Discipline

Algorithm

(3 versions)

CS5201

Stage-2

Sheikh Muhammed Tadeeb (AU19B1014)

❖ Indexing:

An index is an on-disk structure associated with a table or view that speeds retrieval of rows from the table or view. An index contains keys built from one or more columns in the table or view. These keys are stored in a structure (B-tree) that enables SQL Server to find the row or rows associated with the key values quickly and efficiently.

❖ Types of Indexing:

1) Primary Indexing: - Indexing done on primary key of table is called primary indexing.

2) Secondary Indexing: - Indexing done on Alternate key of table is called primary indexing.

3) Cluster Indexing: - Indexing done on some other column of table apart from primary key where records may have repeating values is called cluster indexing.

4) Multi-level Indexing: - Indexing inside indexing is called multi-level indexing. If the data is big machine does this.

❖ Table Insights:

➢ Goods Table:

	Goods_id	Goods_type	Goods_name
☐ 🖉 Edit ⯒ Copy ⊖ Delete	11	Mettalic	Screw

➢ Customer Table:

	C_id	F_name	M_name	L_name
☐ 🖉 Edit ⯒ Copy ⊖ Delete	2001	Somesh	NULL	Anjana

↰ ☐ Check all With selected: 🖉 Edit ⯒ Copy ⊖ Delete 🖼 Export

❖ Indexing on table:

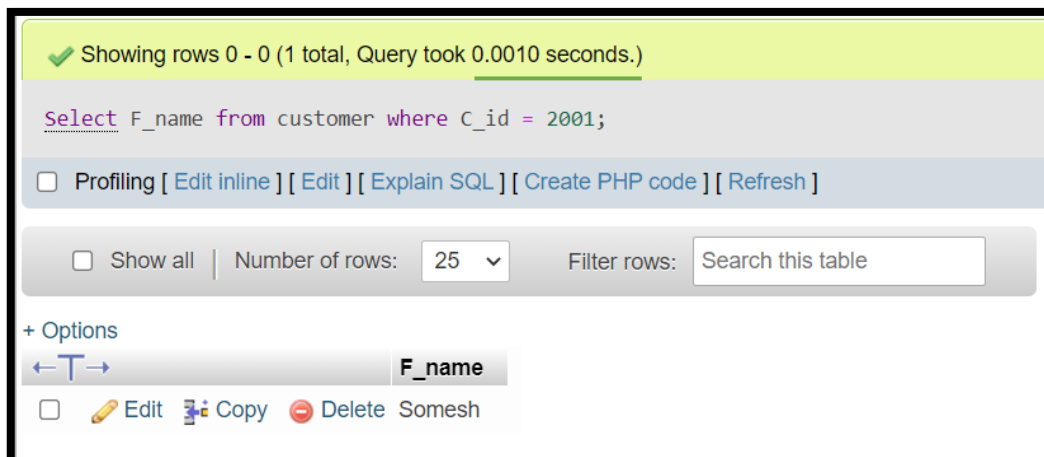| Snos	Primary Indexing	Secondary Indexing	Cluster Indexing
1.	Goods_id	username	-
Description:	Goods_id is the primary key of the table. There will be many goods in the shop whose quantity will grow as the business grows and so will the operations complexity henceforth indexing is very necessary to reduces the time complexity.	username is the alternate key of the table i.e.; it will also have unique plus not null values but it's not declared as the primary key of the table.	-
2.	C_id	-	first_name, last_name
Description	C_id (customer_id) is the primary key of the table. There will be many customers of the business and as business grow so will the customers and operations complexity as well henceforth indexing is very necessary to reduces the time complexity.	-	On column first_name, last_name, status, password and gender as these columns may have repeating values for two different entries.

❖ Queries (Before Indexing):

➢ Goods table:
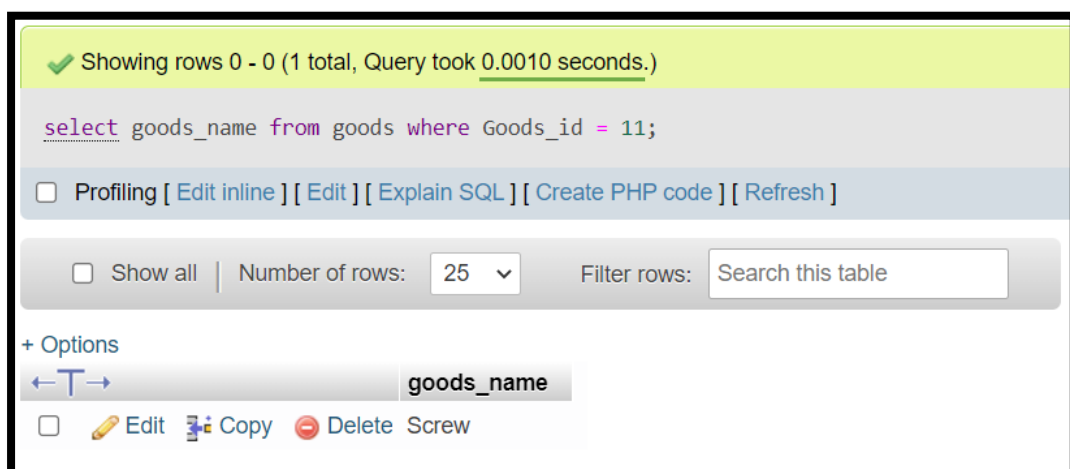
We can see the time complexity before applying indexing in the above image.

> ➤ **Customer table:**

> Showing rows 0 - 0 (1 total, Query took 0.0010 seconds.)
>
> Select F_name from customer where C_id = 2001;
>
> ☐ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
>
> ☐ Show all | Number of rows: 25 ⌄   Filter rows: Search this table
>
> + Options
>
> ←T→   **F_name**
>
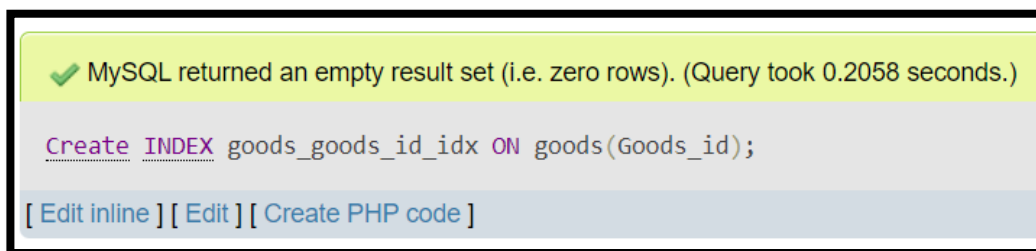> ☐   ✏ Edit   📋 Copy   ⊖ Delete   Somesh

We can see the time complexity before applying indexing in the above image.

❖ Indexing on table:

> ➤ **Goods Table:**

> 1. **Primary indexing:**

> ✔ MySQL returned an empty result set (i.e. zero rows). (Query took 0.2058 seconds.)
>
> Create INDEX goods_goods_id_idx ON goods(Goods_id);
>
> [Edit inline] [Edit] [Create PHP code]

> ➤ **Customer Table:**

> 1. **Primary indexing:**

> ✔ MySQL returned an empty result set (i.e. zero rows). (Query took 0.1248 seconds.)
>
> Create INDEX customer_C_id_idx ON customer(C_id);
>
> [Edit inline] [Edit] [Create PHP code]
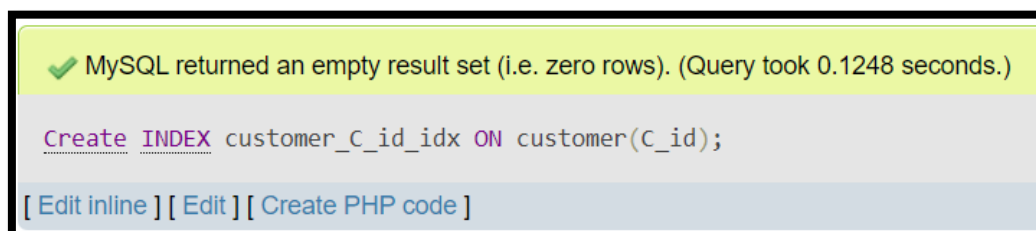
❖ Queries (After Indexing):

As our records will grow then indexing will be very helpful as it will reduce our time complexity.