DBMS/SQL

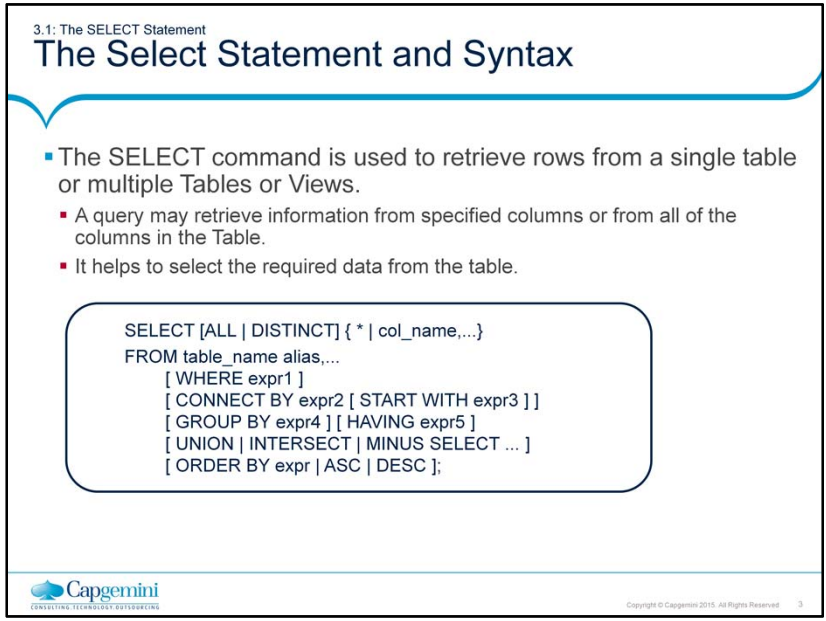
Lesson 03 Data Query Language
(The Select Statement)

Lesson Objectives

- To understand the following topics:
  - The SELECT statement
    - The WHERE clause
    - The DISTINCT clause
    - The Comparison, Arithmetic, and Logical operators
    - The ORDER BY clause

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 2



The SELECT Statement:
The SELECT statement is used to select data from a table. The tabular result is stored in a result table (called the result-set). The statement begins with the SELECT keyword. The basic SELECT statement has three clauses:

SELECT
FROM
WHERE

The SELECT clause specifies the table columns that are retrieved.
The FROM clause specifies the tables accessed.
The WHERE clause specifies which table rows are used. The WHERE clause is optional; if missing, all table rows are used.
Note:
Each clause is evaluated on the result set of a previous clause. The final result of the query will be always a “result table”.
Only FROM clause is essential. The clauses WHERE, GROUP BY, HAVING, ORDER BY, UNION are optional.
All the examples that follow are based on EMP and DEPT tables that are already available.

Selecting Columns

- Displays all the columns from the student_master table.

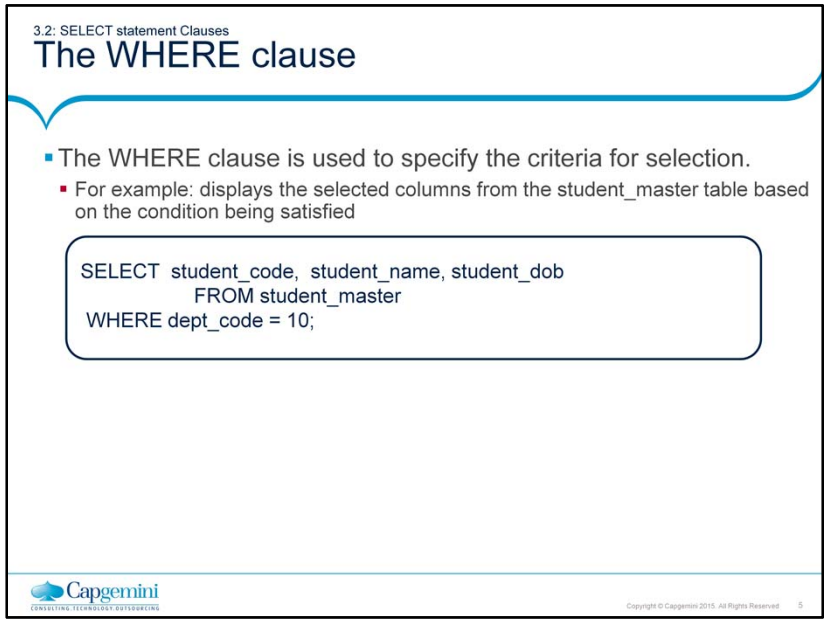
```
SELECT  *  FROM student_master;
```

- Displays selected columns from the student_master table

```
SELECT student_code,
        student_name
          FROM student_master;
```

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 4



The WHERE Clause:
The WHERE clause is used to perform “selective retrieval” of rows. It follows the FROM clause, and specifies the search condition.
The result of the WHERE clause is the row or rows retrieved from the Tables, which meet the search condition.
The clause is of the form:

Comparison Predicates:
The Comparison Predicates specify the comparison of two values.
It is of the form:

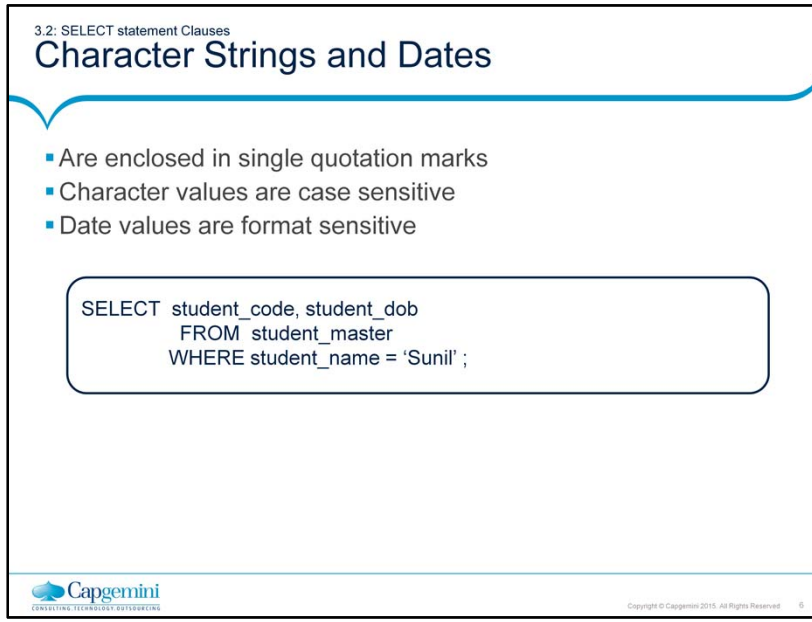
< Expression> < operator > < Expression>

< Expression> <operator> <subquery>

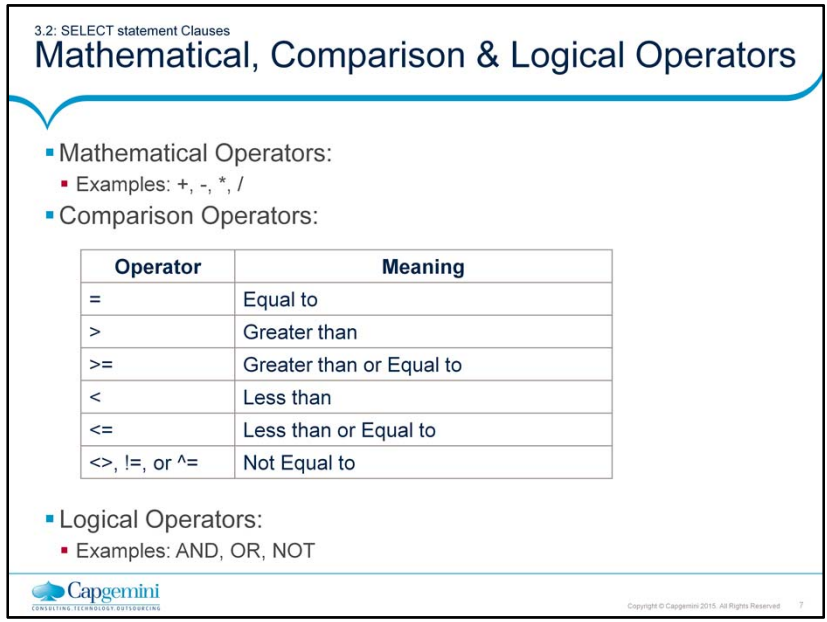
The operators used are shown on the next slide:

```
WHERE <search condition>
```

contd.



Oracle Database store dates in an internal numeric format, representing the century, year, month, day, hours, minutes, and seconds.
The date datatype is covered in detail later.



Operators:
Operators are used in “expressions” or “conditional statements”. They show equality, inequality, or a combination of both.
Operators are of three types:

- mathematical
- logical
- range (comparison)

These operators are mainly used in the WHERE clause, HAVING clause in order to filter the data to be selected.
Mathematical operators:
These operators add, subtract, multiply, divide, and compare equality of numbers and strings. They are +, -, *, /
Comparison Operators:
These operators are used to compare the column data with specific values in a condition. “Comparison Operators” are also used along with the “SELECT statement” to filter data based on specific conditions. The table in the slide describes each Comparison operator. Comparison operators indicate how the data should relate to the given search value.
Logical Operators:
There are three Logical Operators namely AND, OR and NOT. These operators compare two conditions at a time to determine whether a row can be selected for the output or not. When retrieving data by using a SELECT statement, you can use logical operators in the WHERE clause. This allows you to combine more than one condition.

3.2: SELECT statement Clauses

Other Comparison Operators

Other Comparison operators	Description
[NOT] BETWEEN x AND y	Allows user to express a range. For example: Searching for numbers BETWEEN 5 and 10. The optional NOT would be used when searching for numbers that are NOT BETWEEN 5 AND 10.
[NOT] IN(x,y,…)	Is similar to the OR logical operator. Can search for records which meet at least one condition contained within the parentheses. For example: Pubid IN (1, 4, 5), only books with a publisher id of 1, 4, or 5 will be returned. The optional NOT keyword instructs Oracle to return books not published by Publisher 1, 4, or 5.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 8

3.2: SELECT statement Clauses

Other Comparison Operators

Other Comparison operators	Description
[NOT] LIKE	Can be used when searching for patterns if you are not certain how something is spelt. For example: title LIKE 'TH%'. Using the optional NOT indicates that records that do contain the specified pattern should not be included in the results.
IS[NOT]NULL	Allows user to search for records which do not have an entry in the specified field. For example: Shipdate IS NULL. If you include the optional NOT, it would find the records that do not have an entry in the field. For example: Shipdate IS NOT NULL.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 9

3.2: SELECT statement Clauses

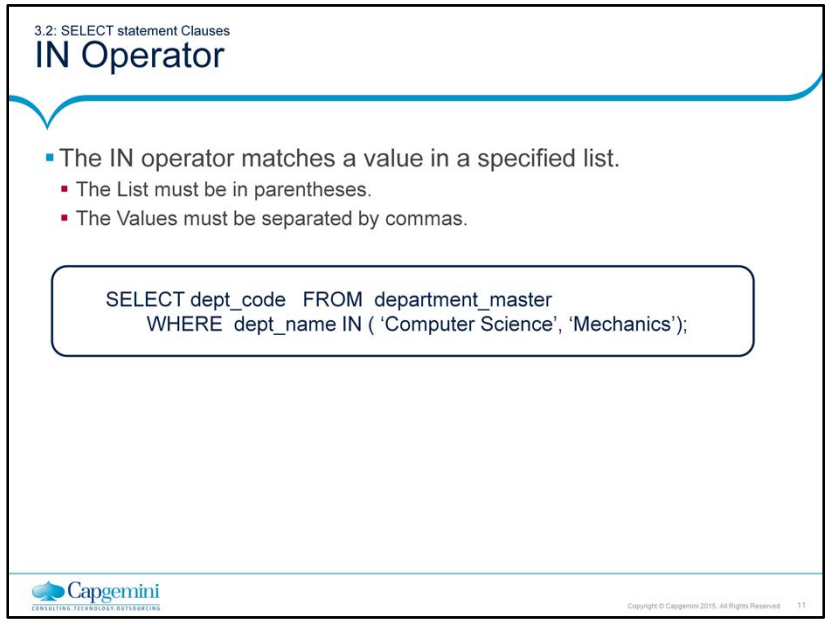
BETWEEN … AND Operator

- The BETWEEN … AND operator finds values in a specified range:

```
SELECT staff_code,staff_name  FROM  staff_master
WHERE  staff_dob
        BETWEEN '01-Jan-1980'
        AND  '31-Jan-1980';
```

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

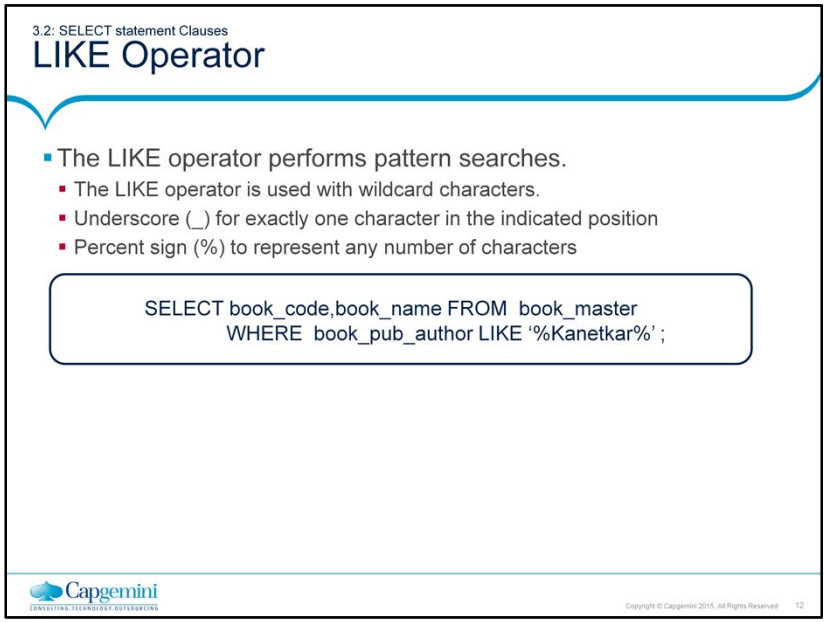
 10



IN predicate:
It is of the form:

<Expression> IN <LIST>
<Expression> IN <SUBQUERY>

The data types should match.

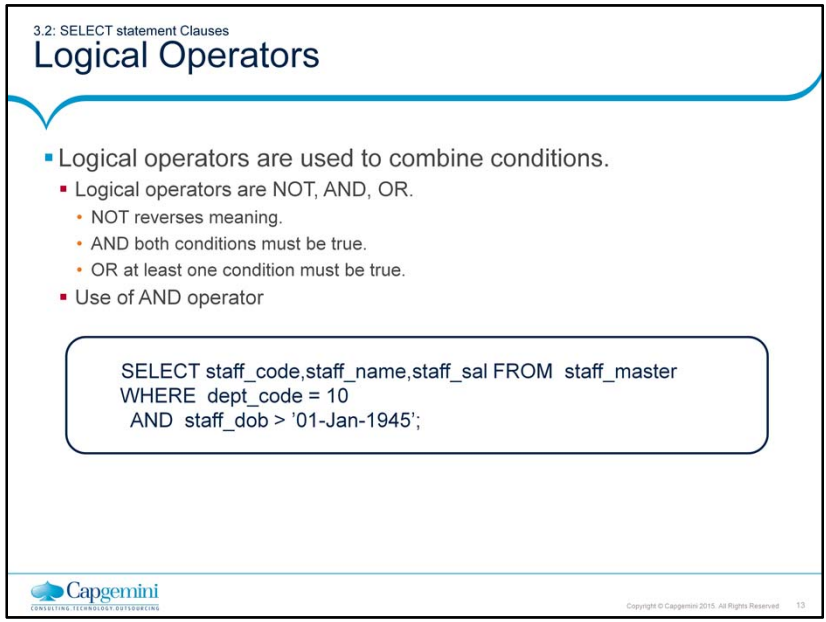


LIKE predicate:
It is of the form:

<COLUMN > LIKE < PATTERN>

The pattern contains a search string along with other special characters % and _. The % character represents a string of any length where as _ (underscore) represents exactly one character. A pattern %XYZ% means search has to be made for string XYZ in any position. A pattern '_XYZ%' means search has to be made for string XYZ in position 2 to 4.
To search for characters % and _ in the string itself we have to use an “escape” character.

For example: To search for string NOT_APP in column status, we have to use the form Status like 'NOT_APP' ESCAPE '\'
The use of \ as escape character is purely arbitrary.

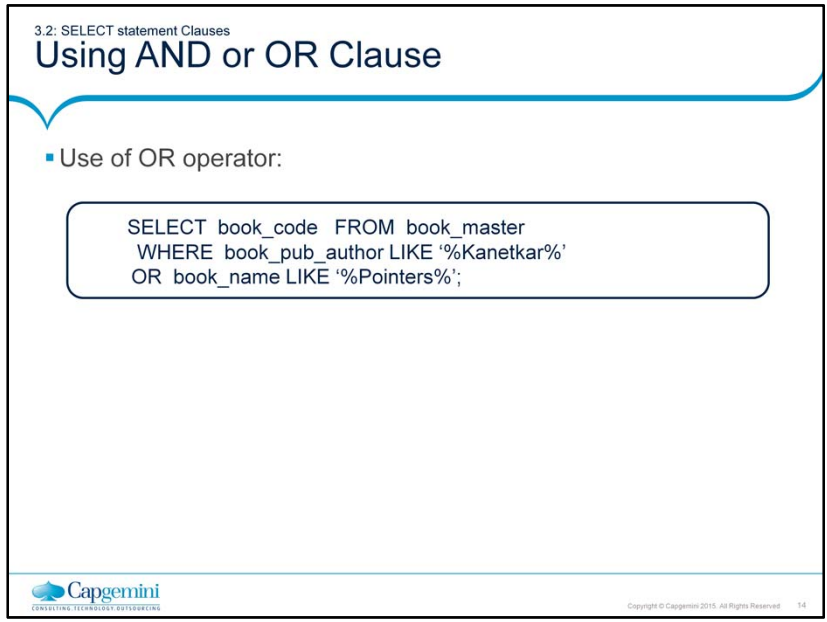


The AND operator displays a record if both the first condition and the second condition is true.
One More Example:

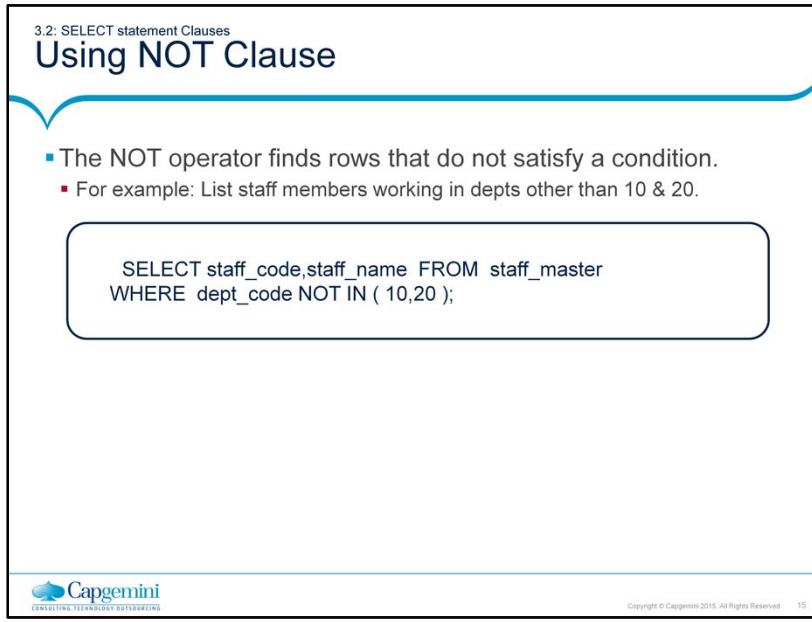
```
SQL> SELECT title, pubid, category
  2  FROM books
  3  WHERE pubid = 3
  4  AND category = 'COMPUTER';
```

Combining Predicates by using Logical Operators:
The predicates can be combined by using logical operators like AND, OR, NOT. The evaluation proceeds from left to right and order of evaluation is:

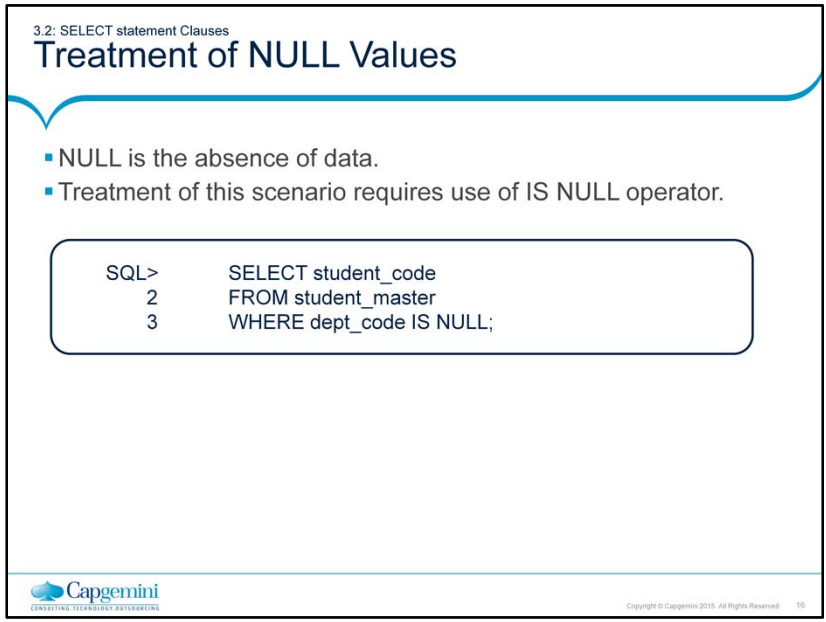
* Enclosed in parenthesis
AND
OR



The OR operator displays a record if either the first condition or the second condition is true.
You can also combine AND and OR as shown in above example. (use parenthesis to form complex expressions).



Note: NOT is a negation operator.



NULL predicate:
The NULL predicate specifies a test for NULL values. The form for NULL predicate is:

< COLUMN SPECIFICATION > IS NULL.
< COLUMN SPECIFICATION > IS NOT NULL.
< COLUMN SPECIFICATION > IS NULL returns TRUE only when column has NULL values.
<COLUMN> = NULL cannot be used to compare null values.

Operator Precedence

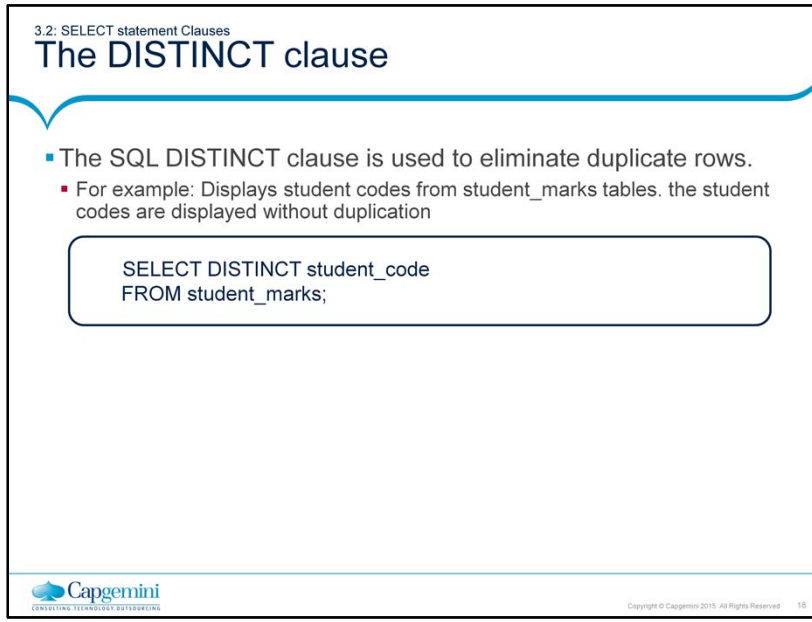
- Operator precedence is decided in the following order:

Levels	Operators
1	* (Multiply), / (Division), % (Modulo)
2	+ (Positive), - (Negative), + (Add), (+ Concatenate), - (Subtract), & (Bitwise AND)
3	=, >, <, >=, <=, <>, !=, !>, !< (Comparison operators)
4	NOT
5	OR
6	AND
7	ALL, ANY, BETWEEN, IN, LIKE, OR, SOME
8	= (Assignment)

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 17

Operator Precedence:
When a complex expression has multiple operators, the operator precedence (or order of execution of operators) determines the sequence in which the operations are performed.
The order of execution can significantly affect the resulting value.
The operators have the precedence levels as shown in the table given in the slide.
An operator on higher levels is evaluated before an operator on lower level.



The DISTINCT clause:
In the examples discussed so far, some of the values have been repeated. However, by default, all values are retrieved. If you wish to remove duplicate values, then use the query as shown in the slide above.

Retrieval of Constant values by using Dual Table

A “dual” is a table, which is created by Oracle along with the data dictionary. It consists of exactly one column, whose name is dummy, and one record. The value of that record is X.

```
SQL>desc dual;
Name                    Null?    Type
DUMMY                            VARCHAR2(1)
Sql>Select * from dual;
D
-
X
```

The owner of dual is SYS. However, “dual” can be accessed by every user.

As “dual” contains exactly one row (unless someone has fiddled with it), it is guaranteed to return exactly one row in SELECT statements. Therefore, dual

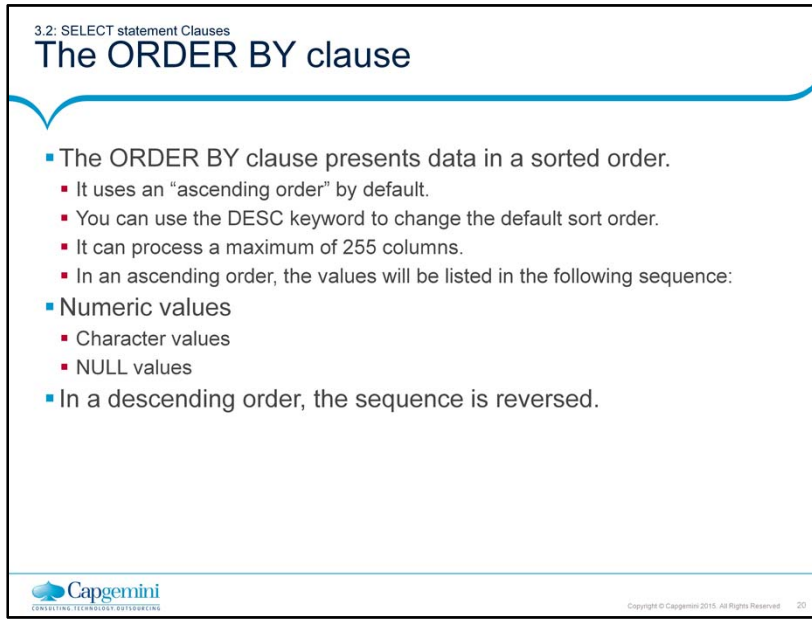
```
SQL>select sysdate from dual;
```

```
SQL>SELECT (319/212)+10 FROM DUAL;
```

For example, you can use it for math:

```
SQL>SELECT employee_seq.NEXTVAL
FROM DUAL;
```

And, you can use it to increment sequences:



The Order By Clause:

- A query with its various clauses (FROM, WHERE, GROUP BY, HAVING) determines the rows to be selected and the columns. The order of rows is not fixed unless an ORDER BY clause is given.
- An ORDER BY clause is of the form:

```
ORDER BY < Sort  list> ASC/DESC
```

- The columns to be used for ordering are specified by using the "column names" or by specifying the "serial number" of the column in the SELECT list.
- The sort is done on the column in "ascending" or "descending" order. By default the ordering of data is "ascending" order.

contd.

3.2: SELECT statement Clauses

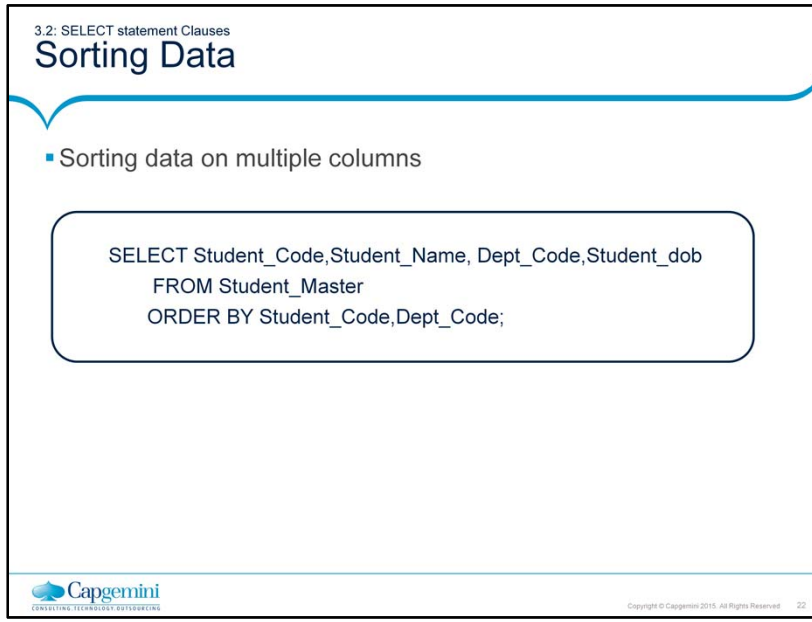
Sorting Data

- The output of the SELECT statement can be sorted using ORDER BY clause
  - ASC : Ascending order, default
  - DESC : Descending order
- Display student details from student_master table sorted on student_code in descending order.

```
SELECT Student_Code,Student_Name,Dept_Code,
       Student_dob
FROM Student_Master
ORDER BY Student_Code DESC ;
```

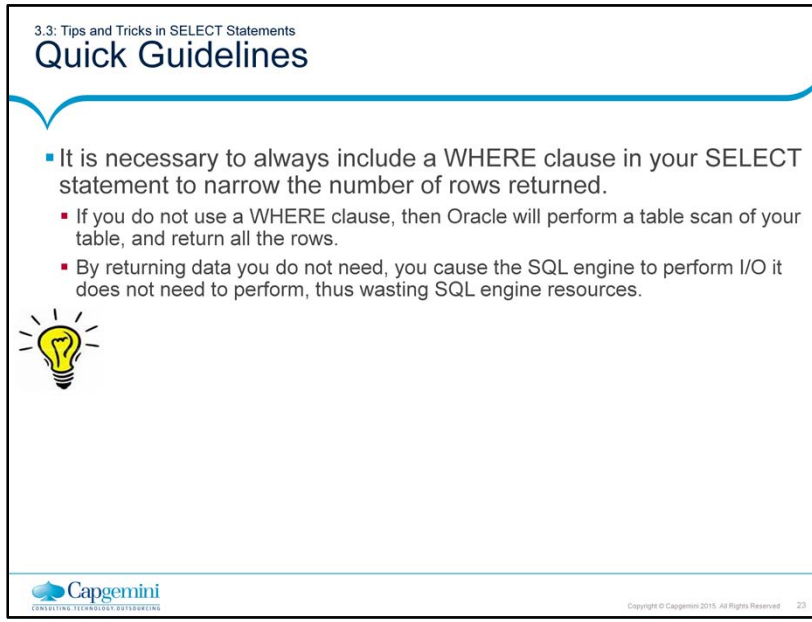
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 21



The query on the slide sorts the data on both the columns in ascending order which is default. But you could also sort the data in different order for the columns.
For Example in the query given below the data is sorted in ascending order on student_code and dept_code is sorted in descending order

```
SELECT
Student_Code,Student_Name,Dept_Code,Student_do
b
FROM Student_Master
ORDER BY Student_Code,Dept_Code DESC;
```



Tips and Tricks in SELECT Statements:

It is necessary to always include a WHERE clause in your SELECT statement to narrow the number of rows returned.

In some case you may want to return all rows. Then not using a WHERE clause is appropriate in this case.

However, if you don't need all the rows to be returned, use a WHERE clause to limit the number of rows returned.

Another negative aspect of a table scan is that it will tend to flush out data pages from the cache with useless data. This reduces ability of the Oracle to reuse useful data in the cache, which increases disk I/O and decreases performance.

Quick Guidelines

- In addition, the above scenario increases network traffic, which can also lead to reduced performance.
- And if the table is very large, a table scan will lock the table during the time-consuming scan, preventing other users from accessing it, and will hurt concurrency.

- In your queries, do not return column data that is not required.
  - For example:
    - You should not use SELECT * to return all the columns from a table if all the data from each column is not required.
    - In addition, using SELECT * prevents the use of covered indexes, further potentially decreasing the query performance.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 24

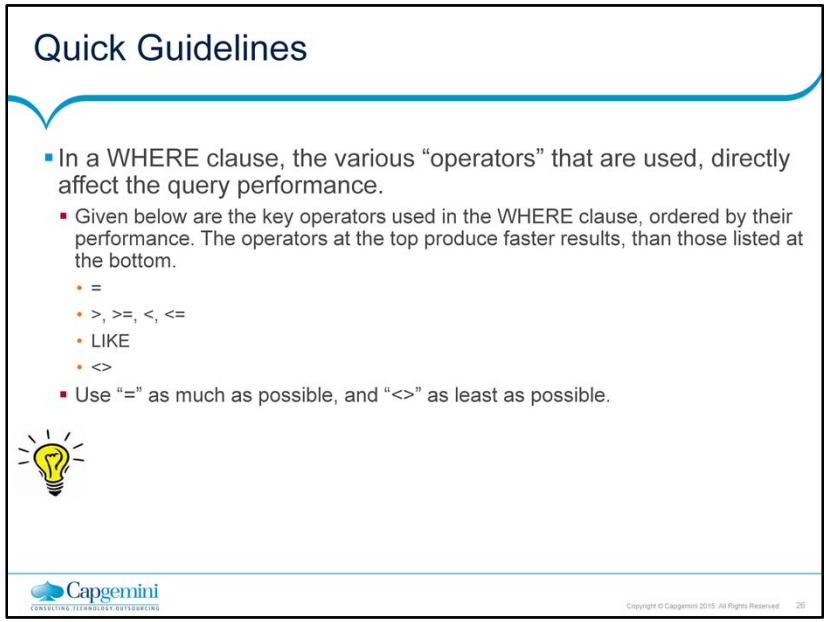
Quick Guidelines

- Carefully evaluate whether the SELECT query requires the DISTINCT clause or not.
  - The DISTINCT clause should only be used in SELECT statements.
    - This is mandatory if you know that "duplicate" returned rows are a possibility, and that having duplicate rows in the result set would cause problems with your application.
  - The DISTINCT clause creates a lot of extra work for SQL Server.
    - The extra load reduces the "physical resources" that other SQL statements have at their disposal.
  - Hence, use the DISTINCT clause only if it is necessary.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 25

Tips and Tricks in SELECT Statements (contd.):
Some developers, as a habit, add the DISTINCT clause to each of their SELECT statements, even when it is not required.
This is a bad habit that should be stopped.



Tips and Tricks in SELECT Statements (contd.):

Use simple operands

Some operators tend to produced speedy results than other operators. Of course, you may not have choice of using an operator in your WHERE clauses, but sometimes you do have a choice.

Using simpler operands, and exact numbers, provides the best overall performance.

If a WHERE clause includes multiple expressions, there is generally no performance benefit gained by ordering the various expressions in any particular order.

This is because the Query Optimizer does this for you, saving you the effort. There are a few exceptions to this, which are discussed further in the lesson.

contd.

Tips and Tricks in SELECT Statements (contd.):

Don't include code that does not do anything

- This may sound obvious. However, this scenario is seen in some off-the-shelf applications.
  - For example, you may see code which is given below:

```
SELECT column_name FROM table_name  WHERE 1
= 0
```

  - When this query is run, no rows will be returned. It is just wasting SQL Server resources.

- By default, some developers routinely include code, which is similar to the one given above, in their WHERE clauses when they make string comparisons.
  - For example:

```
SELECT column_name FROM table_name
    WHERE LOWER(column_name) = 'name'
```

Any use of text functions in a WHERE clause decreases performance.

- If your database has been configured to be case-sensitive, then using text functions in the WHERE clause does decrease performance. However, in such a case, use the technique described below, along with appropriate indexes on the column in question:

```
SELECT column_name FROM table_name
WHERE column_name = 'NAME' or column_name = 'name'
```

- This code will run much faster than the first example.

Quick Guidelines

- If you use LIKE in your WHERE clause, try to use one or more leading character in the clause, if at all possible.
  - For example: Use LIKE 'm%' not LIKE '%m'

- Certain operators in the WHERE clause prevents the query optimizer from using an Index to perform a search.
  - For example: "IS NULL", "<>", "!=", "!>", "!<", "NOT", "NOT EXISTS", "NOT IN", "NOT LIKE", and "LIKE '%500'"

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

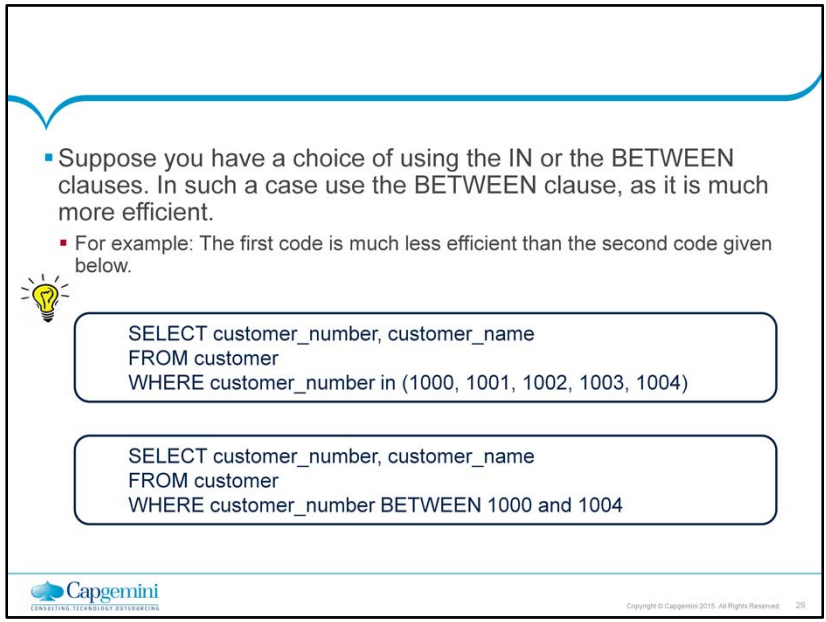
 28

Tips and Tricks in SELECT Statements (contd.):

If you use a leading character in your LIKE clause, then the Query Optimizer has the ability to potentially use an Index to perform the query. Thus speeding performance and reducing the load on SQL engine.

However, if the leading character in a LIKE clause is a "wildcard", then the Query Optimizer will not be able to use an Index. Here a table scan must be run, thus reducing performance and taking more time.

The more leading characters you use in the LIKE clause, it is more likely that the Query Optimizer will find and use a suitable Index.



Tips and Tricks in SELECT Statements (contd.):

Assuming there is a useful Index on customer_number, the Query Optimizer can locate a range of numbers much faster by using BETWEEN clause.

This is much faster than it can find a series of numbers by using the IN clause (which is really just another form of the OR clause).

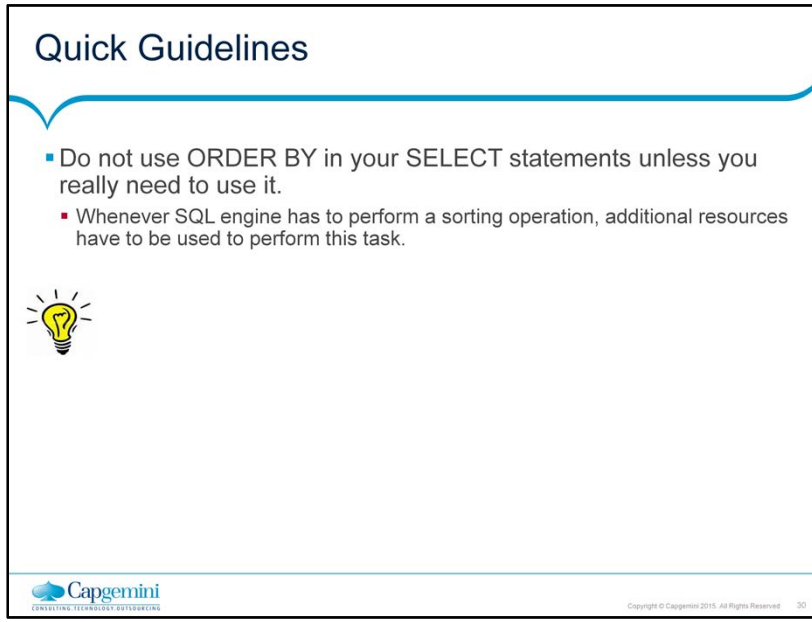
Using Efficient Non-index WHERE clause sequencing:

Oracle evaluates un-indexed equations, linked by the AND verb in a bottom-up fashion. This means that the first clause (last in the AND list) is evaluated, and if it is found TRUE, then the second clause is tested.

Always try to position the most expensive clause first in the WHERE clause sequencing.

Oracle evaluates un-indexed equations, linked by the OR verb in a top-down fashion. This means that the first clause (first in the OR list) is evaluated, and if it is found FALSE, then the second clause is tested.

Always try to position the most expensive OR clause last in the WHERE clause sequencing.



Tips and Tricks in SELECT Statements (contd.):
Don't use ORDER BY in your SELECT statements unless you really need to: The ORDER BY clause adds a lot of extra overhead.

For example: Sometimes it may be more efficient to sort the data at the client than at the server. In other cases, the client does not even need sorted data to achieve its goal. The key here is to remember that you should not automatically sort data, unless you know it is necessary. Whenever SQL Server has to perform a sorting operation, additional resources have to be used to perform this task. Sorting often occurs when any of the following Transact-SQL statements are executed:

ORDER BY
GROUP BY
SELECT DISTINCT
UNION
CREATE INDEX (generally not as critical as happens much less often)

In many cases, these commands cannot be avoided. On the other hand, there are few ways in which sorting overhead can be reduced, like:

Keep the number of rows to be sorted to a minimum. Do this by only returning those rows that absolutely need to be sorted.
Keep the number of columns to be sorted to the minimum. In other words, do not sort more columns than required.
Keep the width (physical size) of the columns to be sorted to a minimum.
Sort column with number datatypes instead of character datatypes.

Summary

- In this lesson, you have learnt:
  - What is SELECT statement?
  - Usage of the following:
    - The WHERE clause
    - The DISTINCT clause
    - The Comparison, Arithmetic, and Logical operators
    - The AND or OR clause
    - The NOT clause
    - The ORDER BY clause

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 31

Review – Questions

- Question 1: The ___ table consists of exactly one column, whose name is “dummy”.
- Question 2: The LIKE operator comes under the ___ category.
  - Option 1: mathematical
  - Option 2: comparison
  - Option 3: logical
- Question 3: The ___ specifies the order in which the operators should be evaluated.

Review – Questions

- Question 4: The NOT NULL operator finds rows that do not satisfy a condition.
  - True / False
- Question 5: More than one column can also be used in the ORDER BY clause.
  - True / False