Index (_ind) → Schema object.

- → Index is a object that associated with table / table cluster to speed up the Query execution process.
- → It will reduce I/O operations.
- → 60% of the performance issue will be cleared by index.

Data Dictionary table } user_indexes, user_ind._columns

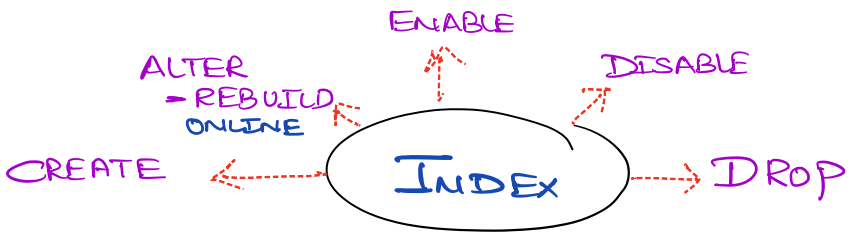
Types of index

- Unique
- Non-Unique

user_indexes → uniqueness → unique / Non Unique (Bitmap)

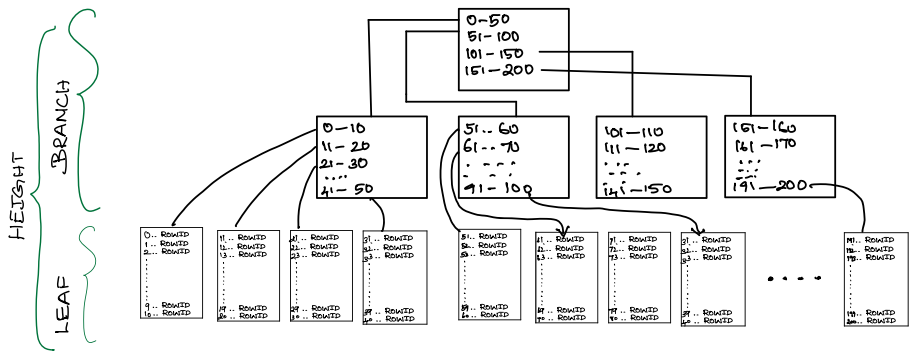
get created while creating pk & uk

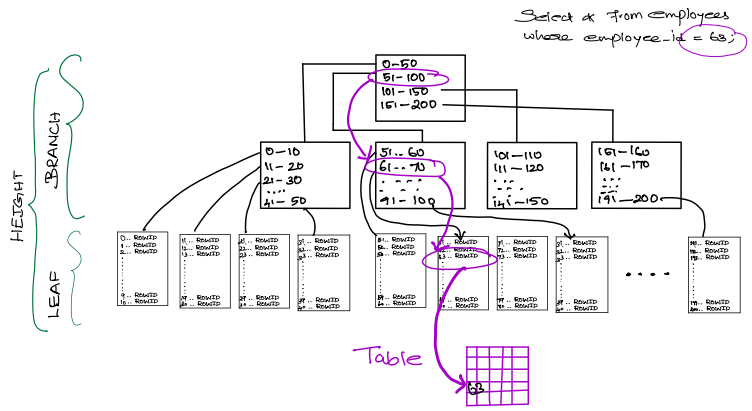
- → B-Tree index (ASC) ✓
  - Balanced index
- → Bitmap index ✓
- → Function Based index ✓
- → Reverse key Index ✓
- → Clustered index
- → Index Organized table.



B-Tree index

- → Balanced index
- → Default : ASC order.
- → Created on high cardinality column.





CREATE INDEX <ind_name>
ON table_name> (column_name);

CREATE Index emp_ind
ON employees (emp_id);

Bit-map index

Created on low cardinality column.

$$\text{Cardinality} = \frac{\text{no. of. possible Values}}{\text{Total no. of. Values}} < 1$$

Created on columns like gender (M/F), office_cd
passport_avai (Y/N)

Employee_name	gender
Rakesh	M
Neera	F
Ram	M
Jeni	F
Sastavanan	M

How the index get stored

	M	F
Rakesh	1	0
Neera	0	1
Ram	1	0
Jeni	0	1
Sastavanan	1	0

Select * from employees where gender = 'F';

	M	F
Rakesh	1	0
Neera	0	1
Ram	1	0
Jeni	0	1
Sastavanan	1	0

No of rows in a table = no. of rows in a index
Bitmap Index will be faster than B-Tree index because here there is no root, branch (or) Leaf like structure instead of that simple scan will happen.

	OLTP	OLAP
Bitmap	✗	✓ → maximum used in DW projects. (o DML operations - table)

(Not preferable)

Due to frequent DML on a table.
(i.e. For every DML operation index will get updated & it causes blocks & deadlocks)

Function Based Index.

If you create index on first_name column means Index will be used while executing the Query.

Where first_name = 'Steven'

Where lower(first_name) = lower(first_name);

while using Function index won't need to fetch data from the table.

So we need to create index with function.

first_name	lower(first_name)
salary	6* salary.

```
Create index my_fn_ind
on employees(lower(first_name));
```

Reverse key index

Clustered Index.

1 – 100 (marks)

Student name	marks	
name1	48	97
	93	99
	99	93
	85	85
	63	81
	72	63
	35	94
	43	74
	85	
	89	
	87	

Index organized table.

20's	40's	50's	60's	70's	80's	90's	100
35	48		63	72	85	93	
	43		63		85	99	
					89	97	
					87	99	
					85	93	
					81	94	
						94	

Less

High.

Volume of data in particular leaf is high.

So we are going to reverse the data & store it in leaf of index

(ie) Steven → nevets

18	20's	40's	50's	60's	70's	80's	90's	100
27	39	49	58		78	84	99	
	36	47	53		79		98	
	34		58				99	
	39		58					
	36							

48	97
93	99
99	93
85	85
63	81
72	63
35	94
43	74
85	
89	
87	

	when to create index	when not to create index
1	Volume of the table ↑ (min to lakhs)	↓
2	where column name =	Not frequently used in a where clause.
3	↓ DML	↑ DML operations
4	< 10% < 4%	> 10% > 4%.

→ [table] → column ← index.

Clustered index.

Create cluster empdep_clust (dep_id Number(10));

Create index clust_ind on cluster empdep_clust (dep_id);

Create table emp
(
=====
) cluster empdep_clust (dept_id);

Create table dept
(
=====
) cluster empdep_clust (dept_id);

IOT:

Create table...
(
~~Organization Oracle~~ } IOT.