|  |
| --- |
|  |
| Oracle Relational Structures |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| REVISION HISTORY | | | | | |
| Ver. | Description of Change | Author | Date | Approved | |
| Name | Effective Date |
| 1.0 | Initial status | Arina Marchenko | 02-11-2017 |  |  |

Contents

[1. Prerequisite Task 3](#_Toc497404090)

[2. Understanding Heap Table Segments 8](#_Toc497404091)

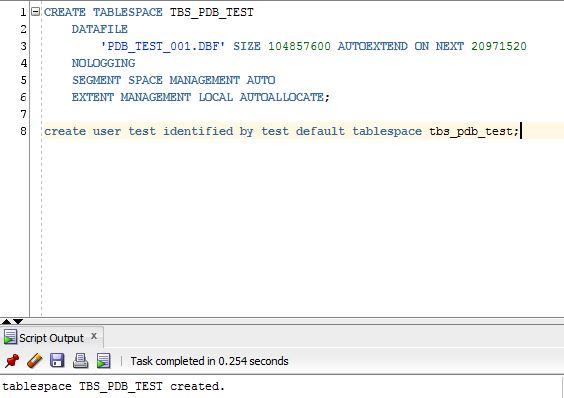
[3. Index Organized Tables 8](#_Toc497404092)

[4. Index Clustered Tables 9](#_Toc497404093)

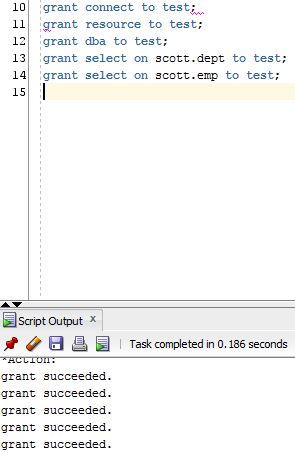
[5. Hash Clustered Tables 9](#_Toc497404094)

[6. Row Migration 10](#_Toc497404095)

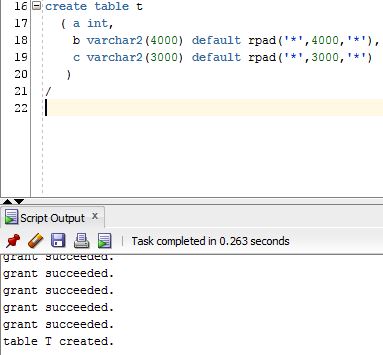
1. Prerequisite Task

Creating tablespace and creating user

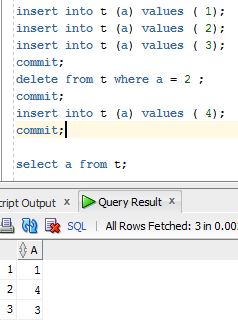
Grants to user



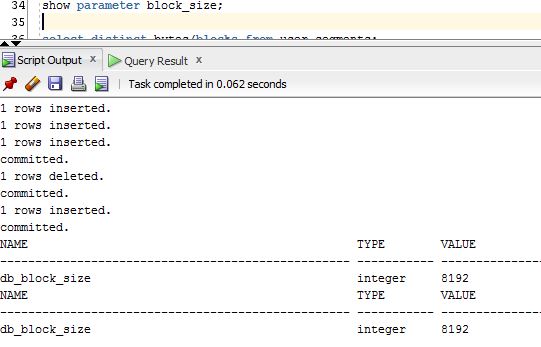
Creating table t



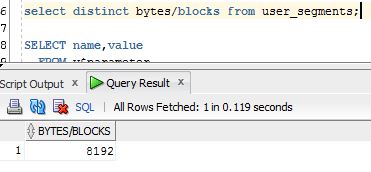
Insert into values



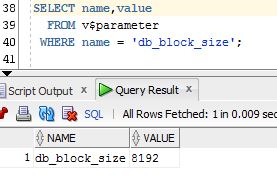
Showing parameter block size



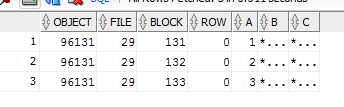
Showing bytes\blocks

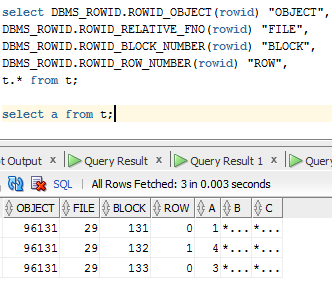


Showing block size



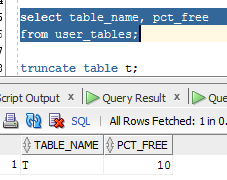
Showing rowid\_object, rowid\_relative\_gno, rowid\_block\_number and rowid\_row\_number





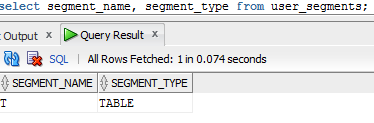
Our db\_block size is 8192. Our row size is 4+3000+4000=7004 and its less then 90% (7372) because 10% using for updates in pcsfree by updates.

Then we decided to delete row where value of t is 2 and then insert value 4. Since data is managed in a heap in a table like this, as space becomes available, it will be reused.



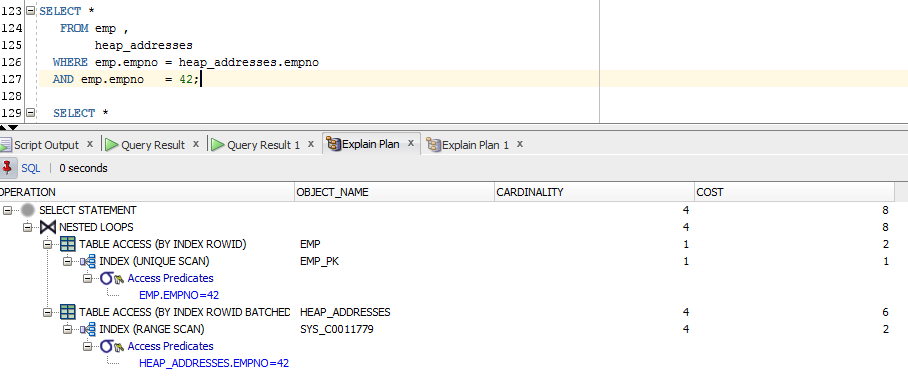
# Understanding Heap Table Segments

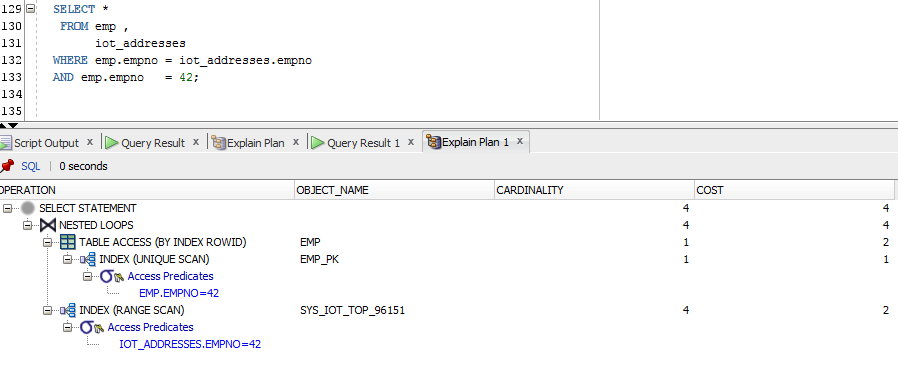
Creating tables: T with segment creation and T2 without segment creation



Segment creation on demand, or deferred segment creation as it is also known, is a space saving feature of Oracle Database 11g Release 2. When non-partitioned tables are created, none of the associated segments (table, implicit index and LOB segments) are created until rows are inserted into the table. If we define segment creation, associated segments create immediately.

# Index Organized Tables





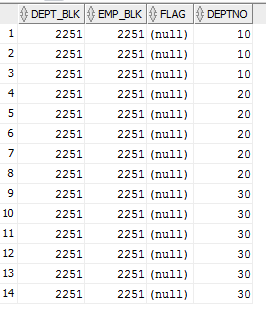
The structure of an index-organized table provides the following benefits:

Fast random access on the primary key because an index-only scan is sufficient. And, because there is no separate table storage area, changes to the table data (such as adding new rows, updating rows, or deleting rows) result only in updating the index structure.

Fast range access on the primary key because the rows are clustered in primary key order.

# Index Clustered Tables

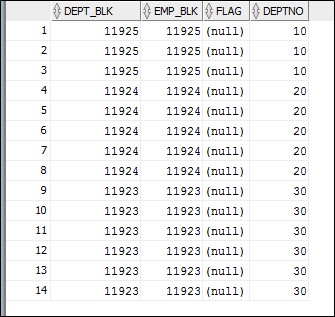
In index clustered tables many tables may be stored physically joined together. Normally, you would expect data from only one table to be found on a database block, but with clustered tables, data from many tables may be stored on the same block. Also, all data that contains the same cluster key value, such as DEPTNO = 10, will be physically stored together. The data is clustered around the cluster key value.



# Hash Clustered Tables

In hash clustered tables instead of using a B\*Tree index to locate the data by cluster key, the hash cluster hashes the key to the cluster to arrive at the database block the data should be on. In a hash cluster, the data is the index.

These tables are appropriate for data that is read frequently via an equality comparison on the key. With an indexed table or indexed cluster, Oracle Database locates table rows using key values stored in a separate index. So we use hash values only for grouping by blocks.



# Row Migration

Migrated rows occur when an UPDATE DML causes the rows to expand onto another data block. This can be avoided by setting PCTFREE to a large enough value to accommodate row expansion, and existing migrated rows can be fixed by reorganizing the tables with the dbms\_redefinition utility. Row migration is when a row is forced to leave the block it was created on because it grew too large to fit on that block with the rest of the rows.

When Oracle migrates the row, it will leave behind a pointer to where the row really is.

Also, migrated rows only degrade performance for single block fetches, and they will not impact the performance of full table scan operations.



