

**Modul Number: 0750362**

**Module Name: Database Applications**

**Teacher: Eman Alnaji**

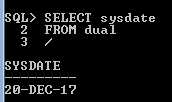
**Part III**

**SQL (Single-row Functions, Views, Sequence)**

**DUAL Table**

* Dummy table
* Consists of one column and one row
* Can be used for table reference in the FROM clause



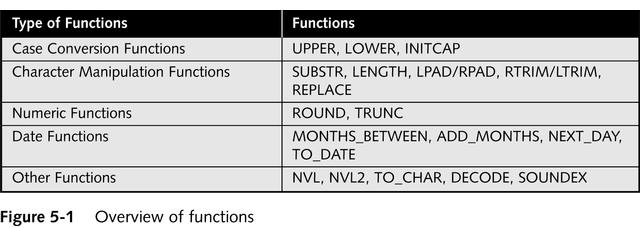


**Sysdate:** Returns the current date stored on your system.

**Single-Row Functions**

**Single-row Function:** returns one row of results for each record processed.

The following are some of the functions that will be discussed shortly.

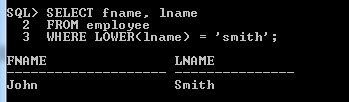


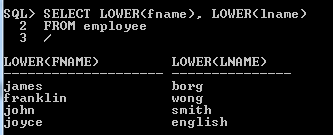
These functions can be used in a Select statement, update or insert. In Select statements, they can be used in the columns part and/or in the where part.

**First: Case Conversion Functions**

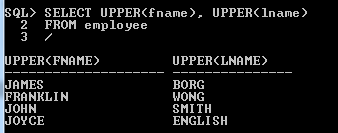
Alter the case of data stored in a column or character string

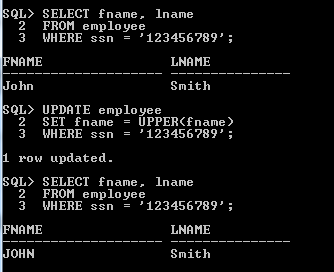
**LOWER:** Used to convert characters to lower-case letters



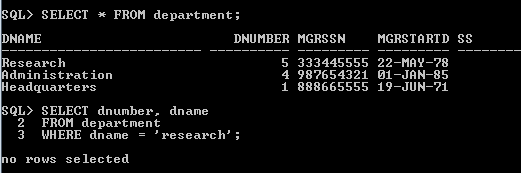


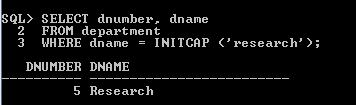
**UPPER:** Used to convert characters to upper-case letters





**INITCAP:** Used to convert characters to mixed-case

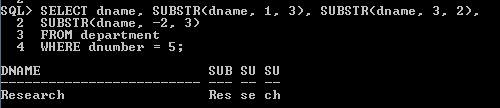




**Second: Character Manipulation Functions**

Manipulates data by extracting substrings, counting number of characters, replacing strings, etc.

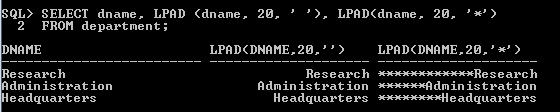
**SUBSTR:** Used to return a substring, or portion of a string



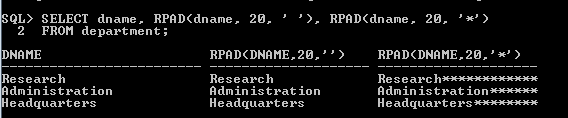
**LENGTH:** Used to determine the number of characters in a string



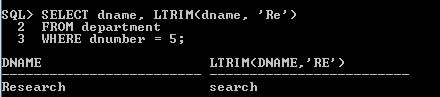
**LPAD:** Used to pad, or fill in, a character string to a fixed width, the filling will be in the left side of the string.



**RPAD:** Used to pad, or fill in, a character string to a fixed width, the filling will be in the right side of the string.



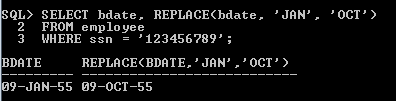
**LTRIM:** Used to remove a specific string of characters from the left of the string.



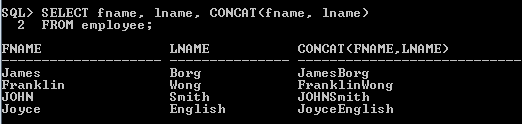
**RTRIM:** Used to remove a specific string of characters from the right of the string.



**REPLACE:** Substitutes a string with another specified string.



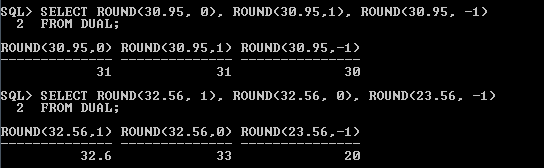
**CONCAT:** Used to concatenate two character strings



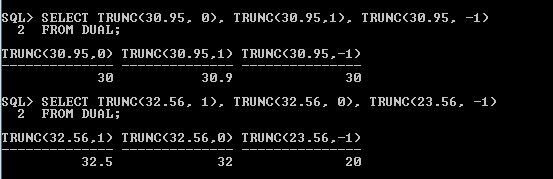
**Third: Number Functions**

Allows for manipulation of numeric data

**ROUND:** Used to round numeric columns to a stated precision

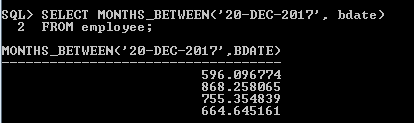


**TRUNC:** Used to truncate a numeric value to a specific position



**Fourth: Date Functions**

**MONTHS\_BETWEEN:** Determines the number of months between two dates



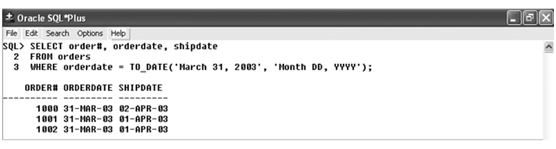
**ADD\_MONTHS:** Adds a specified number of months to a date



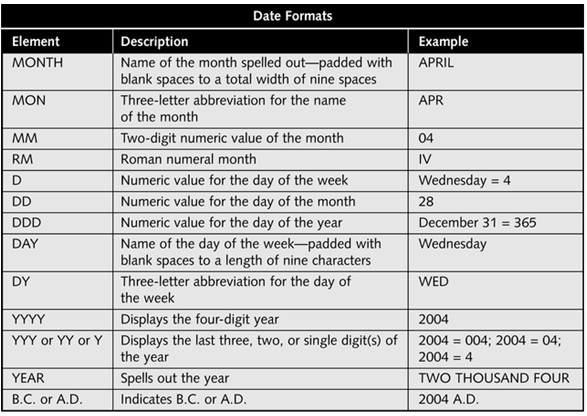
**NEXT\_DAY:** Determines the next occurrence of a specified day of the week after a given date



**TO\_DATE:** Converts various date formats to the internal format (DD-MON-YYYY) used by Oracle9*i*

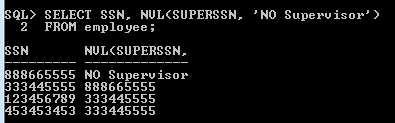


**Format Model Elements – Dates**

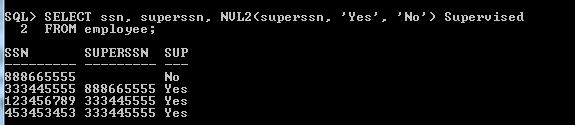


**Fifth: Other Functions**

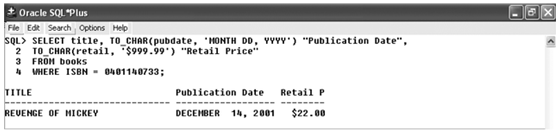
**NVL:** Substitutes a value for a NULL value



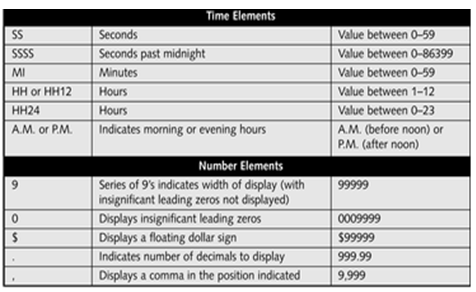
**NVL2:** Allows different actions based on whether a value is NULL



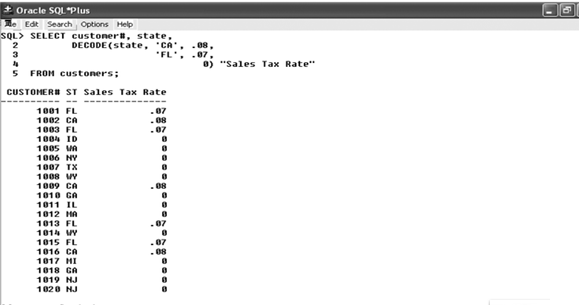
**TO\_CHAR:** Converts dates and numbers to a formatted character string



**Format Model Elements – Time and Number**

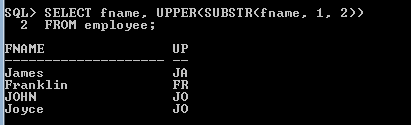


**DECODE:** Determines action based upon values in a list



**Nesting Functions**

* One function is used as an argument inside another function
* Must include all arguments for each function
* Inner function is resolved first, then outer function



**Views**

* A view is a virtual table based on the result-set of a SELECT statement.
* A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.
* You can add SQL functions, WHERE, and JOIN statements to a view and present the data as if the data were coming from one single table.

**Specification of Views**

* a table (view) name
* a possible list of attribute names (for example, when arithmetic operations are specified or when we want the names to be different from the attributes in the base relations)
* a query to specify the table contents

**Example on Views:**

* Specify a different WORKS\_ON table

**CREATE VIEW** WORKS\_ON\_NEW **AS**

**SELECT** FNAME, LNAME, PNAME, HOURS

**FROM** EMPLOYEE, PROJECT, WORKS\_ON

**WHERE** SSN = ESSN

**AND** PNO = PNUMBER;

* We can specify SQL queries on a newly create table (view):

**SELECT** FNAME, LNAME

**FROM** WORKS\_ON\_NEW

**WHERE** PNAME=‘Seena’;

* When no longer needed, a view can be dropped:

**DROP VIEW** WORKS\_ON\_NEW;

If the view definition needs to change, a CREATE OR REPLACE VIEW command is used.

**CREATE OR REPLACE VIEW** WORKS\_ON\_NEW **AS**

**SELECT** FNAME, LNAME, PNAME, HOURS, PNO

**FROM** EMPLOYEE, PROJECT, WORKS\_ON

**WHERE** SSN = ESSN

**AND** PNO = PNUMBER;

**Advantages of Views:**

* To restrict data access
* To make complex queries easy
* To provide data independence
* To present different views of the same data
* You can retrieve data from a view as from a table.

**Sequences**

* Automatically generates unique numbers
* Is typically used to create a primary key

**Create Sequence Syntax:**

**CREATE SEQUENCE *sequence\_name***

**[INCREMENT BY *n*]**

**[START WITH *n*]**

**[{MAXVALUE *n* | NOMAXVALUE}]**

**[{MINVALUE *n* | NOMINVALUE}];**

**Example of Creating a Sequence:**

CREATE SEQUENCE deptid\_seq

INCREMENT BY 10

START WITH 5

MAXVALUE 9999;

* **NEXTVAL** returns the next available sequence

value. It returns a unique value every time it is

referenced, even for different users

* **CURRVAL** obtains the current sequence value.

**NEXTVAL** must be issued for that sequence before

**CURRVAL** contains a value

**Using a sequence:**

INSERT INTO dept(deptno, dname, loc)

VALUES (deptid\_seq.NEXTVAL,'Support', ' HONG KONG' );

**View the Current Value:**

SELECT deptid\_seq.CURRVAL

FROM dual;

**Removing a Sequence:**

DROP SEQUENCE deptid\_seq;