|  |
| --- |
|  |
| SQL - MODULES |
|  |
|  |
|  |
|  |

CONTENTS

1. SQL ( Overview) & Other SQL Platforms
2. SQL Structure
3. Database
4. Table
5. Data Types
6. Constraints
7. Primary Key
8. Foreign Key
9. Check
10. Default
11. Not null
12. Unique
13. Index (Different Types of Index)
14. SQL Commands
15. DDL Commands (CREATE,DROP,ALTER,RENAME,TRUNCATE)
16. Table Backup
17. DML Commands(INSERT,UPDATE,DELETE)
18. Commit
19. Top of the stack (Efftdt, Efftseq)
20. Data Fetch Logic
21. Data Fetch Querying Logic (Basic)
22. Select Clause & Select Distinct
23. Where Clause(AND, OR, NOT, IN, NOT IN, BETWEEN)
24. Where Clause - Wild Cards (Like & Not Like)
25. Order By Clause
26. Data Fetch Querying Logic (Advanced)
27. Case Statements
28. Aggregate Functions (COUNT,MIN,MAX,SUM,AVG),LISTAGG
29. Group By \* Having Clause
30. Order by with Aggregate Functions
31. Join and Union
32. Joins Logic
33. Inner Join
34. Left Outer Join
35. Right Outer Join
36. Full Outer Join
37. Self Join
38. Union Logic
39. Union
40. Union All
41. Difference Between Join and Union
42. Inner Query or Sub Query
43. Aliases
44. Other Inbuilt Functions & Methods
    1. Trim
    2. Round
    3. To\_char, To\_date
    4. Lpad&Rpad
    5. Initcap
    6. Len
    7. sysdate
    8. Row\_number
    9. Coalesce
    10. Minus
    11. Pivot &Unpivot
    12. Substring
    13. Replace
    14. CAST
45. PL/SQL (Dynamic Query)

**SQL ( Overview) & Other SQL Platforms:**

* SQL stands for Structured Query Language.
* Structured Query Language (SQL) is the set of statements with which all programs and users access data in an Oracle database.
* Application programs and Oracle tools often allow users access to the database without using SQL directly, but these applications in turn must use SQL when executing the user's request.

SQL Platforms**:**

* MySQL.
* PostgreSQL.
* Oracle.
* Microsoft's SQL Server.

**SQL Structure:**

1. Database:

A database is an organized collection of structured data stored electronically in a computer system.

1. Create Database Syntax - CREATE DATABASE databasename;
2. Drop Database Syntax - DROP DATABASE databasename;
3. Backup Database Syntax - BACKUP DATABASE databasenameTO DISK = 'filepath';
4. Table:

Tables are the basic unit of data storage in an Oracle Database. Data is stored in rows and columns.

1. Data Types:

When you create a table or cluster, you must specify a datatype for each of its columns.These datatypes define the domain of values that each column can contain.

For example, DATE columns cannot accept the value February 29 (except for a leap year) or the values 2 or 'SHOE'. Each value subsequently placed in a column assumes the datatype of the column. For example, if you insert '01-JAN-98' into a DATE column, then Oracle treats the '01-JAN-98' character string as a DATE value after verifying that it translates to a valid date.

1. Integer
2. Character Varying
3. Date
4. Timestamp
5. Text
6. Nvarchar2(Oracle)
7. Varchar(Oracle)
8. Constraints**:**

SQL constraints are used to specify rules for the data in a table.Constraints are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the table.

Constraints can be column level or table level. Column level (Column-level constraints refer to a single column in the table )constraints apply to a column, and table level (Table-level constraints refer to one or more columns in the table.)constraints apply to the whole table.

Types of constraints:

1. Primary Key - A primary key is a column of a combination of columns in a table that uniquely identifies a row in the table.

Primary Key Rules:

* A primary key column cannot contain a NULL value or an empty string.
* A primary key value must be unique within the entire table.
* A primary key value should not be changed over time.

1. Foreign Key - A foreign key is a way to enforce referential integrity within your Oracle database. A foreign key means that values in one table must also appear in another table. The referenced table is called the parent table while the table with the foreign key is called the child table.
2. Default - The DEFAULT constraint is used to insert a default value into a column. The default value will be added to all new records, if no other value is specified.
3. Check - The CHECK constraint is used to limit the value range that can be placed in a column.

If you define a CHECK constraint on a column it will allow only certain values for this column.

1. NOT NULL - An Oracle NOT NULL constraint specifies that a column cannot contain NULL values.
2. Unique - A unique constraint is an integrity constraint that ensures the data stored in a column, or a group of columns, is unique among the rows in a table.

**Index:**

An index is a performance-tuning method of allowing faster retrieval of records. An index creates an entry for each value that appears in the indexed columns. By default, Oracle creates B-tree indexes.

**SQL Commands:**

1. DDL Commands
2. Create Table

The CREATE TABLE statement is used to create a new table in a database.

Syntax:

CREATE TABLE table\_name (

column1 datatype,

column2 datatype,

column3 datatype,

....);

1. Drop Table

The DROP TABLE statement is used to drop an existing table in a database.

syntax :

DROP TABLE table\_name;

1. Alter Table

**ALTER TABLE statement** to add a column, modify a column, drop a column, rename a column or rename a table

* Add Column - ALTER TABLE table\_name

ADD column\_name datatype;

* Drop Column - ALTER TABLE table\_name

DROP COLUMN column\_name;

* Modify Column - ALTER TABLE table\_name

MODIFY column\_namecolumn\_type;

1. Rename

* Rename Column - ALTER TABLE table\_name

RENAME COLUMN old\_name TO new\_name;

* Rename Table - ALTER TABLE table\_name

RENAME TO new\_table\_name

1. Truncate Table

* It is used to delete the entire data of the table.
* It can’t be used with where clause.
* It is faster than delete

Syntax: TRUNCATE TABLE table\_name;

**Backup table**

Syntax: create table table\_name1 as

select \* from table\_name2;

1. DML Commands
2. Insert:

INSERT statement to insert data into a table.

* First, specify the name of the table into which you want to insert.
* Second, specify a list of comma-separated column names within parentheses.
* Third, specify a list of comma-separated values that corresponds to the column list.

Syntax:

Method 1

1. INSERT INTO table\_name (column\_list)

VALUES(value\_list);

1. INSERT INTO table\_name

VALUES (value\_list);

Method 2

INSERT INTO table\_name1

SELECT \* FROM table\_name2;

2. Update:

The UPDATE statement is used to modify the existing records in a table.

Syntax: UPDATE table\_name

SET

column1 = value1,

column2 = value2,

column3 = value3,

...

WHERE condition;

3. Delete:

* DELETE statement to delete one or more rows from a table.
* First, you specify the name of the table from which you want to delete data.
* Second, you specify which row should be deleted by using the condition in the WHERE clause. If you omit the WHERE clause, the Oracle DELETE statement removes all rows from the table.

Syntax: DELETEFROMtable\_nameWHEREcondition;

1. Commit

Use the COMMIT statement to end your current transaction and make permanent all changes performed in the transaction.

**Top of the stack (Efftdt, Efftseq):**

This logic can be used to fetch latest or top most data for each individual unique records.

**Data Fetch Logic:**

1. Data Fetch Querying Logic(Basic)

* Select:

The SELECT statement is used to select data from a database.

Syntax:

1. SELECT column1, column2, ...

FROM table\_name;

1. SELECT \* FROM table\_name;

* Select Distinct:

SELECT statement to filter duplicate rows in the result set.

Syntax:SELECT DISTINCT column\_1

FROM table;

* Where:

WHERE clause to specify a condition for filtering rows returned by a query.

Syntax:SELECT column1, column2, ...

FROM table\_name

WHERE condition;

* Where Clause - Wild Card

The LIKE& NOT LIKE operator is used in a WHERE clause to search for a specified pattern in a column.

There are two wildcards often used in conjunction with the LIKE operator:

* + - * The percent sign (%) represents zero, one, or multiple characters
      * The underscore sign (\_) represents one, single character

Syntax: SELECT column1, column2, ...

FROM table\_name

WHERE column LIKE pattern;

SELECT column1, column2, ...

FROM table\_name

WHERE column NOT LIKE pattern;

* Order by:

ORDER BY clause to sort the result set by one or more columns in ascending or descending order.

Syntax:SELECT column1, column2, ...

FROM table\_name

ORDER BY column1, column2, ... ASC|DESC;

1. Data Fetch Querying Logic(Advanced)

Aggregate functions:

* Oracle aggregate functions calculate on a group of rows and return a single value for each group.
* We commonly use the aggregate functions together with the GROUP BY clause. The GROUP BY clause divides the rows into groups and an aggregate function calculates and returns a single result for each group.
* If you use aggregate functions without a GROUP BY clause, then the aggregate functions apply to all rows of the queried tables or [views](https://www.oracletutorial.com/oracle-view/).
* We also use the aggregate functions in the HAVING clause to filter groups from the output based on the results of the aggregate functions.

AVG:

The Oracle AVG() function accepts a list of values and returns the average.

Syntax:SELECT AVG(column\_name)FROM table\_nameWHERE condition;

SUM:

SUM() function to calculate the sum of all or distinct values in a set.

Syntax: SUM( [ALL | DISTINCT] expression)

COUNT:

The Oracle COUNT() function is an aggregate function that returns the number of items in a group.

Syntax: COUNT( [ALL | DISTINCT | \* ] expression)

MAX:

MAX() function to return the maximum value from a set of values.

Syntax:MAX( expression );

MIN:

MIN() function to return the minimum value from a set of values.

Syntax: MIN( expression );

LISTAGG:

LISTAGG() function to transform data from multiple rows into a single list of values separated by a specified delimiter.

Syntax:LISTAGG (measure\_column [, 'delimiter'])

WITHIN GROUP (order\_by\_clause) [OVER (query\_partition\_clause)]

Group by:

* The GROUP BY clause is used in a SELECT statement to group rows into a set of summary rows by values of columns or expressions. The GROUP BY clause returns one row per group.
* The GROUP BY clause is often used with aggregate functions such as AVG(), COUNT(), MAX(), MIN() and SUM().

Having:

It is used to filter groups of rows returned by the GROUP BY clause. This is why the HAVING clause is usually used with the GROUP BY clause.

Order by with aggregate functions:

Example: SELECT fruit, COUNT(\*)

FROM plant

GROUP BY fruit

ORDER BY COUNT(\*)

**Join and Union:**

1. Joins:

Oracle join is used to combine columns from two or more tables based on values of the related columns.

Types of Joins:

Right Join

Left Join

Inner Join

Full join

Self Join

**Inner Join**:

The INNER JOIN keyword selects records that have matching values in both tables.

Syntax: SELECT column\_name(s)FROM table1  
 INNER JOIN table2 ON table1.column\_name *=* table2.column\_name;

**Right Join:**

The RIGHT JOIN keyword returns all records from the right table (table2), and the matching records from the left table (table1). The result is 0 records from the left side, if there is no match.

Syntax: SELECT column\_name(s)FROM table1

RIGHT JOIN table2ON table1.column\_name = table2.column\_name;

**Left Join:**

The LEFT JOIN keyword returns all records from the left table (table1), and the matching records from the right table (table2). The result is 0 records from the right side, if there is no match.

Syntax: SELECT column\_name(s)FROM table1

LEFT JOIN table2ON table1.column\_name = table2.column\_name;

**Full Join:**

The FULL OUTER JOIN keyword returns all records when there is a match in left (table1) or right (table2) table records.

Syntax: SELECT column\_name(s)FROM table1

FULL OUTER JOIN table2ON table1.column\_name = table2.column\_name

WHERE condition;

**Self Join:**

A self join is a regular join, but the table is joined with itself.We can use this join when two or more rows needs to be compared in a single table.

Syntax: SELECT column\_name(s)FROM table1a  
 INNER JOIN table1 bON a.column\_name *=* b.column\_name;

1. Union& Union All:

* UNION operator to combine result sets returned by two or more queries.
* UNION operator removes the duplicate rows while UNION ALL operator includes the duplicate rows.

Syntax: SELECTcolumn\_list\_1FROMT1

UNION

SELECTcolumn\_list\_1 FROMT2;

SELECTcolumn\_list\_1FROMT1

UNION ALL

SELECTcolumn\_list\_1 FROMT2;

1. Difference Between Join and Union:

* JOIN in SQL is used to combine data from many tables based on a matched condition between them. The data combined using JOIN statement results into new columns.
* UNION in SQL is used to combine the result-set of two or more SELECT statements. The data combined using UNION statement is into results into new distinct rows.

**Inner Query or sub Query:**

In SQL a Subquery can be simply defined as a query within another query. In other words we can say that a Subquery is a query that is embedded in WHERE clause of another SQL query.

Syntax: SELECT column\_name

FROM table\_name

WHERE column\_name expression operator

( SELECT COLUMN\_NAME from TABLE\_NAME WHERE ... );

**Aliases:**

ALIASES can be used to create a temporary name for columns or tables.

* COLUMN ALIASES are used to make column headings in your result set easier to read.
* TABLE ALIASES are used to shorten your SQL to make it easier to read or when you are performing a self join (ie: listing the same table more than once in the FROM clause).

Syntaxto alias a column:column\_name AS alias\_name

Syntaxto alias a table :table\_namealias\_name

**Other Inbuilt Functions & Methods:**

Trim:

TRIM function removes all specified characters either from the beginning or the end of a string.

Syntax: TRIM( [ [ LEADING | TRAILING | BOTH ] trim\_character FROM ] string1 )

1.Leading - It removes any leading character that equals the specified trim\_character.

2. Trailing - It removes any trailing character that equals the specified trim\_character.

3. Both - It removes any leading and trailing character that equal the specified trim\_character.

Round:

ROUND function returns a number rounded to a certain number of decimal places.

Syntax: ROUND( number [, decimal\_places] )

To\_char:

TO\_CHAR function converts a number or date to a string.

Syntax: TO\_CHAR( value [, format\_mask] [, nls\_language] )

1. Value - A number or date that will be converted to a string.

2.Format\_mask - Optional. This is the format that will be used to convert value to a string.

3. nls\_language - Optional. This is the nls language used to convert value to a string.

Example: TO\_CHAR(sysdate, 'yyyy/mm/dd')

TO\_CHAR(1210.73, '9999.9')

Result: ' 1210.7'

To\_date:

The TO\_DATE function converts a string to a date.

Syntax: TO\_DATE( string1 [, format\_mask] [, nls\_language] )

Example: TO\_DATE('2003/07/09', 'yyyy/mm/dd')

LPAD:

The Oracle LPAD() function returns a string left-padded with specified characters to a certain length.

Syntax: LPAD(source\_string, target\_length [,pad\_string]);

Example: SELECTLPAD( 'ABC', 5, '\*' )FROMdual;

Result :'\*\*ABC'

RPAD:

The Oracle RPAD() function returns a string right-padded with specified characters to a certain length.

Syntax: RPAD(source\_string, target\_length [,pad\_string]);

Example: SELECTRPAD( 'XYZ', 6, '+' )FROMdual;

Result: 'XYZ+++'

INITCAP:

INITCAP function sets the first character in each word to uppercase and the rest to lowercase.

Syntax: INITCAP( string1 )

Example: INITCAP('tech on the net');

Result: 'Tech On The Net'

LEN:

The Oracle LENGTH() function returns the number of characters of a specified string. It measures the length of the string in characters as defined by the input character set.

Syntax: LENGTH(string\_expression);

SYSDATE:

The Oracle/PLSQL SYSDATE function returns the current system date and time on your local database.

Syntax: SYSDATE

Example: SELECT SYSDATEdateFROM dual;

ROW\_NUMBER**:**

The ROW\_NUMBER() is an analytic function that assigns a sequential unique integer to each row to which it is applied, either each row in the partition or each row in the result set.

Syntax: ROW\_NUMBER() OVER ( [query\_partition\_clause] order\_by\_clause)

Example: SELECT

ROW\_NUMBER() OVER(

ORDER BY list\_price DESC

) row\_num,

product\_name,

list\_price

FROM

products;

COALESCE:

The Oracle/PLSQL COALESCE function returns the first non-null expression in the list. If all expressions evaluate to null, then the COALESCE function will return null.

Syntax:COALESCE( expr1, expr2, ... expr\_n )

MINUS:

The Oracle MINUS operator is used to return all rows in the first SELECT statement that are not returned by the second SELECT statement. Each SELECT statement will define a dataset. The MINUS operator will retrieve all records from the first dataset and then remove from the results all records from the second dataset.

Syntax: SELECT expression1, expression2, ...expression\_n

FROM tables

[WHERE conditions]

MINUS

SELECT expression1, expression2, ...expression\_n

FROM tables

[WHERE conditions];

PIVOT:

PIVOT clause to transpose rows to columns to generate result sets in crosstab format.

Syntax: SELECT \* FROM

(

SELECT column1, column2

FROM tables

WHERE conditions

)

PIVOT

(

aggregate\_function(column2)

FOR column2

IN ( expr1, expr2, ... expr\_n) | subquery

)

ORDER BY expression [ ASC | DESC ];

Example: SELECT \* FROM

(

SELECT customer\_ref, product\_id

FROM orders

)

PIVOT

(

COUNT(product\_id)

FOR product\_id IN (10, 20, 30)

)

ORDER BY customer\_ref;

UNPIVOT:

The Oracle UNPIVOT clause allows you to transpose columns to rows. The UNPIVOT clause is opposite to the PIVOT clause except that it does not de-aggregate data during the transposing process.

Syntax: SELECT

select\_list

FROM table\_name

UNPIVOT [INCLUDE | EXCLUDE NULLS](

unpivot\_clause

unpivot\_for\_clause

unpivot\_in\_clause

);

Example: SELECT \* FROM sale\_stats

UNPIVOT(

quantity --unpivot\_clause

FOR product\_code-- unpivot\_for\_clause

IN ( --unpivot\_in\_clause

product\_a AS 'A',

product\_b AS 'B',

product\_c AS 'C'

)

);

Substring:

The SUBSTR functions allows you to extract a substring from a string.

Syntax: SUBSTR( string, start\_position [, length ] )

Replace:

The REPLACE function replaces a sequence of characters in a string with another set of characters.

Syntax: REPLACE( string1, string\_to\_replace [, replacement\_string] )

Example: REPLACE('123123tech', '123');

Result: 'tech'

Cast:

The Oracle CAST function converts one data type to another. The CAST function can convert built-in and collection-typed values into other built-in or collection typed values.

Syntax: cast(column\_name as required\_datatype)

Example: cast('123' as nvarchar2(20))

**PL/SQL (Dynamic Query):**

With the help of PL/SQL, we can generate and execute DML (INSERT,UPDATE,DELETE ) queries dynamically.

Task: create a dynamic query to drop table if exists and create table automatically with a particular name and insert data.