**PRACTICAL - 3**

**AIM:To perform various data manipulation commands, aggregate functions and sorting concept on all created tables.**

**THEORY:**

**ORACLE FUNCTIONS**

Oracle functions serve the purpose of manipulating data items and returning a result. Functions are capable of accepting user supplied variables or constants and operating on them. Such variable or consonants are termed as arguments. Any number of arguments can be passed to a function in following format:

Function\_Name(argument1, argument2,….)

**AGGREGATE FUNCTIONS – AVG()**

Returns a minimum value of expr.

**Syntax:**

MIN([<DISTINCT>|<ALL>]<expr>)

**Example:**

SELECT MIN(SALARY) “MINIMUM SALARY” FROM EMP;

**AGGREGATE FUNCTIONS – COUNT()**

Returns number of rows where expr is not null.

**Syntax:**

COUNT([<DISTINCT>|<ALL>]<expr>)

**Example:**

SELECT COUNT(FNAME) “EMP COUNT” FROM EMP;

OR  
SELECT COUNT(\*) “NO OF RECORDS” FROM EMP; **//It includes duplicates and null**

**AGGREGATE FUNCTIONS – MAX()**

Returns maximum value of expr.

**Syntax:**

MAX([<DISTINCT>|<ALL>]<expr>)

**Example:**

SELECT MAX(SALARY) “MAXIMUM SALARY” FROM EMP;

**AGGREGATE FUNCTIONS – SUM()**

Returns sum of the values of ‘n’.

**Syntax:**

SUM([<DISTINCT>|<ALL>]<n>)

**Example:**

SELECT SUM(SALARY) “TOTAL SALARY” FROM EMP;

**NUMERIC FUNCTIONS – ABS()**

Returns the absolute value of ‘n’.

**Syntax:**

ABS(n)

**Example:**

SELECT ABS(-15) “ABSOLUTE” FROM DUAL;

**NUMERIC FUNCTIONS – POWER()**

Returns m raised to the nth power. n must be an integer.

**Syntax:**

POWER(m,n)

**Example:**

SELECT POWER(3,2) “RAISED” FROM DUAL;

**NUMERIC FUNCTIONS – ROUND()**

Returns n, rounded to m places to the right of a decimal point. If m is omitted, n is rounded to 0 places. m can be negative to round off digits to left of the decimal point. m must be an integer.

**Syntax:**

ROUND(n[,m])

**Example:**

SELECT ROUND(15.19,1) “ROUNDED VALUE” FROM DUAL;

**NUMERIC FUNCTIONS – SQRT()**

Returns square root of n. If n<0, Null SQRT returns a real result.

**Syntax:**

SQRT(n)

**Example:**

SELECT SQRT(25) “SQUARE ROOT” FROM DUAL;

**NUMERIC FUNCTIONS – EXP()**

Returns e raised to the nth power, where e=2.7182

**Syntax:**

EXP(n)

**Example:**

SELECT SQRT(5) “EXPONENT” FROM DUAL;

**EXTRACT**

Returns a value extracted from a date or an interval value. A DATE can be used only to extract YEAR, MONTH and a DAY, while a timestamp with a time zone data type can be used only to extract TIMEZONE\_HOUR and TIMEZONE\_MINUTE.

**Syntax:**

EXTRACT ({year | month | day | hour | minute | timezone\_hour | timezone\_minute | timezone\_region} FROM {date\_value | interval\_value})

**Example:**

SELECT EXTRACT (YEAR FROM DATE ‘2017-07-02’) “YEAR” FROM DUAL;

**GREATEST**

Returns a greatest value in list of expressions.

**Syntax:**

GREATEST (expr1, expr2………..expr n);

**Example:**

SELECT GREATEST(10,40,20) FROM DUAL;

**LEAST**

Returns a least value in list of expressions.

**Syntax:**

LEAST (expr1, expr2………..expr n);

**Example:**

SELECT LEAST(10,40,20) FROM DUAL;

**MOD**

Returns the remainder of first number divided by second number passed as parameter. If the second number is zero, the result is same as first number.

**Syntax:**

MOD(m, n)

**Example:**

SELECT MOD(17,6) “Mod1”, MOD(12,3) “Mod2” FROM DUAL;

**TRUNC**

Returns a number truncated to a certain number of decimal places. The decimal place value must be an integer. It this parameter is omitted, the TRUNC function will truncate the number to 0 decimal places.

**Syntax:**

TRUNC(number, [decimal\_places])

**Example:**

SELECT TRUNC(123.323,1) “TRUNC1” TRUNC(123.323,-1) “TRUNC2” FROM DUAL;

**FLOOR**

Returns largest integer value that is equal to or less than a number.

**Syntax:**

FLOOR(n)

**Example:**

SELECT FLOOR(24.8) “Floor value” FROM DUAL;

**CEIL**

Returns smallest integer value that is equal to or less than a number.

**Syntax:**

CEIL(n)

**Example:**

SELECT CEIL(24.8) “Floor value” FROM DUAL;

**STRING FUNCTION – LOWER()**

Returns string with all in lower case.

**Syntax:**

LOWER(string)

**Example:**

SELECT LOWER(“HARRY”) FROM DUAL;

**STRING FUNCTION – UPPER()**

Returns string with all in upper case.

**Syntax:**

UPPER(string)

**Example:**

SELECT UPPER(“harry”) FROM DUAL;

**STRING FUNCTION – INITCAP()**

Returns string with first letter of each word in upper case.

**Syntax:**

INITCAP(string)

**Example:**

SELECT INITCAP(“HARRY POTTER”) FROM DUAL;

**STRING FUNCTION – SUBSTR()**

Returns a position of characters, beginning at character m, and going upto character n. If n is omitted, the result returned is upto the last character in the string. The first position of char is 1.

**Syntax:**

SUBSTR(<string>, <start\_position>, [<length>])

**Example:**

SELECT SUBSTR(‘SECURE’,3,4) FROM DUAL; // will give output as SURE

**STRING FUNCTION – TRANSLATE()**

Replaces a sequence of characters in a string with another set of characters. However, it replaces a single character at a time.

**Syntax:**

TRANSLATE(<string>, <string\_to\_replace>, <replacement\_string>)

**Example:**

SELECT TRANSLATE(‘1sct523’,123,,7a9’) FROM DUAL; //will give output as 7scta9

**STRING FUNCTION – LENGTH()**

Returns the length of the word.

**Syntax:**

LENGTH(string)

**Example:**

SELECT LENGTH(‘HARRY’) FROM DUAL;

**PROGRAM EXECUTION:**

**Queries to be performed:**

* 1. List total deposit from deposit.

**Output:**

|  |
| --- |
|  |

* 1. List total loan from karolbagh branch.

**Output:**

|  |
| --- |
|  |

* 1. Give maximum loan from branch vrce.

**Output:**

|  |
| --- |
|  |

* 1. Count total number of customers.

**Output:**

|  |
| --- |
|  |

* 1. Count total number of customer’s cities.

**Output:**

|  |
| --- |
|  |

* 1. Create table supplier from employee with all the columns.

**Output:**

|  |
| --- |
|  |

* 1. Create table sup1 from employee with first two columns.

**Output:**

|  |
| --- |
|  |

* 1. Create table sup2 from employee with no data.

**Output:**

|  |
| --- |
|  |

* 1. Insert the data into sup2 from employee whose second character should be ‘n’ and string should be 5 characters long in employee name field.

**Output:**

|  |
| --- |
|  |

* 1. Delete all the rows from sup1.

**Output:**

|  |
| --- |
|  |

* 1. Delete the detail of supplier whose sup\_no is 103.

**Output:**

|  |
| --- |
|  |

* 1. Rename the table sup2

**Output:**

|  |
| --- |
|  |

* 1. Destroy table sup1 with all the data.

**Output:**

|  |
| --- |
|  |

* 1. Update the value dept\_no to 10 where second character of emp. name is ‘m’.

**Output:**

|  |
| --- |
|  |

* 1. Update the value of employee name whose employee number is 103.

**Output:**

|  |
| --- |
|  |

**VIVA QUESTIONS:**

* Which function returns the remainder in a division operation?
* What is the purpose of the NVL function?
* Say True or False. Give explanation if False.

COUNT(\*) returns the number of columns in a table.

* How do you copy rows from one table to another?