**Лабораторная работа №3**

**Невейков Андрей, 2022**

**/\*Prerequisites\*/**

create tablespace tbs\_lab datafile 'db\_lab\_001.dat'

size 5M autoextend ON next 5M MAXSIZE 100M;

create user scott identified by 123456;

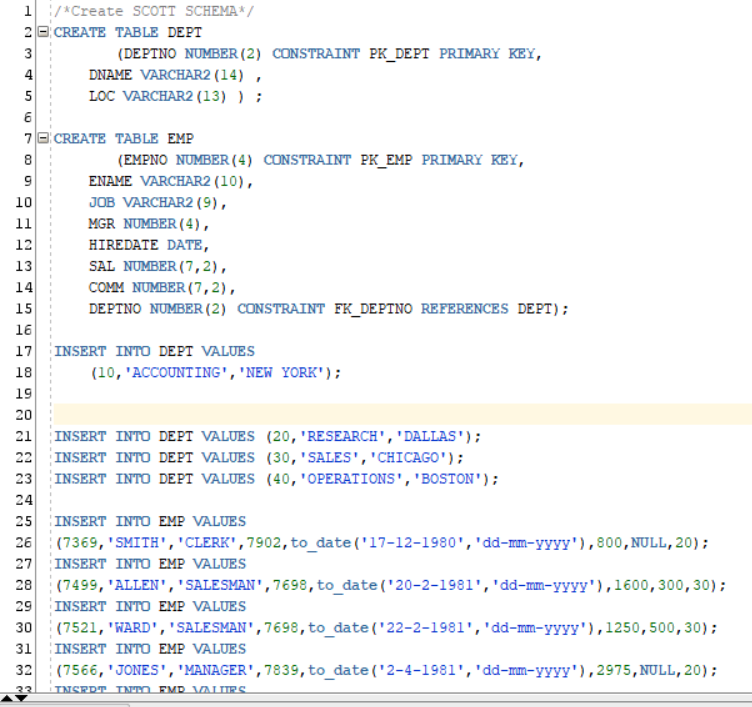
grant connect to scott;

grant resource to scott;

GRANT UNLIMITED TABLESPACE TO scott;

grant select on scott.dept to ANeveykov;

grant select on scott.emp to ANeveykov;



**SUMMARY “Prerequisites”**

Пользователей создает администратор. Чтобы пользователь мог вносить информацию в базу данных, администратор должен сначала наделить его правами. Чтобы зайти за пользователя, надо создать отдельное подключение.

**/\*Task\_1\*/**

-- Step 1

create table t

( a int,

b varchar2(4000) default rpad('\*',4000,'\*'),

c varchar2(4000) default rpad('\*',4000,'\*')

)

/\*

RPAD вставка с дозаполнение справа.

Общий формат этой функции:

RPAD(string, length\_to\_make\_string, what\_to\_add\_to\_right\_of\_string)

SQL> SELECT RPAD( 'Буквы' , 20, '.' ) ОТ двойного;

RPAD( 'БУКВЫ' ,20,'.')

--------------------

Буквы.............

\*/

-- Step 2

insert into t (a) values ( 1);

insert into t (a) values ( 2);

insert into t (a) values ( 3);

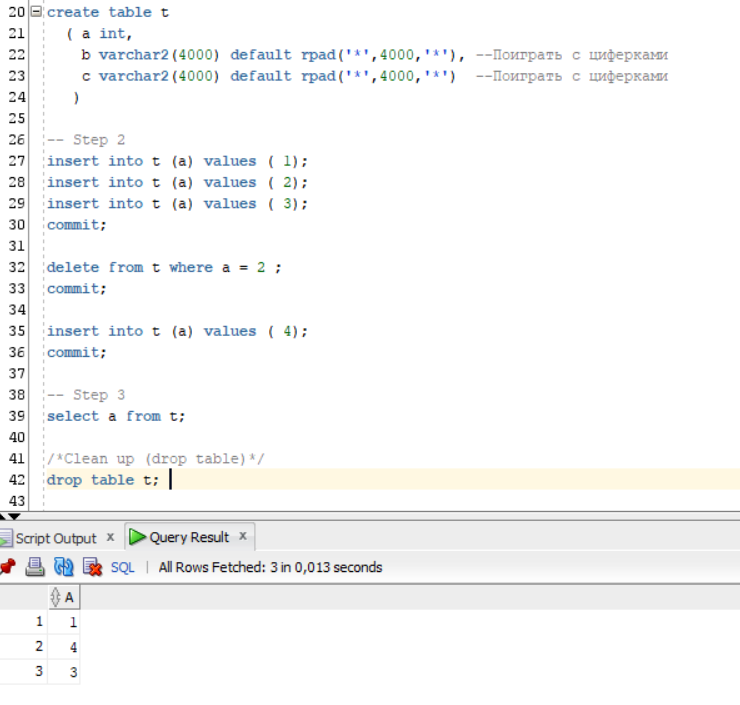
commit;

delete from t where a = 2 ;

commit;

insert into t (a) values ( 4);

commit;

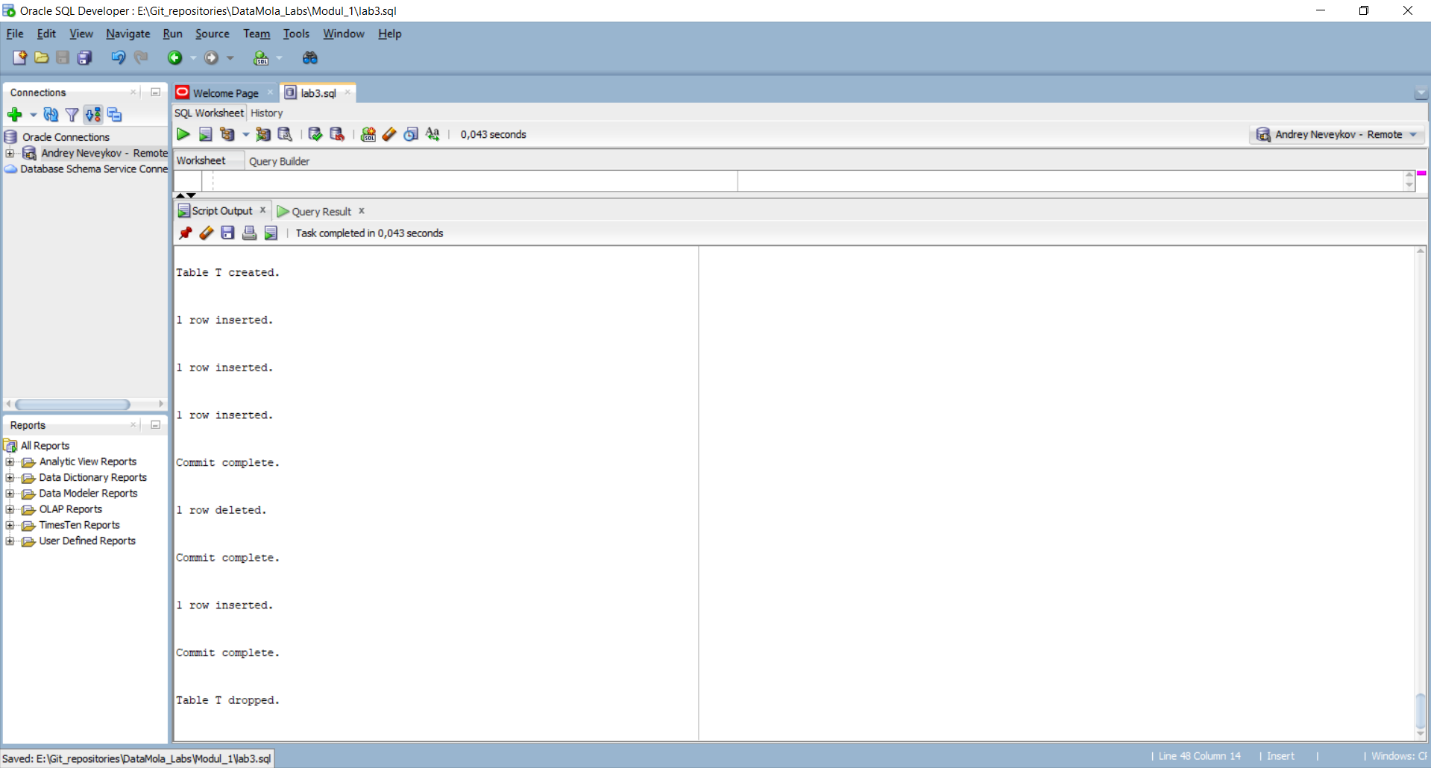


-- Step 3

select a from t;

/\*Clean up (drop table)\*/

drop table t;



**SUMMARY “TASK\_1”**

1. **Я научился** создавать Heap organized таблицы, использовать функцию

«RPAD» (вставка с дозаполнением справа), изменять максимальный размер

колонки, содержащей строку.

1. **Я узнал**, что в Heap organized таблицах, Oracle старается максимально эффективно использовать память и поэтому последнее вставленное значение может оказаться в середине «кучи» и порядок вывода значений изменится.
2. **Я понял** принцип использования памяти в Oracle, особенности использования и область применения (для данных, которые должны занимать минимум места и для которых редко выполняется поиск по условиям) Heap organized таблиц.

**/\*Task\_2\*/**

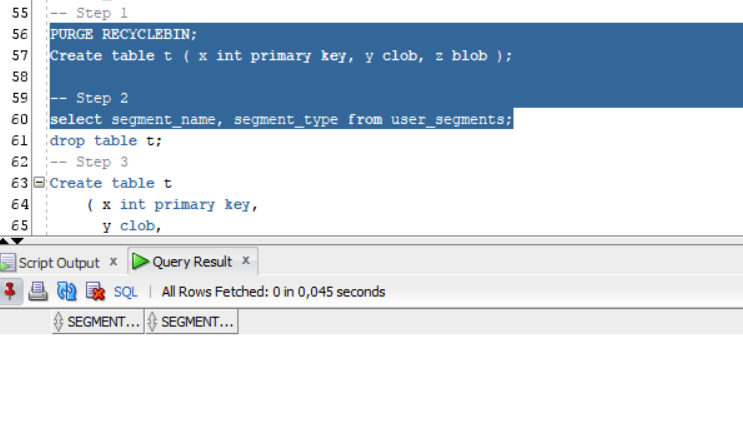
-- Step 1

PURGE RECYCLEBIN;

Create table t ( x int primary key, y clob, z blob )

-- Step 2

select segment\_name, segment\_type from user\_segments;



drop table t;

-- Step 3

Create table t

( x int primary key,

y clob,

z blob )

SEGMENT CREATION IMMEDIATE

-- Step 4

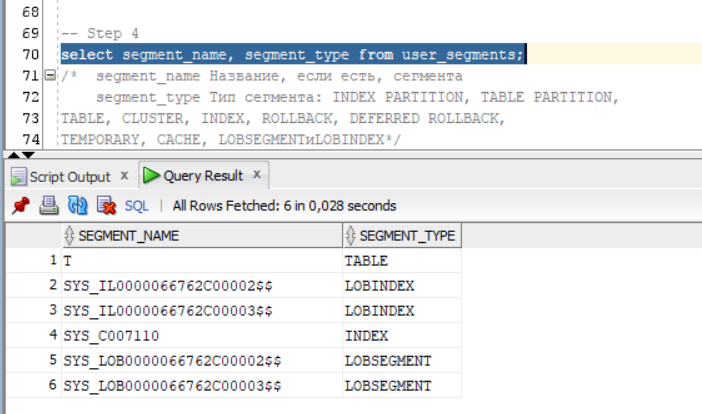
select segment\_name, segment\_type from user\_segments;

/\* segment\_name Название, если есть, сегмента

segment\_type Тип сегмента: INDEX PARTITION, TABLE PARTITION,

TABLE, CLUSTER, INDEX, ROLLBACK, DEFERRED ROLLBACK,

TEMPORARY, CACHE, LOBSEGMENTиLOBINDEX\*/



-- Step 5

SELECT DBMS\_METADATA.GET\_DDL('TABLE','T') FROM dual

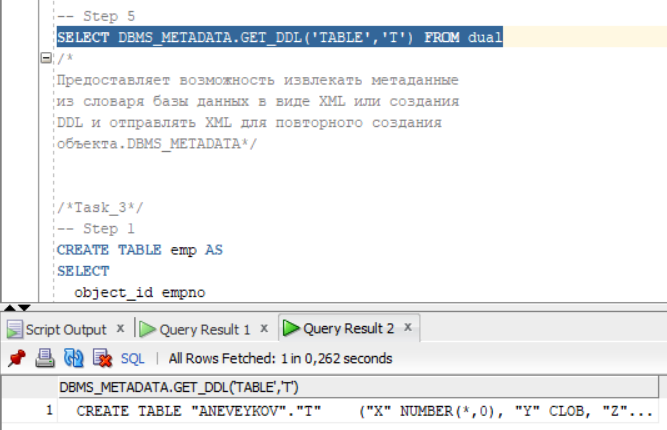
/\*

Предоставляет возможность извлекать метаданные

из словаря базы данных в виде XML или создания

DDL и отправлять XML для повторного создания

объекта.DBMS\_METADATA\*/



**SUMMARY “TASK\_2”**

1. **Я научился** создавать извлекать метаданные из словаря базы данных.
2. **Я узнал** разницу между SEGMENT CREATION IMMEDIATE и DEFERRED.
3. **Я понял**, что поиск по метаданным может помочь при дебагинге.

**/\*Task\_3\*/**

-- Step 1

CREATE TABLE emp AS

SELECT

object\_id empno

, object\_name ename

, created hiredate

, owner job

FROM

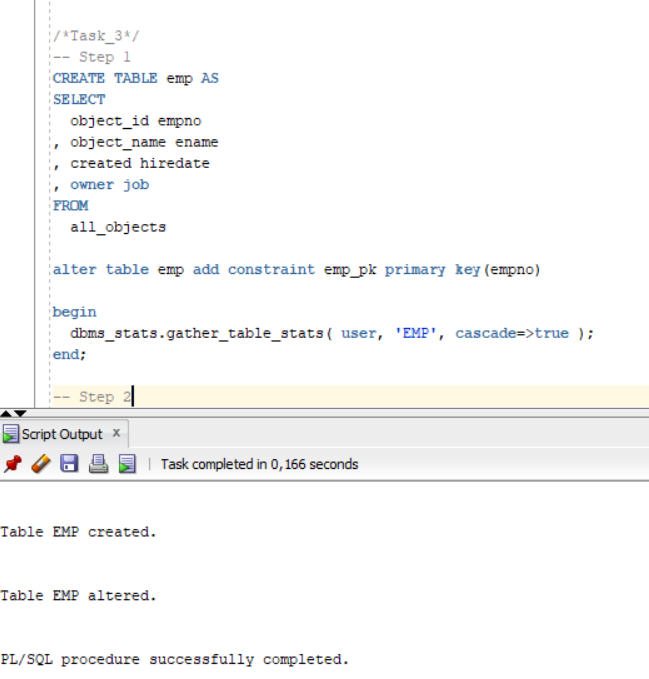
all\_objects

alter table emp add constraint emp\_pk primary key(empno)

begin

dbms\_stats.gather\_table\_stats( user, 'EMP', cascade=>true );

end;



-- Step 2

CREATE TABLE heap\_addresses

(

empno REFERENCES emp(empno) ON DELETE CASCADE

, addr\_type VARCHAR2(10)

, street VARCHAR2(20)

, city VARCHAR2(20)

, state VARCHAR2(2)

, zip NUMBER

, PRIMARY KEY (empno,addr\_type)

)

-- Step 3

CREATE TABLE iot\_addresses

(

empno REFERENCES emp(empno) ON DELETE CASCADE

, addr\_type VARCHAR2(10)

, street VARCHAR2(20)

, city VARCHAR2(20)

, state VARCHAR2(2)

, zip NUMBER

, PRIMARY KEY (empno,addr\_type)

)

ORGANIZATION INDEX

-- Step 4

INSERT INTO heap\_addresses

SELECT empno, 'WORK' , '123 main street' , 'Washington' , 'DC' , 20123 FROM emp;

INSERT INTO iot\_addresses

SELECT empno , 'WORK' , '123 main street' , 'Washington' , 'DC' , 20123 FROM emp;

--

INSERT INTO heap\_addresses

SELECT empno, 'HOME' , '123 main street' , 'Washington' , 'DC' , 20123 FROM emp;

INSERT INTO iot\_addresses

SELECT empno, 'HOME' , '123 main street' , 'Washington' , 'DC' , 20123 FROM emp;

--

INSERT INTO heap\_addresses

SELECT empno, 'PREV' , '123 main street' , 'Washington' , 'DC' , 20123 FROM emp;

INSERT INTO iot\_addresses

SELECT empno, 'PREV' , '123 main street' , 'Washington' , 'DC' , 20123 FROM emp;

--

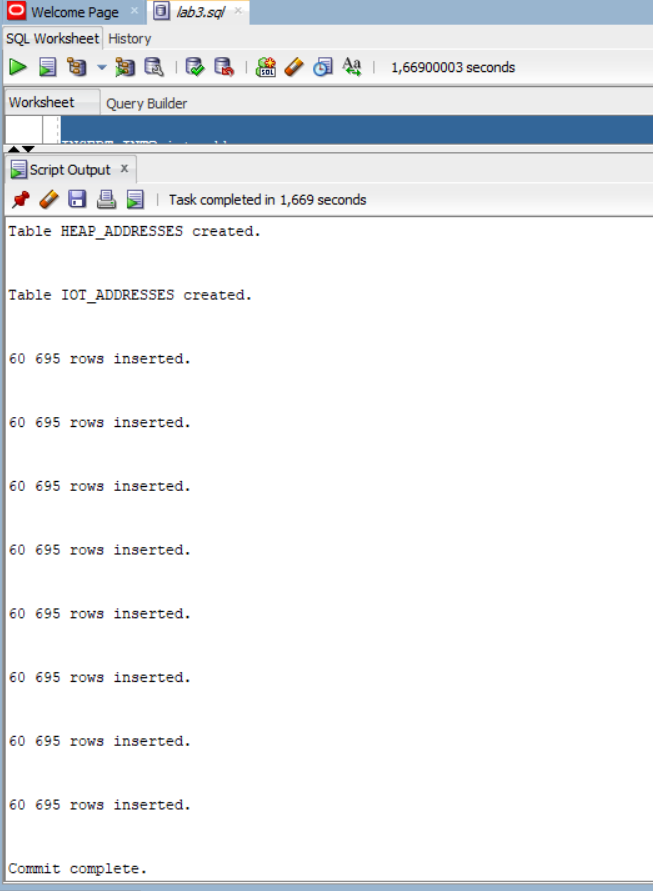
INSERT INTO heap\_addresses

SELECT empno, 'SCHOOL' , '123 main street' , 'Washington' , 'DC' , 20123 FROM emp;

INSERT INTO iot\_addresses

SELECT empno, 'SCHOOL' , '123 main street' , 'Washington' , 'DC' , 20123 FROM emp;

Commit;



-- Step 5

exec dbms\_stats.gather\_table\_stats(ANeveykov, 'HEAP\_ADDRESSES' );

exec dbms\_stats.gather\_table\_stats(ANeveykov, 'IOT\_ADDRESSES' );

-- Step 6

SET AUTOTRACE OFF

set autotrace traceonly

explain plan for

SELECT \*

FROM emp ,

heap\_addresses

WHERE emp.empno = heap\_addresses.empno

AND emp.empno = 42;

explain plan for

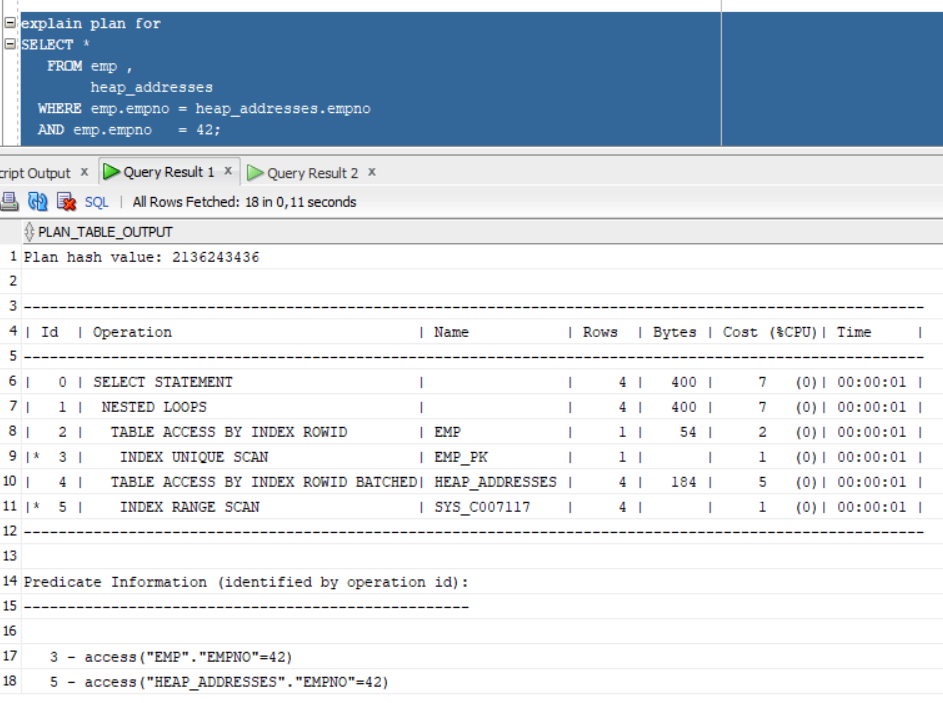
SELECT \*

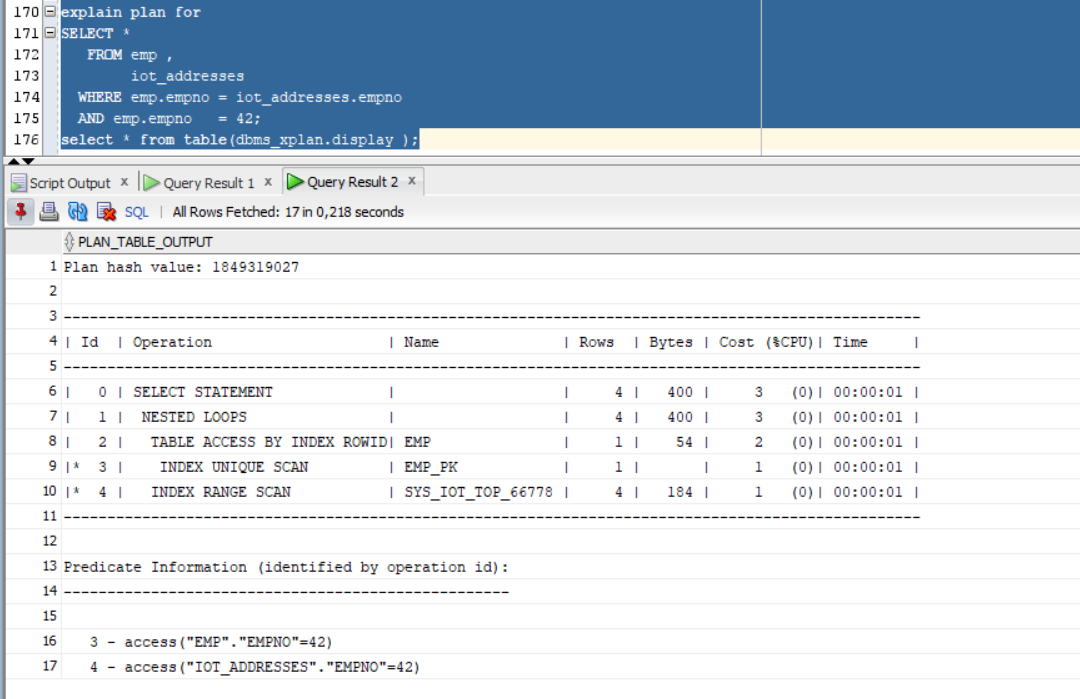
FROM emp ,

iot\_addresses

WHERE emp.empno = iot\_addresses.empno

AND emp.empno = 42;



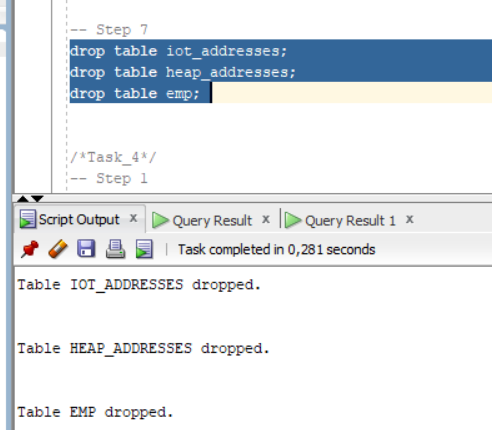


-- Step 7

drop table iot\_addresses;

drop table heap\_addresses;

drop table emp;



**SUMMARY “TASK\_3”**

1. **Я научился** создавать Index organized таблицы, выводить статистические

данные о затраченных ресурсах на выполнение запроса.

1. **Я узнал**, что в Index organized таблицах, таблицы хранятся в balance

tree (не в heap, а внутри индексов, в листике вся строка, а string в отдельных сегментах). Также Index organized таблицы используют меньше

ресурсов, при выполнении запроса т.к. данные структурируются при помощи индексов.

1. **Я понял**, что если в будущем понадобится делать различные выборки из

хранящихся данных, то лучше использовать Index organized таблицы, чем

Heap organized таблицы. На больших объемах данных разница будет еще

более заметна т.к. если oracle использует весь RAM, то будет хранить

временные файлы на диске, а скорость его чтения меньше.

**/\*Task\_4\*/**

-- Step 1

CREATE cluster emp\_dept\_cluster( deptno NUMBER( 2 ) )

SIZE 1024

STORAGE( INITIAL 100K NEXT 50K );

-- Step 2

CREATE INDEX idxcl\_emp\_dept on cluster emp\_dept\_cluster;

-- Step 3

CREATE TABLE dept

(

deptno NUMBER( 2 ) PRIMARY KEY

, dname VARCHAR2( 14 )

, loc VARCHAR2( 13 )

)

cluster emp\_dept\_cluster ( deptno ) ;

CREATE TABLE emp

(

empno NUMBER PRIMARY KEY

, ename VARCHAR2( 10 )

, job VARCHAR2( 9 )

, mgr NUMBER

, hiredate DATE

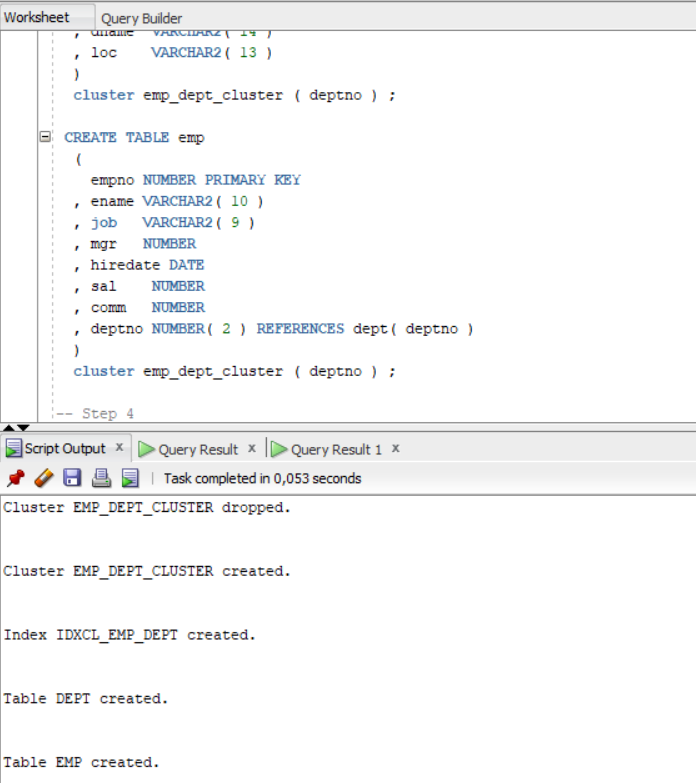
, sal NUMBER

, comm NUMBER

, deptno NUMBER( 2 ) REFERENCES dept( deptno )

)

cluster emp\_dept\_cluster ( deptno ) ;



-- Step 4

INSERT INTO dept( deptno , dname , loc)

SELECT deptno , dname , loc

FROM scott.dept;

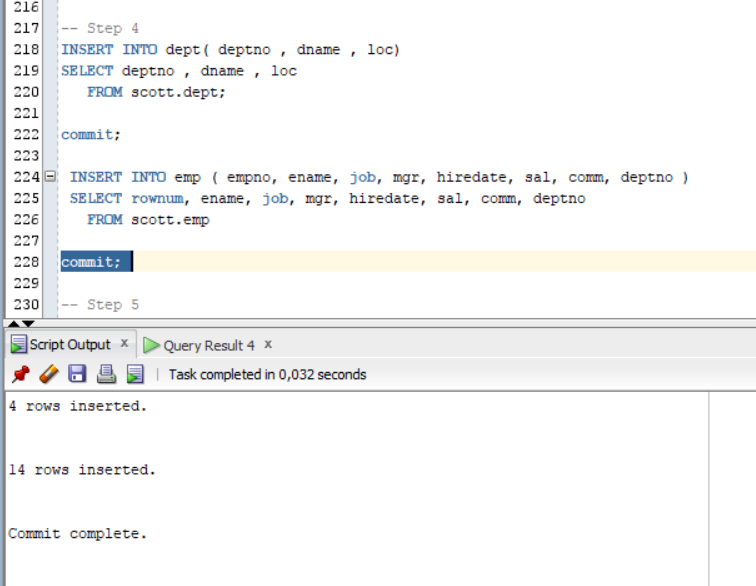
commit;

INSERT INTO emp ( empno, ename, job, mgr, hiredate, sal, comm, deptno )

SELECT rownum, ename, job, mgr, hiredate, sal, comm, deptno

FROM scott.emp

commit;



-- Step 5

SELECT \*

FROM

(

SELECT dept\_blk, emp\_blk, CASE WHEN dept\_blk <> emp\_blk THEN '\*' END flag, deptno

FROM

(

SELECT dbms\_rowid.rowid\_block\_number( dept.rowid ) dept\_blk, dbms\_rowid.rowid\_block\_number( emp.rowid ) emp\_blk, dept.deptno

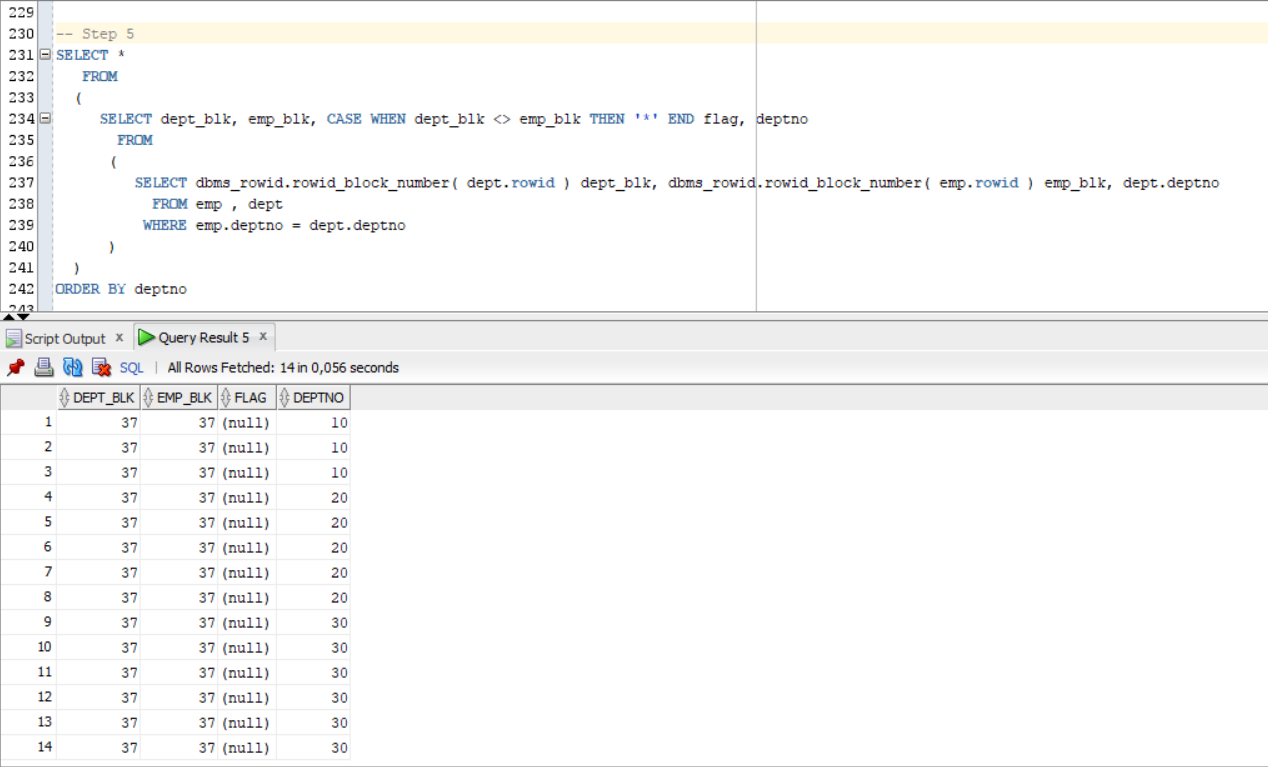
FROM emp , dept

WHERE emp.deptno = dept.deptno

)

)

ORDER BY deptno

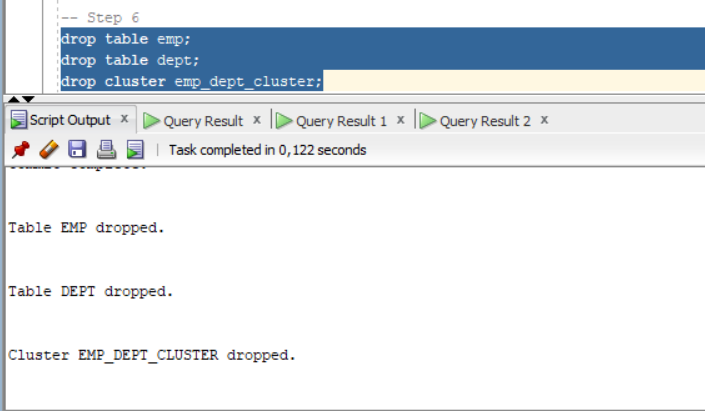


-- Step 6

drop table emp;

drop table dept;

drop cluster emp\_dept\_cluster;



**SUMMARY “TASK\_4”**

1. **Я научился** анализировать кластерное хранилище по блокам.
2. **Я узнал**, что суть подхода состоит в том, чтобы разрезать данные на сегменты, где каждый сегмент уникальный dept.id для быстрого поиска

соответствующих данных между таблицами т.к. данные находятся в одном

блоке;

1. **Я понял**, Index Clustered таблицы надо использовать, только если таблицы соединены ВСЕГДА!

**/\*Task\_5\*/**

-- Step 1

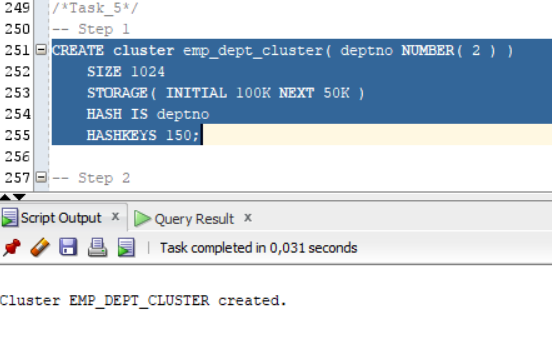
CREATE cluster emp\_dept\_cluster( deptno NUMBER( 2 ) )

SIZE 1024

STORAGE( INITIAL 100K NEXT 50K )

HASH IS deptno

HASHKEYS 150;



-- Step 2

-- Невозможно создать индекс в хэш-кластере

--CREATE INDEX idxcl\_emp\_dept on cluster emp\_dept\_cluster;

-- Step 3

CREATE TABLE dept

(

deptno NUMBER( 2 ) PRIMARY KEY

, dname VARCHAR2( 14 )

, loc VARCHAR2( 13 )

)

cluster emp\_dept\_cluster ( deptno ) ;

CREATE TABLE emp

(

empno NUMBER PRIMARY KEY

, ename VARCHAR2( 10 )

, job VARCHAR2( 9 )

, mgr NUMBER

, hiredate DATE

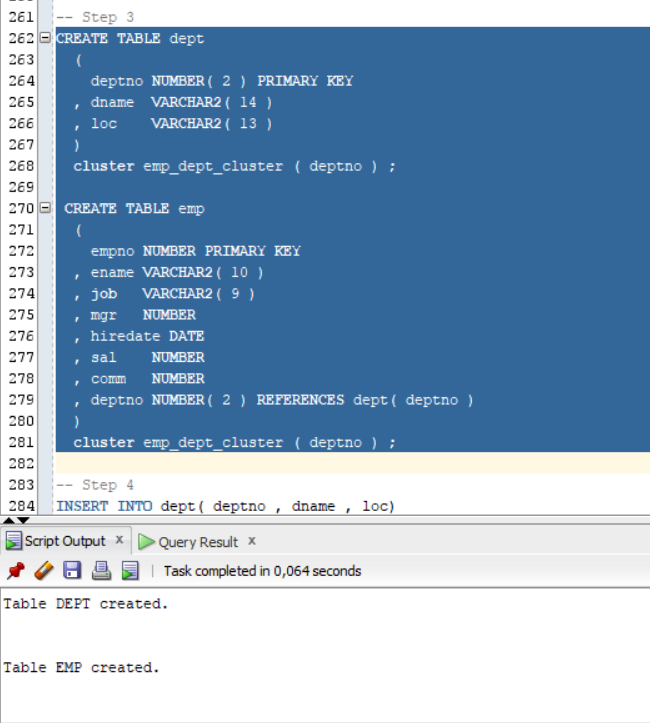
, sal NUMBER

, comm NUMBER

, deptno NUMBER( 2 ) REFERENCES dept( deptno )

)

cluster emp\_dept\_cluster ( deptno ) ;



-- Step 4

INSERT INTO dept( deptno , dname , loc)

SELECT deptno , dname , loc

FROM scott.dept;

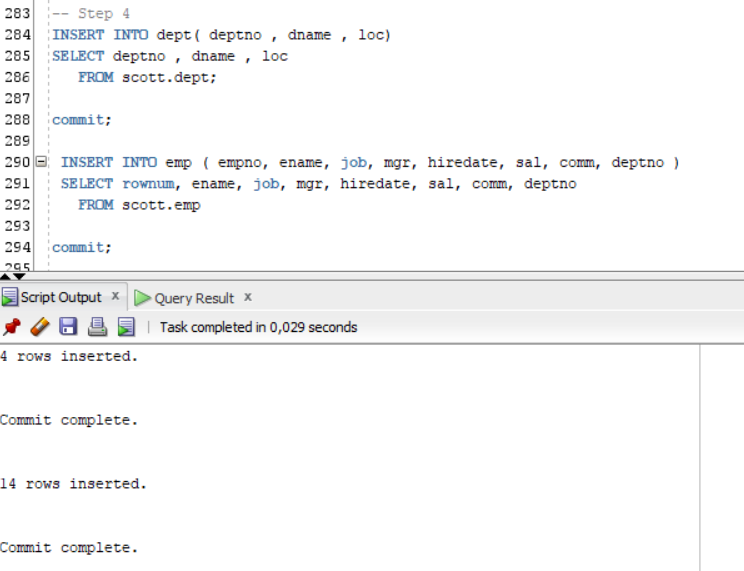
commit;

INSERT INTO emp ( empno, ename, job, mgr, hiredate, sal, comm, deptno )

SELECT rownum, ename, job, mgr, hiredate, sal, comm, deptno

FROM scott.emp

commit;



-- Step 5

SELECT \*

FROM

(

SELECT dept\_blk, emp\_blk, CASE WHEN dept\_blk <> emp\_blk THEN '\*' END flag, deptno

FROM

(

SELECT dbms\_rowid.rowid\_block\_number( dept.rowid ) dept\_blk, dbms\_rowid.rowid\_block\_number( emp.rowid ) emp\_blk, dept.deptno

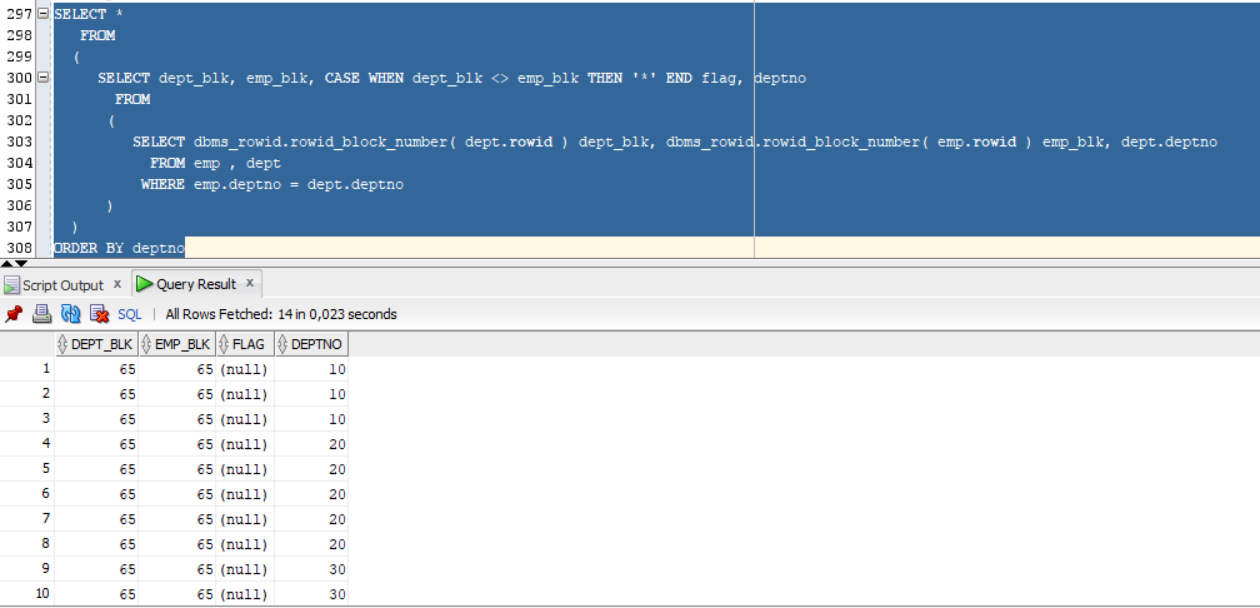
FROM emp , dept

WHERE emp.deptno = dept.deptno

)

)

ORDER BY deptno



-- Step 6

drop table emp;

drop table dept;

drop cluster emp\_dept\_cluster;



**SUMMARY “TASK\_5”**

1. **Я научился** использовать Hash Clustered таблицы для анализа кластерного хранилища по блокам.
2. **Я узнал**, что суть подхода аналогична предыдущему заданию – обеспечить быстрый поиск соответствующих данных между связанными таблицами т.к.

данные находятся в одном блоке;

1. **Я понял**, Hash Clustered таблицы надо использовать, только если таблицы соединены ВСЕГДА!

**FINAL SUMMARY**

Данная лабораторная работа …

1. … показывает положительные и отрицательные стороны различных типов таблиц и учит их использовать с учетом особенностей хранимых данных.
2. … наглядно показывает способы сегментации данных и взаимодействие сегментированных данных с Oracle DB.
3. … учит использовать метаданные, получать статистику, для принятия решения об оптимальном способе хранения данных.
4. … позволяет проработать создание пользователей, наделение его правами, извлекать данные из таблиц других пользователей (в качестве администратора)