**Oracle Vs Sql Server**

DDL COMMANDS

|  |  |  |  |
| --- | --- | --- | --- |
| **Ser** | **Concept** | **ORACLE** | **SQL Server** |
| 1 | Create  Database/  Schema | We can create schema(user) only  Grant DBA touserName identified by (password); | Wecan create multiple databases in SQL Server  1. createDatabase Library  2. Go to Object Explorer  =>Right Click on Databases  =>Select New Database  =>Give Database Name  =>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  (  BI\_ID number(2)primarykey,  B\_ID number(2)  references Book(B\_ID),  C\_ID number(2)  references Customer(C\_ID),  Issue\_dt timestamp(0),  Return\_dt timestamp(0),  Remarks varchar(250)  );  2. Go to Connection in ORACLE SQL Developer  =>Expand Schemas  =>Right Click on Tables  =>Select New Table  =>Give Table Name  =>Tick on Primary Key Button if it is Primary Key field.  Type Column Name, Data Type, Size Not Null, Default and Comment  =>Click on + button to add columns  =>Save Table | 1. Createtable Book\_Issue  (  BI\_ID intprimarykeyidentity (1,1),  B\_ID intforeignkey (B\_ID)references Book(B\_ID),  C\_ID intforeignkey(C\_ID)references Customer(C\_ID),  Issue\_dt Datetime,  Return\_dt Datetime,  Remarks varchar(250)  )  2. Go to Object Explorer in SQL SERVER  =>Expand Databases  =>Expand Database Objects  =>Right Click on Tables  =>Select New Table  =>Type Column Name, Data Type and Allow Null  =>Save Table |
| 3 | Add Column | ALTER TABLE Customer  ADD dob DATE,  ADD address VARCHAR (250); | ALTERTABLE Customer  ADD dob DATE,  Address VARCHAR(200) |
| 4 | Drop Column | ALTER TABLE Customer drop column address ; | ALTERTABLE Cust  DROPCOLUMN dob,  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  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  **Result : SMITH WORKING AS CLERK** |
| 16 | Mathematical expression in select statement | SELECT 10 + 20 FROM DUAL;  SELECT 10 \* 20FROM DUAL;  SELECT 5/2 FROM DUAL;  (Will give result with decimals)  SELECT 10 + 5 \* 6 M1, (10+5)\* 6 M2 FROM DUAL; | SELECT 10 + 20-- {FOR ADDITION}  SELECT 10 \* 20-- {FOR MULTIPLICATION}  SELECT 5/2 -- WILL TREAT AS INTEGER (Will removes the decimals)  SELECT 5/2.0 –Will give result with decimals  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;  or  SELECT \* FROM DEPT WHERE DEPTNO = '10'; | SELECT \* FROM DEPT WHERE DEPTNO = 10  or  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';  SELECT ENAME, JOB FROM EMP WHERE JOB IN ('CLERK','ANALYST'); | SELECT ENAME, JOB FROM EMP WHERE JOB = 'CLERK' OR JOB = 'ANALYST'  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)  UPDATE TOP(3) FROM EMP SET SAL = SAL + 1000  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';  SELECT ENAME, JOB FROM EMP WHERE JOB IN ('CLERK','ANALYST'); | SELECT ENAME, JOB FROM EMP WHERE JOB = 'CLERK' OR JOB = 'ANALYST'  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;  SELECT ENAME, SAL FROM EMP WHERE SAL BETWEEN 2000 AND 3000; | SELECT ENAME, SAL FROM EMP WHERE SAL >= 2000 AND SAL <=3000  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;  (or)  SELECT \* FROM EMP WHERE JOB IN ('CLERK','ANALYST') AND SAL >1000; | SELECT \* FROM EMP WHERE JOB = 'CLERK' OR 'ANALYST' AND SAL >1000  (or)  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;  SELECT \* FROM EMP WHERE NOT COMM IS NULL; | SELECT \* FROM EMP WHERE COMM IS NOT NULL  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  Select \* from source | Insert into emp\_tgt  Select \* from emp\_src |
| To create the table and populate records from another table | Create table emp\_new  As select \* from scott.emp; | Select \* into emp\_new from emp; |
| To Update the columns | Update emp set sal=sal+1000  Where empno=1010 | Same as Oracle |
| To delete the records | Delete from emp  Where sal>1000; | Same as Oracle |
| TCL | Commit(To make change permanent)  Rollback(To undo change)  Savepoint a;(Create a marker)  Rollback to savepoint a; | Begin tran  Commit;  Rollback tran s;  Save tran F; |

DCL Commands

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

Scalar Datatypes

|  |  |  |
| --- | --- | --- |
| Integer Data | Number | INT  SMALLINT  BIGINT  TINYINT  DECIMAL  NUMERIC  MONEY  SMALLMONEY |
| REAL Numbers |  | FLOAT  REAL |
| Date And Time | Date | DATE  DATETIME  SMALLDATETIME |
| STRING Datatype | Varchar  Varchar2 | CHAR  NCHAR  VARCHAR  NVARCHAR |
| Unstructured DATA |  | BINARY  VARBINARY  IMAGE  XML |

DATA DICTIONARY VIEWS(METADATA)

|  |  |  |
| --- | --- | --- |
| To display list of User Defined Tables | SELECT\*FROM USER\_TABLES;  SELECT\*FROM USER\_OBJECTS WHERE OBJECT\_TYPE=‘TABLE’; | SELECT NAME FROMSYS.TABLES  SELECT NAME FROMSYS.SYSOBJECTSWHERE XTYPE ='U'  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'  SELECT NAME FROM SYS.VIEWS  SP\_HELPTEXT<View\_name>  SP\_HELP<View\_name>  SP\_DEPENDS<Table\_name>/<View\_name>  SP\_REFRESHVIEW<View\_name>  Select \* from information\_schema.views |
| To display STRUCTURE or DEFINITION or METADATA of a Table | Desc< TABLE\_NAME > | SP\_HELP < TABLE\_NAME >  SP\_HELP'EMP' |
| To Display List of Synonyms | Select \* from User\_synonyms | Select name,base\_object\_name from sys.synonyms  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>  SELECT NAME FROM SYS.INDEXES;  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  Entity Integrity  Referential Integrity | Not null  Check  Default  Primary key  Unique  Foreign Key | Not null  Check  Default  Primary key  Unique  Foreign Key |
|  | Same | CREATE TABLE EMP1(  ENO INT NOT NULL, ENAME VARCHAR(10) NOT NULL,  JOB VARCHAR(15) CHECK (JOB IN ('CLERK','MANAGER','OPERATIONS')),  SAL INT CONSTRAINT SAL\_CHK CHECK (SAL BETWEEN 15000 AND 20000),  DOJ SMALLDATETIME DEFAULT GETDATE(),  DNO INT CONSTRAINT DF\_DNO DEFAULT 20) |
|  | Same | **COLUMN LEVEL**  CREATE TABLE STUDENT  (RNO INT UNIQUE, SN VARCHAR(10) CONSTRAINT SN\_UQ UNIQUE)  **TABLE / ENTITY LEVEL**  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 (  ENO INT PRIMARY KEY,  ENAME VARCHAR(15) NOT NULL UNIQUE,  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.  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.  SELECT POWER (2,5) |
| 5 | Same as in SQL Server | SIGN ( NUMBER )  Returns 1 if a number is positive  Returns -1 if a number is negative  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  SELECT SIN (30\* PI()/180) |
| 8 | Same as in SQL Server | ROUND ( NUMBER, NUMBER , [NUMBER])  Argument1, Argument2, Argument3 (optional for TRUNCATE)  By default Argument3 is ZERO  ROUND(1234.5678,2,1) --- 1234.5600  ROUND(5/2.0,0) --- 3.00  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.  CEILING (123.000) ---- 123  CEILING (123.010) ---- 124  CEILING (123.456) ---- 124 |
| 10 | Same as in SQL Server | FLOOR (NUMBER) – It decreases a nearest integer  FLOOR (-123.456) ---- {-124} |
| 11 | SELECT MOD(25,5)FROM DUAL;  ---- will get the remainder | SELECT 25%5 |
| 12 | SELECT TRUNC(5.92,0)FROM DUAL; --5  SELECT TRUNC(5.92,1)FROM DUAL;--5.9  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);  SELECT ENAME, LENGTH(ENAME) FROM EMP WHERE LENGTH(ENAME) >5; | LEN(TEXT)  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);  SELECT rtrim('100%','%') FROM DUAL;  Result : 100 | RTRIM(STRING) |
| 10 | No concept in ORACLE | REPLICATE(STRING, NUMBER)  REPLICATE('COMM',2) |
| 11 | REVERSE(STRING);  SELECT REVERSE('COMM') | REVERSE(STRING)  SELECT REVERSE('COMM')--- MMOC |
| 12 | Same as SQL Server | REPLACE (STRING1, STRING2, STRING3)  REPLACE ('COMPUTER','UT','IR') |
| 13 | No concept in ORACLE | STUFF(STRING1, NUMBER1, NUMBER2, STRING2)  STUFF('COMPUTER',5,2,'IL') -- COMPILER |
| 14 | INSTR ('COMPUTER','R',1,1) | CHARINDEX(STRING1, STRING2, [NUMBER])  SELECT CHARINDEX('R','COMPUTER',7) |
| 15 | SELECT LPAD('100',6,'0')FROM DUAL;  Result : 000100 | No concept in SQL SERVER |
| 16 | SELECT RPAD('COMPUTER',10,'A')FROM DUAL;  Result : COMPUTERAA | No concept in SQL SERVER |
| 17 | SELECT TRANSLATE(COMPUTER','PT','NQ')FROM DUAL;  Result : COMNUQER | No concept in SQL SERVER |

**DATE FUNCTIONS**

|  |  |  |
| --- | --- | --- |
| **Ser** | **ORACLE** | **SQL Server** |
| 1 | SELECT SYSDATE FROM DUAL;  --It returns the Current Sys Date 20-Apr-18 | GETDATE()  --It returns current system DATE & TIME of the Server |
| 2 | SELECT TO\_CHAR(SYSDATE,'DD')FROM DUAL;--23  SELECT TO\_CHAR(SYSDATE,'MM')FROM DUAL;--04  SELECT TO\_CHAR(SYSDATE,'YYYY')FROM DUAL;--2018 | SELECTDAY(GETDATE())-------20  SELECTMONTH(GETDATE())------ 4  SELECTYEAR(GETDATE())-----2018 |
| 3 | SELECT SYSDATE+10 FROM DUAL;  SELECT ADD\_MONTHS(SYSDATE,10)FROM DUAL;  SELECT TO\_CHAR(SYSDATE,'YYYY')+10 FROM DUAL; | SELECT DATEADD(DAY,10,GETDATE())  SELECT DATEADD(DD,10,GETDATE())  SELECT DATEADD(MONTH,10,GETDATE())  SELECT DATEADD(MM,10,GETDATE())  SELECT DATEADD(YEAR,10,GETDATE())  SELECT DATEADD(YY,10,GETDATE()) |
| 4 | SELECT TO\_DATE('10-FEB-2018')-TO\_DATE('10-FEB-17')FROM DUAL;  SELECT MONTHS\_BETWEEN('10-FEB-2018','10-FEB-17')FROM DUAL;  SELECT TO\_CHAR(TO\_DATE(SYSDATE),'YYYY')-TO\_CHAR(TO\_DATE('10-FEB-17'),'YYYY')FROM DUAL; | SELECTDATEDIFF(DAY,'2016-04-20',GETDATE())  SELECTDATEDIFF(MONTH,'2016-04-20',GETDATE())  SELECTDATEDIFF(YEAR,'2016-04-20',GETDATE()) |
| 5 | SELECT EXTRACT(DAYFROM SYSDATE)FROM DUAL; –-23  SELECT EXTRACT(MONTHFROM SYSDATE)FROM DUAL; –-04  SELECT EXTRACT(YEARFROM SYSDATE)FROM DUAL; –-2018 | DATEPART(DD,GETDATE())  –-can use MM:YY:HH:MI:SS:DW instead of DD  SELECTDATEPART(DD,GETDATE())–-23  SELECTDATEPART(MM,GETDATE())–-04  SELECTDATEPART(YY,GETDATE())–-2018  SELECTDATEPART(DW,GETDATE())–-2  SELECTDATEPART(DY,GETDATE())–-113(Day of Year) |
| 6 | SELECT TO\_CHAR(SYSDATE,'YEAR')FROM DUAL; --TWENTY EIGHTEEN  SELECT TO\_CHAR(SYSDATE,'Month')FROM DUAL;--April  SELECT TO\_CHAR(SYSDATE,'MONTH')FROM DUAL;--APRIL | DATENAME(DATEPART,DATE)  --In this function month name, day name will be extracted other date parts providing same output  SELECTDATENAME(MM,GETDATE())--April  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;  SELECT TRUNC(SYSDATE,'MM')FROM DUAL;  SELECT TRUNC(SYSDATE,'DY')FROM DUAL;  SELECT TRUNC(SYSDATE,'Q')FROM DUAL; | SELECTDATEADD(YEAR, DATEDIFF(YEAR,0,GETDATE()),0)  --to get 1st Day of the Current Year |

**DATACONVERSIONS**

|  |  |  |
| --- | --- | --- |
| **Ser** | **ORACLE** | **SQL Server** |
| 1 | SAME  SELECT 5/CAST(2 ASFLOAT)FROM DUAL; | CAST(SOURCE\_DATA AS TARGET\_DATA\_TYPE)  SELECT 5/ CAST(2 AS FLOAT) |
| 2 | SELECT TO\_CHAR(6)FROM DUAL;  SELECT TO\_NUMBER('6')FROM DUAL;  SELECT TO\_DATE('12','DD')FROM DUAL; -- 12-APR-2018  SELECT TO\_DATE('12','YYYY')FROM DUAL; -- 01-APR-2012  SELECT TO\_CHAR(SYSDATE,'MM')FROM DUAL; --04  SELECT TO\_CHAR(SYSDATE,'MONTH')FROM DUAL; --APRIL | CONVERT(TARGET\_DATA\_TYPE, SOURCE\_DATA [NUMBER])  SELECT 5/ CONVERT(FLOAT,2)  SELECT CONVERT(VARCHAR(30), GETDATE(),0)  -- Apr 24 2018 1:51PM  SELECT CONVERT(VARCHAR(30), GETDATE(),8)  -- 13:48:53  **CONVERTION WOULDN’T WORK IF WE GIVE DATEPART AS INPUT AS IT WORKS IN ORACLE**  SELECTCONVERT(VARCHAR(30),GETDATE(),103)  -- 24/04/2018  SELECTCONVERT(VARCHAR(30),GETDATE(),105)  -- 24-04-2018  SELECTCONVERT(VARCHAR(30),GETDATE(),106)  --24 Apr 2018 |

**RANKING OR WINDOW FUNCTIONS**

|  |  |  |
| --- | --- | --- |
| **Ser** | **ORACLE** | **SQL Server** |
| 1 | Same as in SQL Server | ROW\_NUMBER()  -– 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)  SELECT ENAME, SAL, COMM, NVL(COMM,0) RS FROM EMP;  SELECT ENAME, SAL, COMM, NVL(ENAME,'UNKNOWN') UN FROM EMP;  SELECT ENAME ,SAL, COMM, NVL(CAST(COMM ASVARCHAR(4)),'NC') RS FROM EMP;  SELECT ENAME,SAL,COMM,SAL+ NVL(COMM,0) RS FROM EMP; | ISNULL(EXPRESSION1, EXPRESSION2)  SELECT ENAME, SAL, COMM, ISNULL(COMM,0) RS FROM EMP  SELECT ENAME, SAL, COMM, ISNULL(ENAME,'UNKNOWN') UN FROM EMP  SELECT ENAME ,SAL, COMM, ISNULL ( CAST(COMM AS VARCHAR(4),'NC') RS  FROM EMP  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** |
| 3 | Same as in SQL Server | NULLIF( EXPRESSION1, EXPRESSION2)  -- It is used to compare two expressions of any datatype. If equal it returns NULL, if not equal returns value of Expression1  SELECT NULLIF(10,10) -----NULL  SELECT NULLIF(10,90) -----10  SELECT NULLIF(‘X’,’Y’)--- X |
| 4 | Same as in SQL Server  and we have one more option like  SELECT SAL, DECODE(SAL, 5000,'A',4000,'B','C')FROM EMP; | CASE EXPRESSION WHEN CONDITION1 THEN RESULT1  [ WHEN CONDITION2 THEN RESULT2 ]  ELSE DEFAULT\_RESULT END [ALIAS\_NAME]  SELECT ENAME, DEPTNO, SAL,  CASE DEPTNO  WHEN 10 THEN SAL\* 20/100  WHEN 20 THEN SAL \* 18/100  WHEN 30 THEN SAL\*15/100  END INCR FROM EMP |

**MULTIROW FUNCTIONS OR GROUP FUNCTIONS**

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

|  |  |  |
| --- | --- | --- |
| **ROLLUP:** | | |
| 3 | SELECT DEPTNO,SUM(SAL),COUNT(\*)FROM EMP GROUPBY**ROLLUP(DEPTNO)**  SELECT DEPTNO, JOB,SUM(SAL),COUNT(\*)FROM EMP GROUPBYROLLUP (DEPTNO, JOB)  SELECT DEPTNO, JOB,SUM(SAL),COUNT(\*),GROUPING(DEPTNO) gpdpt,GROUPING(JOB) gpjob FROM EMP GROUPBYROLLUP(DEPTNO, JOB)  SELECT DEPTNO ,CASEGROUPING(JOB) WHEN 0 THEN JOB  WHEN 1 THEN'ALL JOBS' END DEP,SUM(SAL),COUNT(\*)  FROM EMP GROUPBYROLLUP (DEPTNO, JOB) | SELECT DEPTNO,SUM(SAL),COUNT(\*)FROM EMP GROUPBY**DEPTNO WITHROLLUP**  SELECT DEPTNO, JOB,SUM(SAL),COUNT(\*)FROM EMP GROUPBY DEPTNO, JOB WITHROLLUP  SELECT DEPTNO, JOB,SUM(SAL),COUNT(\*),GROUPING(DEPTNO) gpdpt,GROUPING(JOB) gpjob FROM EMP GROUPBY DEPTNO, JOB WITHROLLUP  SELECT DEPTNO ,  CASEGROUPING(JOB)  WHEN 0 THEN JOB  WHEN 1 THEN'ALL JOBS'--Replaces NULLS With Value  END DEP,SUM(SAL),COUNT(\*)  FROM EMP GROUPBY DEPTNO, JOB WITHROLLUP |
| **CUBE:** | | |
| 4 | SELECT DEPTNO, JOB,SUM(SAL),COUNT(\*)FROM EMP GROUPBY**CUBE(DEPTNO, JOB)**;  SELECT DISTINCT JOB,  SUM (CASE DEPTNO WHEN 10 THEN SAL END) "D10",  SUM (CASE DEPTNO WHEN 20 THEN SAL END) "D20",  SUM (CASE DEPTNO WHEN 30 THEN SAL END) "D30",  SUM (SAL) "TOTAL SAL"  FROM EMP GROUP BY JOB | SELECT DEPTNO, JOB, SUM(SAL), COUNT(\*) FROM EMP  GROUP BY **DEPTNO, JOB WITHCUBE**  SELECTDISTINCT JOB,  SUM(CASE DEPTNO WHEN 10 THEN SAL END)'D10',  SUM(CASE DEPTNO WHEN 20 THEN SAL END)'D20',  SUM(CASE DEPTNO WHEN 30 THEN SAL END)'D30',  SUM(SAL)'TOTAL SAL'  FROM EMP GROUPBY JOB |
| **HAVING CLAUSE:** | | |
| 5 | Same as in SQL Server | SELECT DEPTNO,SUM(SAL) FROM EMP GROUP BY DEPTNO  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  Condition is made by using relational operators.  It works like “OR” LOGICAL OPERATOR.  It Can be used with all RELATIONAL OPERATORS [ > ANY , < ANY, = ANY, !=ANY , >= ANY, <=  ANY ]  SELECT \* FROM EMP WHERE SAL < ANY (SELECT DISTINCT SAL FROM EMP  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.  It works like “AND” LOGICAL OPERATOR.  It can also be used with ALL Relational Operators.  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:**  It Returns Boolean value ie.. True or False  If Condition at inner query is satisfied than it will written True else written with False  Q) Display DEPTNO, DNAME of those Department’s where atleast one Employe is working.  SELECT DEPTNO, DNAME FROM DEPT DWHERE EXISTS ( SELECT 1 FROM EMP WHERE DEPTNO = D.DEPTNO ) |
| 10 | Same as in SQL Server | **NOT EXISTS:**  This operator also writtens Boolean Value i.e. TRUE or FALSE  If Condition at INNER Query is FALSE then it returns TRUE  Q) Display DEPTNO, DNAME of those Department’s where no Employee is working.  SELECT DEPTNO, DNAME FROM DEPT DWHERE NOT EXISTS (SELECT 1 FROM EMP WHERE DEPTNO = D.DEPTNO)  OR  SELECT DEPTNO,DNAME FROM DEPTWHERE DEPTNO = (( SELECT DEPTNO FROM DEPT )  EXCEPT (SELECT DISTINCT DEPTNO FROM EMP)) |

Database Objects

Synonyms

|  |  |  |
| --- | --- | --- |
| Synonyms | Create synonym syn\_emp  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  Nonclustered  Unique  Non Unique  Composite  Function based Index | Clustered  Nonclustered  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 \*  FROM sys.indexes  WHERE object\_id = OBJECT\_ID('schema.MyTableName') |

Views

|  |  |  |
| --- | --- | --- |
| Creating Views | CREATE VIEW < VIEW\_NAME >  AS  SELECT QUERY  [ WITH CHECK OPTION ] | CREATE VIEW < VIEW\_NAME >  [ WITH ENCRYPTION ]/ [ WITH SCHEMA BINDING ]  AS  SELECT QUERY  [ WITH CHECK OPTION ] |
| To refresh the view |  | SP\_REFRESHVIEW < VIEW\_NAME > |
| Creating Views | Same | CREATE VIEW V1  AS  SELECT \* FROM EMP |
| Check Option View | Same | CREATE VIEW V4  AS  SELECT EMPNO, ENAME FROM EMP WHERE DEPTNO = 10 WITH CHECK OPTION |
| Read Only View | CREATE VIEW V4  AS  SELECT EMPNO, ENAME FROM EMP 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  AS  SELECT \* FROM EMP  create view vw\_emp\_encry with encryption  as select empno,ename,sal from emp |
| Schema binding View | No Concept | CREATE VIEW V10 WITH SCHEMA BINDING  AS  SELECT \* FROM EMP  --Error & also cannot drop base tables  create view vw\_emp\_schema with schemabinding  as select empno,ename,sal from dbo.emp |
| Materialized Or Indexed View Syntax | Materialized View | Indexed View  Creating unique clustered index on views  create unique clustered index inx\_emp1 on vw\_emp\_schema(empno); |
| Information about materialized view/Indexed Views | User\_mviews | select  OBJECT\_SCHEMA\_NAME(object\_id) as [SchemaName],  OBJECT\_NAME(object\_id) as [ViewName],  Name as IndexName  from sys.indexes  where object\_id in  (  select object\_id  from sys.views  )  The inner join version  select  OBJECT\_SCHEMA\_NAME(si.object\_id) as [SchemaName],  OBJECT\_NAME(si.object\_id) as [ViewName],  si.Name as IndexName  from sys.indexes AS si  inner join sys.views AS sv  ON si.object\_id = sv.object\_id |

Sequences Or Identity

|  |  |  |
| --- | --- | --- |
| Creating sequence | Create sequence seq\_emp  Start with 1  Increment by 1  Minvalue 1  Maxvalue 20  Cycle  Cache 5;  To get nextvalue  Select seq\_emp.nextval from dual  To get current value  Select seq\_emp.currval from dual | CREATE SEQUENCE Test.DecSeq  AS int  START WITH 125  INCREMENT BY 25  MINVALUE 100  MAXVALUE 200  CYCLE  CACHE 3  To get nextvalue  SELECT NEXT VALUE FOR Test.DecSeq;  To get current value  SELECT current\_value  FROM sys.sequences  WHERE name = 'Seq\_emp' ; |
| Altering Sequences | alter sequence seq\_emp  increment by 5  cycle;  Start with value cannot be altered. | ALTER SEQUENCE Test. TestSeq  START WITH 100  INCREMENT BY 50  MINVALUE 50  MAXVALUE 200  NO CYCLE  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 \*  from sys.sequences  where object\_id = object\_id('seq\_test')  SYS.Sequences Or SYS.IDENTITY\_COLUMNS |
| start value of the IDENTITY COLUMN | INSERT Test.Orders (OrderID, Name, Qty)  VALUES (Seq\_name.nextval,’Tire’,2);  Seq\_name.nextval(Nextvalue)  Seq\_name.currval(Current Value) | INSERT Test.Orders (OrderID, Name, Qty)  VALUES (NEXT VALUE FOR Test.CountBy1, 'Tire', 2) ;  IDENT\_SEED ( < TABLE\_NAME > )  SELECT IDENT\_SEED ('PRODUCT')  SET IDENTITY\_INSERT < TABLE\_NAME > ON / OFF [ DEFAULT IS OFF ] |
| incremented value of the IDENTITY  COLUMN |  | IDENT\_INCR ( < TABLE\_NAME > )  SELECT IDENT\_INCR ('PRODUCT') |
| To restart the sequence of Numbers for IDENTITY COLUMN | Cycle | DBCC CHECKIDENT ( <TABLE\_NAME>, RESEED , VALUE )  DBCC CHECKIDENT('PRODUCT',RESEED,3) |

With Clause Or CTE

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

Pseudocolumns

|  |  |  |
| --- | --- | --- |
| Rownum | Select \* from emp  Where rownum<=3; | Row\_number() |
| Rowid(To delete duplicate records) | Delete from emp  Where rowid not in(select max(rowid) from tab  Group by col1,col2) | delete from emp  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  Connect by level<=10  with cte(n)  as  (select 1 as n from dual  union all  select n+1 from cte where n<12)  select n from cte | with cte  as  (select 1 as n  union all  select n=n+1 from cte where n<12)  select n from cte |  | User\_name |
| To display month names | with cte(n)  as  (select 1 as n from dual  union all  select n+1 from cte where n<12)  select to\_char(add\_months(trunc(sysdate,'yy'),n-1),'month') from cte  select to\_char(to\_date(level,'mm'),'month')  from dual  connect by level<=12; | with cte  as  (select 1 as n  union all  select n=n+1 from cte where n<12)  select DateName(m,dateadd(m,n-1,0)) from cte |
| To display Day names | select to\_char(trunc(sysdate,'dy')+level-1,'day') from dual  connect by level<=7  with cte(n) as  (select 1 as n from dual  union all  Select n+1 from cte where n<7 )  select to\_char(trunc(sysdate,'dy')+n-1,'day') from cte | with cte as  (select 1 as n  union all  Select n=n+1 from cte where n<7 )  Select datename(dw,dateadd(d,n-1,Datename(yy,getdate()) +'-01-01')) from cte |
| To display multiples of 5 | select level  from dual  where mod(level,5)=0  connect by level<=50  with cte(n) as  (select 1 as n from dual  union all  Select n+1 from cte where n<10 )  Select 5\*n from cte | with cte as  (select 1 as n  union all  Select n=n+1 from cte where n<10 )  Select 5\*n from cte |
| To display ‘Aroha’ vertically | with cte(n)  as  (select 1 as n from dual  union all  select n+1 from cte where n<length('AROHA'))  select substr('AROHA',n,1) from cte | with cte  as  (select 1 as n  union all  select n=n+1 from cte where n<len('AROHA'))  select substring('AROHA',n,1) from cte |
| To display AROHA | select substr('aroha',1,level)  from dual  connect by level<=length('aroha')  with cte(n)  as  (select 1 as n from dual  union all  select n+1 from cte where n<length('AROHA'))  select substr ('AROHA',1,n) from cte | with cte  as  (select 1 as n  union all  select n=n+1 from cte where n<len('AROHA'))  select substring ('AROHA',1,n) from cte |
| To display December dates | with cte(n) as  (select 1 as n from dual  union all  Select n+1 from cte where n<31 )  Select (add\_months(trunc(sysdate,'yy'),11))+n-1 from cte  select to\_date('01-dec-18')+level-1  from dual  connect by level<=to\_char(last\_day('01-dec-18'),'dd'); | with cte as  (select 1 as n  union all  Select n=n+1 from cte where n<31 )  Select dateadd(dd,n-1,dateadd(m,-1,Datename(yy,getdate()) +'-01-01')) from cte |
| Delete duplicate records | Only select and no delete works here. | WITH DuplicateCTE(col1,col2,Row\_num) AS  (  SELECT col1,col2,  ROW\_NUMBER()OVER(PARTITION BY col1,col2 ORDER BY col1,col2)AS Row\_num  FROM dup  )  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  cursor c1 is select \* from emp  Begin  Open c1  Loop  Fetch c1 into variable  Exit when c1%notfound  Dbms\_output.put\_line(variable);  End loop;  Close c;  End; | Declare  C1 cursor for select \* from emp  Open c1  Fetch next from c1 into variables  While @@fetch\_status <>-1  Begin  Print variables  Fetch next from c1 into variables  End  Close c1  Deallocate c1 |

Procedures

|  |  |  |
| --- | --- | --- |
| Creating Procedures | Create or replace procedure sp\_p1(p\_no number, p\_name out varchar2) as  Begin  Select ename into p\_name from emp where empno=p\_no  End; | create procedure sp\_p1(@p\_no int,@p\_name varchar(15) output) as  begin  select @p\_name=ename from emp where empno=@p\_no;  end |
| Executing Procedures With exec command | Exec sp\_p1(7788,:b\_name)  Print :b\_name | Exec sp\_p1 7788 if there is no out parameter |
| Executing procedures with Ananymous block | Declare  V\_name varchar2(20);  Begin  Sp\_p1(7788,v\_name);  Dbms\_output.put\_line(v\_name);  End; | declare  @name varchar(15)  exec sp\_p1 7788,@name output  print @name |
| Altering the procedures |  | Alter procedure sp\_p1  As  Begin  Statements  End |
| Dropping the procedure | Drop procedure sp\_p1 | Drop procedure sp\_p1 |
| Procedures returning values | create or replace procedure sp\_return is  v\_no int:=1;  begin  dbms\_output.put\_line(v\_no);  return;  end; | create procedure sp\_return(@x int,@y int)as  begin  declare @z int  set @z=@x+@y  return @z  end |
| Executing Procedures with returning values | exec sp\_return; | declare @r int  exec @r=sp\_return 10,20  print @r |

Scalar Valued Functions which returns single value

|  |  |  |
| --- | --- | --- |
| Creating Functions | Create function fn\_emp (p\_no number) return number is  V\_cnt number(4);  Begin  Select count(\*) into v\_cnt from emp where deptno=p\_no;  Return p\_no;  End; | create function fn\_emp(@p\_no int) returns int as  begin  declare @v\_cnt int  select @v\_cnt=count(\*) from emp where deptno=@p\_no  return @v\_cnt  end; |
| Executing Functions using select | Select fn\_emp(10) from dual | Select dbo.fn\_emp(10) |
| Executing functions using anonymous blocks | Declare  V\_count number(4);  Begin  V\_count:=fn\_emp(10);  Dbms\_output.put\_line(v\_count);  End; | declare @v\_count int  set @v\_count=dbo.fn\_emp(10)  print @v\_count |
| Executing using exec command | Exec :b\_cnt:=fn\_emp(10)  Print :b\_cnt | No option---- |

Table Valued Functions returning multiple values

|  |  |  |
| --- | --- | --- |
| Creating Functions | Create function fn\_emp(p\_dno number) return sys\_refcursor as  P\_ref sys\_refcursor;  Begin  Open p\_ref for select \* from emp where deptno=p\_dno;  Return p\_ref; | create function fn\_emp\_table(@dno int) returns table as  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  Before insert or update or delete on emp  Begin  If to\_char(sysdate,’dy’) in(‘sat’,’sun’) or to\_char(sysdate,’hh24’) not between 10 and 17 then  Raise\_application\_error(-20001,’not business hours’)  End if;  End; | create trigger trig\_emp  on emp  for insert,delete,update  as  begin  if datename(dw,getdate())='sunday' or datepart(hh,getdate()) not between 10 and 17  begin  rollback  raiserror('Invalid Time',15,16)  end  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;  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

|  |  |  |
| --- | --- | --- |
| Dynamic SQL | Create or replace procedure sp\_dyn(p\_table\_nm varchar2) as  Begin  Execute immediate ‘select \* from ‘||p\_table\_nm;  End; | create procedure sp\_dyn(@table\_nm varchar(15)) as  begin  exec('select \* from '+@table\_nm)  end  exec sp\_dyn emp |
|  | Will not Support | declare  @s varchar(100)='select \* from emp'  exec (@s) |