# References

<https://www.dcodeman.com/docs/oracle-sql/>

# Basics

## List of Objects present in the Database?

* Tables
* Views
* Constraints
* Sequences
* Indexes
* Packages
* Operators
* Synonyms
* Materialized views
* Materialized view logs
* Object types
* Stored functions, stored procedures
* Database triggers
* Database links

## Important points about order by?

ORDER BY clause is only used with SELECT statement.

ORDER BY clause should be the last clause in a SELECT statement.

ORDER BY clause can take ASC OR DESC keywords to sort the data either in ascending or descending order. Default order for columns specified in a ORDER BY clause is ascending (ASC).

## What is the default order of the ORDER BY clause?

ASC

## What is the difference between where and having?

The difference between WHERE clause and HAVING clause is, WHERE is executed before GROUP BY. But having is performed after GROUP BY.

# Parts of oracle query

## In what order oracle processes query ?

FROM

WHERE

GROUP BY

SELECT

HAVING

ORDER BY

## In What order oracle process query when we use join ?

You specify multiple tables in the query FROM clause and join condition in WHERE clause.

* A join, is evaluated first, whether the join is specified in the FROM clause (ANSI join) or with WHERE clause predicates
* WHERE Clause other filter conditions if any
* GROUP BY Clause
* HAVING Clause
* ORDER BY Clause

## When you have Hierarchical query

* A join, if present, is evaluated first.
* START WITH and CONNECT BY conditions are evaluated.
* WHERE Clause filter conditions other than join condition if any
* GROUP BY Clause
* HAVING Clause
* ORDER BY Clause

## When you have Pseudo Columns

* A join, if present, is evaluated first.
* START WITH and CONNECT BY conditions are evaluated.
* WHERE Clause filter conditions other than join condition if any
* GROUP BY Clause
* HAVING Clause
* Pseudo Columns are applied.
* ORDER BY Clause

# Oracle functions

## What are Analytical functions ?

Analytical functions, also known as Windowing functions, are introduced in Oracle 8i. Analytical functions are designed to address problems in real life such as

* Getting top ‘N’ results
* Find percentage within a group.
* Calculate running total.
* Compute moving averages.

## Analytic Functions Syntax and Features

AnalyticFunction(arguments)

OVER( PARTITION BY clause

ORDER BY clause

Windowing clause

)

## How Analytic functions work?

* Analytic functions are the “last set of operations performed in a query, except for the final ORDER BY clause”.
* All JOINs, WHERE clause, GROUP BY and HAVING clauses are completed before the Analytic functions are processed. Therefore, analytic functions can appear only in the select list or ORDER BY clause.
* The **OVER** clause indicates the function is analytic.
* The **PARTITION BY** clause logically breaks a single result set into “N” groups. Analytic function is applied for each group independently, and they are reset for each group defined by the PARTITION BY clause.
* When you do not provide PARTITION BY to the OVER clause, entire result set is considered as a single group.
* Analytic functions computes or aggregates based on the identified group but return for each row.
* The ORDER BY clause specifies how data must be processed within each group (partition). For example, when finding first value, last value for a partition, the order of rows becomes crucial.
* The Windowing clause gives a definition for how many numbers of rows are to be considered by the Analytic function while it is operating.

## Working with Analytical Functions in Oracle SQL

## Important Analytical functions

### RANK

RANK calculates the rank of a value in a group of values. The return type is NUMBER.

### DENSE\_RANK

DENSE\_RANK computes the rank of a row in an ordered group of rows and returns the rank as a NUMBER.

### ROW\_NUMBER

ROW\_NUMBER assigns a unique number to each row to which it is applied (either each row in the partition or each row returned by the query), in the ordered sequence of rows specified in the order\_by\_clause, beginning with 1.

### FIRST\_VALUE

FIRST\_VALUE returns the first value in an ordered set of values. If the first value in the set is null, then the function returns NULL unless you specify IGNORE NULLS.

### LAST\_VALUE

LAST\_VALUE returns the last value in an ordered set of values.

### LAG

LAG provides access to a row at a given physical offset prior to the position of the current row.

### LEAD

LEAD provides access to a row at a given physical offset beyond the position of the current row.

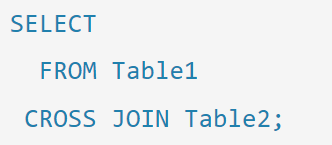
### NTILE

NTILE divides an ordered data set into a number of buckets indicated by expr and assigns the appropriate bucket number to each row.

# JOIN

## Cross join

Cross join is nothing but cartesian product.

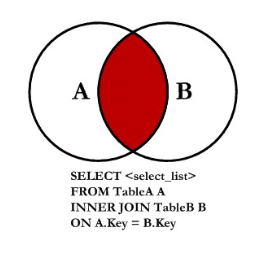


In SQL, a Cartesian product is a cross-join that returns all the rows in all the tables listed in a query. It's also called a cross-product.

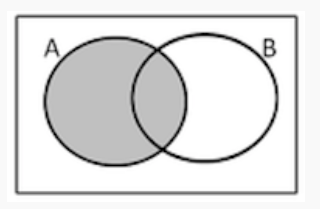
Cartesian product comes when there is no join condition and

## Inner Join

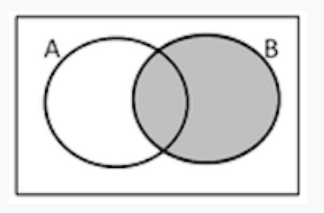
An inner join (also called a simple join) is a join that returns only rows that satisfy the join condition. Inner joins are either equi joins or non-equi joins or self-join.



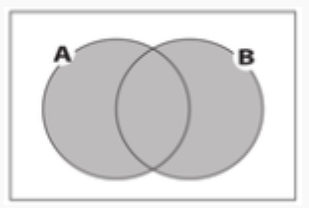
## Left Outer Join



## Right Outer Join



## Full join



# Sub query

## Can we use order by in sub query?

No, we can not use order by in sub query.

# SET

## UNION

## UNION ALL

## INTERSECT

## MINUS

# INDEX

### What is index ?

Index is a schema object which contains an entry for each value that appears in the indexed column(s) of the table or cluster. Index provides direct, fast access to rows.

The data stored in index column will be in ordered way for the fast access.

### Why do we create index ?

Index is created to speed up query execution.

If there is some column in the table which needs to be queried multiple times in multiple scenarios like join (special in where clause). Then we need to create index to have better execution.

If we do not create index then entire table will be scanned to check where conditions which is time consuming

### Btree index

* Syntax

Create index <INDEX\_NAME> ON TABLE\_NAME (EMP NO)

No need to mention the index type explicitly here.

* In What scenario should we create index?

B trees index we can create on the column which has wide range of spread.

Also, it is better to have column values unique.

* When the index will be used ?

Index will be used when you mention column on which index is created in the where clause.

### Unique index

* Syntax

Create UNIQUE index <INDEX\_NAME> ON TABLE\_NAME (EMP NO)

No need to mention the index type explicitly here.

* In What scenario should we create index?

When we are 100% sure that the column values will always be unique then we can create unique index. If we have doubt that data range will be wide, but it may not be unique.

### Composite Index

It is index on multiple columns of the same table.

* Syntax

Create index <INDEX\_NAME> ON TABLE\_NAME (**COL1,COL2**)

Multiple columns will automatically identify as composite. No need to mention explicitly.

### Reverse key index

* Syntax

Create index <INDEX\_NAME> ON TABLE\_NAME (COL) **REVERSE**

Create **UNIQUE** ndex <INDEX\_NAME> ON TABLE\_NAME (COL) **REVERSE**

* In what scenario we should use

If the data is not distributed widely then we must use reverse key index in such situations.

### Function based index.

* Syntax

Create index <INDEX\_NAME> ON TABLE\_NAME (**UPPER(**COL**)**)

Create index <INDEX\_NAME> ON TABLE\_NAME (**TO\_CHAR(HIREDATE,’YYYY’)**)

### Bit map Index

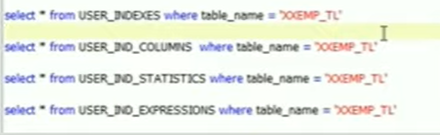
* Syntax

Create **BITMAP** index <INDEX\_NAME> ON TABLE\_NAME (COL)

* In what scenario we should use

When all columns will have data, but when we take distinct of records, we will receive very less data. For example, gender (M/F), dept (few departments )

### Where will meta data of the index be stored ?



# TODO

## What is SAVEPOINT in TCL?

## Data types in Oracle?

Character Datatypes

* NCHAR
* NVARCHAR2
* CLOB
* NCLOB
* LONG

binary datatypes

* BLOB
* BFILE
* RAW
* LONG RAW

## Check if you know how to use all operators ?

## PARTITION BY ?

## INDEX ?

## NVL in oracle?

## STATISTICS ?