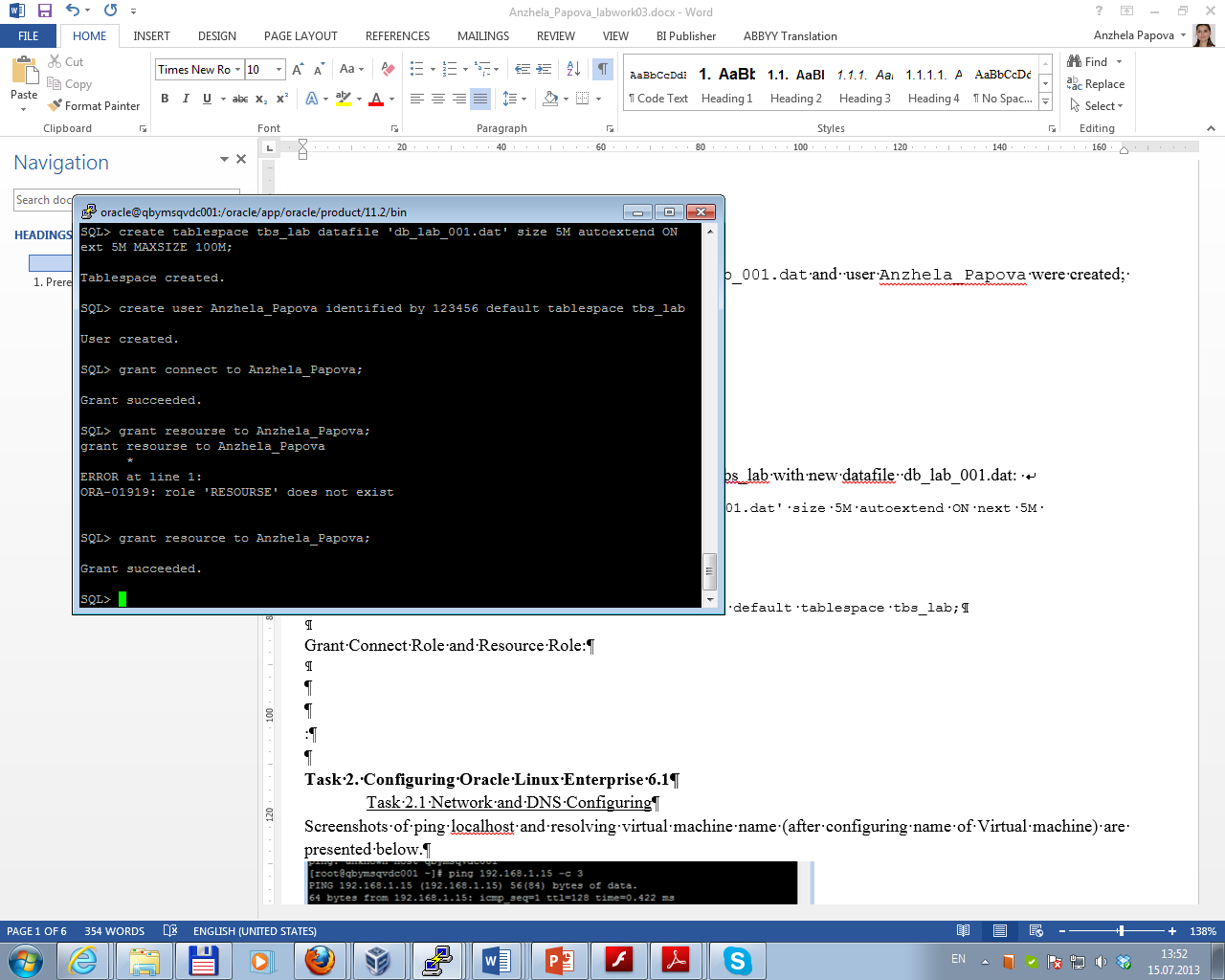
Here are the results of labwork 03:

**Task 1. Prerequisites**

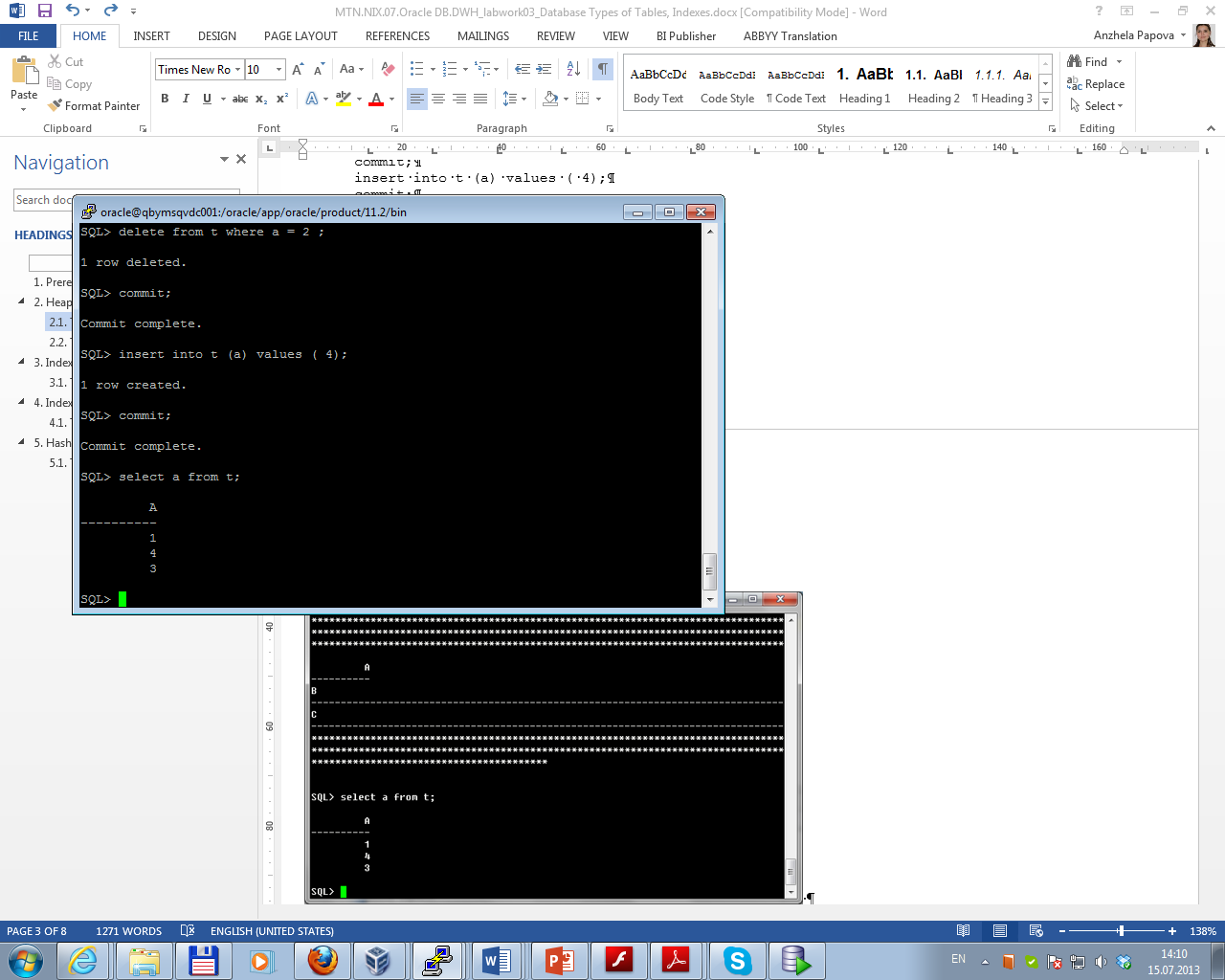
At first new tablespace tbs\_lab with new datafile db\_lab\_001.dat and user Anzhela\_Papova were created; then connect role and resource role were granted:

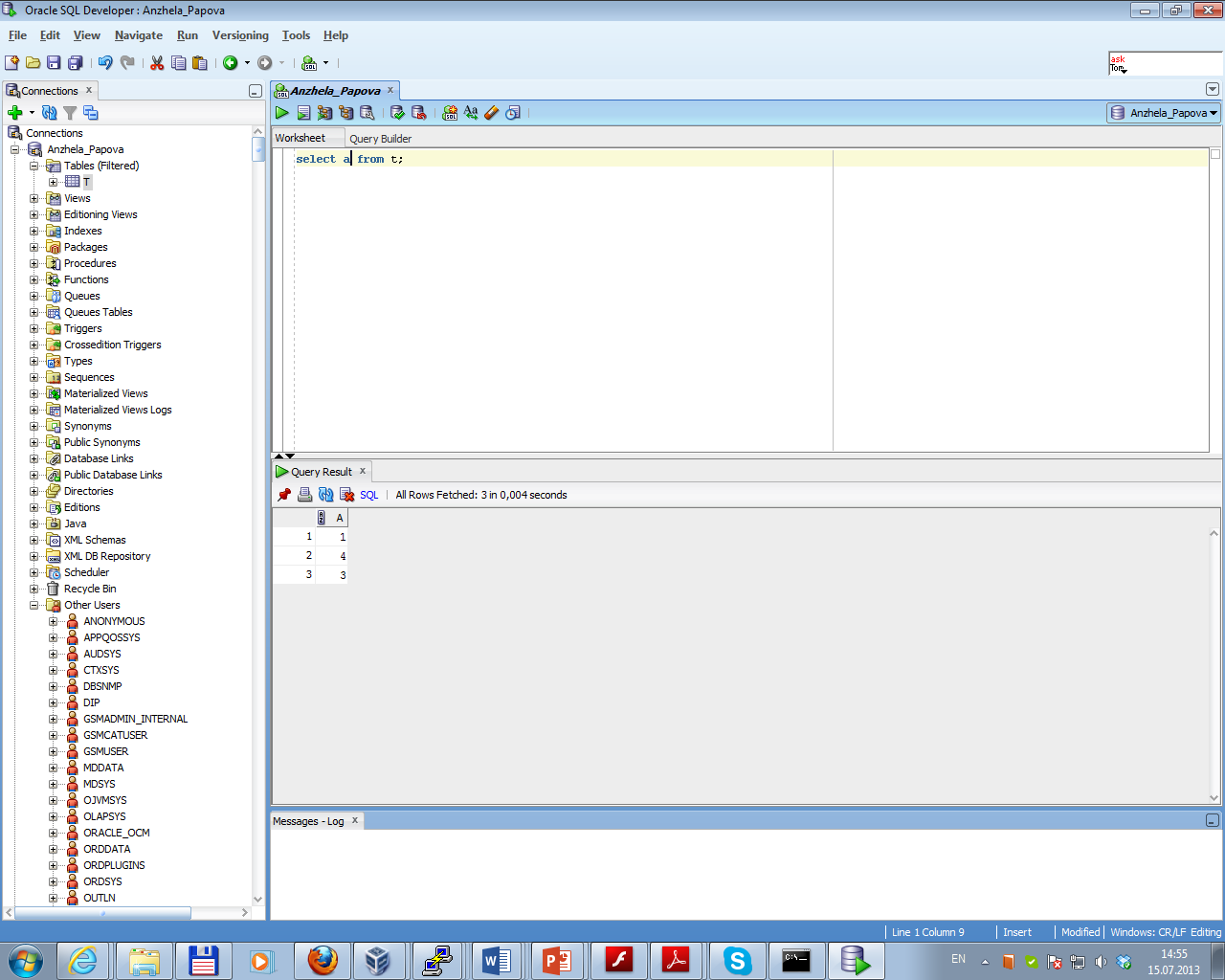


**Task 2.** **Heap Organized Tables**

Task 2.1 Heap Understanding

Screenshot of creating table t and data manipulations is presented below/



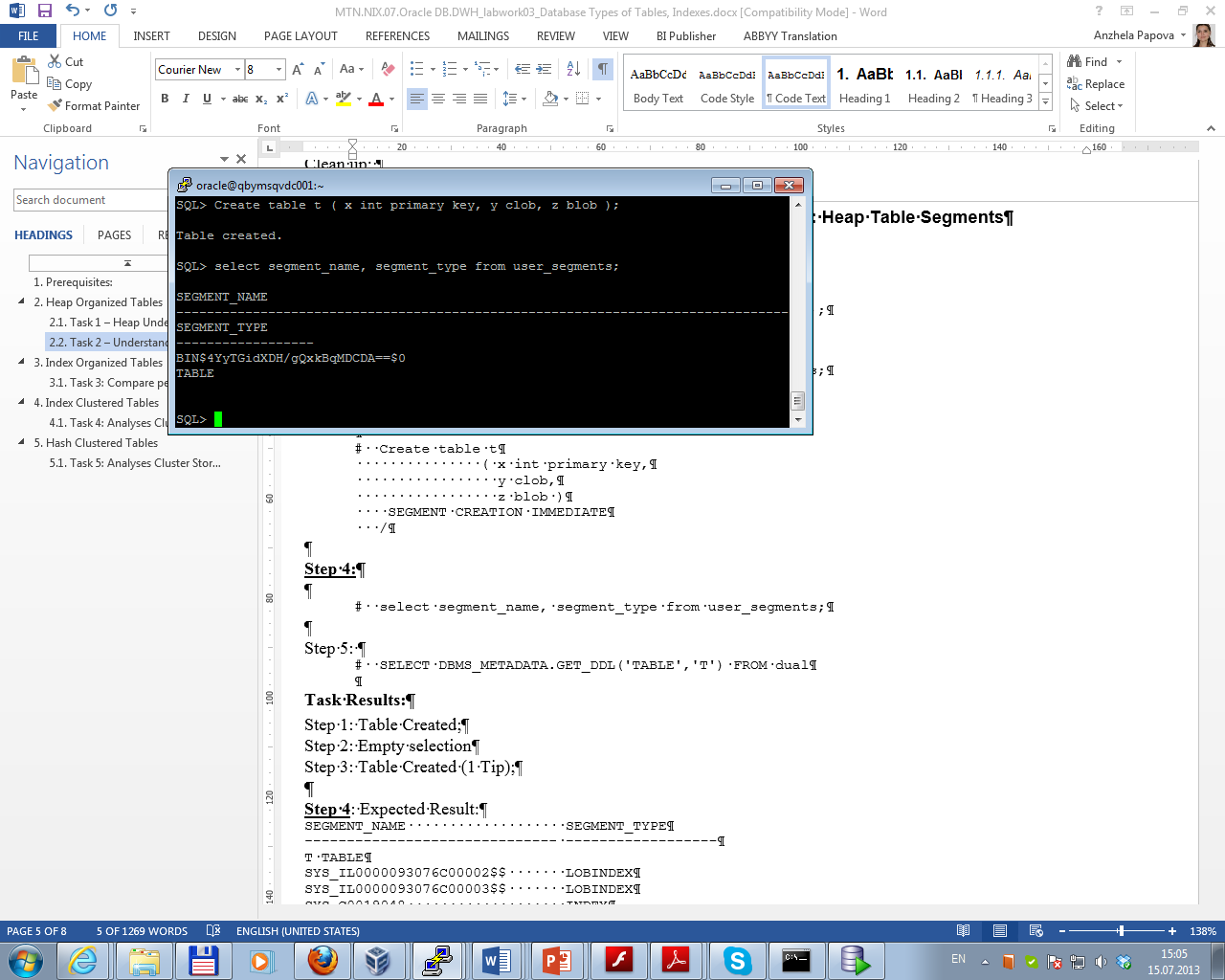


So, heap organized table is a big unordered collection of rows. These rows will come out in a random order, and depending on other options being used; they may come out in a different order with the same query.

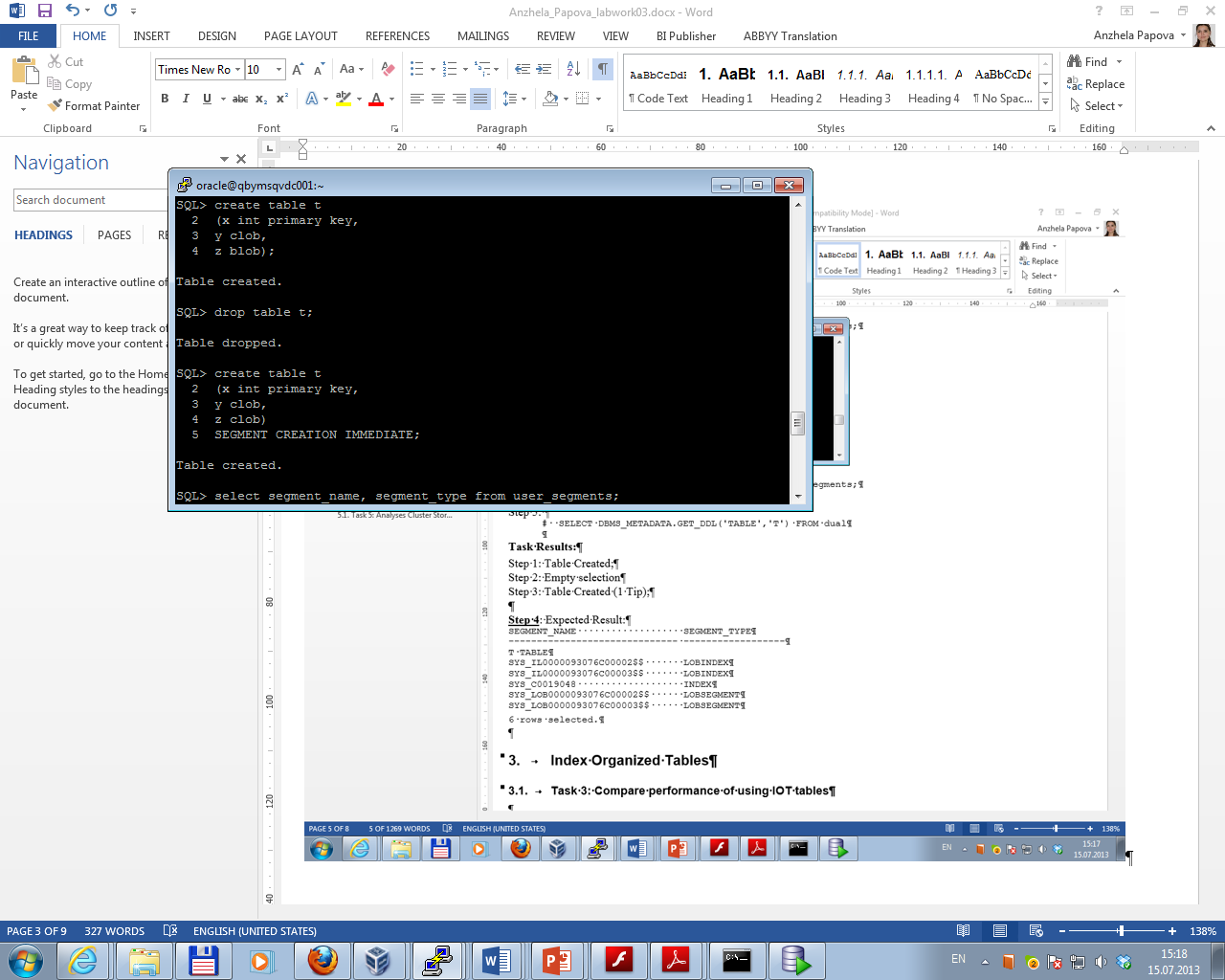
Task 2.2 Understanding Low level of data abstraction: Heap Table Segments

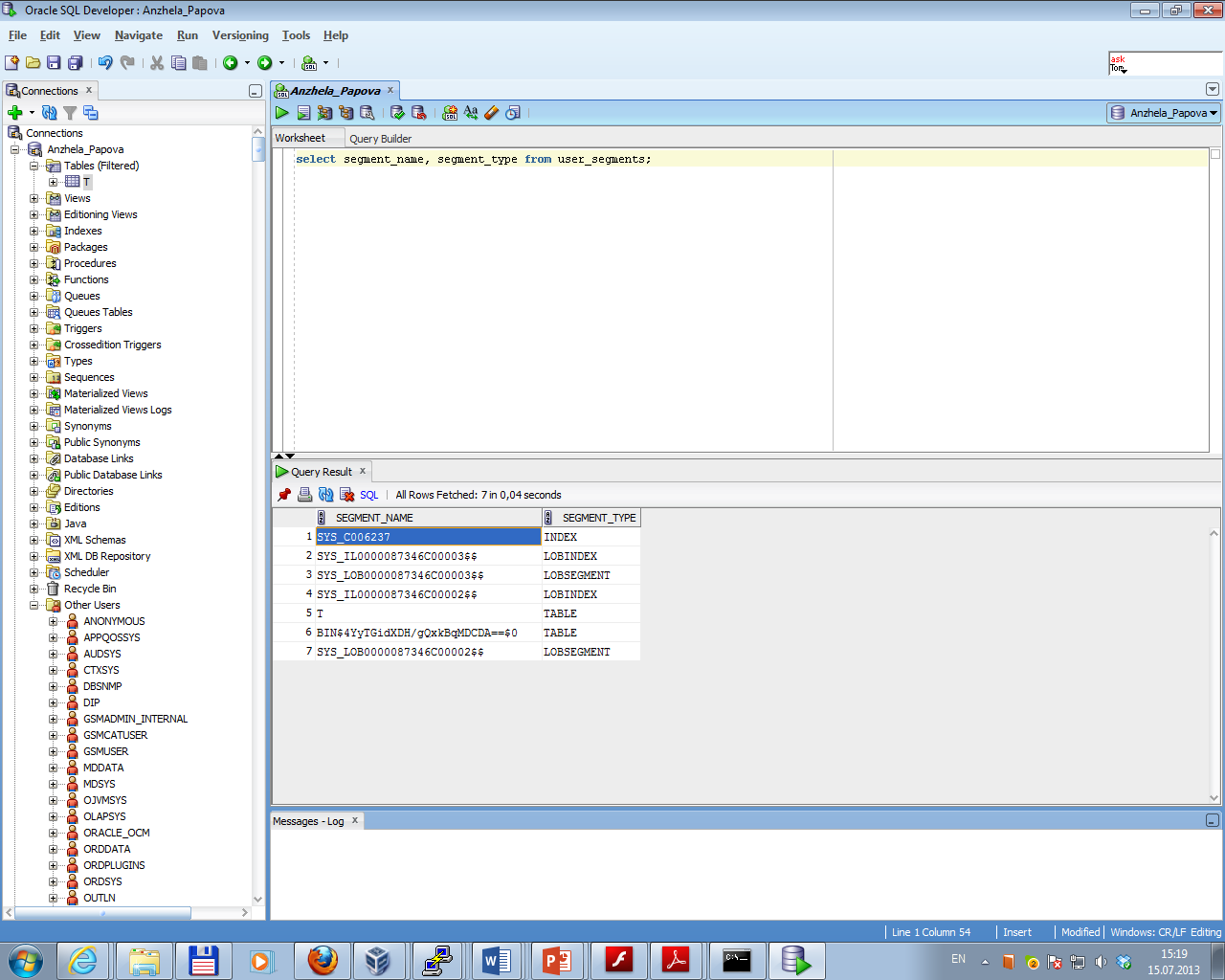
Oracle stores data in data **blocks** (that correspond to a specific number of bytes of physical database space on disk). The next level of logical database space is an **extent** (a specific number of contiguous data blocks allocated for storing a specific type of information). The level of logical database storage above an extent is a **segment** (a set of extents, each of which has been allocated for a specific data structure and all of which are stored in the same tablespace; each table's data is stored in its own data segment, while each index's data is stored in its own index segment; if the table or index is partitioned, each partition is stored in its own segment).

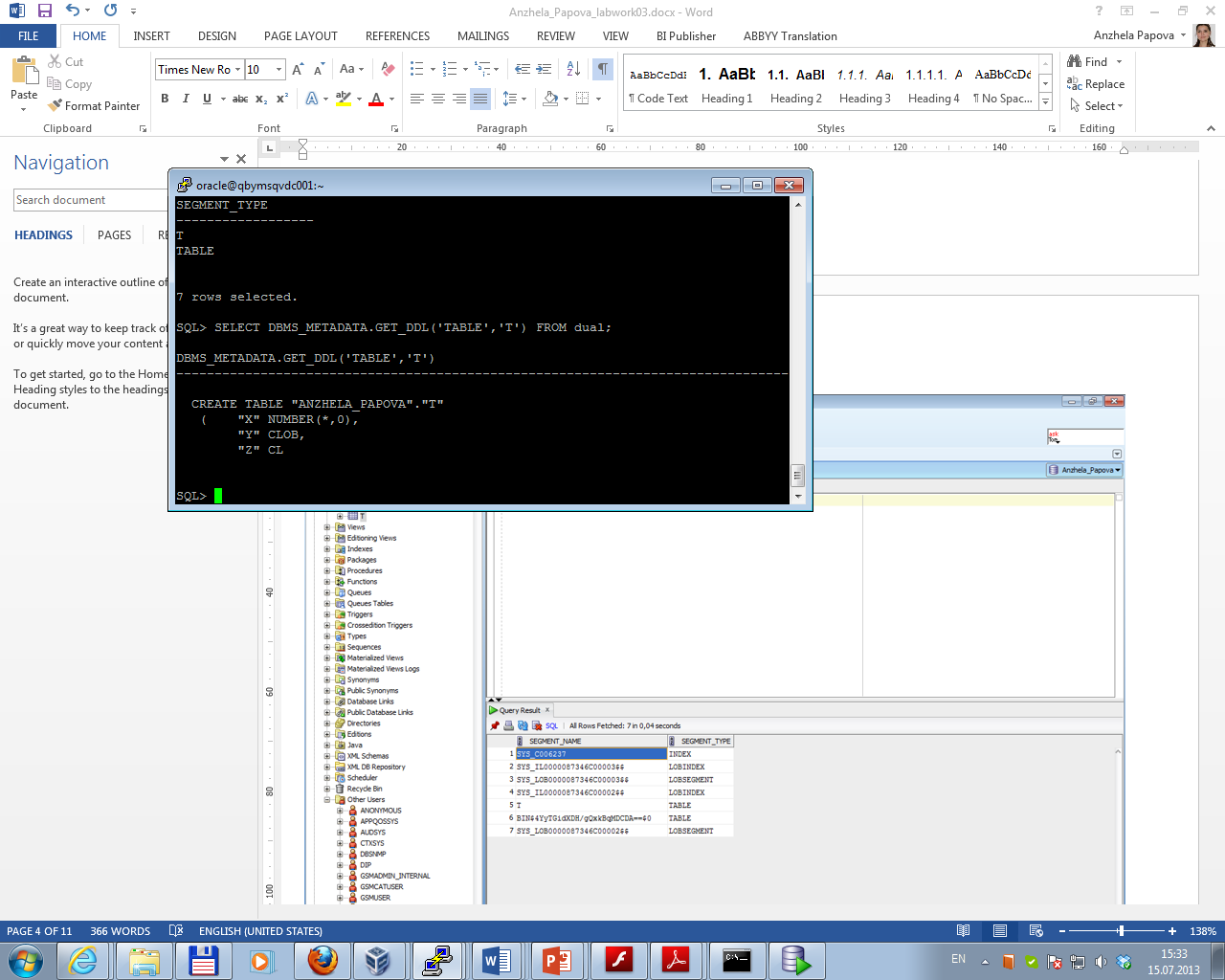
When non-partitioned tables are created, none of the associated segments (table, implicit index and LOB segments) are created until rows are inserted into the table. The functionality can be controlled by the DEFERRED\_SEGMENT\_CREATION initialization parameter, which is set to TRUE by default.



The default behavior is altered by using the IMMEDIATE clause (in this case segments for data table, index table, LOB were created immediately):





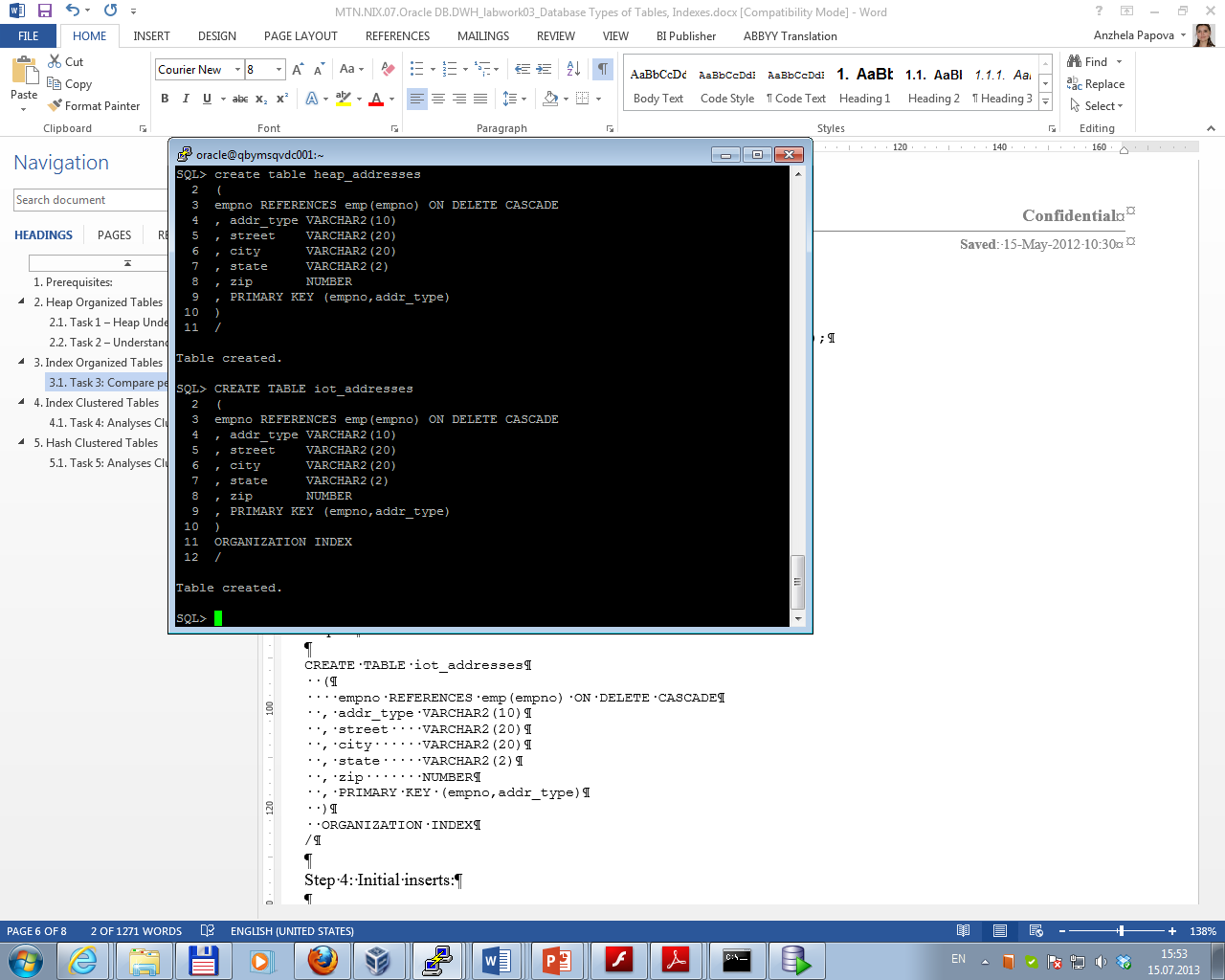


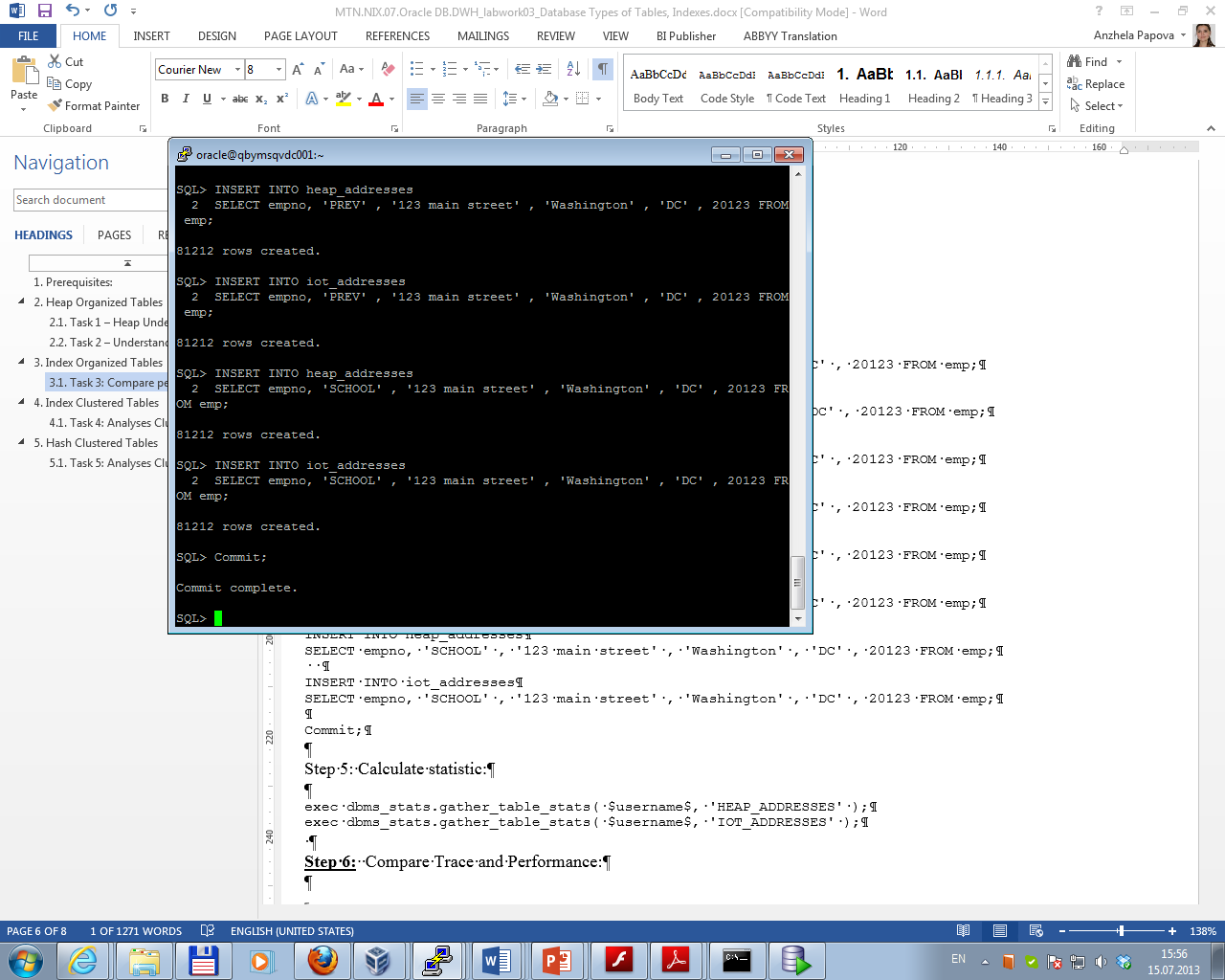
**Task 3. Index Organized Tables**

Task 3.1 Compare performance of using IOT tables

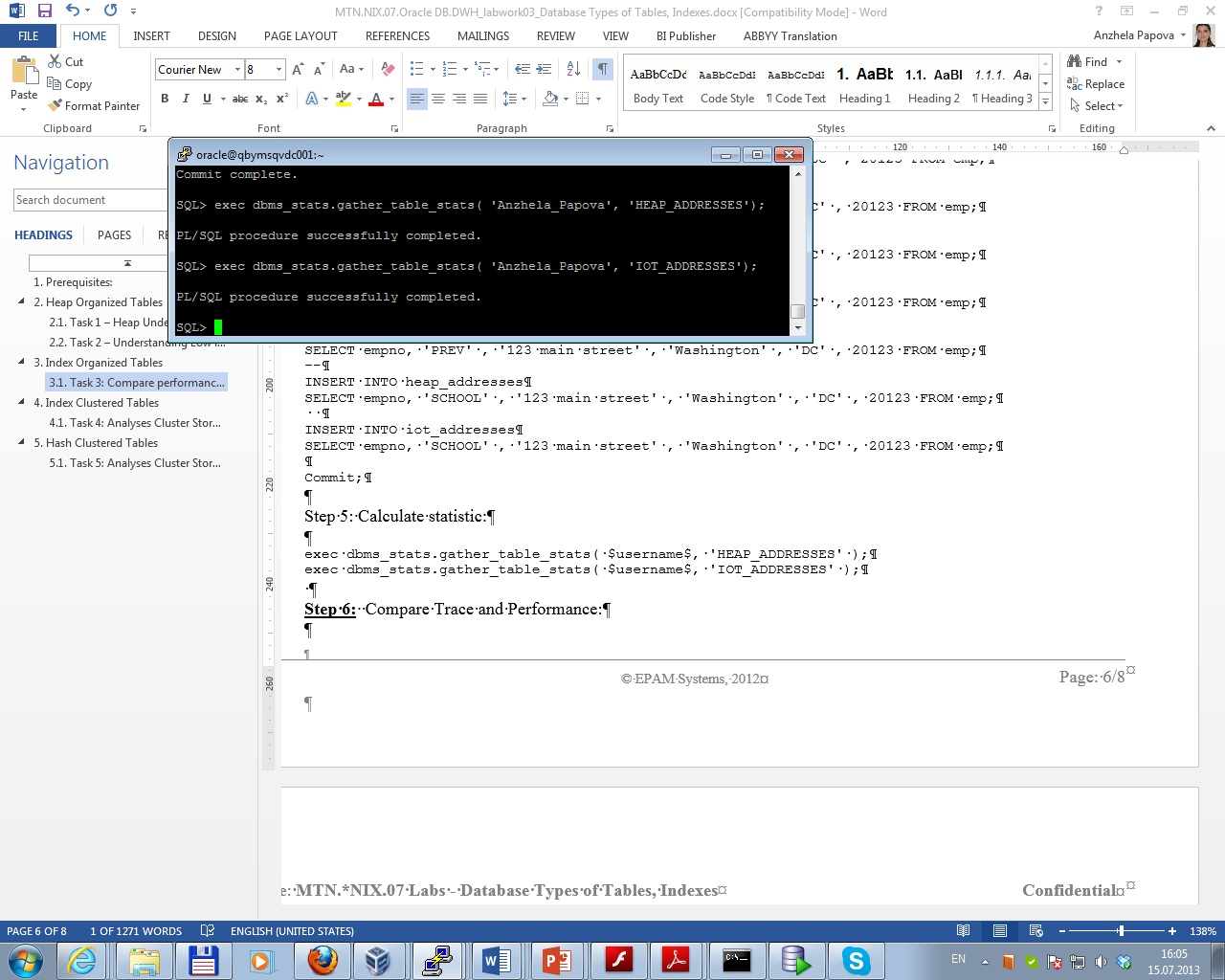
Index organized tables (IOTs) are quite simply tables stored in an index structure. Whereas a table stored in a heap is unorganized, data in an IOT is stored and sorted by primary key.

To compare performance of using Heap Organized and Index Organized Tables two tables were created; after that some data was inserted in each of them:

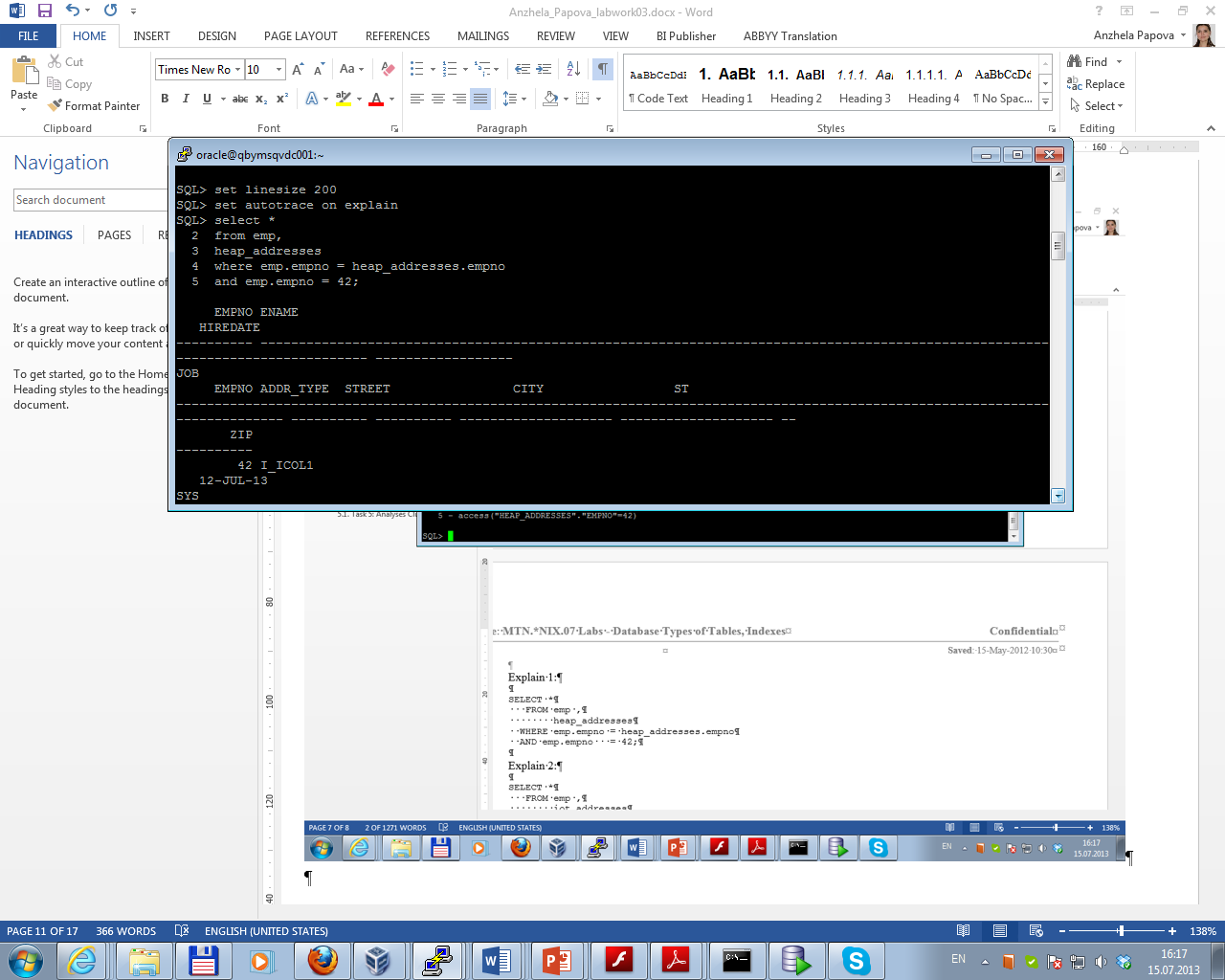




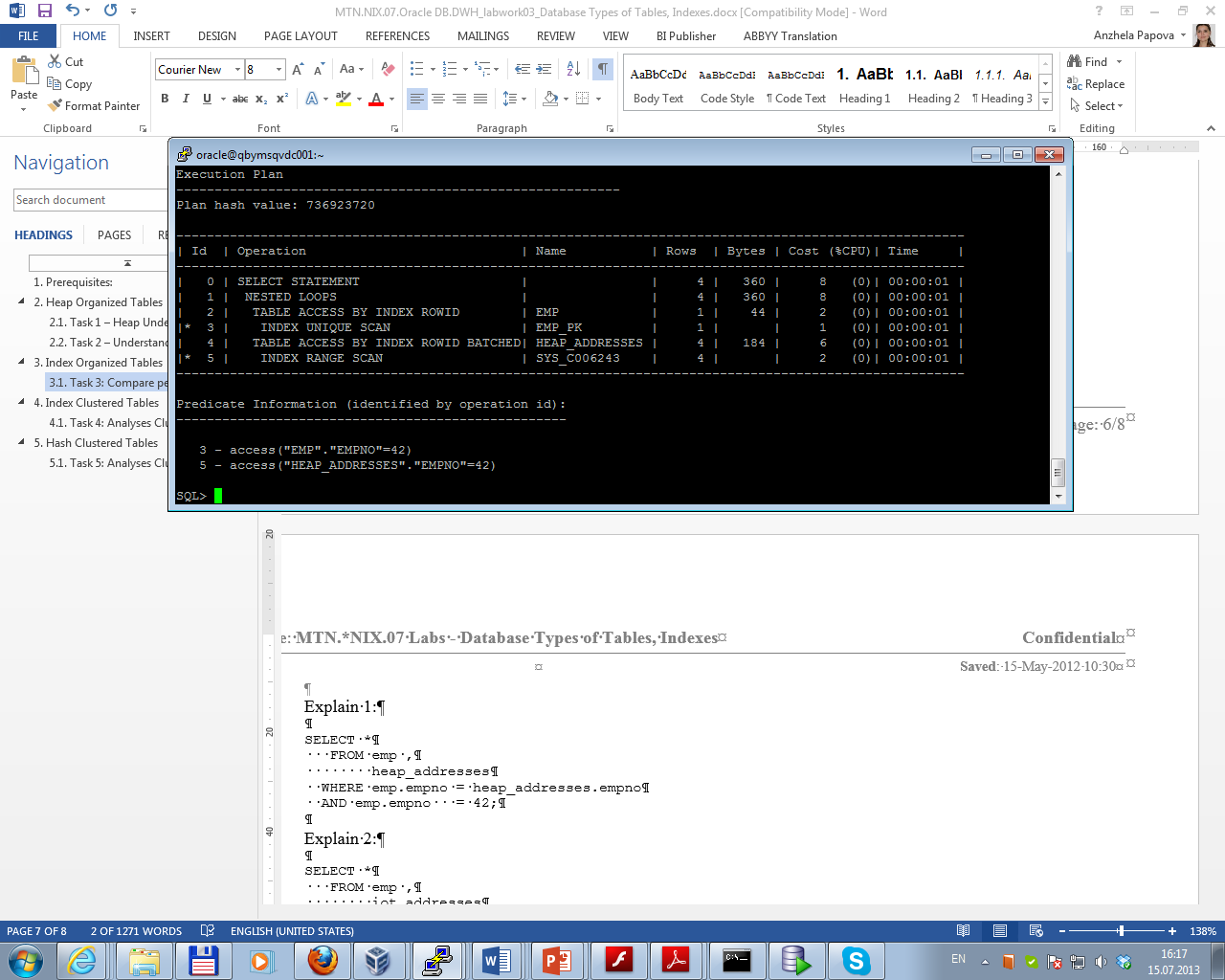
Then statistic was gathered:



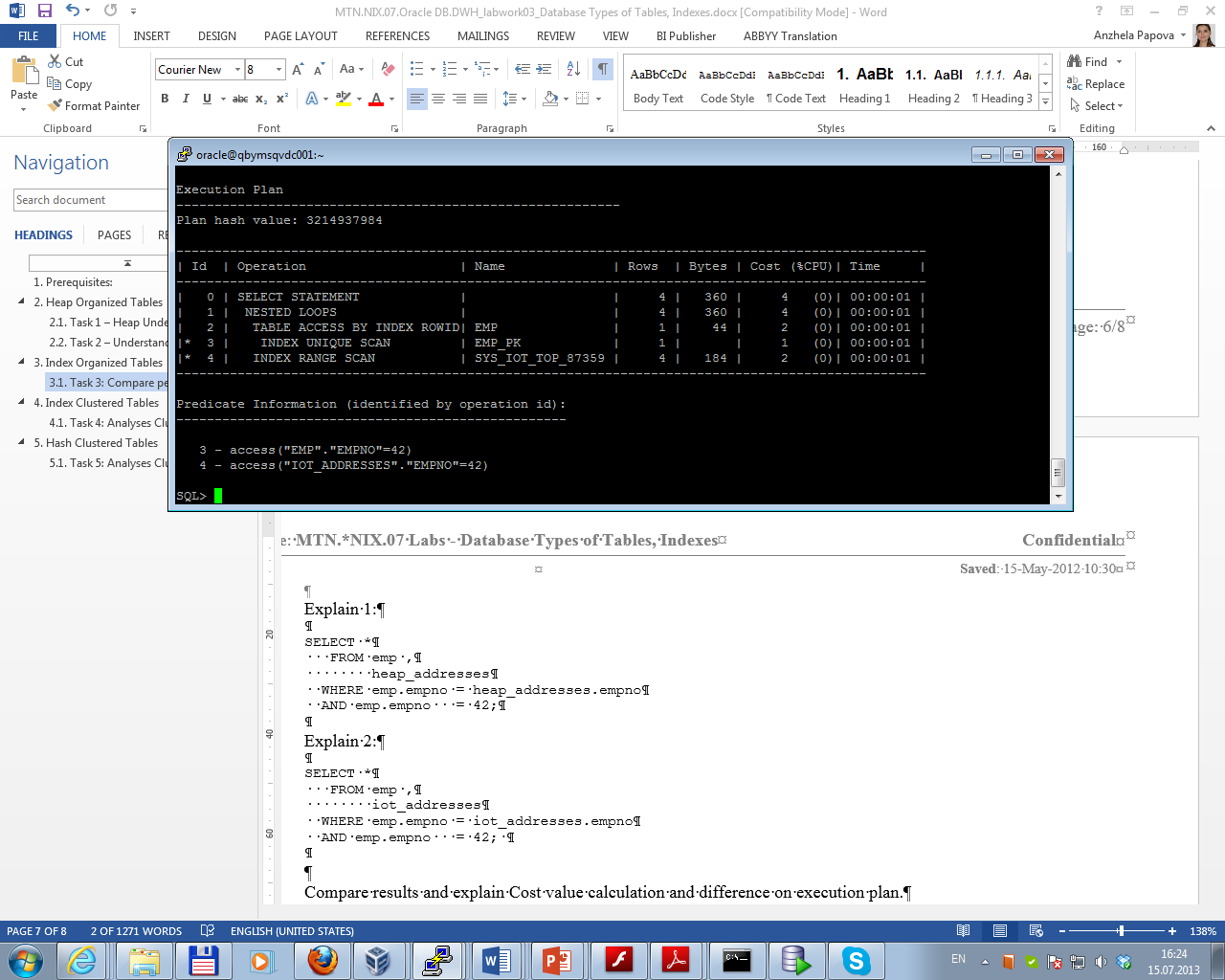
The next step is getting Execution plan by running command AUTOTRACE:



The result for Heap Organized table:



The result for Index Organized table:



Execution Plan is to go to the EMP table by primary key (EMP\_PK); get the row (EMP.EMPNO = 42); then using this EMPNO, go to the address table; and using the index, pick up the child records.

To get the result of the query, in case of Heap Organized Table it was necessary to get additionally four TABLE ACCESS (BY INDEX ROWID BATCHED); cost value total 8.

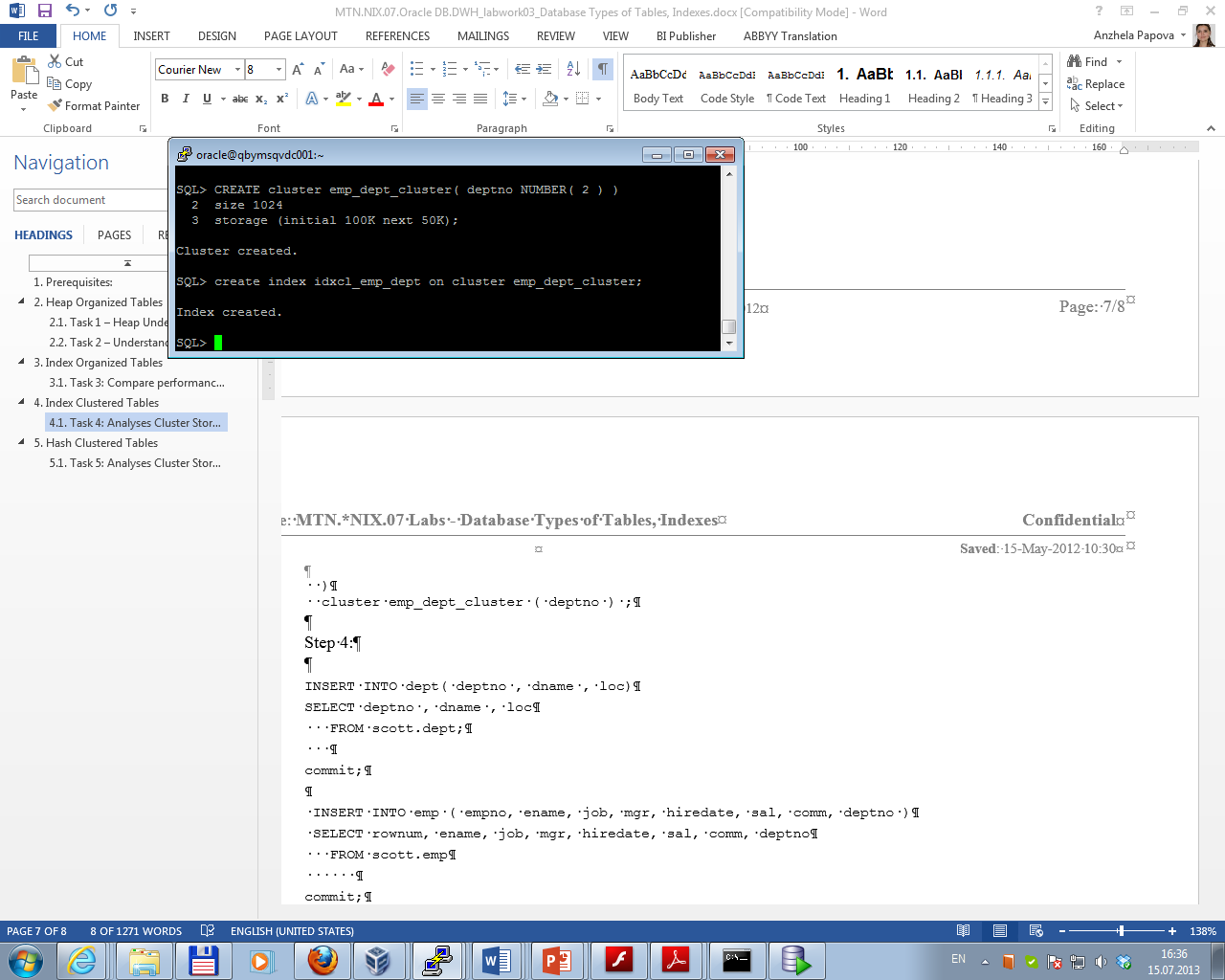
In case of Index Organized Table, since the data is sorted by index and data about each of employee is located close there is no need to browse entire child table; cost value total 4.

So the difference in performance is measurable.

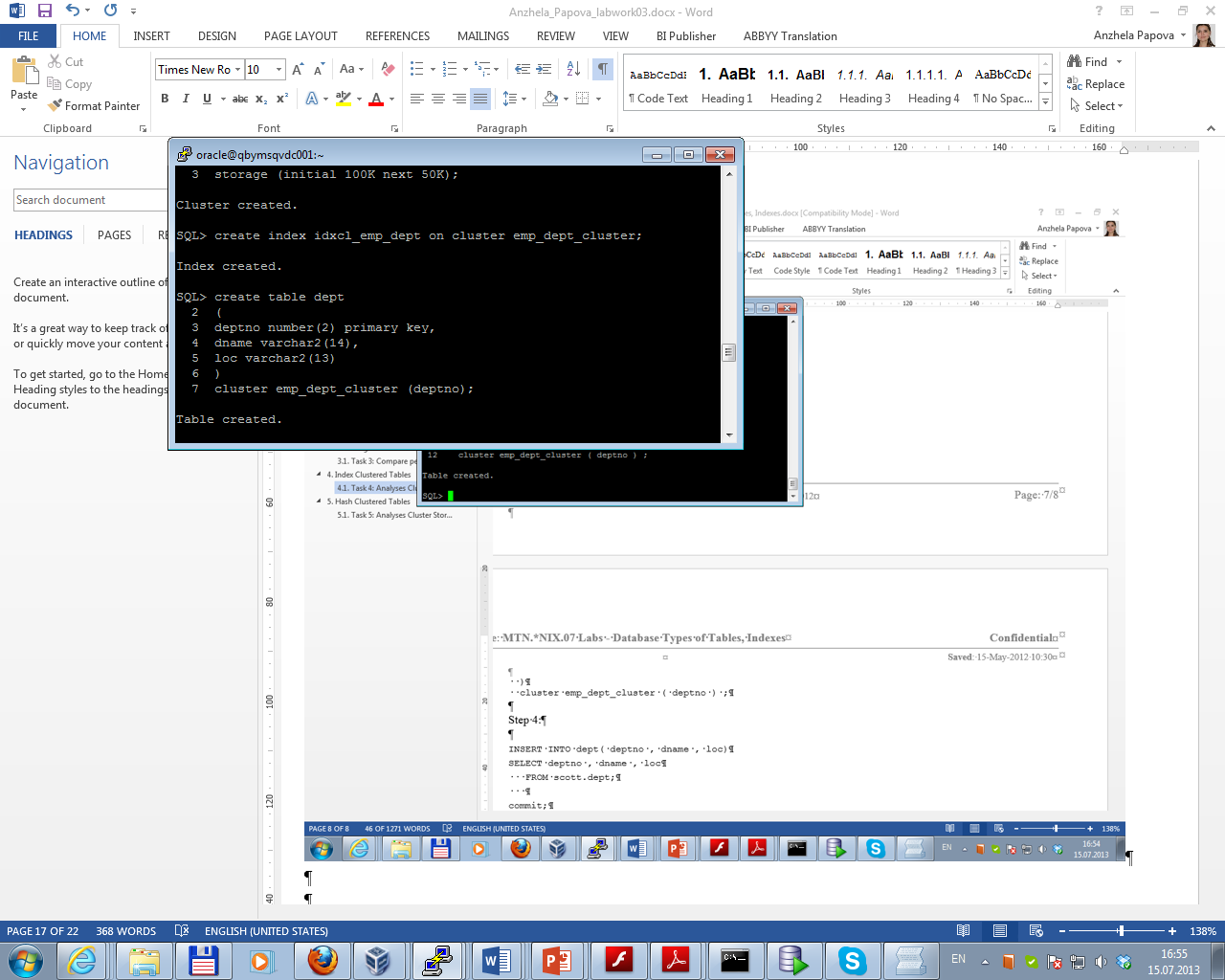
**Task 4. Index Clustered Tables**

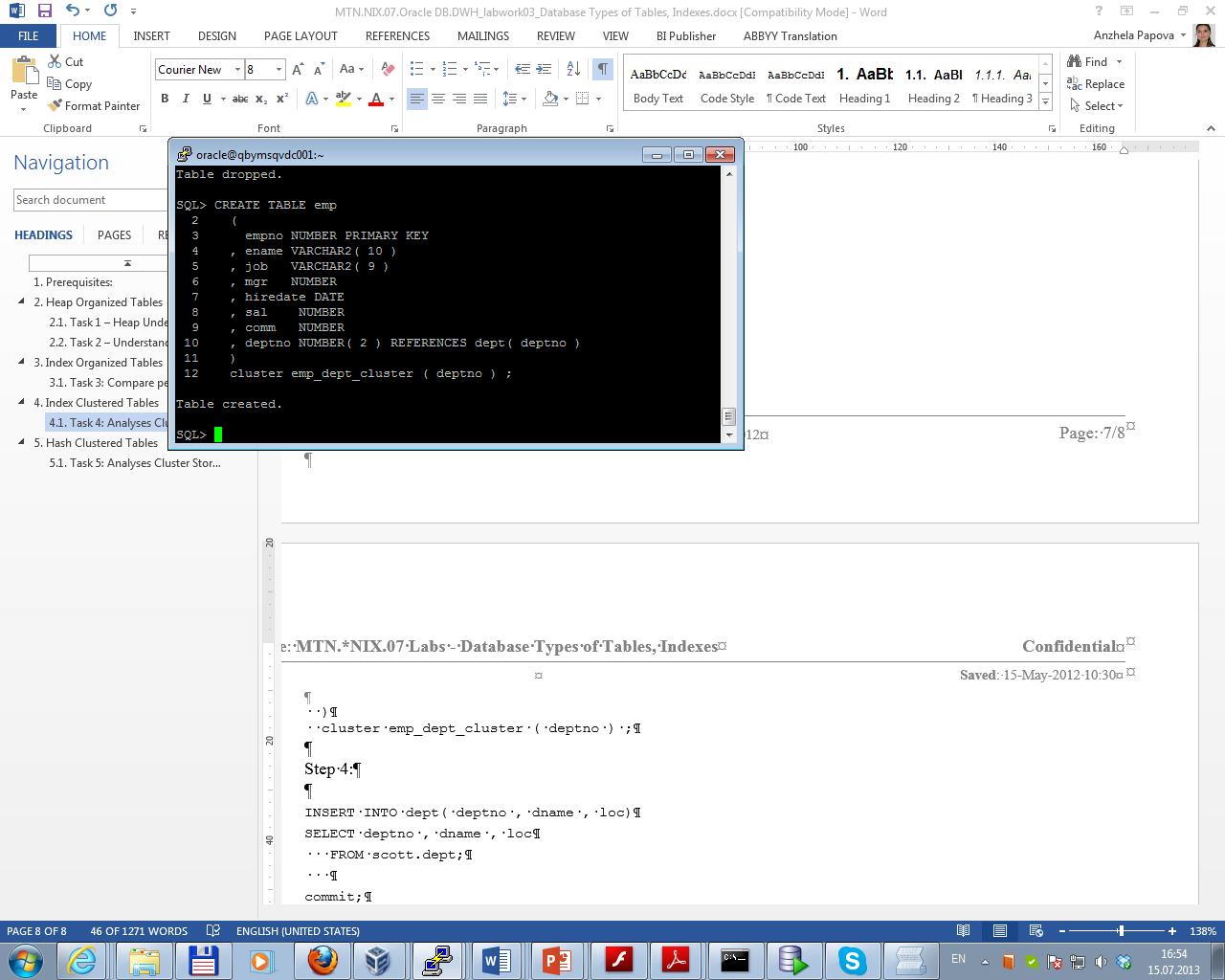
Task 4.1 Analyses Cluster Storage by Blocks

At the first cluster and index (before inserting data) were created:

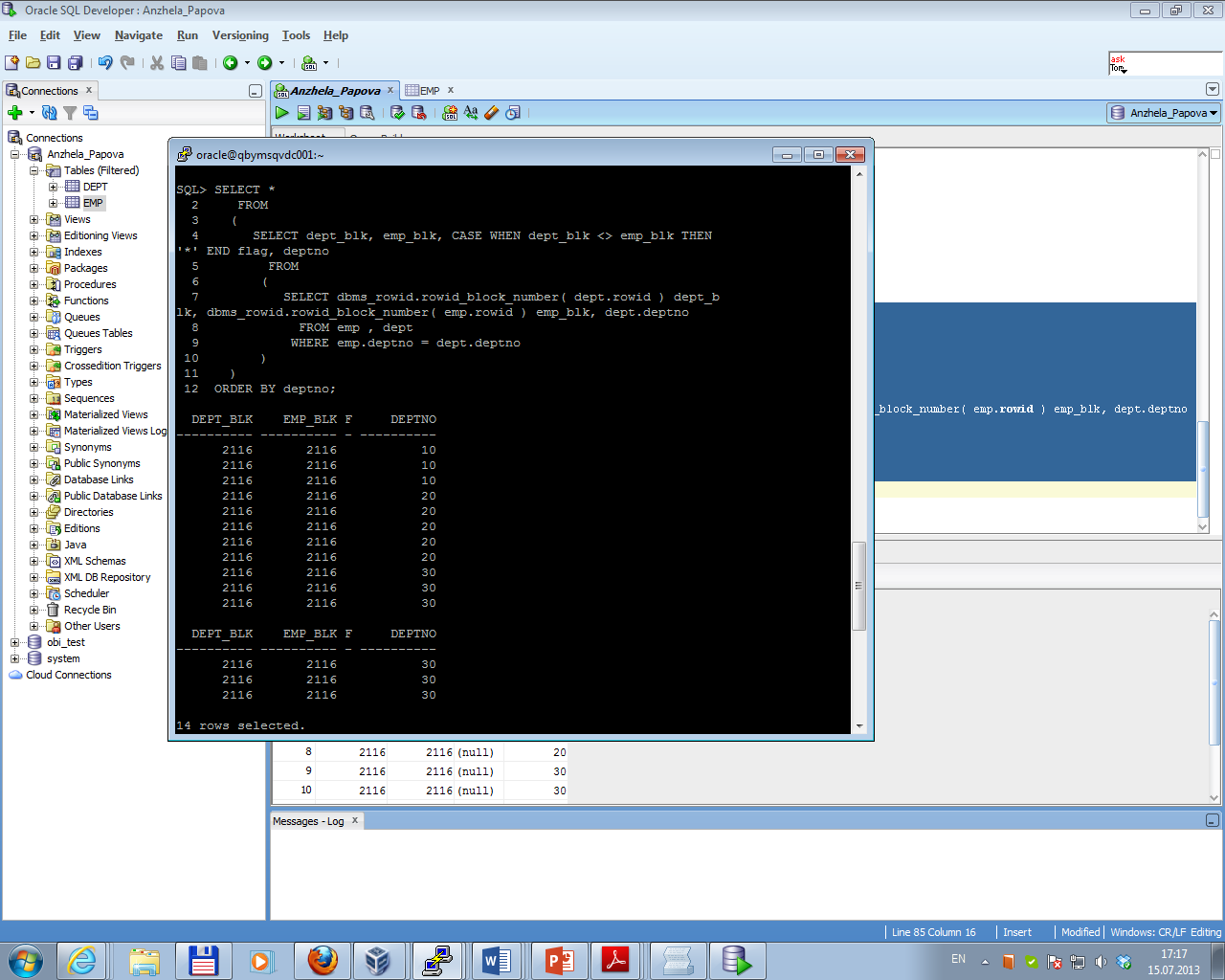


Then tables were created and data was inserted:





As shown below, all data is stored on the same block:



With a cluster, a single block of data may contain data from many tables (the data is “prejoined”). But it is not storing the data sorted—that is the role of the IOT; it is storing the data clustered by some key, but in a heap. We’ve defined cluster on these two tables (EMP\_DEPT\_CLUSTER), that’s why they are stored on the same physical database block.

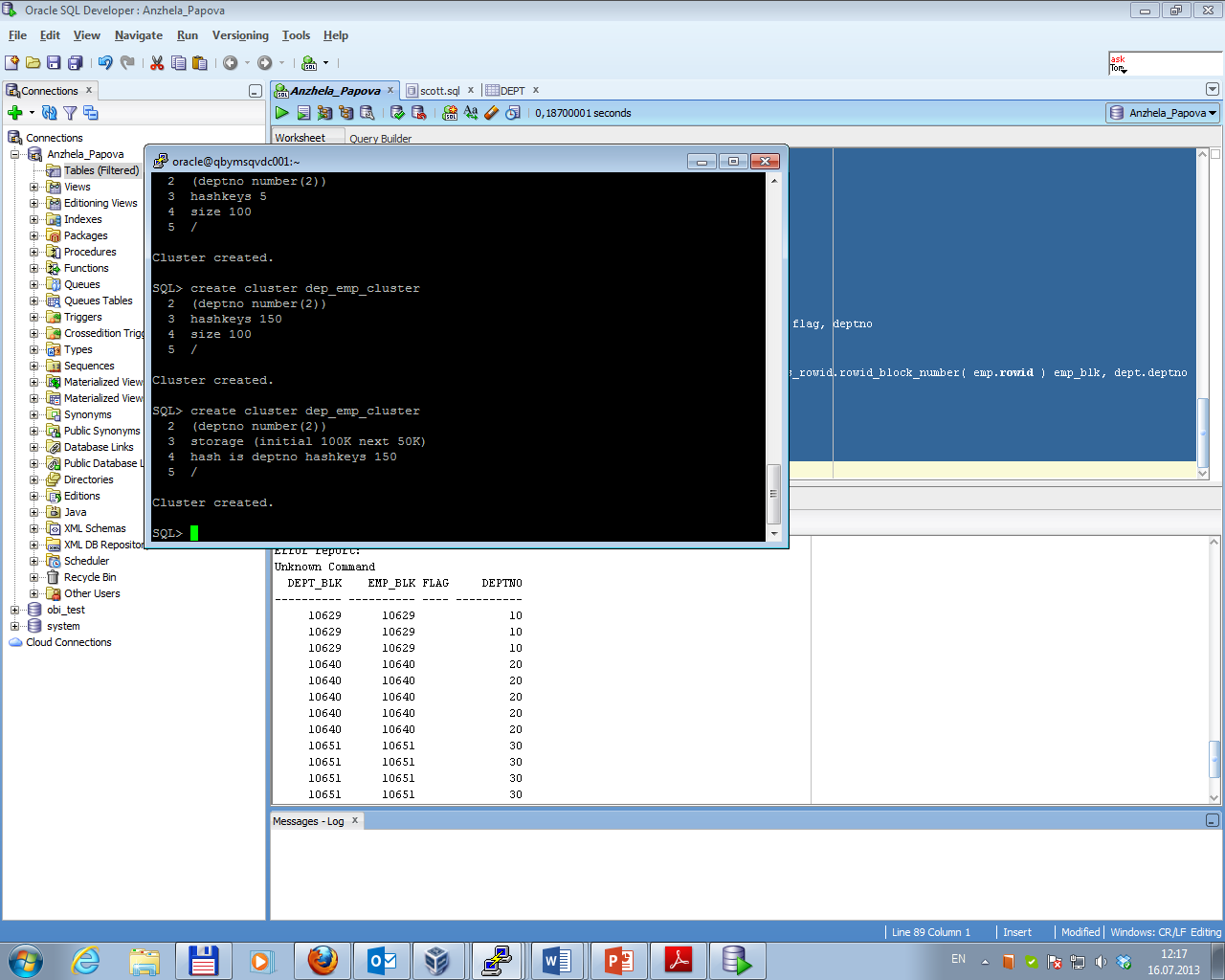
The advantages of this type of storage are:

* clustered tables give the ability to physically prejoin data together;
* clusters can help read-intensive operations that always join data together or access related sets of data;
* clustered tables reduce the number of blocks that Oracle must cache.

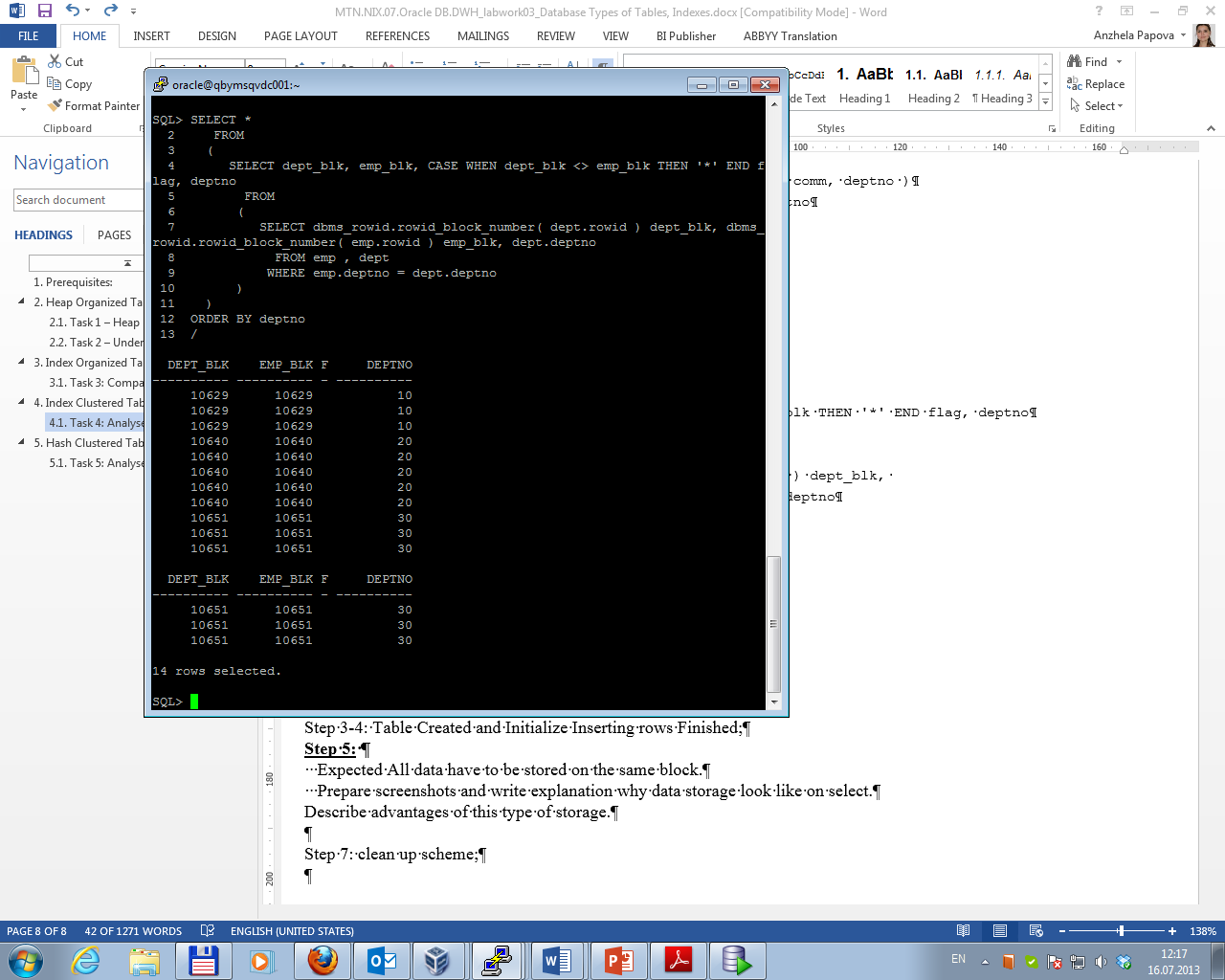
**Task 5. Hash Clustered Tables**

Task 5.1 Analyses Cluster Storage by Blocks

At the first cluster was created (cluster index is not used; the data is the index in this case):



Then tables were created and data was inserted. As shown below, data about employees by their departments is stored on the different blocks:



The advantage of this type of storage is light data retrieval. But to use hash clustered tables it’s necessary to know with a good degree of accuracy how many rows the table will have over its life (or some reasonable upper bound).