# 1. Prerequisites:

Connect as system user and Create new tablespace tbs\_lab with new datafile db\_lab\_001.dat:

create tablespace tbs\_lab datafile 'db\_lab\_001.dat' size 5M autoextend ON next 5M MAXSIZE 100M;

Create new user:

create user Saveli identified by plsdonotpwndme default tablespace tbs\_lab;

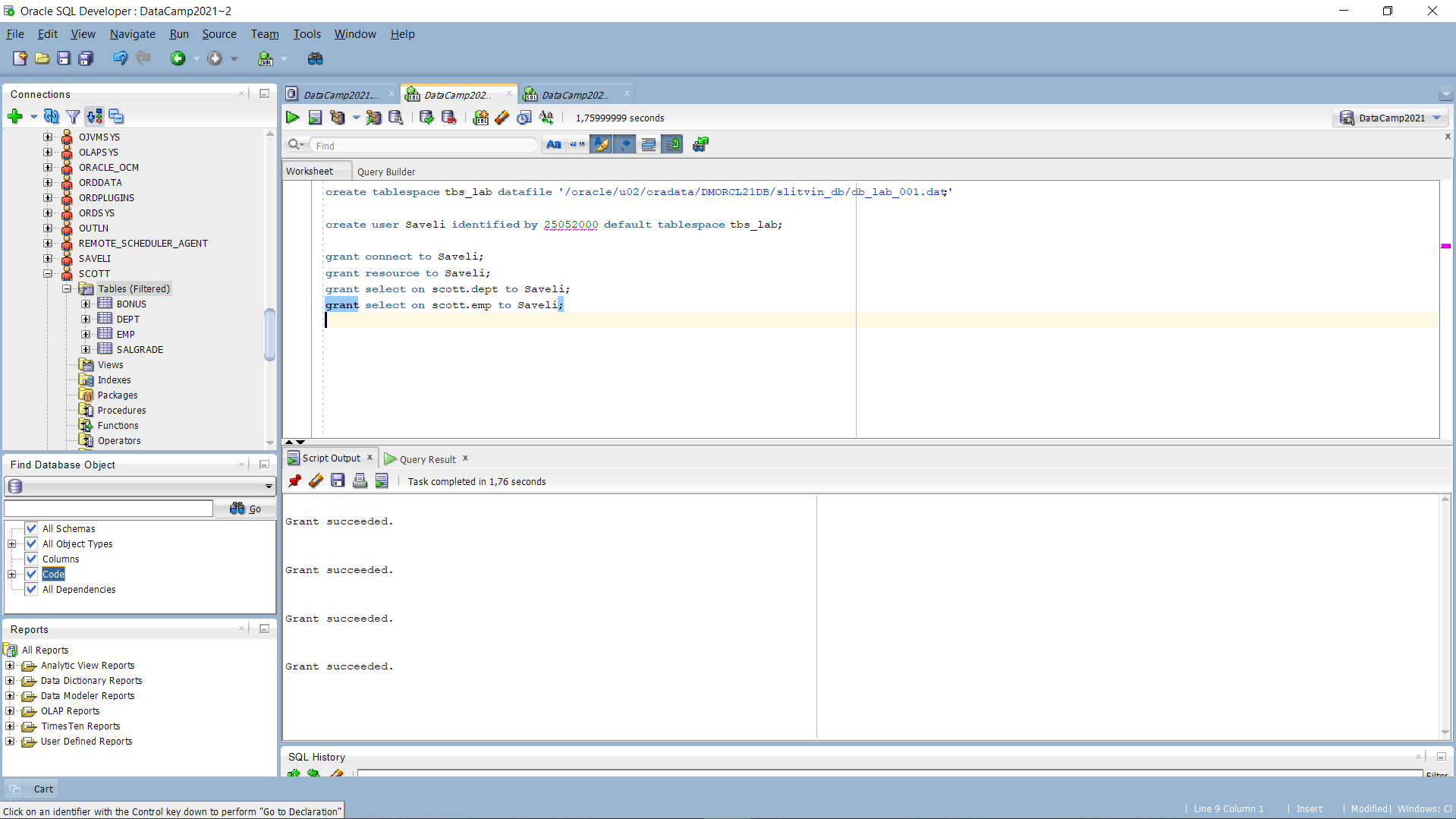
Grant Connect Role and Resource Role:

grant connect to Saveli;

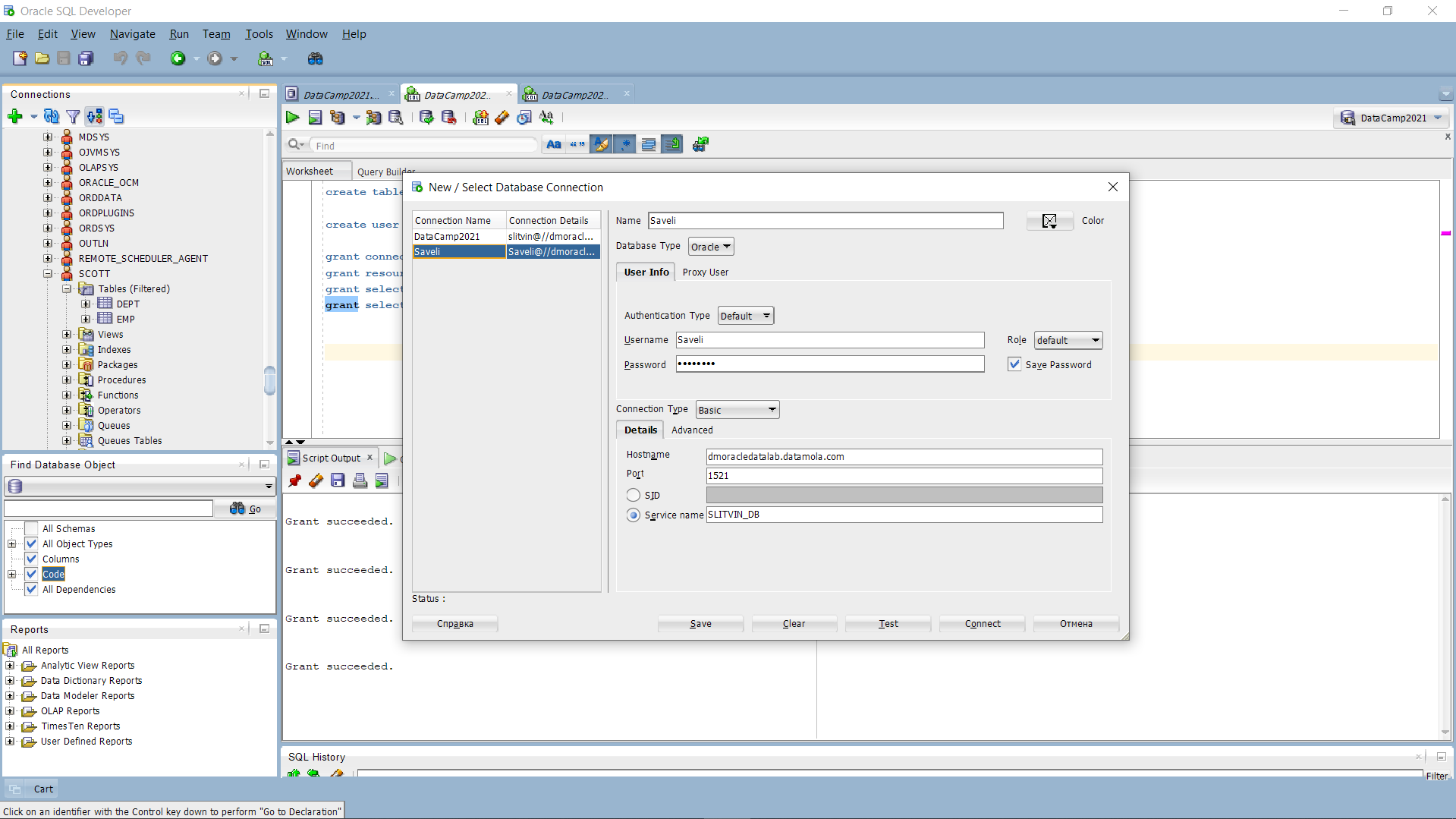
grant resource to Saveli;

grant select on scott.dept to Saveli;

grant select on scott.emp to Saveli;



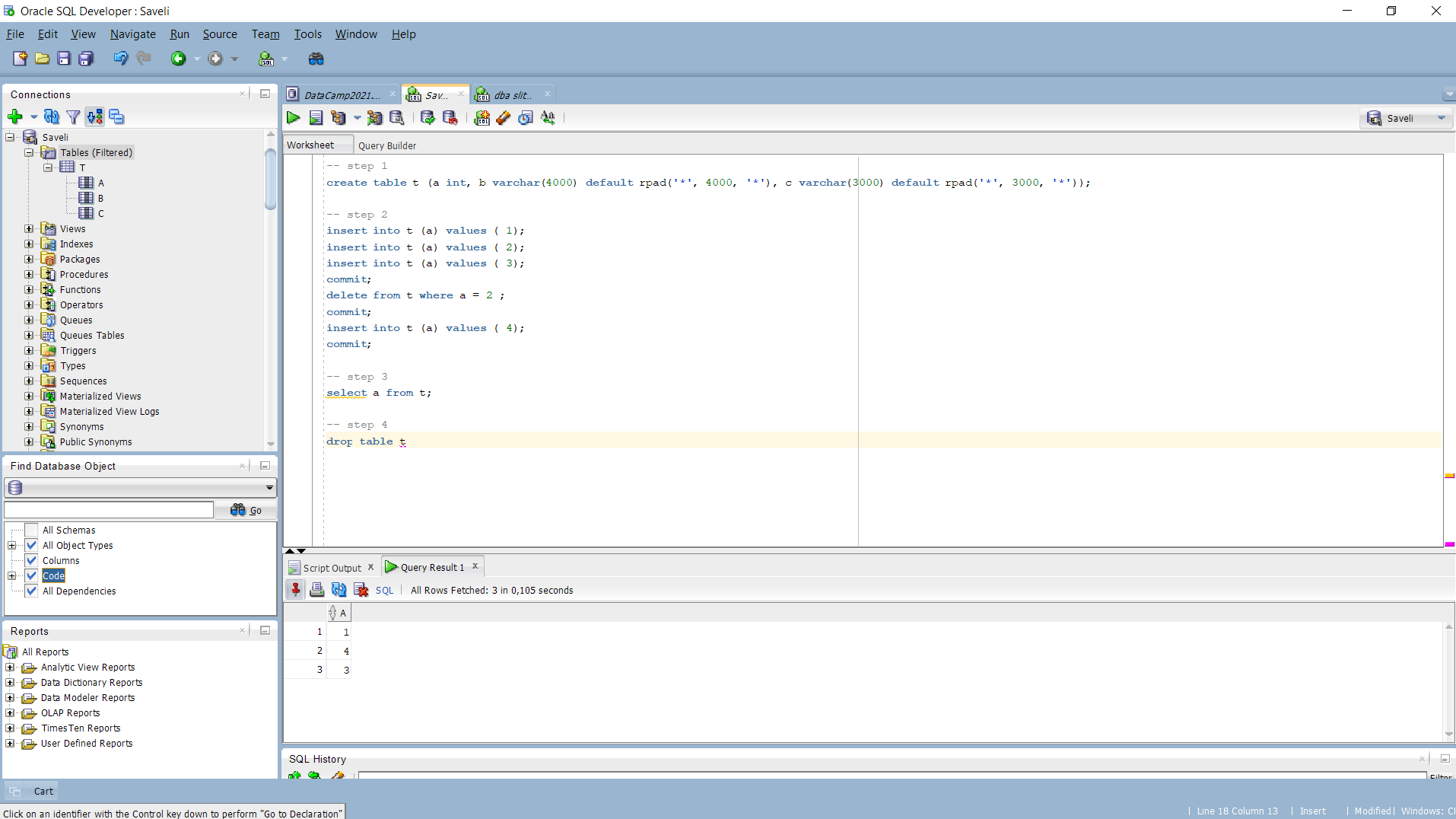
I used web to find scott.dept and scott.emp creation steps. As you can see, at the left side of attached screenshot – Scott user is defined and has 4 tables in our tablespace. All grants were successful, at this way we can enjoy our next task.

Finally I estable new connection via created credentials, and again, at the left side you can see, that there’s only 2 Tables from Scott user granted is available.

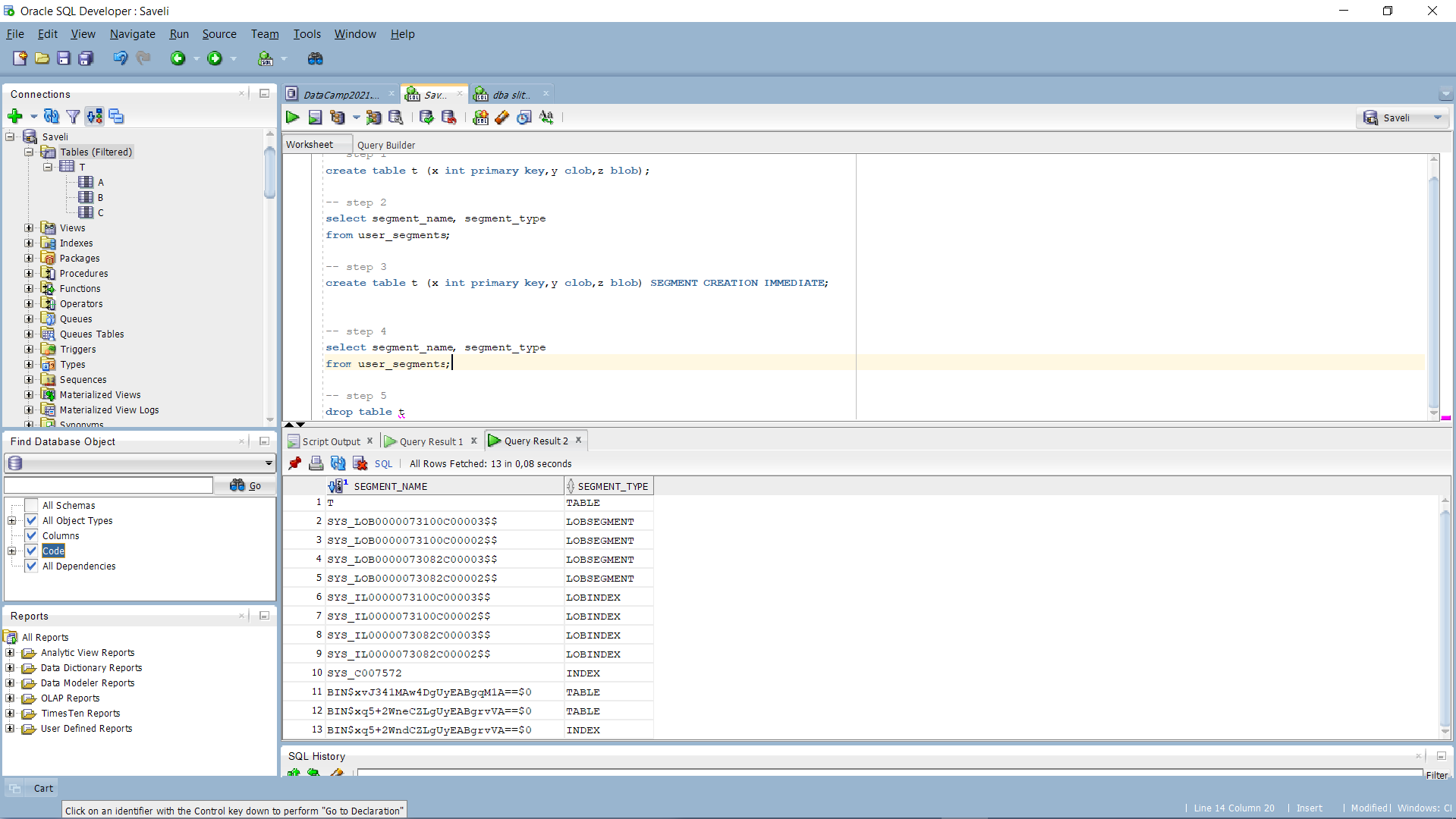
# 2. Heap Organized Tables

## 2.1. Task 1 – Heap Understanding

First output was successful after using command “ALTER USER Saveli quota unlimited on tbs\_lab;” in dba account window. Code output until step 4 attached via screenshot.

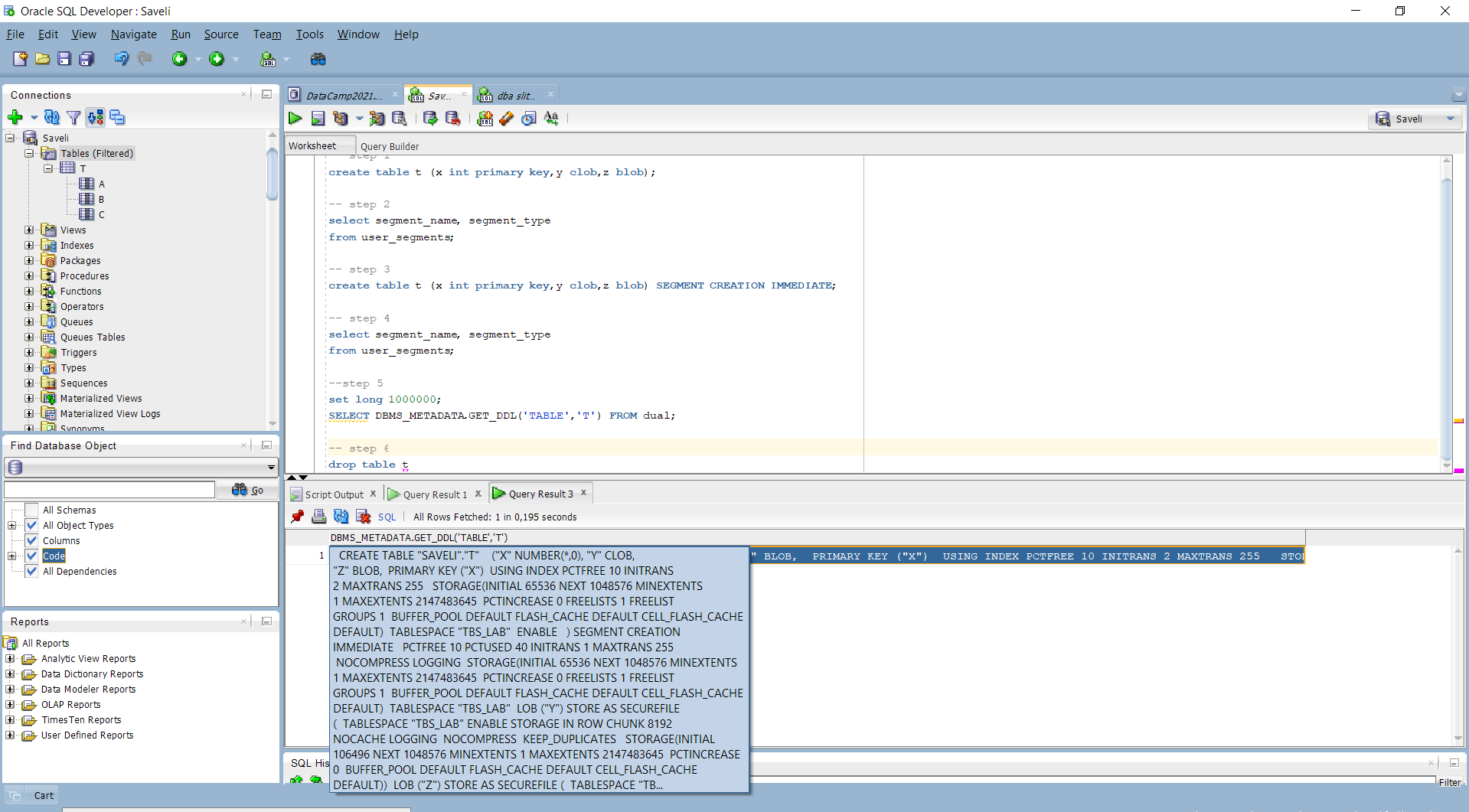


## 2.2. Task 2 – Understanding Low level of data abstraction: Heap Table Segments



The difference between output till step 3 and in steps 3-4 is that T has been included into the Segmenation block.

To check metadata of our table we’ve using next function SELECT DBMS\_METADATA.GET\_DDL('TABLE','T') FROM dual



We can assume that our data stored in predefined above tablespace tbs\_lab. Also, we’re retrieving our metadata as dictionary in xml.

# 3. Index Organized Tables

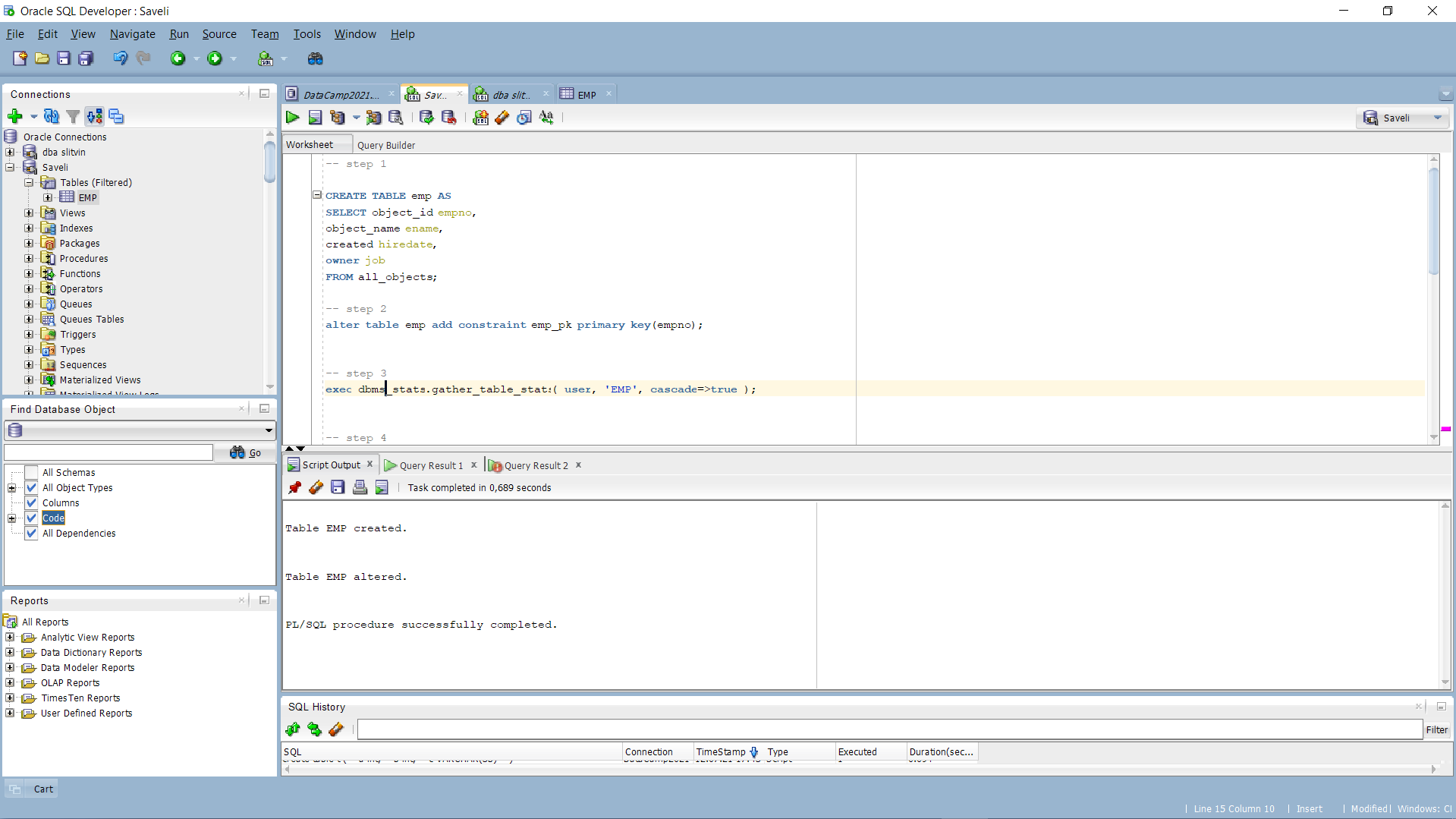
Task 3: Compare performance of using IOT tables

Table is created and primary key is applied. By initiate a second step I’ve got an error ORA-01658, the solution was to init this command from dba connection :

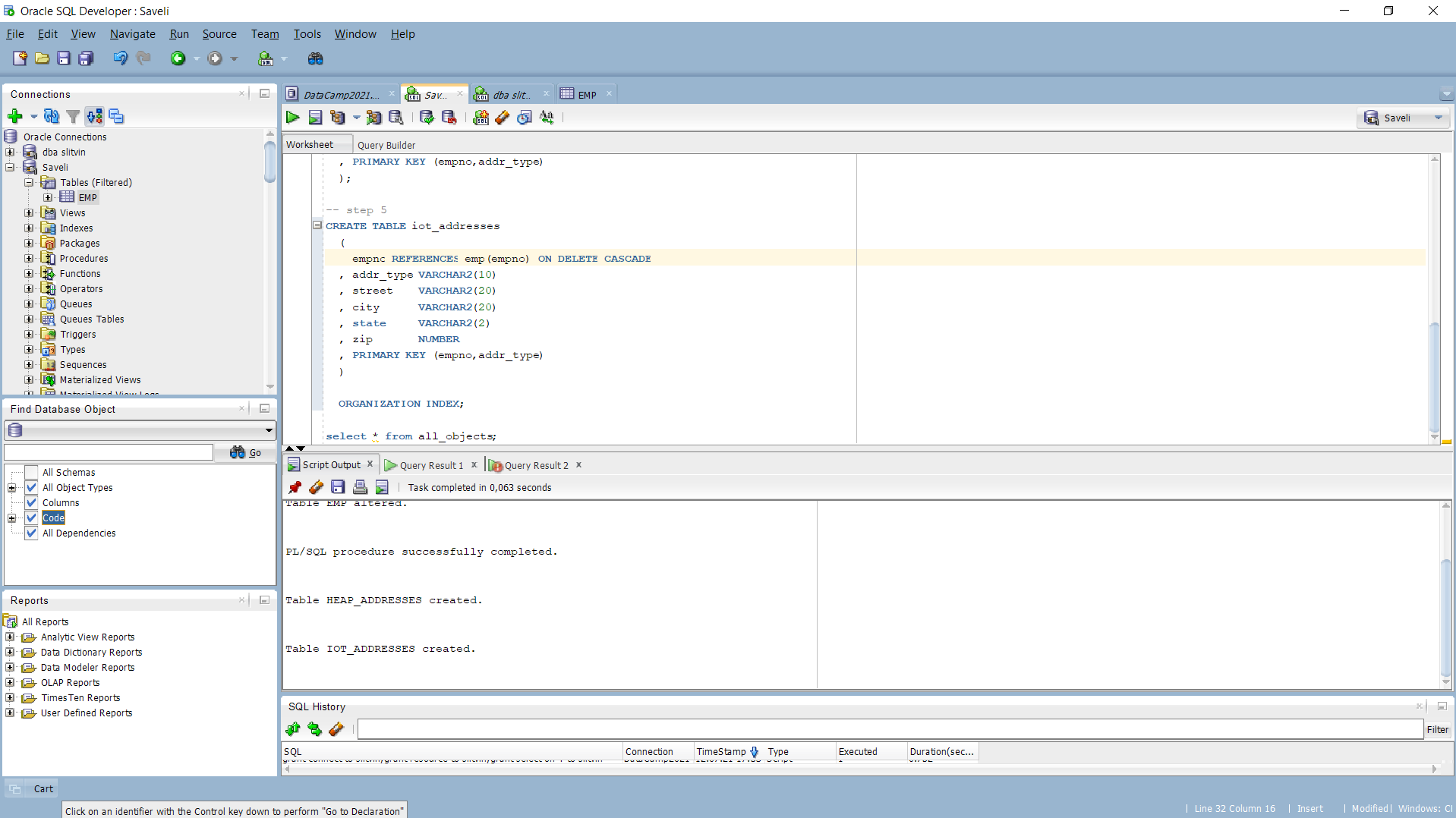
*alter database*

*datafile '/oracle/u02/oradata/DMORCL21DB/slitvin\_db/db\_lab\_001.dat'*

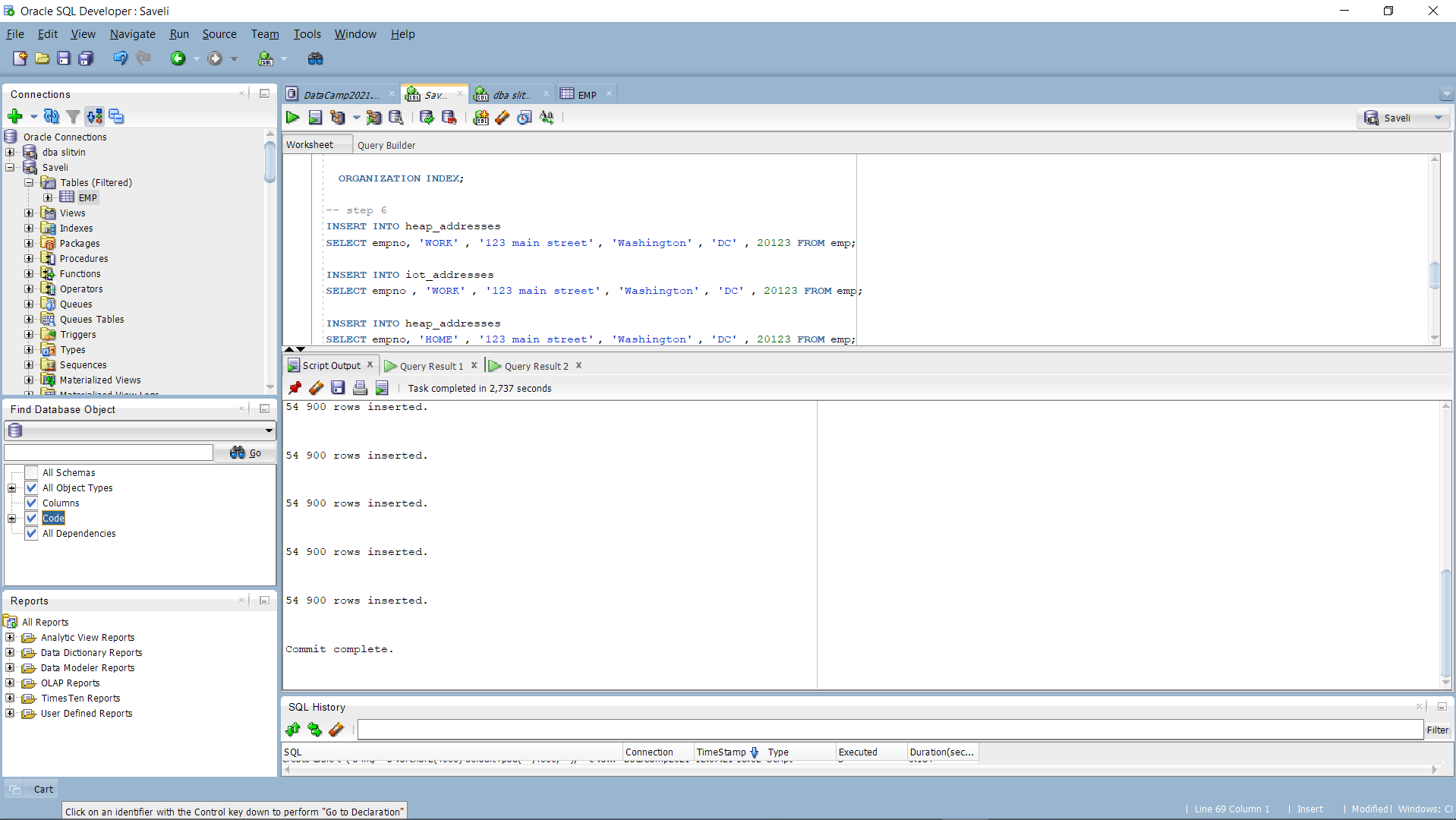
*AUTOEXTEND ON*

*MAXSIZE UNLIMITED;*

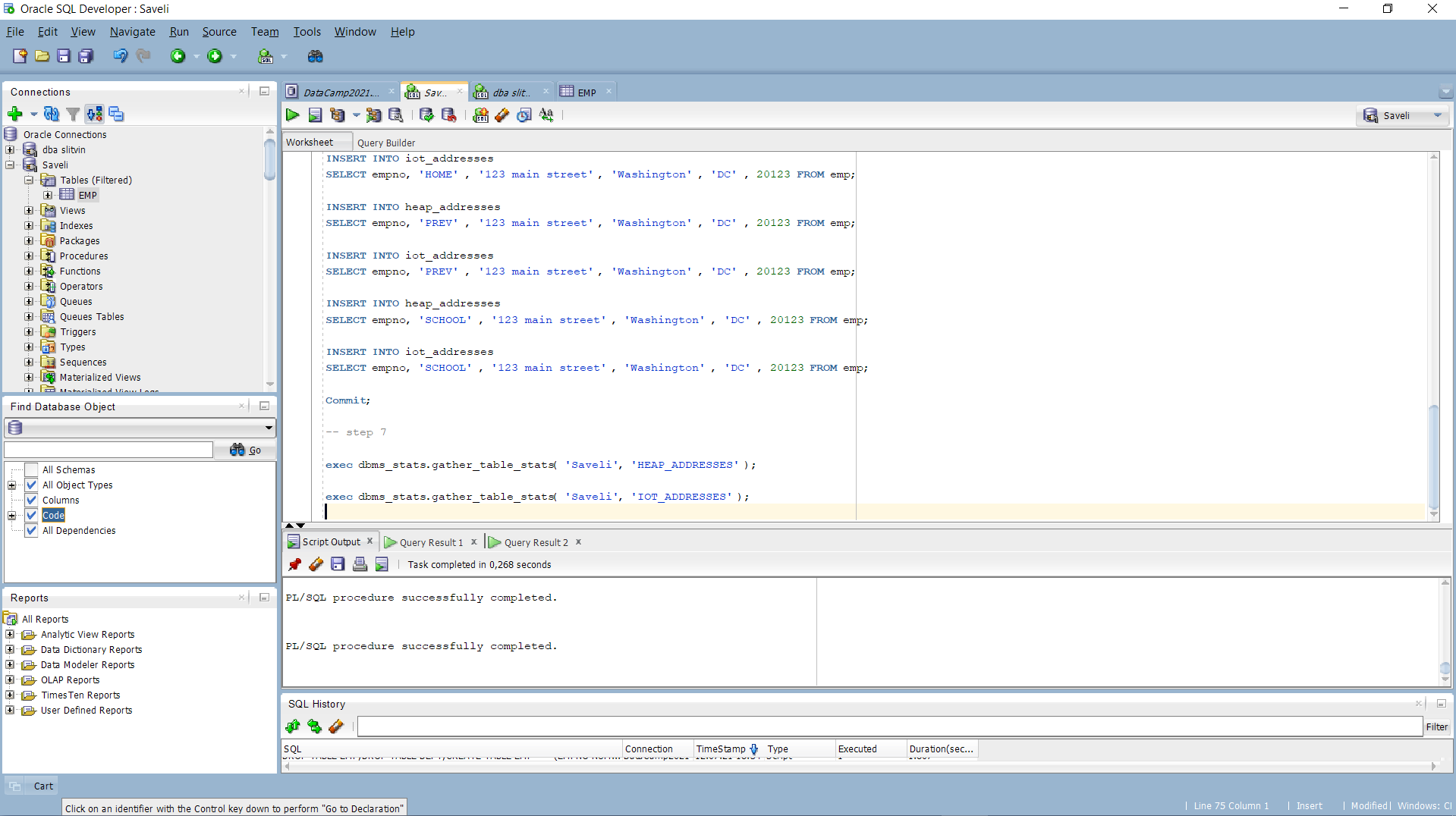
The command increased maximum size of tablespace file created with upper bound of 100mb to unlimited size.



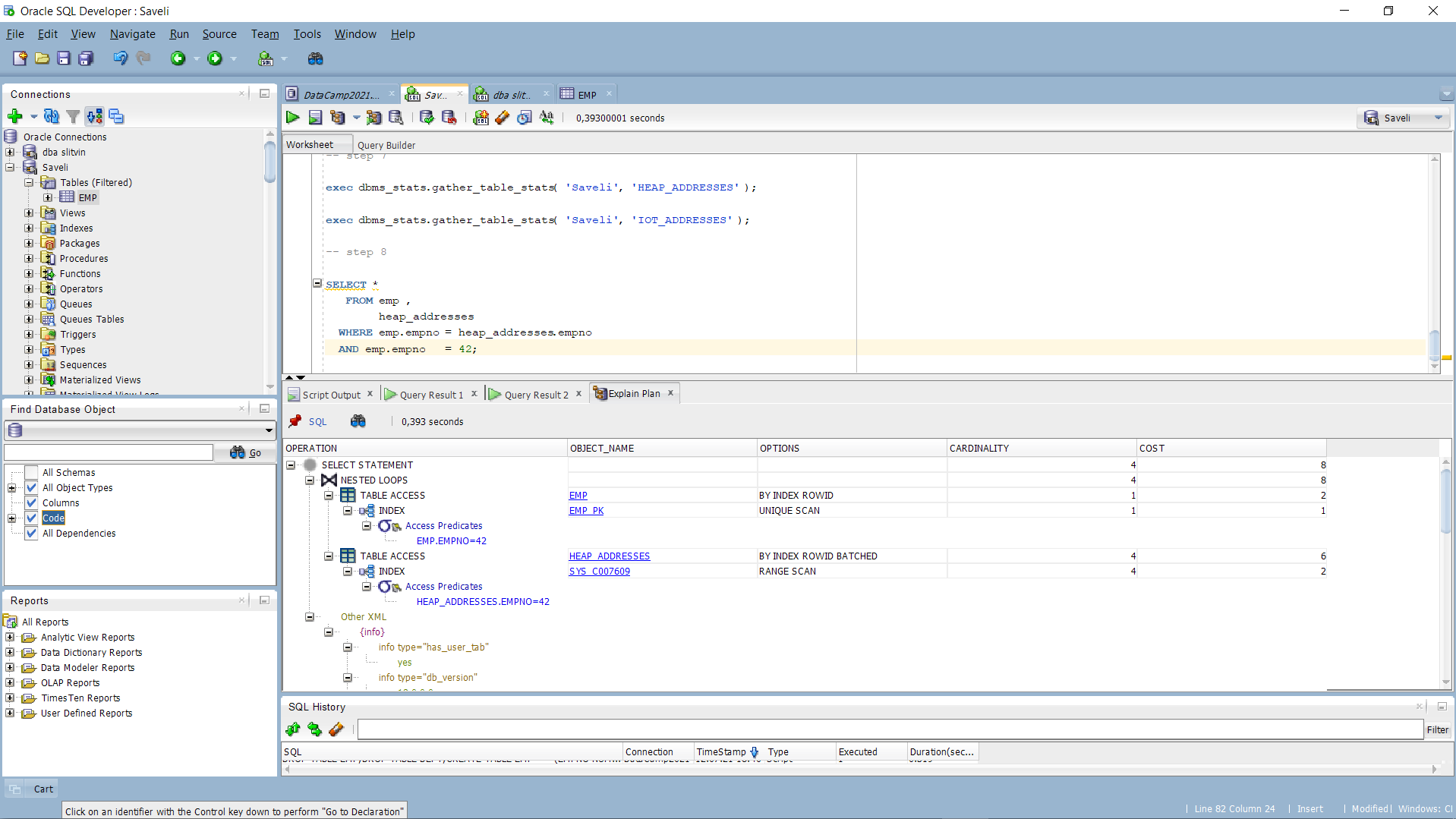
Tables heap\_adresses and iot\_addresses were created without issues.



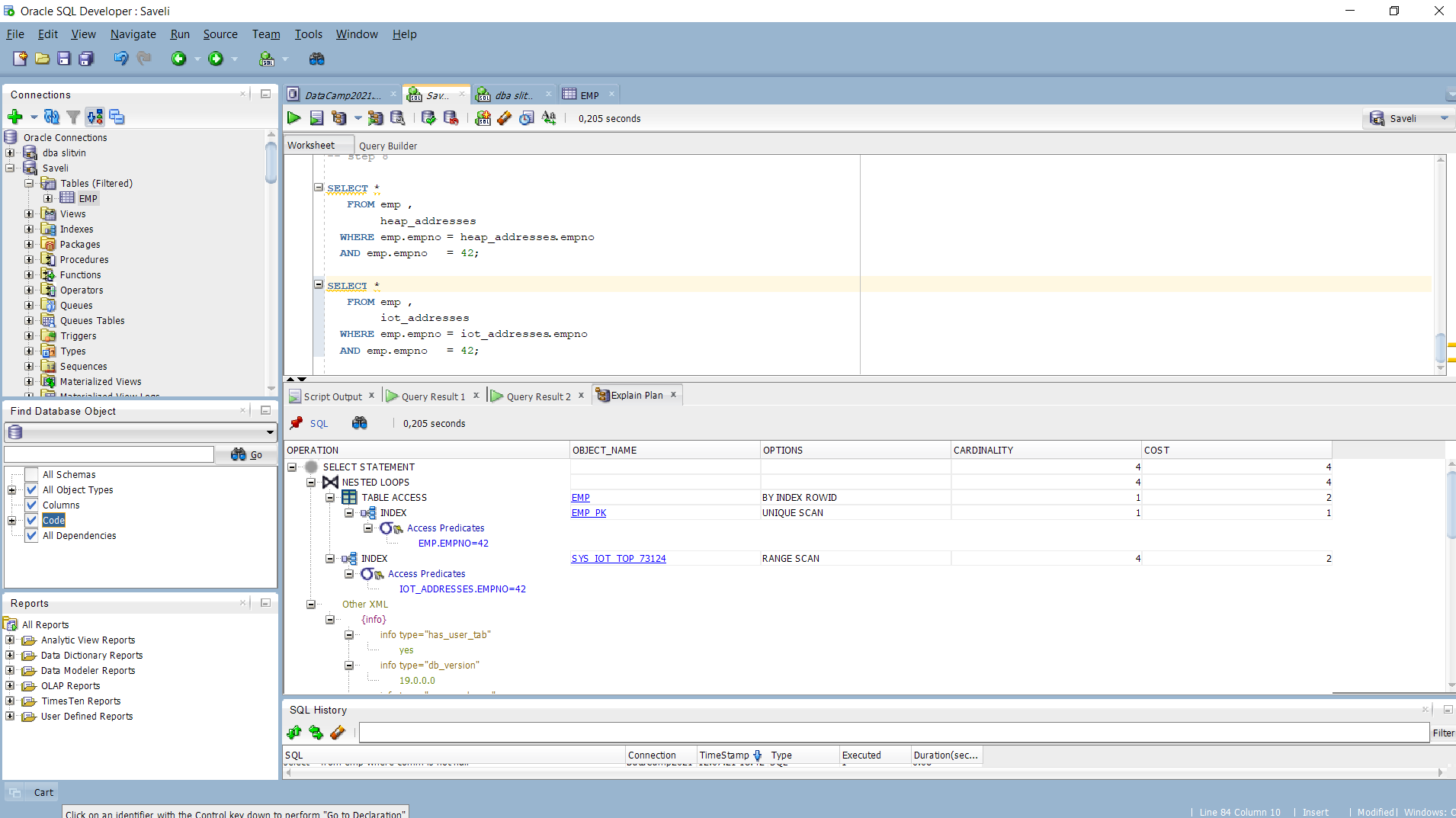
Insert was completely done



Stats is calculated



Explain Plan for heap\_adresses table



Explain plan for iot\_adresses table

Well, we can attempt Attention at Cost-Based-Oriented(COST) column, from which we can assume that iot table is much more faster and cheaper for our sql querry.

It’s because we’ve used index organization table type. The sorting goes trough the primary key, the meaning of accessfull is much more faster.

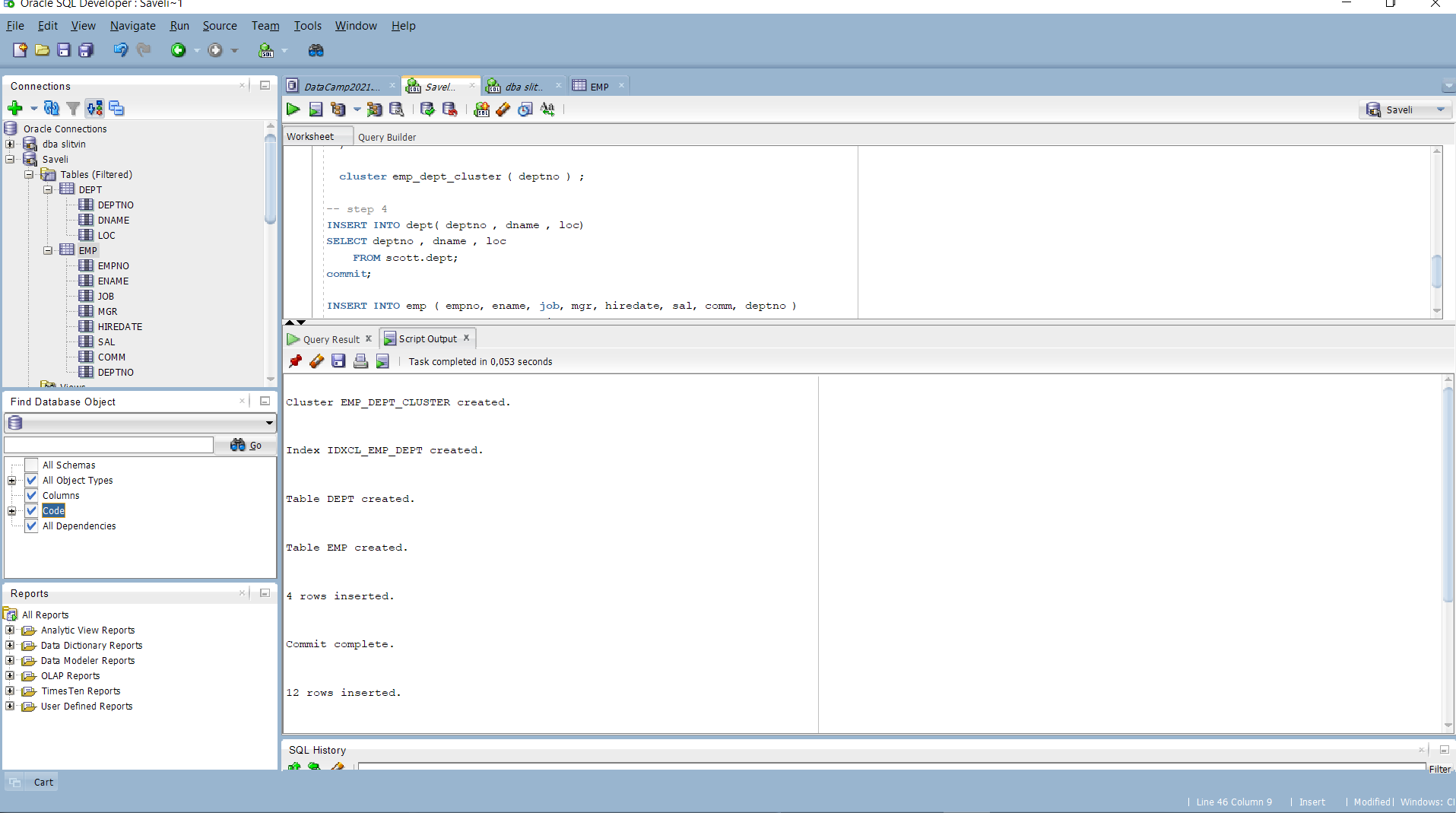
At this way using an index organization could decrease cost and time to access data inside table and the main plus is that, cause of index and the table are created in the same segment, less storage is needed.

Final step is to drop tables we’ve created before.

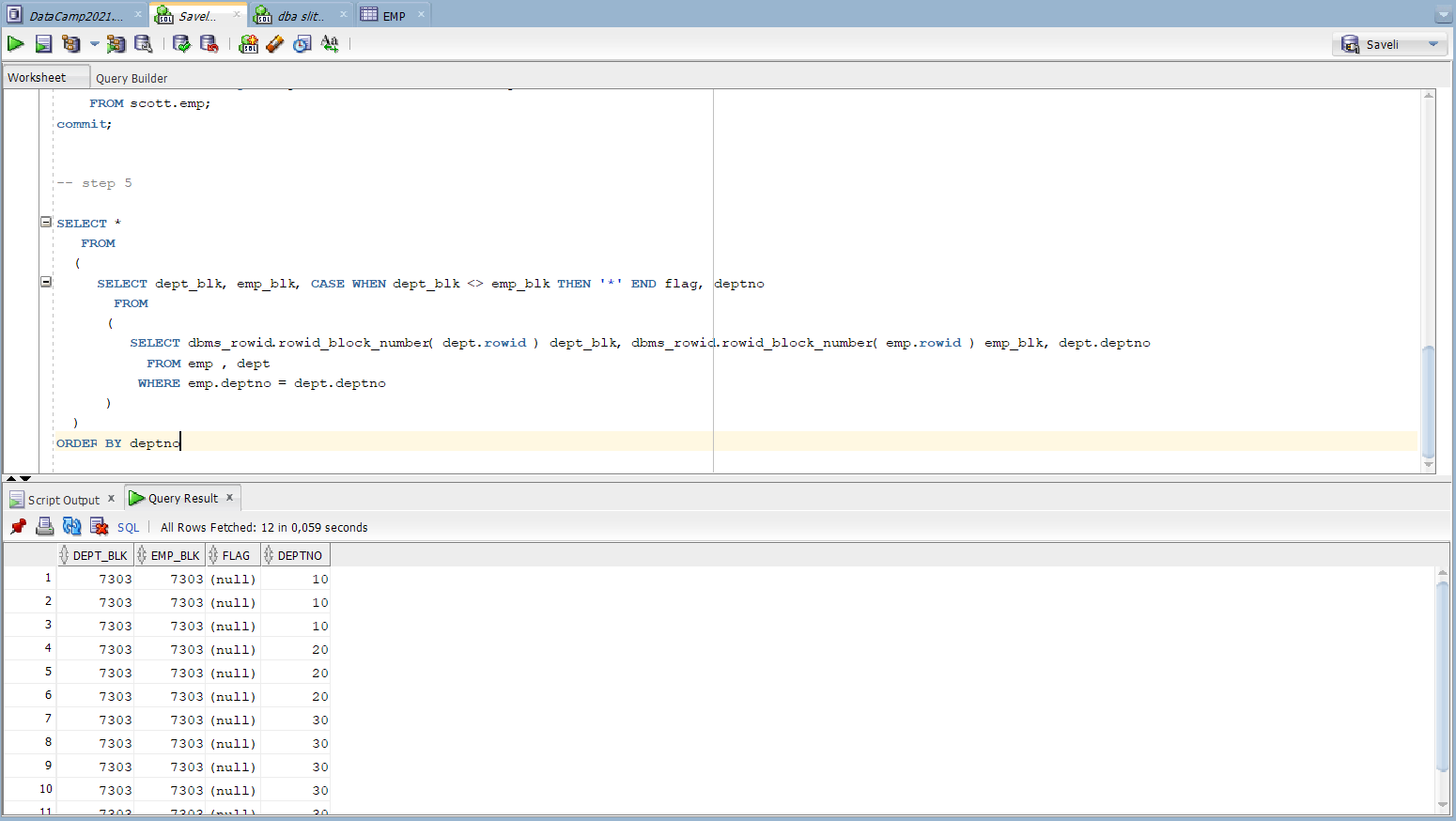
Funny, that ‘emp’ table could be dropped only when all other tables with unique or primary keys referenced by foreign keys in another table will be dropped. But it truly fair.

# 4. Index Clustered Tables

## Task 4: Analyses Cluster Storage by Blocks



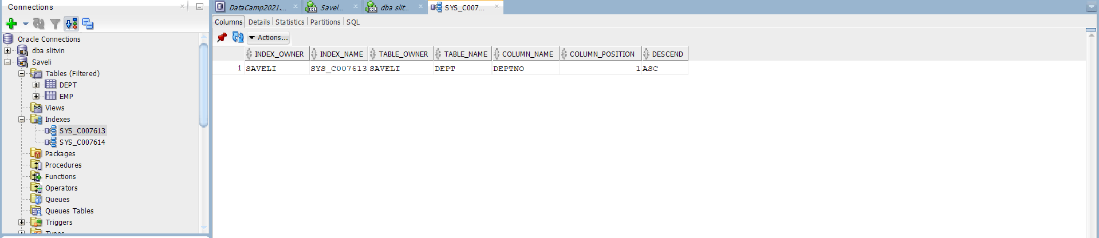
Steps 1-4



All the data stored in the same block with only dept increasing. The meaning of index clustered tables is that they store values with using cluster as sort helpful tool that sort values in ascendig order. That means no matter how you describe your inputs for example (insert into tablename values( (7, true) , (2, false) ...and so on till row 12) – the value with index 7 will be stored in 7th row.

At this way clustered index table is much more faster than index tables, therefore they do not consume extra storage.

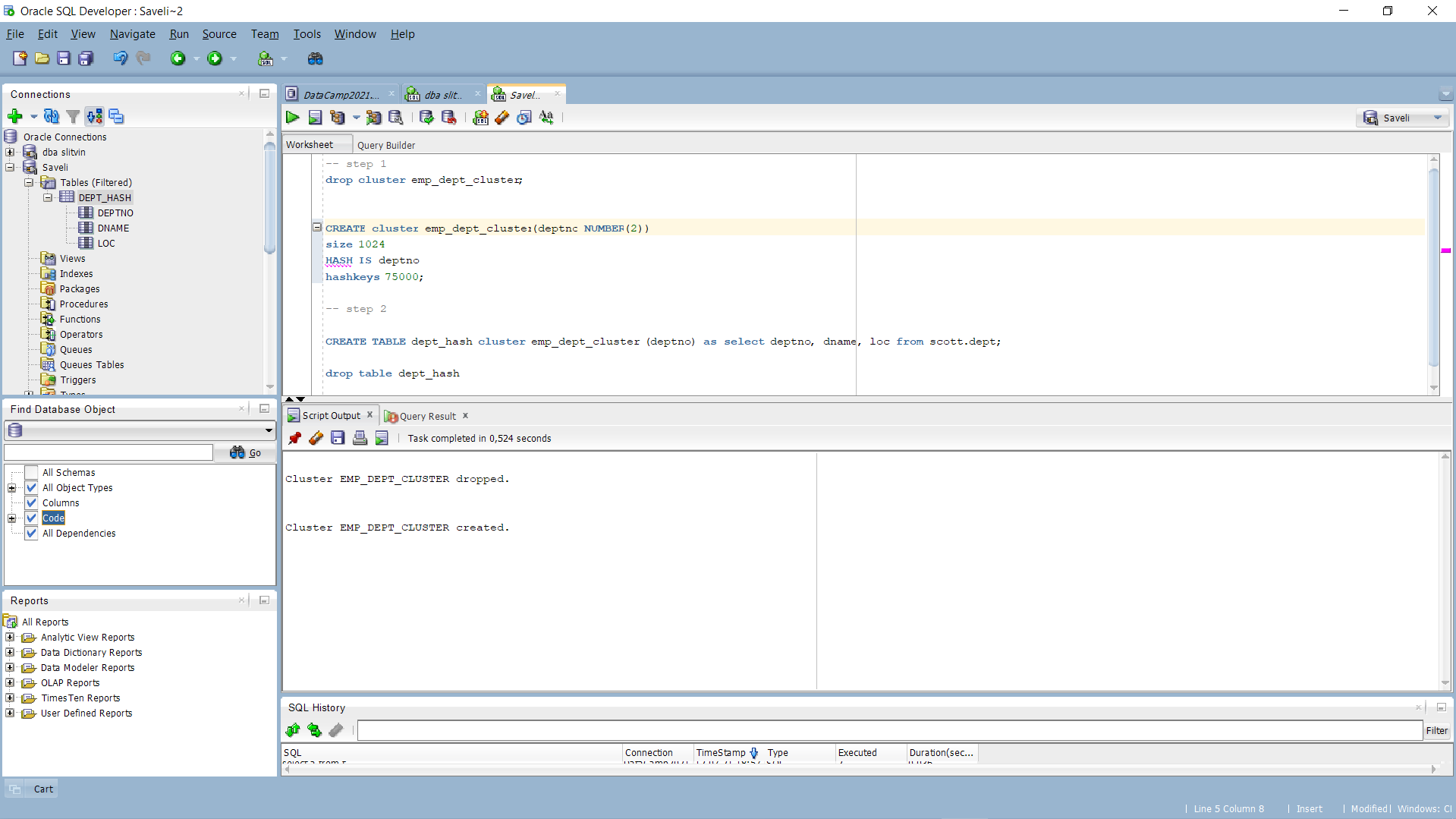
Also clustered index creates new index for each cluster index tables that contains in index folder



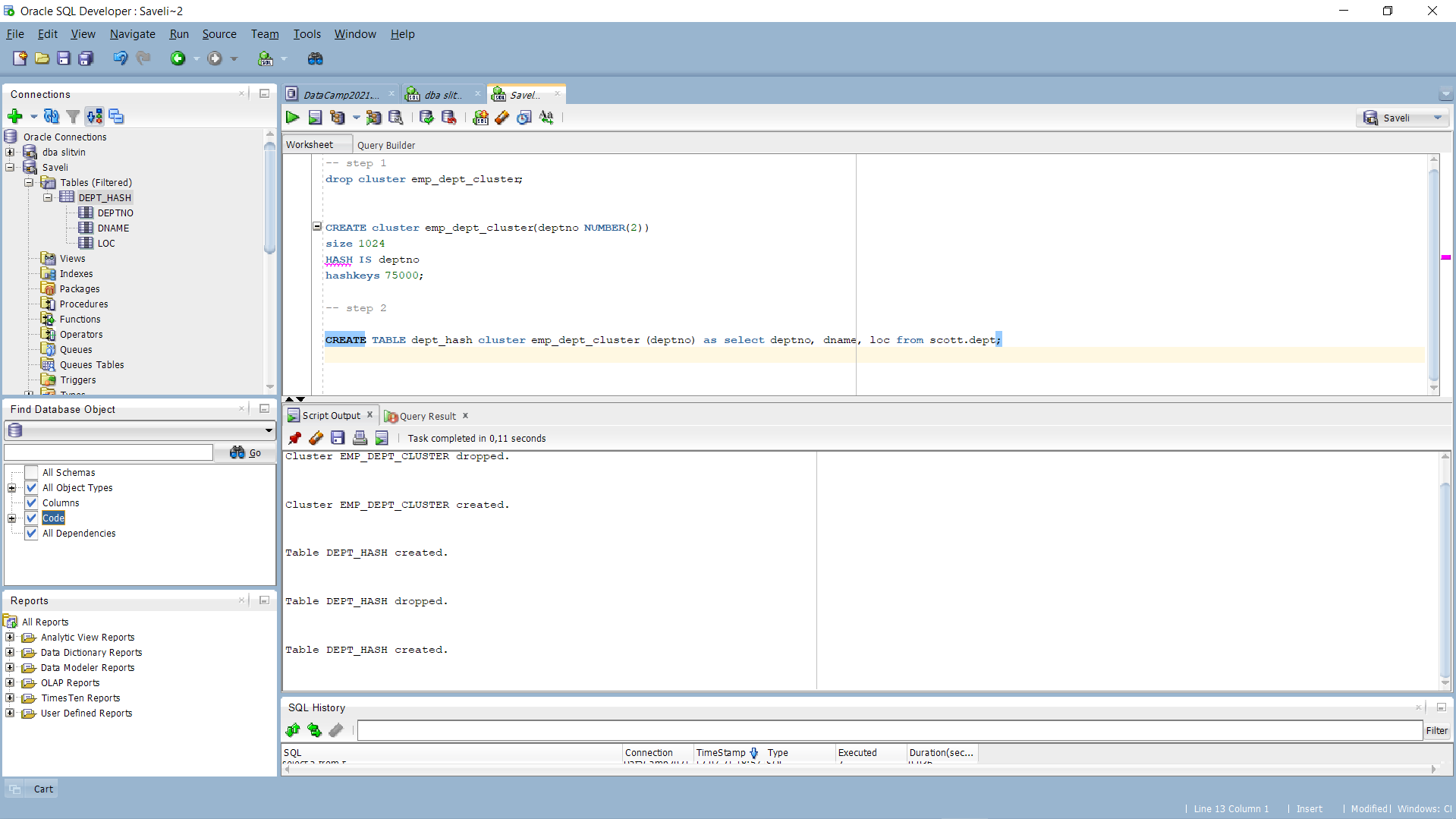
Back to our tables, despite awaited 0 values in output, there’s 12 values that each tables stores In different blocks (unique blocks). We can consume that storage capacity was not enough, maybe recreation with much more value can fix this and we will get expected results. The cluster index table is good opportunity to create different tables and get up speed access to them both. But to do not get wrong results - via the setting steps needed to consciously approach the issue of memory allocation. Otherwise it will make data base and operations slowly to executing.

# 5. Hash Clustered Tables

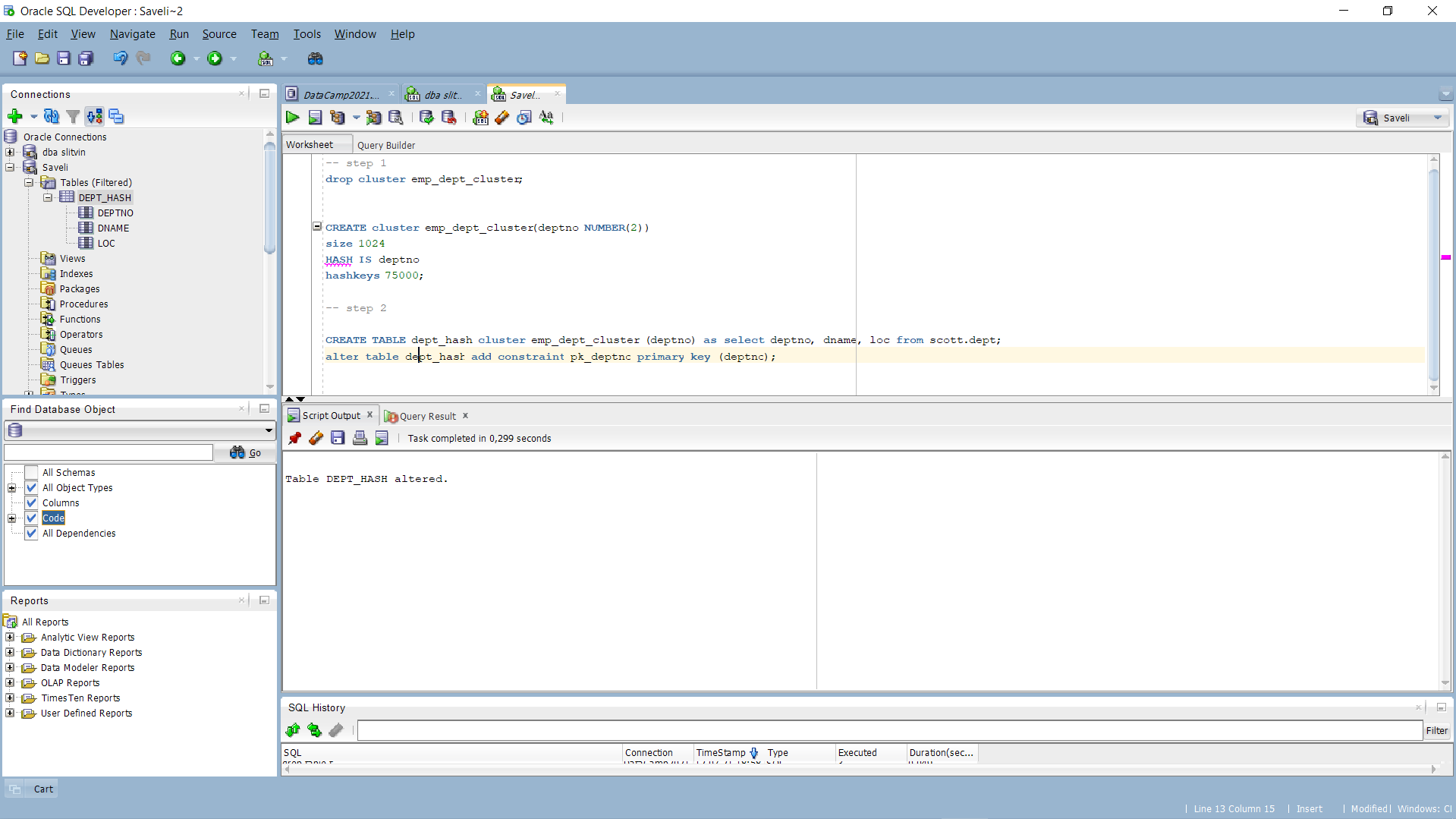
## Task 5: Analyses Cluster Storage by Blocks



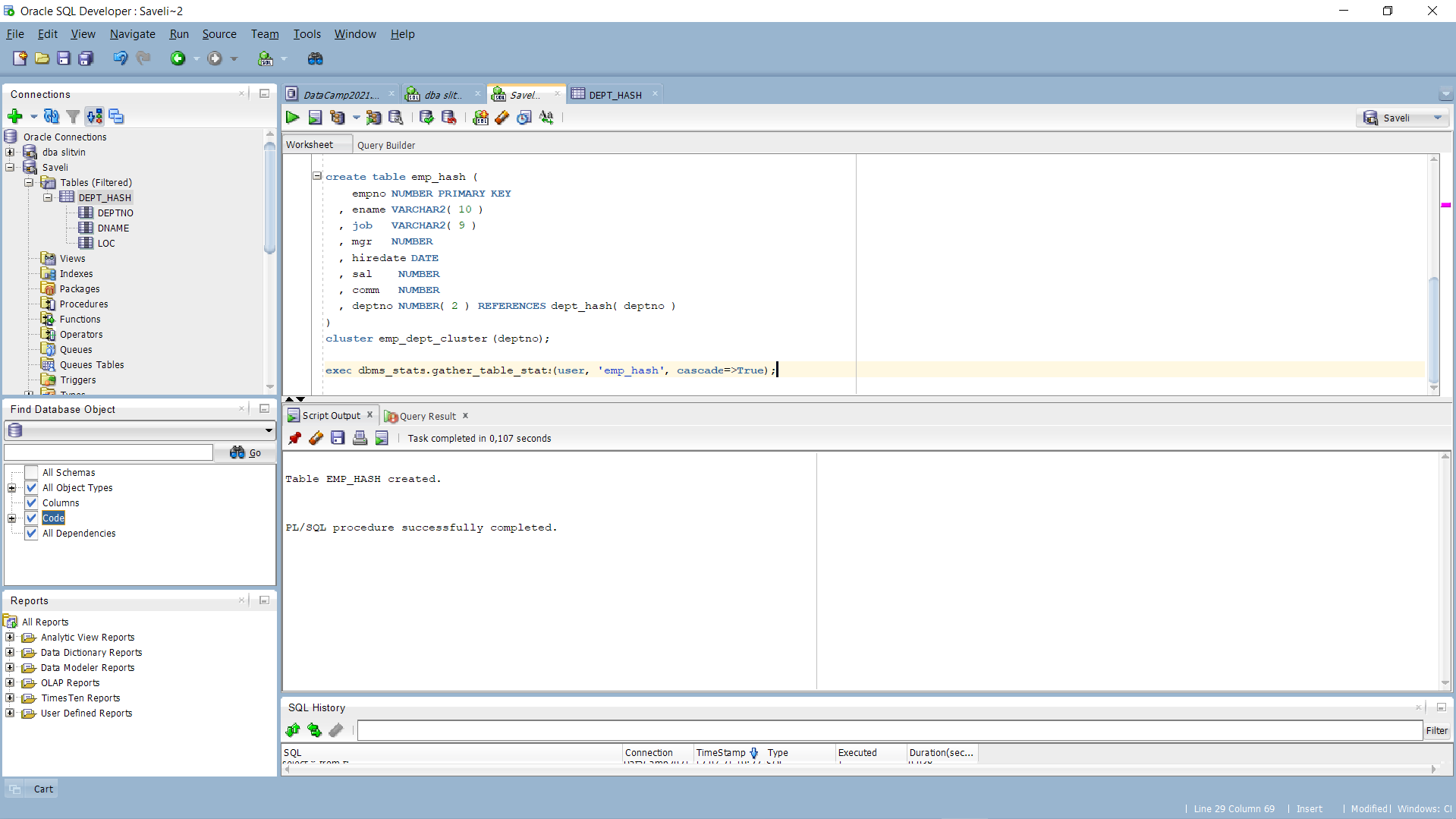
Firstly we’ve recreating cluster to use it in next steps.



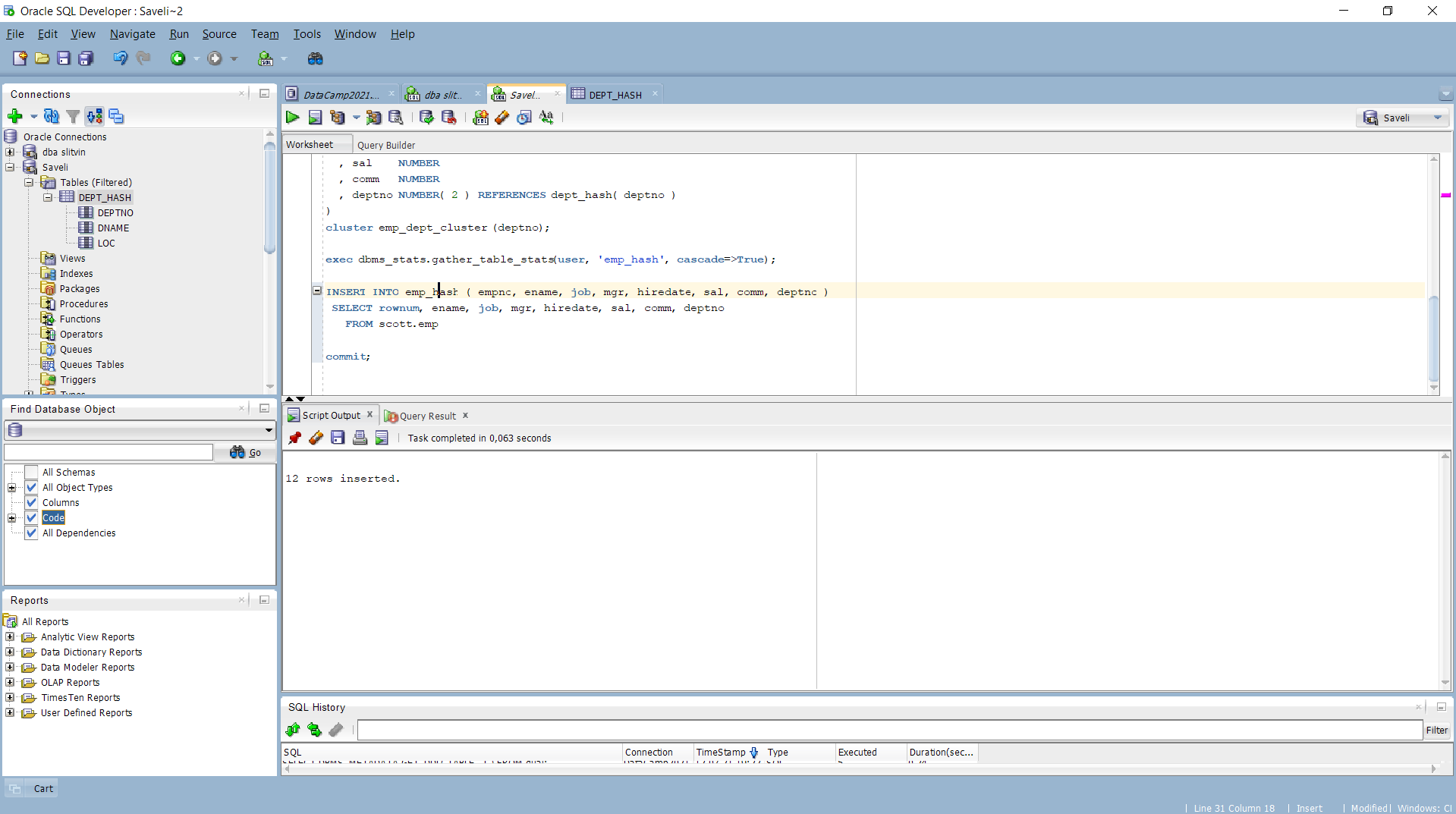
After that we creating table associated to our cluster



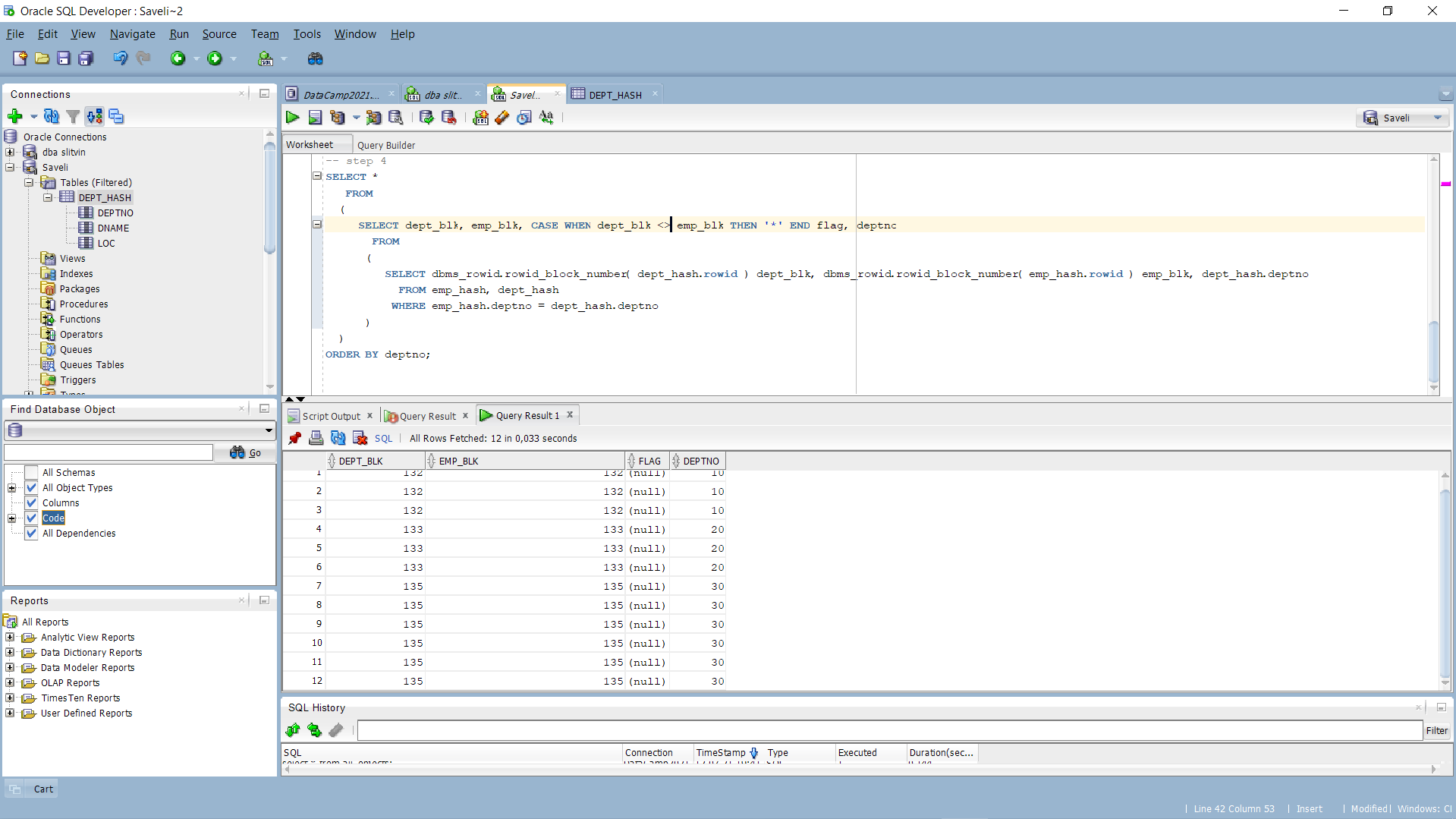
Then adding primary key to column deptno in table dept\_hash



Next step is to create emp\_hash table with using our cluster and gathering stats



The data from scott.emp is inserted



final output

* Expected All data have to be stored on the same block.
* Prepare screenshots and write explanation why data storage look like on select.
* Describe advantages of this type of storage.

From the output we can assume that hash tables is much more complex than index tables. Data being stored at the different blocks. Also we can see correlation between blocks and deptno -> each new block increased dept at 10.

The advantages:

* The cluster key is hashed to the address of the block where the data should be located.
* You know with a good degree of accuracy how many rows the table will have over its life, or you have some reasonable upper bound. Getting the size of the HASHKEYs and SIZE parameters right is crucial to avoid a rebuild.
* DML, especially inserts, is light with respect to retrieval. This means you have to balance optimizing data retrieval with new data creation. Light inserts might be 100,000 per unit of time for one person and 100 per unit of time for another—all depending on their data retrieval patterns. Updates do not introduce significant overhead, unless you update the HASHKEY, which would not be a good idea as it would cause the row to migrate.

• You access the data by the HASHKEY value constantly. For example, say you have a table of parts, and these parts are accessed by part number. Lookup tables are especially appropriate for hash clusters.