**LESSON 3 Introduction to SQL**

**1. What is SQL?**

SQL is a language that allows to express operations of definition of data structures, of authorizations of access, of query and updating of data stored in relational databases.

The word SQL is formed by the English initials of "Structured Query Language".

The first version of SQL was developed by IBM in 70s. Given the success of this first version, other computer companies such as Oracle, Unisys and Digital developed their own SQL languages for all of microcomputers, minicomputers and mainframes. This has led SQL to be a widely used tool in the World of relational databases. This trend was consolidated because ANSI (American National Standard Institute) developed its own specifications for the SQL language, which were subsequently accepted by the ISO (International Standard Organization).

[There are still small differences among the various SQL products on the market, that are easily understood when de programmers look up the reference manual of the product being used.]

The main features of SQL versus traditional systems are:

* It allows to handle record sets instead of record-to-record.
* It has a strong theoretical-mathematical base which gives a great expressive capacity with a very simple structure. This allows for a single sentence to express complex queries that traditionally required one or more programs.
* It can be used interactively allowing to perform unplanned queries (i.e. not included in programs), so the end users can directly access the data.

**2. How to use SQL**

Sentences can be directed to the system in a two-fold way:

* Interactive mode: The sentence is written directly on the computer screen, executed and the results of the query appear on the same screen.
* SQL embedded in a programming language: SQL statements are inserted in a program to access data from a relational database. The accessed results are collected by the program that will treat them. For example, to list them.

The language in which SQL statements are inserted is called *host language,* and it can be C++, Java, PL/SQL, etc....

There are two techniques for using SQL embedded in programs:

* + Static SQL: The SQL statements included in the program cannot be changed during execution.
  + Dynamic SQL: A SQL statement can be completely or partially modified by the program itself during execution.

**3. What is SQL used for?**

The SQL consists of three sublanguages:

**Data Definition language (DDL)**: that allows to create, alter and delete objects from the database such as tables, views, indexes and constraints. These tasks are the database administrator'. The sentences are:

* **CREATE:** To create objects.
* **ALTER**: To alter already created objects in the database.
* **DROP**: To delete objects from the database.

**Data Manipulation Language (DML)**: that allows to add, update, query and delete data from the database.

* **SELECT**: To query data from one or more tables.
* **INSERT:** To add one or more rows in a table.
* **UPDATE:** To modify the column value of one or more rows in a table.
* **DELETE**: To delete rows from a table.

**Data Control Language** (**DCL**): that allows to assign and remove authorizations of access to the data, and to control transactions.

* **GRANT**: To grant access authorizations.
* **REVOKE:** To remove access authorizations.
* **COMMIT**: To confirm transactions.
* **ROLLBACK**: To abort transactions.

**4. Elements of a SQL statement**

SQL statements have the following components:

* Reserved words: Words that have a predefined meaning in the language. They can be a sentence, a clause or a function. All sentences begin with a reserved word.
* Names of tables, views and columns: they are words invented by the creators of the database and are assigned in the creation of the table.
* Names of host variables: they are names of program variables that are inserted within a sentence.
* Constants: they are alphanumeric character sequences (single quoted) or numbers.
* Delimiting signs: parentheses, relational operators, arithmetic operators, blank spaces, etc...

Example:

Select nomcli into nombre\_cliente from clientes where codcli = 20;

Reserved words: Select, into, from, where.

Names of tables, views and columns: nomcli, codcli, clientes.

Names of host variables: nombre\_cliente.

Constants: 20

Delimiting signs: , = espacio en blanco.

**First use of SQL Developer: Environment presentation**

* SQL Developer is the Oracle tool for interactive SQL.
* There are several predefined users:
  + SYS (which is the owner of the Data dictionary)
  + SYSTEM (which is DBA)
  + ANONYMOUS (which is user for testing and guests)
* Connect as ANONYMOUS
* The SELECT statement is used to query data. Its simplest format is

SELECT columns FROM table;

* There is a virtual table to be able to use a SELECT statement without consulting any table called DUAL, it is used for quick calculations and tests.
* There is a default SQL function SYSDATE, that returns the system date.

Example: Get the system date

Select SYSDATE from DUAL system;

**5. Columns Data types.**

The data types that a column of a table may be declared in Oracle are:

1. **NUMERIC VALUES** include integer numbers of various sizes (INTEGER or INT, and SMALLINT) and floating-point (real) numbers of various precision (FLOAT or REAL, and DOUBLE PRECISION).

**NUMBER(p, s)**: to declare numeric values. The p and s parameters are optional.

* **p** parameter indicates the maximum amount of total digits that the number can have.
* **s** parameter indicates the maximum number of digits the decimal part will have.

If *scale* s is not specified, the numeric data will be an integer.

If *precision* p is not specified, the numeric data will default to 38 digits (*implementation-defined*) and the value is stored as it is assigned to the column.

The **numeric operators** are: + - \* /

Example: How Oracle stores the number value 6123.89 with different precisions and scales in the numeric column definition:

NUMBER 6123.89

NUMBER(6) 6124 rounds

NUMBER(6,2) 6123.89

NUMBER(6,1) 6123.9 rounds

NUMBER(6,-2) 6100 rounds hundreds

1. **CHARACTER-STRING DATA TYPES** are either fixed length or varying length:

* **CHAR(n)** fixed length where n is the number of characters. (For fixed-length strings, a shorter string is padded with blank characters to the right).
* **VARCHAR(n)** varying length where n is the maximum number of characters.

When specifying a literal string value, it is placed between single quotation marks (apostrophes), and it is case sensitive (a distinction is made between uppercase and lowercase).

* **LONG** is another variable-length string data type called *Character Large Object* or *CLOB* to specify columns that have large text values, such as documents.

Maximum length can be specified in kilobytes (K), megabytes (M), or gigabytes (G).

The **character operator** is ***concatenation*** denoted by || (double vertical bar) that can concatenate two strings in SQL.

1. **BIT-STRING DATA TYPES** are either of fixed length or varying length:

* **RAW**(n): contains binary information. Its maximum length is 255 bytes.
* **LONG RAW**: contains binary information. It is used to store graphic information, sounds, etc... *Binary Large Object* or *BLOB* is available to specify columns that have large binary values, such as images, videos, sounds and other kind of multimedia values.

Maximum length can be specified in kilobytes (K), megabytes (M), or gigabytes (G).

1. **DATE:** Stores date and time data.

**6. DATE Data-type Programming**

* It stores date and time type data.
* It permits value ranges from June 1st, 4712 BC to December 30th, 4712 D.C.
* In the queries the data are presented in the default format DD-MM-YY

For example, 03-10-18 indicates October 3rd, 2018.

* The stored datum is actually a numerical value that is applied to a conversion algorithm and the presentation format is given by the NLS\_DATE\_FORMAT environment variable found in the REGEDIT at HKEY\_LOCAL\_MACHINE\Software\ORACLE\KEY\_OraDB10g-home1.

(Run REGEDIT to change the variable NLS\_DATE\_FORMAT, press Windows + R)

(It could be also changed by SQL Developer properties)

* This is the information displayed that contains the date data-type in its default format, but it actually stores the year, the month, the day, the time, the minute, and the second.

To see it completely, you would have to use the format 'YYYY-MM-DD/HH:MI:SS'

**6.1 FUNCTIONS FOR DATE TREATMENT**

* **SYSDATE** returns the system date.
* **TO\_CHAR (data [, mask [, ' nlsparams '])** converts the DATE type data to the VARCHAR2 type according to the format specified in the mask.

The nlsparams parameter specifies the language in which the names and abbreviations of the months and days of the week appear, with the following format 'NLS\_DATE\_LANGUAGE = Language'.

**Mask Items for dates:**

The separators can be : - / . ; “Text”

MM month number 1..12

MON abbreviation of the month. Jan

MONTH name of the month filled with spaces up to a length of 9 characters.

RM month in Roman numerals. I

DDD number of the day in the year 1..365

DD number of the day in the month 1..31

D number of the day in the week 1..7

DY abbreviation of the day Mon

DAY complete name of the day Monday

YYYY year using 4 digits. 2018

YY year using 2 digits. 18

YEAR year using words

Q quadrimester number 1..3

WW week in the year . 1 al 53.

W week in the month . 1 a 5.

HH Hour . 1..12

HH12 Hour 1-12

HH24 Hour 1-24

MI Minute . 1..60

SS Second . 1..60

AM Shows AM or PM depending on the time

\* Exercise: Display the current year.

\* Exercise: Display the name of the month in which we are, first in Spanish and then in English.

\* Exercise: Display the system date and time.

* **TO\_DATE (data [, mask [, 'nlsparams']]** converts the char-type data or varchar-type data to the DATE type. The mask is a date format in which the datum is represented. If the mask is omitted, the datum must be in the default date format.

**6.2. Assignment of value to a DATE-type column**

* You can directly assign alphanumeric value to a column using the default format, because the system automatically makes the conversion.

- Create Table TEST\_TABLE (ORDEN number, FECHA date);

- Insert Into TEST\_TABLE Values(1, '04-10-2018');

- Select FECHA From TEST\_TABLE;

- Insert Into TEST\_TABLE Values(2, '2018-10-04'); Error

* If a value is assigned to a date-type column, in a format other than the default format, it must use the TO\_DATE function that converts a String-type datum to a date-type datum.

\* Exercise: Insert a row with the values ‘2018-10-04/21:30:45’.

* If a value is assigned to a date-type column without indicating hour, minute and second, it is stored by default 00:00:00, that is to say, midnight.

\* Exercise: Insert a row with the date of the system.

\* Exercise: Select the records inserted today, using the date of the system.

\* Exercise: Insert a row with the current date, omitting the hours, minutes and seconds.

\* Exercise: Select the records inserted in the current month.

**6.3. Date operators + , –**

You can use the + and – operators with date data-type.

* If an integer *n* is added to a date, it returns the date after *n* days.
* If an integer n is subtracted from a date, it returns the date before n days.
* If you subtract two dates returns the number of days between them.

Select sysdate - to\_date (' 12-10-2018', ' DD-MM-YYYY ') From dual;

Note: This sentence returns decimals because of hours and minutes.

\* Exercise: Display the day of the week (in letters) that will be in 5 days

* If you subtract or add fractions of integer, it returns the date after the hours or minutes or seconds. Integer fractions for
  + - * hours 1/24
      * minutes 1/1440
      * seconds 1/86400

\* Exercise: Display date and time ten minutes before

\* Exercise: Display date and time two hours before

\* Exercise: Display date and time 80 seconds later

\* Exercise: Calculate how many days you have lived up until now.