>Deallocate c1 |

Procedures

| | | |
|---|---|---|
| Creating Procedures | Create or replace procedure sp_p1(p_no number, p_name out varchar2) as<br>Begin<br>Select ename into p_name from emp where empno=p_no<br>End; | create procedure sp_p1(@p_no int,@p_name varchar(15) output)<br>as<br>begin<br>select @p_name=ename from emp where empno=@p_no;<br>end |
| Executing Procedures With exec command | Exec sp_p1(7788,:b_name)<br>Print :b_name | Exec sp_p1 7788 if there is no out parameter |
| Executing procedures with Ananymous block | Declare<br>V_name varchar2(20);<br>Begin<br>Sp_p1(7788,v_name);<br>Dbms_output.put_line(v_name);<br>End; | declare<br>@name varchar(15)<br>exec sp_p1 7788,@name output<br>print @name |
| Altering the procedures | | Alter procedure sp_p1<br>As<br>Begin<br>Statements<br>End |
| Dropping the procedure | Drop procedure sp_p1 | Drop procedure sp_p1 |
| Procedures returning values | create or replace procedure sp_return is<br>v_no int:=1;<br>begin<br>dbms_output.put_line(v_no);<br>return;<br>end; | create procedure sp_return(@x int,@y int)as<br>begin<br>declare @z int<br>set @z=@x+@y<br>return @z<br>end |
| Executing Procedures with returning values | exec sp_return; | declare @r int<br>exec @r=sp_return 10,20<br>print @r |

Scalar Valued Functions which returns single value

| | | |
|---|---|---|
| Creating Functions | Create function fn_emp (p_no number) return number is<br>V_cnt number(4);<br>Begin | create function fn_emp(@p_no int)<br>returns int as<br>begin<br>declare @v_cnt int<br>select @v_cnt=count(*) from emp<br>where deptno=@p_no |

| | | |
|---|---|---|
| | Select count(*) into v_cnt from emp where deptno=p_no;<br>Return p_no;<br>End; | return @v_cnt<br>end; |
| Executing Functions using select | Select fn_emp(10) from dual | Select dbo.fn_emp(10) |
| Executing functions using anonymous blocks | Declare<br>V_count number(4);<br>Begin<br>V_count:=fn_emp(10);<br>Dbms_output.put_line(v_count);<br>End; | declare @v_count int<br>set @v_count=dbo.fn_emp(10)<br>print @v_count |
| Executing using exec command | Exec :b_cnt:=fn_emp(10)<br>Print :b_cnt | No option---- |

Table Valued Functions returning multiple values

| | | |
|---|---|---|
| Creating Functions | Create function fn_emp(p_dno number) return sys_refcursor as<br>P_ref sys_refcursor;<br>Begin<br>Open p_ref for select * from emp where deptno=p_dno;<br>Return p_ref; | create function fn_emp_table(@dno int) returns table as<br>return(select * from emp where deptno=@dno); |
| Executing Functions | | SELECT * FROM dbo.fn_emp_table(20) |

Triggers

| | | |
|---|---|---|
| Creating Triggers | Create or replace trigger trig_name<br>Before insert or update or delete on emp<br>Begin<br>If to_char(sysdate,’dy’) in(‘sat’,’sun’) or to_char(sysdate,’hh24’) not between 10 and 17 then<br>Raise_application_error(-20001,’not business hours’)<br>End if;<br>End; | create trigger trig_emp<br>on emp<br>for insert,delete,update<br>as<br>begin<br>if<br>datename(dw,getdate())='sunday'<br>or datepart(hh,getdate()) not between 10 and 17<br>begin<br>rollback<br>raiserror('Invalid Time',15,16)<br>end<br>end |
| List of triggers created in a database | Select object_name from user_objects where object_type=’TRIGGER’ | Select name from sysobjects where xtype=’TR’ |
| Viewing Trigger Information | User_triggers | SP_HELPTEXT<Trigger_name> |
| Disable/Enable the triggers | Alter table emp disable trigger trig_emp;<br>Alter table emp disable all triggers; | Alter table emp disable/enable trigger<trig_name> |
| To disable Database or DML Triggers | Alter trigger trig_emp disable; | Disable/Enable trigger <trigger_name>[ALL] on database/Object_name |
| Dropping a Trigger | Drop trigger trig_emp | Drop trigger <trigger_name> on <database name> |

Dynamic Queries

<table>
<tr><td>Dynamic SQL</td><td>Create or replace procedure sp_dyn(p_table_nm varchar2) as<br>Begin<br>Execute immediate ‘select * from ‘||p_table_nm;<br>End;</td><td>create procedure sp_dyn(@table_nm varchar(15)) as<br>begin<br>exec('select * from '+@table_nm)<br>end<br><br>exec sp_dyn emp</td></tr>
<tr><td></td><td>Will not Support</td><td>declare<br>@s varchar(100)='select * from emp'<br>exec (@s)</td></tr>
</table>