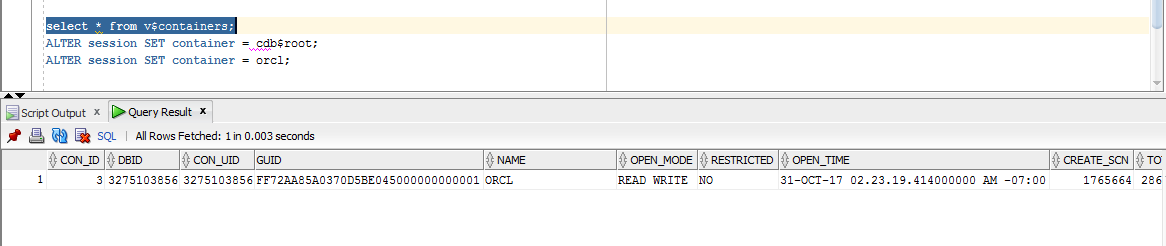
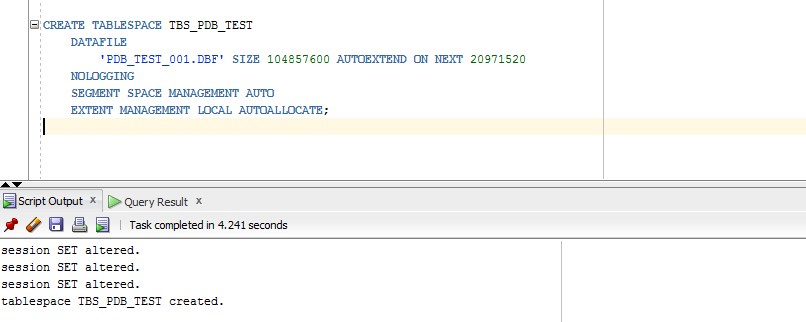
# Prerequisite Task

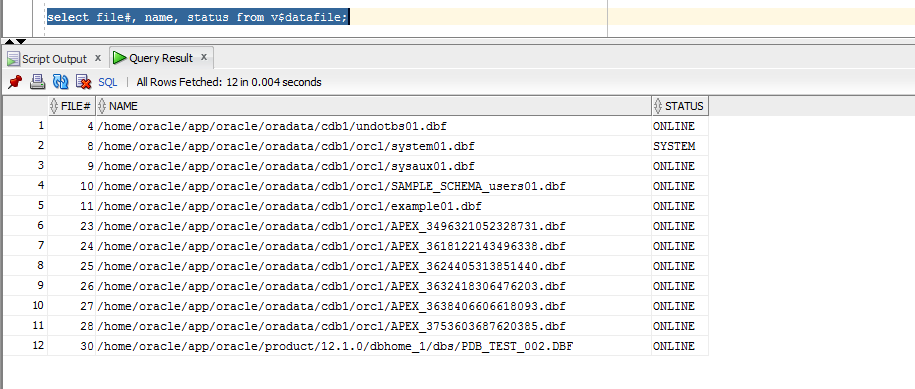
Just try to understand what pluggable database means. As I understand, it’s our stored data. Because there is another one, root db, if I can name it like this. Pluggable db depends on root db.



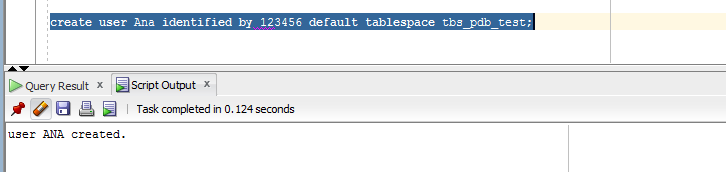
Well. A new tablespace was created.



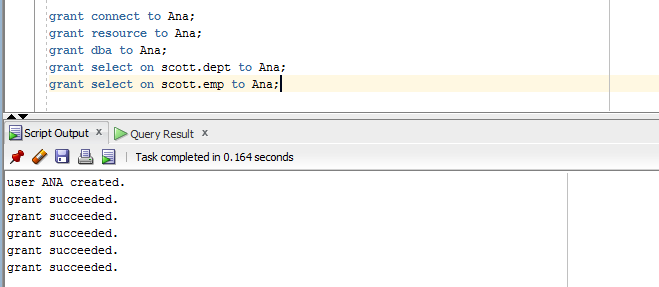
After that I would like to find created file .dbf.



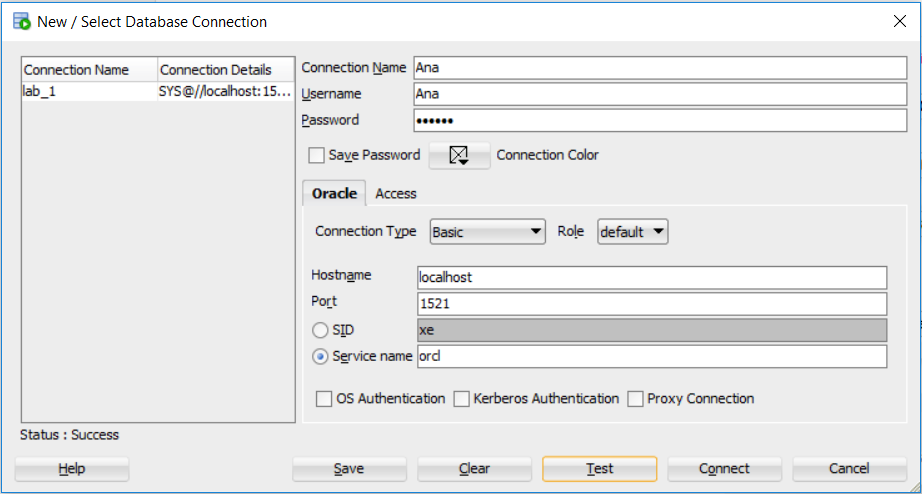
A new user Ana was created.



And necessary privileges were granted.

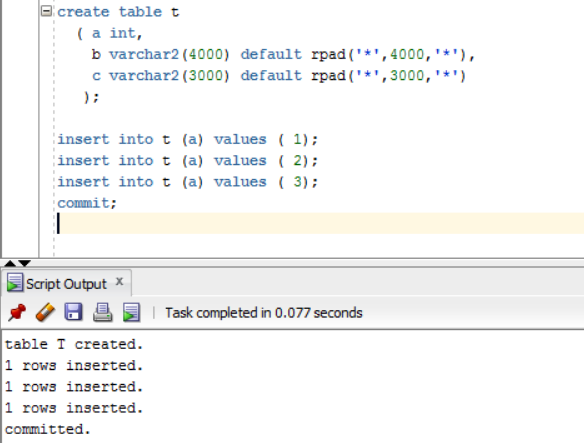


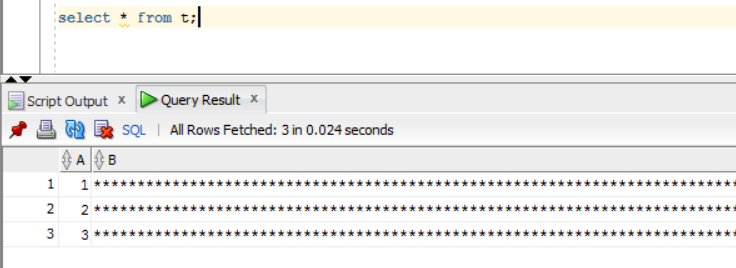
After creation new user it’s able to connect , using the name and password .



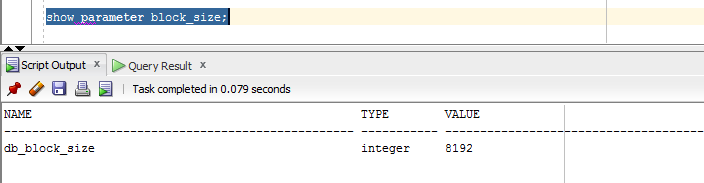
# Heap Organized Tables

A table **T** was created. Then the appropriate data were inserted , result was commited. .

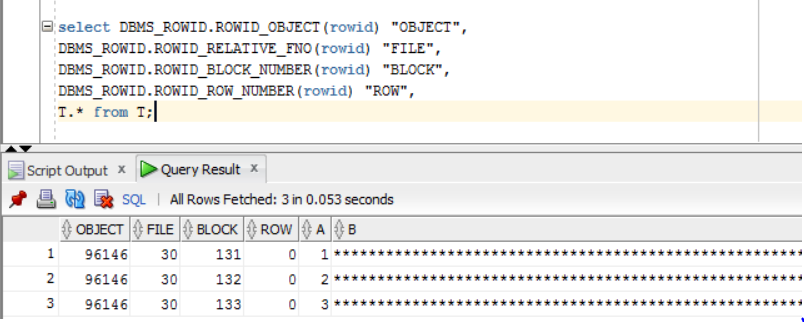




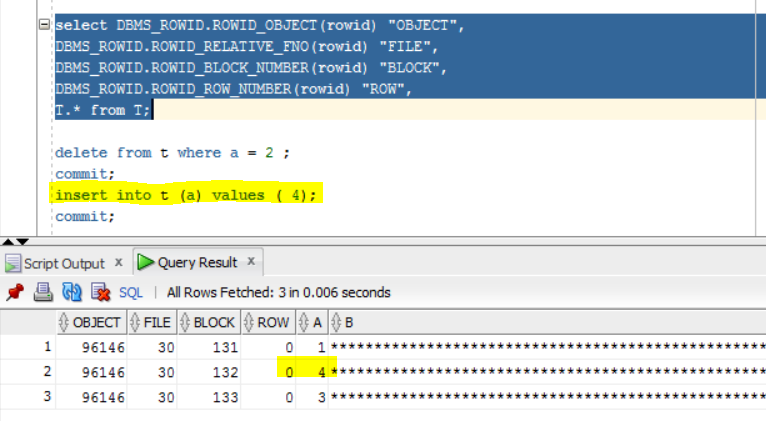
Checking a value of DB\_BLOCK\_SIZE.



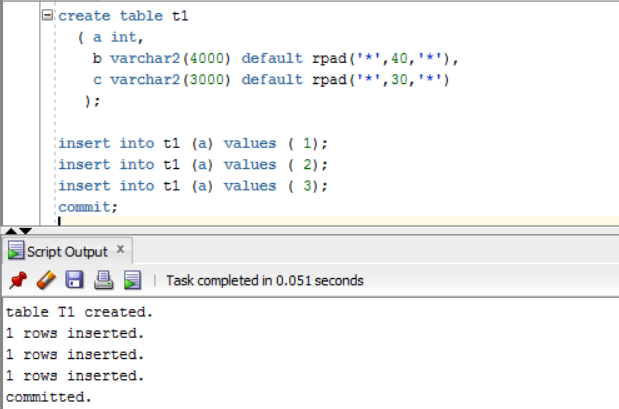
Looking for how our data is stored. The screenshot below shows that out data stored in different blocks. The reason is that Column B and C in the total sum equal to 7000 symbols or 7000 B and plus column A it is equal to 7002(it is 85% of total size block, another space is used by PCTFREE, which coefficient is 10% ), our block-size is 8192 B. Only one row can be stored in one block.



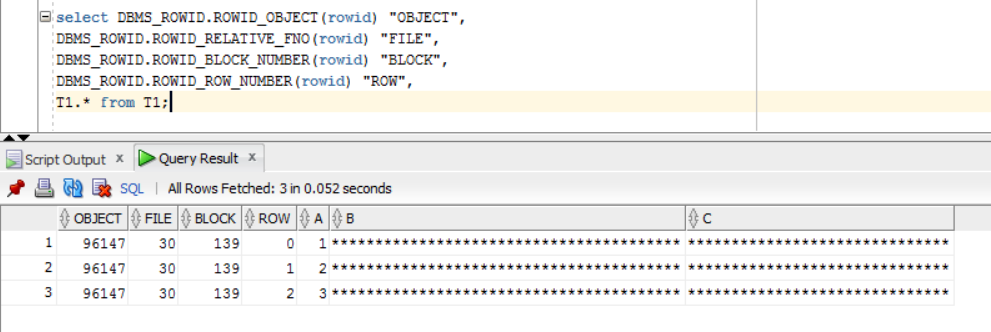
The row with the value 2 was deleted. And new row with the value 4 was inserted. The result is new value replaced the old value. All of these is about heap-organized table: rows are not ordered, any new row take the first available free space, when row is moved the place become available to reuse by new rows and the ROWID changes.



Some experiments to prove the theory about block size. A table T2 was created with new size of columns B and C.

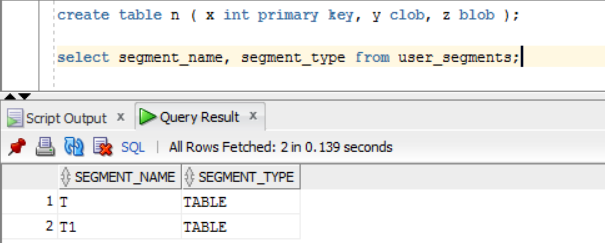


The screenshot below shows the expected result. All rows occupy one block. Now there is enough space in one block.

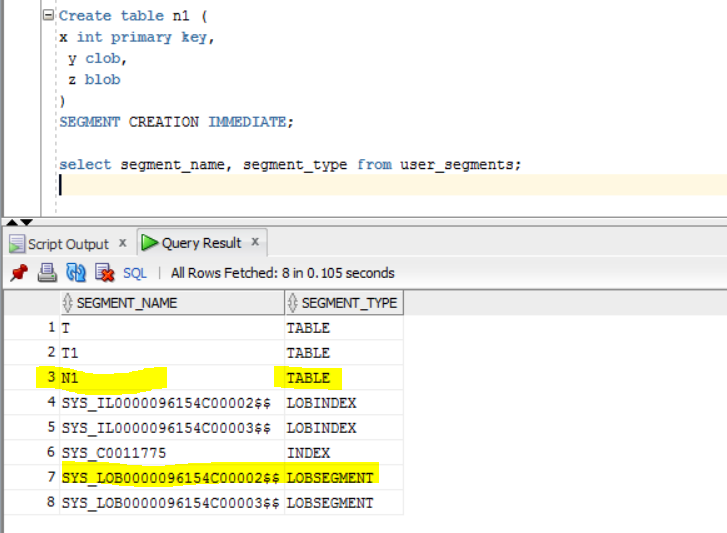


## Understanding Heap Table Segments

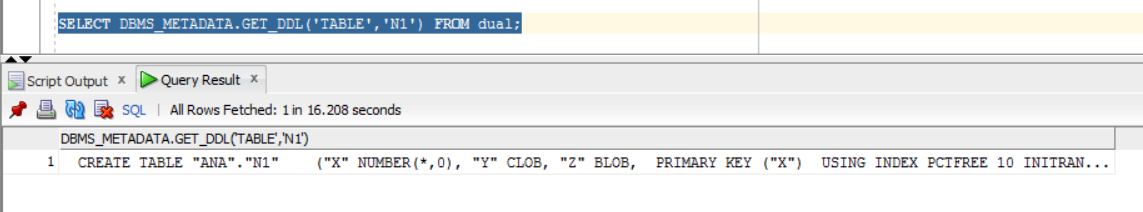
Step 1-2: only table segments from previous tables are visible, because a new table N is empty for this moment.



Step 3-4: For an immediately creation segments the **segment creation immediate** statement is used.



Getting information about N2:

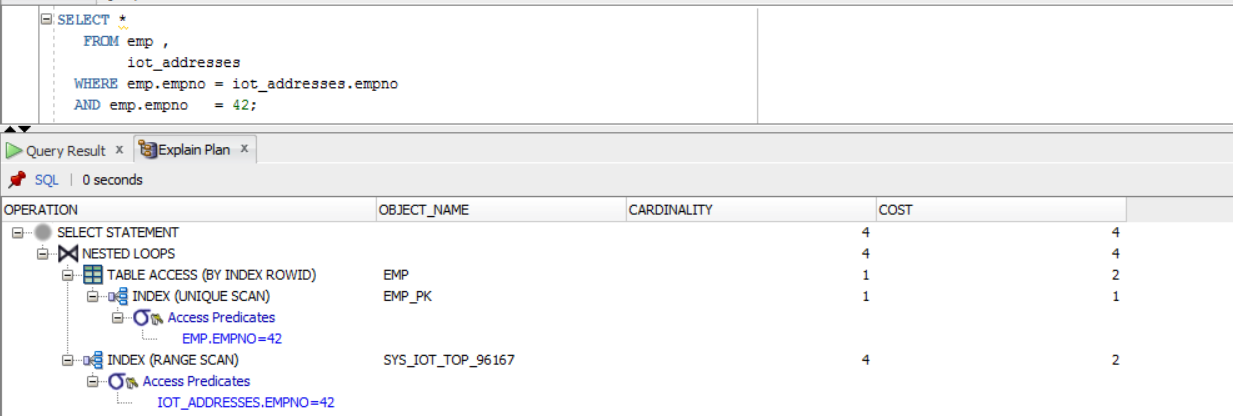


# Index Organized Tables

Explain 1:



Explain 2:



In index-organized tables, records are placed in the order specified by the primary key that you set.

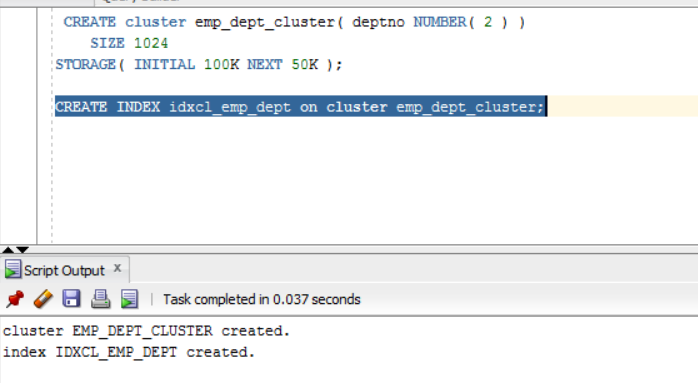
In fact, the index-organized table is actually an index. The difference between this index and the usual B \* Tree index is that all the columns (fields) of the record are written to it, and not only indexed as in a normal index. When you use a normal (HEAP) table and create an index on it, that index uses the physical rowid as a pointer to the address of the record itself.

For index-organized tables, a slightly different approach is used: an index is immediately created based on the primary key and includes other columns of the record. In fact, there is only an index, and there is no table in the form of a heap. Therefore, for this type of index, the Oracle does not use the physical ROWID.

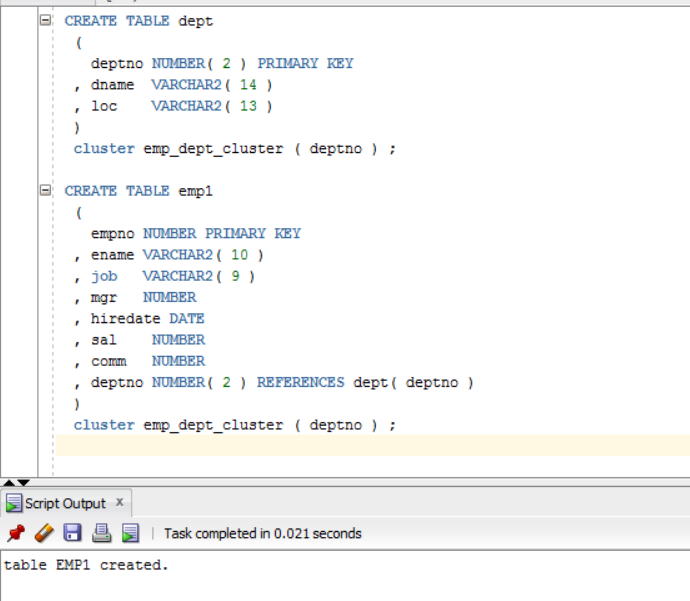
That’s why the second plan shows more fast execution.

# Index Organized Tables

Step 1-2:

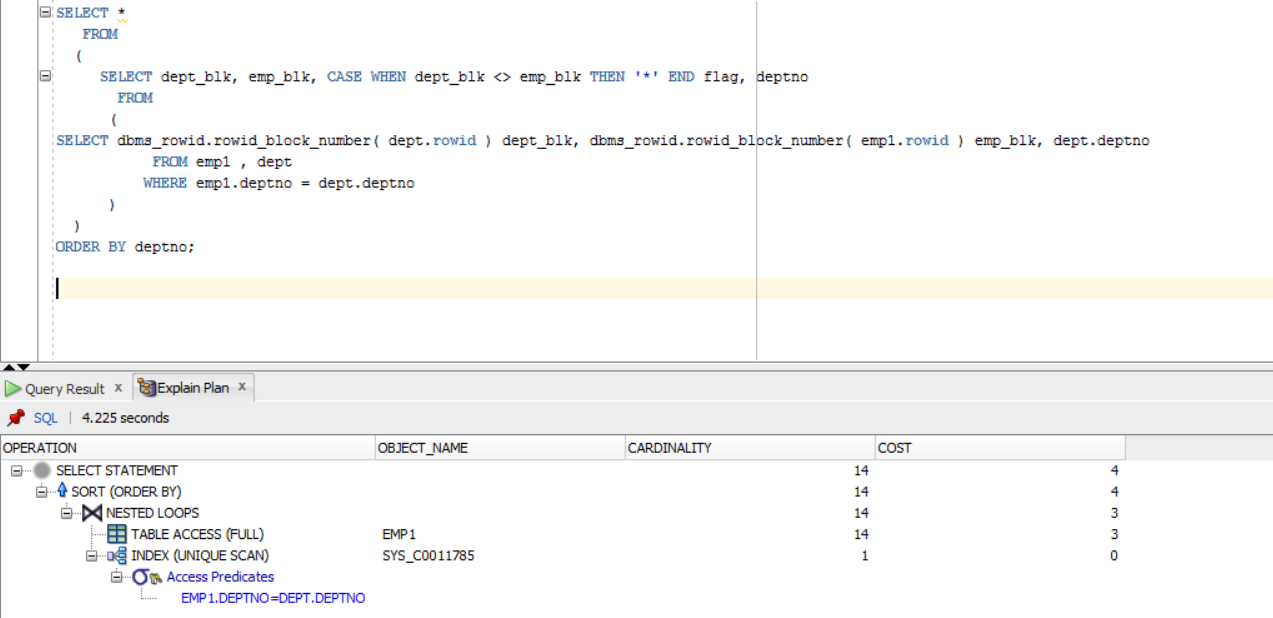


Step 3:



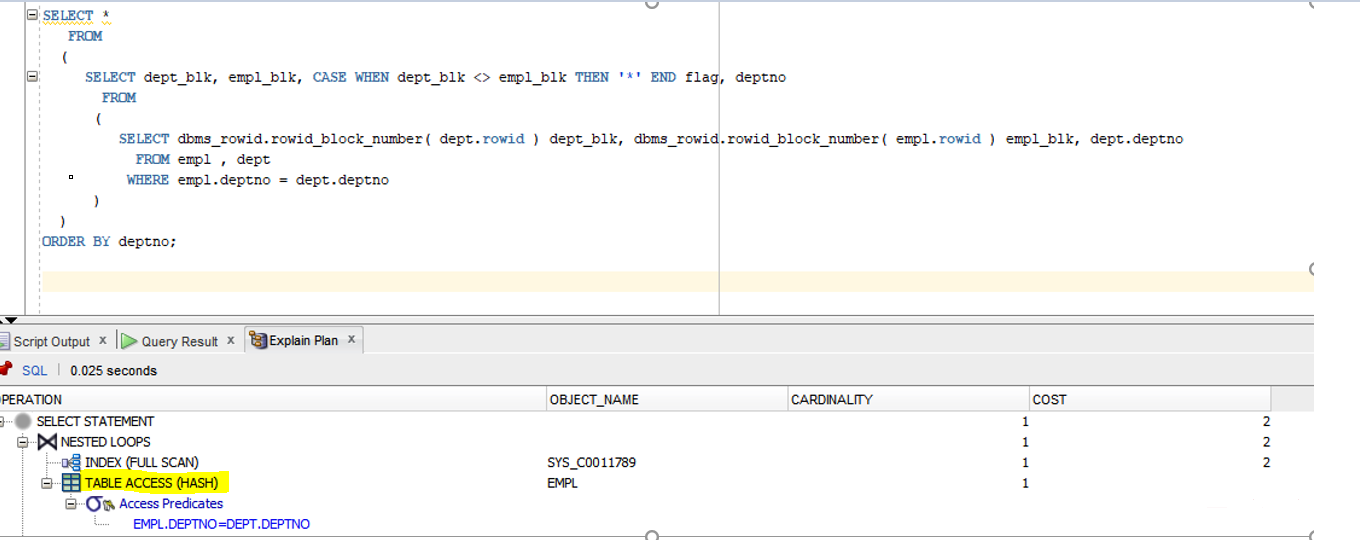
# Index Clustered Tables

Clustered index tables reduce the number of blocks read from the hard disk, thereby speeding up performance.





# Hash Clustered Tables



The main difference between hash clustered tables and index clustered tables are that tables are merged into a cluster, then each hash of data is assigned a hash function, which will subsequently be searched. To find any string value, all you need to do is get a hash value for the cluster key - strings, so the only I/O operation will output the data of the required string and provide more efficient performance.