Chapter 2: SELECT

Objectives

- Introduce the SQL language.
- Introduce SQL*Plus
- Examine Oracle datatypes
- Write simple queries
- Restrict and sort data

SQL FUNDAMENTALS

- SQL is a data sub-language
- SQL statements are instructions to the database
- Features of SQL include:
  - Processes sets of data as groups
  - Automatic navigation to the data
  - Stand alone statements
- SQL lets you work with data at the logical level
- SQL provides commands for tasks such as:
  - Querying data
  - Inserting, updating & deleting rows in a table
  - Creating, replacing, altering and dropping objects
  - Controlling access to the database and its objects
  - Guaranteeing database consistency and integrity

SQL FUNDAMENTALS

- 6 SQL Statement Categories
  - **Data Definition (DDL)** – Used to define, alter or drop database objects. (e.g. CREATE, ALTER, DROP etc.)
  - **Data Control (DCL)** –Used to control the database including security issues (e.g GRANT, REVOKE etc.)
  - **Data Manipulation (DML)** – Used to access, create or manipulate data in existing structures of the database (e.g. SELECT, INSERT, UPDATE, DELETE etc.)
  - **Transaction Control** – Used to manage changes made by the DML statements, whether to save them or discard them (e.g. COMMIT, ROLLBACK, SAVEPOINT etc.)
  - **Session Control** – Used to manage the properties of a user's session (e.g. SET ROLE, ALTER SESSION)
  - **System Control** – Used to manage the properties of the database (e.g. ALTER SYSTEM)

SQL vs SQL*Plus

SQL

- A language
- ANSI standard
- Keyword cannot be abbreviated
- Statements manipulate data and table definitions in the database

SQL*Plus (includes iSQL Plus)

- An environment
- Oracle proprietary
- Keywords can be abbreviated
- Commands do not allow manipulation of values in the database

SQL*Plus

Some SQL*Plus commands include:

- ACCEPT
- BREAK
- COLUMN
- DEFINE
- DESCRIBE
- EDIT
- GET
- PROMPT
- REPLACE
- SAVE
- SET
- SHOW
- SPOOL
- START
- STORE SET
- TITLE

SQL*Plus

- Logging in to SQL*Plus from a Windows environment

  User Name:
  Password:
  Host string:

- Use the SQL*Plus DESCRIBE command to display the structure of a table.

  DESC[RIBE] ***tablename***

- Displaying a table structure

```
SQL> DESCRIBE dept

Name              Null?    Type
----------------- -------- ------------
DEPTNO            NOT NULL NUMBER(2)
DNAME                      VARCHAR2(14)
LOC                        VARCHAR2(13)
```

SQL*Plus

```
SELECT     [DISTINCT] {*,column [alias],...}
FROM       table;
```

- Use SQL*Plus as an environment to:
  - Execute SQL statements
  - Edit SQL statements

Oracle SQL Datatypes

Datatype	Description
VARCHAR2(*size*)	Variable-length character data
CHAR(*size*)	Fixed-length character data
NUMBER(*p*,*s*)	Variable-length numeric data
DATE	Date and time values
LONG	Variable-length character data up to 2 gigabytes
CLOB	Single-byte character data up to 4 gigabytes
RAW and LONG RAW	Raw binary data
BLOB	Binary data up to 4 gigabytes
BFILE	Binary data stored in an external file; up to 4 gigabytes

Basic SELECT Statement

```
SELECT     [DISTINCT] {*,column [alias],...}
FROM       table;
```

- SELECT identifies *what* columns
- FROM identifies *which* table

SQL Statements

- SQL statements are not case sensitive
- SQL statements can be on one or more lines
- Keywords cannot be abbreviated or split across lines
- Clauses are usually placed on separate lines
- Tabs and indents are used to enhance readability

SELECT all columns

```
SQL> SELECT *
  2  FROM dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON
```

SELECT specific columns

```
SQL> SELECT deptno, dname
  2  FROM   dept;

DEPTNO DNAME
------------- --------------
        10 ACCOUNTING
        20 RESEARCH
        30 SALES
        40 OPERATIONS
```

Arithmetic Expression

- Can create expressions on NUMBER and DATE fields by using arithmetic operators.

Operator	Description
+	Add
-	Subtract
*	Multiply
/	Divide

Using Arithmetic Operators

```
SQL> SELECT ename, sal, sal*12
  2  FROM   emp;

ENAME           SAL     SAL*12
----------------- ---------- --------------
SMITH             800        9600
ALLEN            1600       19200
WARD             1250       15000
JONES            2975       35700
MARTIN           1250       15000
. . .
14 rows selected.
```

Operator Precedence

- Multiplication and division take priority over addition and subtraction
- Operators of the same priority are evaluated from left to right
- Parentheses are used to force prioritized evaluation and to clarify statements.

*	/	+	-

Operator Precedence

```
SQL> select ename, sal, 12*sal+500
  2  from emp;

ENAME           SAL 12*SAL+500
--------------- ----------- ---------------
SMITH             800         10100
ALLEN            1600         19700
WARD             1250         15500
JONES            2975         36200
MARTIN           1250         15500
```

Using Parentheses

```
 SQL> select ename, sal, 12*(sal+500)
     2   from emp;

ENAME           SAL 12*(SAL+500)
--------------- ----------- ----------------
SMITH             800          15600
ALLEN            1600          25200
WARD             1250          21000
JONES            2975          41700
MARTIN           1250          21000
```

Defining a Column Alias

- Renames a column heading
- Is useful with calculations
- Immediately follows the column name
- Can use the option AS keyword between the column name and alias
- Requires double quotation marks if it contains spaces or special characters or is case sensitive

Using Column Aliases

```
SQL> select ename AS surname, sal salary
  2  from emp;

SURNAME       SALARY
---------- ----------
SMITH            800
ALLEN           1600
WARD            1250
JONES           2975
MARTIN          1250
```

Concatenation Operator

- Concatenate columns or character strings to other columns
- Is represented by two vertical bars (||)
- Creates a resultant column that is a character expression

Using the Concatenation Operator

```
SQL> select deptno || dname AS "Department"
  2  from dept;

Department
--------------------------------------------------
10ACCOUNTING
20RESEARCH
30SALES
40OPERATIONS
```

Literal Character Strings

- A literal is a character, expression, or number included in the SELECT list.
- Date and character literal values must be enclosed within single quotation marks.
- Each character string is output once for each row returned.

Using Literal Character Strings

```
SQL> select deptno || ' is the number for the ' || dname || ' department.'
  2  AS "Department"
  3  from dept;

Department
--------------------------------------------------------------------------------
10 is the number for the ACCOUNTING department.
20 is the number for the RESEARCH department.
30 is the number for the SALES department.
40 is the number for the OPERATIONS department.
```

Null Values

- Is an unknown value
- Is NOT a blank string
- It is a special value
- Very important to handle Null values
- Checking for a null value

```
SELECT ename, comm
FROM emp
WHERE comm IS NULL;
```

- To distinguish between a blank string and a null value, can use one of Oracle's built-in functions to convert a null value to another value.

```
SELECT ename, NVL(comm, 'No commission') "Commission"
FROM emp;
```

Duplicate Rows

– The default display of queries is all rows including duplicate rows

```
SQL> select job
  2  from emp;

JOB
---------
CLERK
SALESMAN
SALESMAN
MANAGER
SALESMAN
…
```

Eliminating Duplicate Rows

– Eliminate duplicate rows by using the DISTINCT keyword in the SELECT clause.

```
SQL> select DISTINCT job
  2  from emp;

JOB
-----------------
ANALYST
CLERK
MANAGER
PRESIDENT
SALESMAN
```

Restricting and Sorting Data

Limiting Rows Selected

- Restrict the rows returned by using the WHERE clause
- The WHERE clause follows the FROM clause

```
SELECT  [DISTINCT] {*| column [alias], …}
FROM    table
[WHERE  condition(s)];
```

Using the WHERE clause

```
SQL> select ename, job
  2  from emp
  3  where job = 'CLERK';

ENAME      JOB
---------- ---------
SMITH      CLERK
ADAMS      CLERK
JAMES      CLERK
MILLER     CLERK
GATES      CLERK
```

Character Strings and Dates

- Character strings and date values are enclosed in single quotation marks
- Character values are case sensitive and date values are format sensitive
- The default date format is DD-MON-YY

```
SQL> select ename, job, hiredate
    2    from emp
    3    where ename = 'SMITH';

ENAME      JOB       HIREDATE
---------- --------- ----------
SMITH      CLERK     17-DEC-80
```

Comparison Operators

Operator	Meaning
=	Equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
<>	Not equal to

Using the Comparison Operators

```
SQL> select ename, sal
  2  from emp
  3  where sal >= 3000;

ENAME           SAL
----------- ----------
SCOTT          3000
KING           5000
FORD           3000
```

Other Comparison Operators

Operator	Meaning
BETWEEN …AND…	Between two values (inclusive)
IN (list)	Match any of a list of values
LIKE	Match a character pattern
IS NULL	Is a null value

BETWEEN

```
SQL> select ename, sal
 2  from emp
 3  where sal BETWEEN 900 AND 1100;

ENAME           SAL
---------- ----------
ADAMS          1100
JAMES           950
GATES           900
```

IN

```
SQL> select ename,mgr
 2  from emp
 3  where mgr IN (7566, 7788);

ENAME           MGR
---------- ----------
SCOTT          7566
ADAMS          7788
FORD           7566
```

Using the LIKE Operator

- Use the LIKE operator to perform wildcard searches of valid search string values
- Search conditions can contain either literal characters or numbers
  - % denotes zero or many characters
  - _ denotes one character
- You can combine pattern-matching characters

LIKE

```
SQL> select ename
  2  from emp
  3  where ename LIKE 'S%';

ENAME
----------
SMITH
SCOTT
```

Logical Operators

Operator	Meaning
AND	Returns TRUE if both component conditions are TRUE
OR	Returns TRUE if either component condition is TRUE
NOT	Returns TRUE if the following condition is false

AND

```
 SQL> select ename, sal, job
  2  from emp
  3  where job = 'SALESMAN'
  4  AND sal >= 1500;

ENAME           SAL JOB
--------------- ------------ ----------------
ALLEN           1600 SALESMAN
TURNER          1500 SALESMAN
```

OR

```
SQL> select ename, sal, job
  2  from emp
  3  where job = 'SALESMAN'
  4  OR sal >= 1500;

ENAME            SAL JOB
---------- ---------- ---------
ALLEN           1600 SALESMAN
WARD            1250 SALESMAN
JONES           2975 MANAGER
MARTIN          1250 SALESMAN
. . .
10 rows selected.
```

NOT

```
SQL> select ename, job
  2  from emp
  3  where job NOT IN ('SALESMAN','CLERK','MANAGER');

ENAME      JOB
---------- ---------
SCOTT      ANALYST
KING       PRESIDENT
FORD       ANALYST
```

Rules of Precedence

Order Evaluated	Operator
1	All comparison operators
2	NOT
3	AND
4	OR

Example 1.

```
 SQL> select ename, job, sal
  2  from emp
  3  where job = 'SALESMAN‘
  4  OR    job = 'PRESIDENT‘
  5  AND   sal > 1500;

ENAME      JOB                 SAL
---------- ---------- ----------
ALLEN      SALESMAN          1600
WARD       SALESMAN          1250
MARTIN     SALESMAN          1250
KING       PRESIDENT         5000
TURNER     SALESMAN          1500
```

Example 2.

```
SQL> select ename, job, sal
  2  from emp
  3  where (job = 'SALESMAN'
  4  OR    job = 'PRESIDENT')
  5  AND   sal > 1500;

ENAME      JOB                 SAL
---------- ----------------- ----------
ALLEN      SALESMAN      1600
KING       PRESIDENT     5000
```

ORDER BY Clause

- Sort rows with the ORDER BY clause
    - ASC: ascending order, default
    - DESC: descending order
- The ORDER BY clause comes last in the SELECT statement

ORDER BY

```
SQL> select ename, job, sal
  2  from emp
  3  ORDER BY sal;

ENAME      JOB                 SAL
---------- ----------------- ----------
SMITH      CLERK              800
GATES      CLERK              900
JAMES      CLERK              950
ADAMS      CLERK             1100
WARD       SALESMAN          1250
```

ORDER BY - DESC

```
SQL> select ename, job, sal
  2  from emp
  3  ORDER BY sal DESC;

ENAME      JOB                 SAL
---------- ----------------- ----------
KING       PRESIDENT         5000
SCOTT      ANALYST           3000
FORD       ANALYST           3000
JONES      MANAGER           2975
BLAKE      MANAGER           2850
```

Sorting by Column Alias

```
SQL> select ename, sal*12 "Annual Salary"
  2  from emp
  3  ORDER BY "Annual Salary";

ENAME      Annual Salary
---------- --------------------
SMITH                 9600
GATES                10800
JAMES                11400
. . .
```

Sorting by Multiple Columns

- The order of the ORDER BY list is the order of the sort
- You can sort by a column that is not in the SELECT list

Sorting

```
SQL> select ename, job, sal
  2  from emp
  3  ORDER BY job, sal DESC;

ENAME      JOB               SAL
---------- --------------- ------------
SCOTT      ANALYST          3000
FORD       ANALYST          3000
MILLER     CLERK            1300
ADAMS      CLERK            1100
JAMES      CLERK             950
GATES      CLERK             900
SMITH      CLERK             800
```

Summary

- Introduced the SQL language.
- Introduced SQL*Plus
- Examined Oracle datatypes
- Wrote simple queries using SELECT
- Restricted and sorted data in a SELECT