Bitmap Index: Numerical Example

A **Bitmap Index** is a data structure that represents the presence of a particular value in a set of records using a bit array. Each bit corresponds to a record, with `1` indicating the presence of a specific value and `0` indicating its absence.

Example Dataset

Consider a dataset with the following schema:

Record ID	Gender	Marital Status
1	Male	Single
2	Female	Married
3	Male	Married
4	Female	Single
5	Male	Single

Bitmap Index Construction

Let's construct bitmap indices for the **Gender** and **Marital Status** columns.

Gender:

- **Male:** `[1, 0, 1, 0, 1]`
- **Female:** `[0, 1, 0, 1, 0]`

Marital Status:

- **Single:** `[1, 0, 0, 1, 1]`
- **Married:** `[0, 1, 1, 0, 0]`

Query Using Bitmap Index

Suppose you want to find records where **Gender** = **Male** and **Marital Status** = **Single**.

- **Gender = Male:** `[1, 0, 1, 0, 1]`
- **Marital Status = Single:** `[1, 0, 0, 1, 1]`

To find the records satisfying both conditions, perform a **bitwise AND** operation on the two bitmaps:

- **Result:** `[1, 0, 0, 0, 1]`

This result corresponds to **Record ID 1** and **Record ID 5**. These records match the criteria of being Male and Single.

Join Index: Numerical Example

A **Join Index** is used to speed up join operations between two tables by precomputing the relationships between them. It stores pairs of record identifiers from two tables that would be joined in a query.

Example Dataset

Let's consider two tables:

Customers Table:

Customer ID	Name
1	John
2	Mary
3	Alice

Orders Table:

Order ID	Customer ID	Amount
101	1	200
102	2	150
103	1	100
104	3	250
105	2	300

Join Index Construction

Suppose you frequently need to join the `Customers` and `Orders` tables on `Customer ID`. The Join Index would store the relationship between the `Customer ID` in the `Customers` table and the `Order ID` in the `Orders` table.

Join Index:

Customer ID	Order ID
1	101
1	103
2	102
2	105
3	104

Query Using Join Index

Now, if you want to find the total order amount for each customer:

- Instead of performing the join operation between the `Customers` and `Orders` tables, you can directly use the Join Index to identify relevant `Order ID`s for each `Customer ID`.
- Use the Join Index to quickly find which orders belong to which customer.

For example:

- **Customer ID 1** (John) has orders with **Order IDs** 101 and 103.
  - The total amount = 200 + 100 = 300
- **Customer ID 2** (Mary) has orders with **Order IDs** 102 and 105.
  - The total amount = 150 + 300 = 450
- **Customer ID 3** (Alice) has orders with **Order ID** 104.
  - The total amount = 250

This approach significantly reduces the need to scan and match rows from both tables, thus speeding up the query.

Summary

- **Bitmap Index**: Efficiently answers queries by performing bitwise operations on indexed columns.
- **Join Index**: Optimizes join operations by precomputing and storing the relationships between tables.