CSIS05I

Database Systems II

Lab (4)

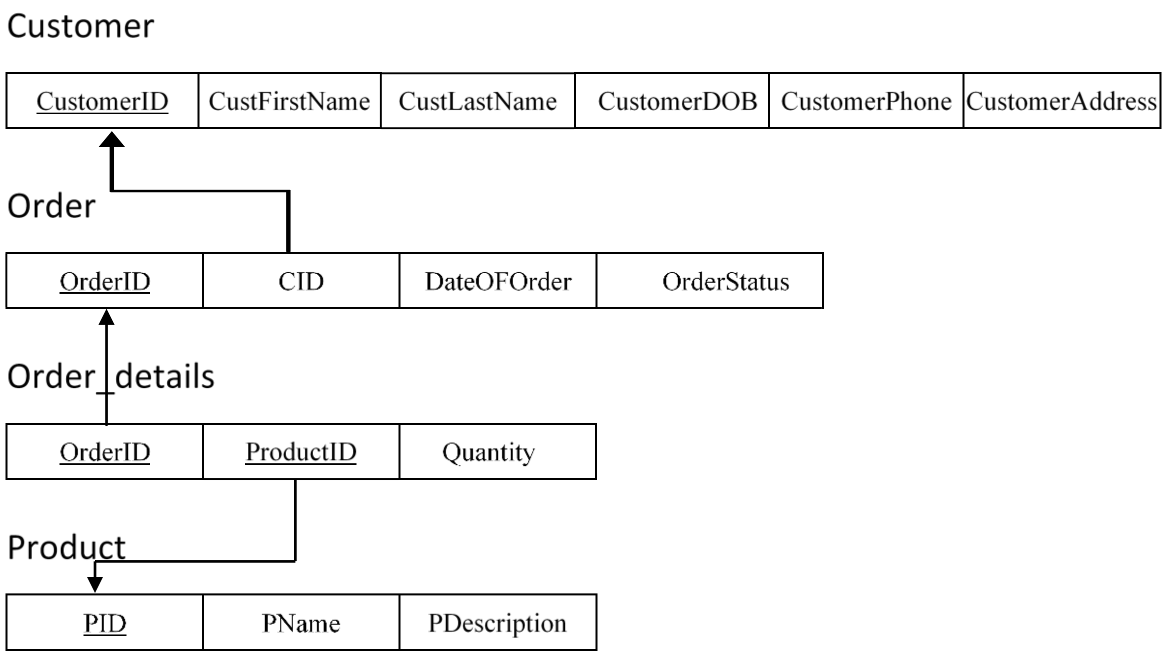
INDEXING
Cluster and non-cluster

Lab (4)

The Overview

In the previous lab, we mentioned the indexing and its type. The types of indexing are single index or composite index. Also, the usage procedure of retrieving information about the index was mentioned. Through this lab we will discuss the cluster and non-cluster indexing. Moreover, the unique and filtered indexes will be explained.

Store Schema:



Customer

CustomerID	CustFirstName	CustLastName	CustomerDOB	CustomerPhone	CustomerAddress

Order

OrderID	CID	DateOFOrder	OrderStatus

Order_details

OrderID	ProductID	Quantity

Product

PID	PName	PDescription

Clustered index

A clustered index defines the order in which data is physically stored in a table. Table data can be sorted in only one way, therefore, there can be only one clustered index per table. In SQL Server, the primary key constraint automatically creates a clustered index on that particular column. When a table has a clustered index, the table is called a clustered table. If a table has no clustered index, its data rows are stored in an unordered structure called a heap.

Non-clustered index

Non-clustered index does not sort the physical data inside the table. In fact, a non-clustered index is stored at one place and table data is stored in another place. This is similar to a textbook where the book content is located in one place and the index is located in another. This allows for more than one non-clustered index per table.

Clustered vs Non-Clustered index:

- In a table, there can be only one clustered index, one or more than one non-Clustered index.
- In Clustered index there is no separate index storage but in non-Clustered index there is separate index storage for the index.
- Clustered index is faster than non-Clustered index.

Clustered Indexes syntax

```
CREATE CLUSTERED INDEX Index_name
ON table_name (column Condition);
```

Note:

- ✓ The cluster index done by default on the primary key constraint in the table. Therefore, you cannot do more than one cluster in the table.
- ✓ To do a clustered index other than the Primary key you should drop the PK constraint and all relationships associated to it (FKs), then create another cluster index for the table.
- ✓ The Condition means to be in Ascending order or Descending order.

Example 1:

Create an index named "idx_lastname" on the " CustLastName " column in the "Customer" in which the last name is arranged in ascending order.

```
CREATE CLUSTERED INDEX idx_lastname
ON Customer (CustLastName ASC);
```

Example 2:

Create an index named "idx_fullname" for the Customer in which the last name is arranged in ascending order and the First name in descending.

```
CREATE CLUSTERED INDEX idx_fullname
ON Customer (CustFirstName DESC , CustLastName ASC);
```

Non-Clustered Indexes syntax

```
CREATE NONCLUSTERED INDEX Index_name
ON table_name (column Condition);
```

Note:

- ✓ Condition could be Ascending order or Descending order.
- ✓ We can create one or more non-Clustered index in a table.

Example 3:

Create an index named "idx_Orderinfo" on the " orderID " and the customer related to this order with the status of it in which the order ID is arranged in descending order.

```
CREATE NONCLUSTERED INDEX idx_Orderinfo
ON Order (OrderID DESC, CID, OrderStatus);
```

Example 4:

Create an index named "idx_ProductName " for the Products in which the PID is arranged in ascending order along with its name in descending order.

```
CREATE NONCLUSTERED INDEX idx_ProductName
ON Product (PID ASC, PName DESC);
```

Unique Indexes syntax

```
CREATE UNIQUE INDEX index_name
 ON table_name (column);
```

Note:

- ✓ Duplicate values are not allowed.

Example 5:

Create an index named "idx_lastname" on the " CustLastName " column in the "Customer" table without duplication in the records.

```
CREATE UNIQUE INDEX idx_lastname
ON Customer (CustLastName);
```

Detailed table information

To get all the details of the table and its information you can write the following command.

```
EXECUTE SP_HELP tablename;
```

Example 6:

Retrieve all the information related to the table product.

```
EXECUTE SP_HELP Product;
```

Filtered index

- Non clustered index along with where condition.
- Filtered index maintenance cost is less because it uses smaller part of an index.

Filtered index syntax

```
CREATE NONCLUSTERED INDEX Index_name
ON table_name (column_name)
WHERE predicate;
```

Example 7:

Create a filtered index named idx_orderStatus where the order status is delivered.

```
CREATE NONCLUSTERED INDEX idx_orderStatus
ON Orderr(OrderStatus)
WHERE OrderStatus='Delivered';
```

Index with included columns

- It improves the speed of the query.
- It is called covering index.
- SQL server allows extending the functionality of non-clustered index by including non-key columns to reduce key look up cost.

Syntax

```
CREATE INDEX idx_name
ON table_name(Col1)
INCLUDE (Col2);
```

Example 8:

Create an index named idx_CustomerAddressAndPhone on customer Address column and then include customer phone to it.

```
CREATE INDEX idx_CustomerAddressAndPhone
ON Customer(CustomerAddress)
INCLUDE (CustomerPhone);
```

Exercises

1. Create a non-clustered index that has all customers IDs in descending order.

2. Create a non-clustered index that have all the order information where both order ID and date of order arranged in descending order.

3. Create a non-clustered index that have the quantity of the products in each order arranged in descending order along with the order ID.

4. Create an index that has all the customer's names without any duplication.

6. Show all the index information related to table Customer.

7. Use the stored procedure to retrieve all the information of table order.

8. Create an index on product quantity in ascending order where quantity is less than 11.

9. Create an index on order's date and then covering with its status.