



ETL AND SQL REVIEW

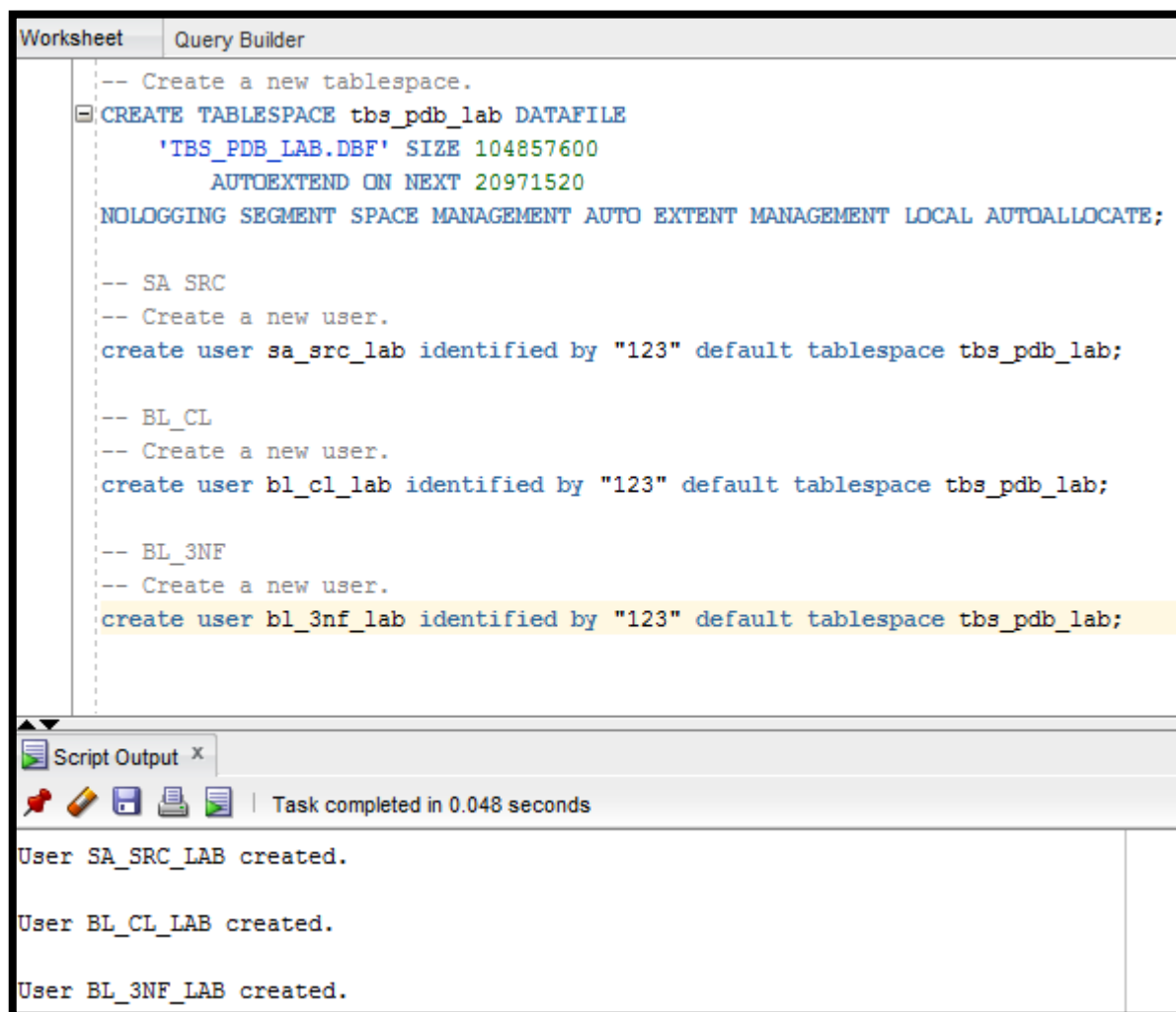
СПИСОК ИЗМЕНЕНИЙ					
Версия	Описание изменений	Автор	Дата	Подтверждено	
				Имя	Дата
1.0	Initial status	<u>Valeryia Lupanova</u>	17-NOV-2017		

ОГЛАВЛЕНИЕ

1	СОЗДАНИЕ ОБЪЕКТОВ ХРАНИЛИЩА.....	4
2	ГЕНЕРАЦИЯ ДАННЫХ ДЛЯ STAGING AREA.....	12
3	SQL*PLUS.....	19
3.1	КОМАНДА - EXECUTION PLANS SQL*PLANS.....	19
3.2	КОМАНДА - SET TIMING ON.....	21
3.3	ЗАПУСК СКРИПТА ИЗ SQL-ФАЙЛА.....	21
3.4	СОХРАНЕНИЕ ИНФОРМАЦИИ В ФАЙЛ	22

1 СОЗДАНИЕ ОБЪЕКТОВ ХРАНИЛИЩА

Сначала были созданы пользователи для каждого слоя в STAGING AREA.



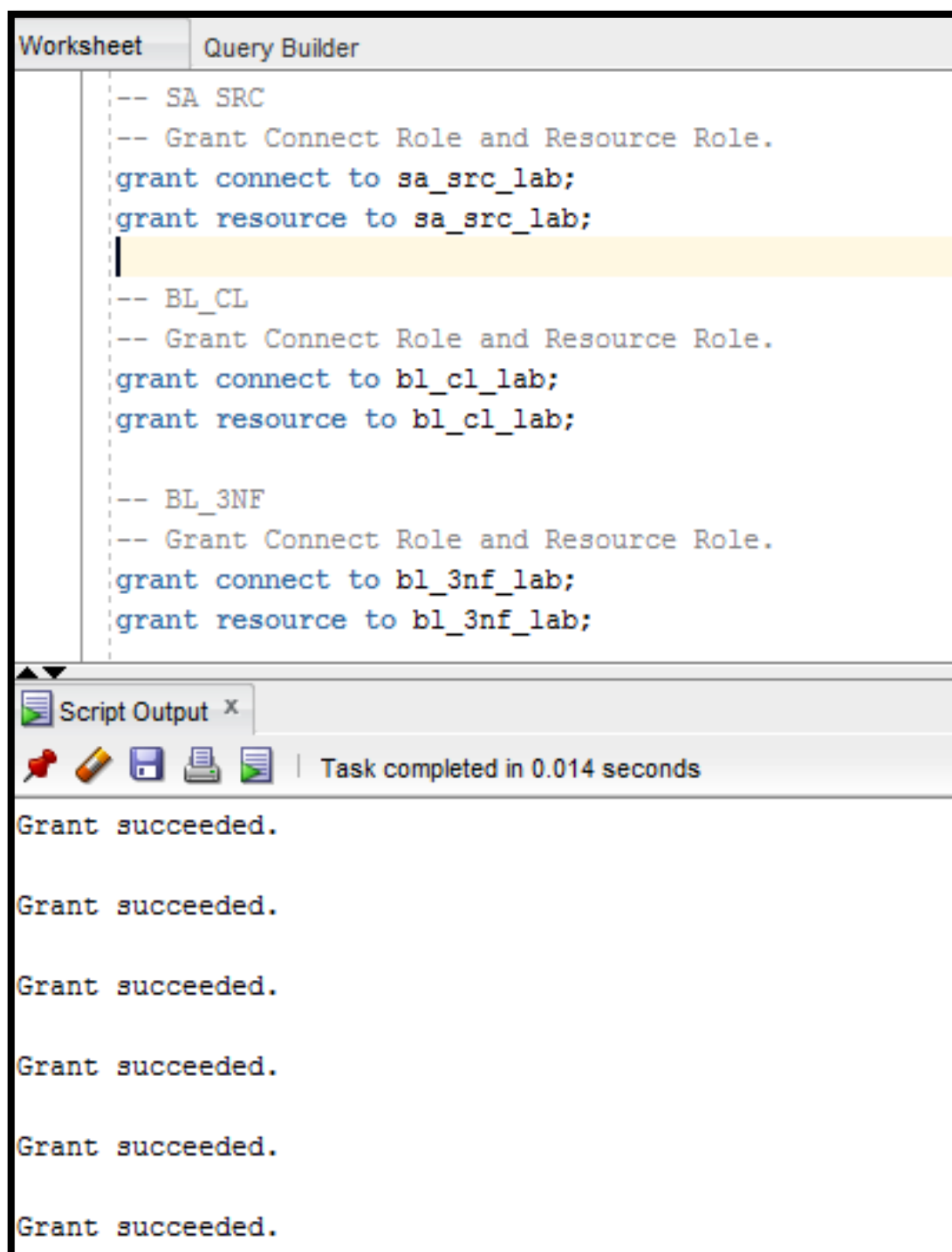
The screenshot shows a SQL Query Builder window with a 'Worksheet' tab. The SQL script in the editor is as follows:

```
-- Create a new tablespace.  
CREATE TABLESPACE tbs_pdb_lab DATAFILE  
    'TBS_PDB_LAB.DBF' SIZE 104857600  
    AUTOEXTEND ON NEXT 20971520  
    NOLOGGING SEGMENT SPACE MANAGEMENT AUTO EXTENT MANAGEMENT LOCAL AUTOALLOCATE;  
  
-- SA_SRC  
-- Create a new user.  
create user sa_src_lab identified by "123" default tablespace tbs_pdb_lab;  
  
-- BL_CL  
-- Create a new user.  
create user bl_cl_lab identified by "123" default tablespace tbs_pdb_lab;  
  
-- BL_3NF  
-- Create a new user.  
create user bl_3nf_lab identified by "123" default tablespace tbs_pdb_lab;
```

Below the editor, a 'Script Output' window shows the execution results:

```
Task completed in 0.048 seconds  
  
User SA_SRC_LAB created.  
  
User BL_CL_LAB created.  
  
User BL_3NF_LAB created.
```

Затем были даны гранты пользователям на создание соединения и объектов в схеме.



Скрипты на создание пользователей и раздачу грантов в папке SYSTEM.
Создала пакет на создание грантов в SYSTEM, SA_SRC_LAB и BL_CL_LAB, поскольку из этих схем потребуется выдавать гранты другим схемам.

```

Worksheet | Query Builder
CREATE OR REPLACE PACKAGE BODY pkg_grants AS

  PROCEDURE user_grant (grant_name IN VARCHAR2,
                        schema_name IN VARCHAR2,
                        object_name IN VARCHAR2,
                        user_name IN VARCHAR2)
  IS
  BEGIN
    EXECUTE IMMEDIATE ('GRANT ' || grant_name || ' ON ' || schema_name || '.' || object_name || ' TO ' || user_name);
  END user_grant;

  PROCEDURE user_grant (grant_name VARCHAR2,
                        user_name VARCHAR2,
                        comments IN boolean := FALSE)
  IS
  BEGIN
    IF comments THEN
      EXECUTE IMMEDIATE ('GRANT ' || grant_name || ' TO ' || user_name || ' WITH ADMIN OPTION');
    END IF;
    EXECUTE IMMEDIATE ('GRANT ' || grant_name || ' TO ' || user_name);
  END;
END pkg_grants;

```

Script Output x

Task completed in 0.121 seconds

Package PKG_GRANTS compiled

Затем дали грант на создание директории SA_SRC_LAB.

```

Worksheet | Query Builder
BEGIN
  pkg_grants.user_grant(grant_name => 'CREATE ANY DIRECTORY', user_name => 'SA_SRC_LAB');
END;

```

Script Output x

Task completed in 0.112 seconds

PL/SQL procedure successfully completed.

Затем создаем директорию в SA_SRC_LAB, которая является логической ссылкой в базе данных на каталог с файлами-источниками для внешних таблиц.

```

Worksheet | Query Builder
-- Creating directories.

CREATE OR REPLACE DIRECTORY external_tables AS '/media/sf_Valeryia_Lupanova/Task 01/Task 01/SA_SRC';

```

Затем создаем таблицы в SA_SRC_LAB для загрузки внешних таблиц.

```

-- Creating tables.
-- ext_geo_countries_iso3166.
create table EXT_COUNTRIES
  (COUNTRY_ID      number ( 10 ),
   COUNTRY_DESC    varchar2 ( 200 char ),
   COUNTRY_CODE    varchar2 ( 3 )
  )
organization external
  (type ORACLE LOADER
   default directory EXTERNAL_TABLES
   access parameters
     (RECORDS DELIMITED by 0x'0D0A'
      NOBADFILE NODISCARDFILE NOLOGFILE FIELDS TERMINATED by ';'
      MISSING FIELD values ARE null
      (COUNTRY_ID integer external (4),
       COUNTRY_DESC char(200),
       COUNTRY_CODE char(3)
      )
   )
  location ('iso_3166.tab')
)
reject limit unlimited;

```

В 1-м пункте создаем колонки для данных из внешнего источника, в 2-м пункте указываем директорию, в 3-м - указывает внешний файл, который нужно взять из директории. Аналогично создаются остальные таблицы.

```

Table EXT_COUNTRIES created.

Table EXT_STRUCTURES created.

Table EXT_FULL_DATA created.

```

Создаем таблицы WRK, в которые будут грузиться грязные данные, в CLEANSING LAYER - BL_CL_LAB. Была выбрана структура CONTINENTS -> REGIONS -> COUNTRIES.



```
--Creating WRK tables.

CREATE TABLE wrk_full_data
(
    country_id    NUMBER(10,0),
    county_desc   VARCHAR2(200 CHAR),
    structure_code NUMBER(10,0),
    structure_desc VARCHAR2(200 CHAR)
);

CREATE TABLE wrk_structures
(
    child_code     NUMBER(10,0),
    parent_code    NUMBER(10,0),
    structure_desc  VARCHAR2(200 CHAR),
    structure_level VARCHAR2(200 CHAR)
);

CREATE TABLE wrk_countries
(
    country_id    NUMBER ( 10 ),
```

Script Output x

Task completed in 0.009 seconds

Table WRK_FULL_DATA created.

Table WRK_STRUCTURES created.

Table WRK_COUNTRIES created.

Создаем таблицы CLS, в которые будут грузиться очищенные данные, в CLEANSING LAYER - BL_CL_LAB.


```
Worksheet | Query Builder

--Creating CLS tables.

CREATE TABLE cls_continents
(
    continent_id    NUMBER ( 10 ),
    continent_desc  VARCHAR2 ( 200 CHAR )
);

CREATE TABLE cls_regions
(
    region_id       NUMBER ( 10 ),
    region_desc     VARCHAR2(200 CHAR),
    continent_id    NUMBER ( 10 )
);

CREATE TABLE cls_countries
(
    country_id      NUMBER ( 10 ),
    country_desc    VARCHAR2 ( 200 CHAR ),
    country_code    VARCHAR2 ( 3 ),
    region_id       NUMBER ( 10 )
);
```

Script Output x


Task completed in 0.022 seconds

Table CLS_CONTINENTS created.

Table CLS_REGIONS created.

Table CLS_COUNTRIES created.

Создаем таблицы CE, в которые будут грузиться итоговые данные в нормализованном виде из CLEANSING LAYER - BL_CL_LAB, в 3NF LAYER - BL_3NF_LAB. В таблицы также добавляем ограничения - PRIMARY KEY и FOREIGN KEY.



Worksheet | Query Builder

```
--Creating CE tables.  
  
CREATE TABLE ce_continents  
(  
    continent_id    NUMBER ( 10 ),  
    continent_desc VARCHAR2 ( 200 CHAR )  
);  
  
CREATE TABLE ce_regions  
(  
    region_id      NUMBER ( 10 ),  
    region_desc    VARCHAR2(200 CHAR),  
    continent_id   NUMBER ( 10 )  
);  
  
CREATE TABLE ce_countries  
(  
    country_id     NUMBER ( 10 ),  
    country_desc   VARCHAR2 ( 200 CHAR ),  
    country_code   VARCHAR2 ( 3 ),  
    region_id      NUMBER ( 10 )
```

Script Output x

Task completed in 0.011 seconds

Table CE_CONTINENTS created.

Table CE_REGIONS created.

Table CE_COUNTRIES created.

```

-- Adding constraints.

ALTER TABLE ce_continents
ADD CONSTRAINT continent_id_pk PRIMARY KEY (continent_id);

ALTER TABLE ce_regions
ADD CONSTRAINT region_id_pk PRIMARY KEY (region_id);

ALTER TABLE ce_regions
ADD CONSTRAINT region_id_fk FOREIGN KEY (continent_id)
REFERENCES ce_continents (continent_id)
ON DELETE CASCADE;

ALTER TABLE ce_countries
ADD CONSTRAINT country_id_pk PRIMARY KEY (country_id);

ALTER TABLE ce_countries
ADD CONSTRAINT cn_region_id_fk FOREIGN KEY (region_id)
REFERENCES ce_regions (region_id)
ON DELETE CASCADE;

```

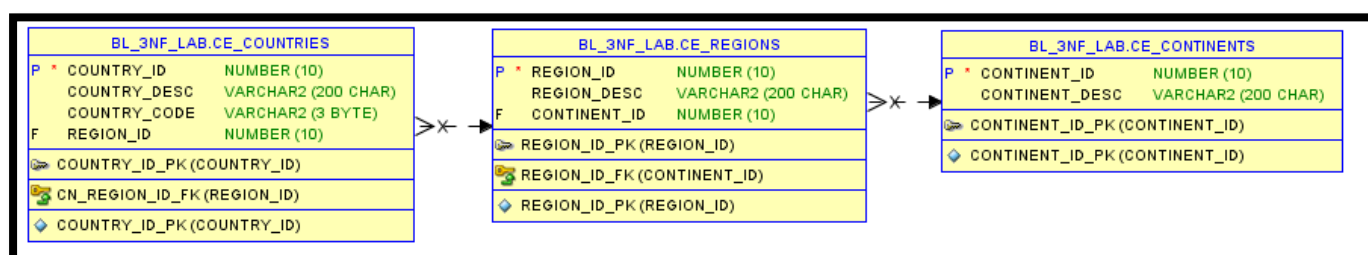
Script Output x

Task completed in 0.043 seconds

Table CE_COUNTRIES altered.

Table CE_COUNTRIES altered.

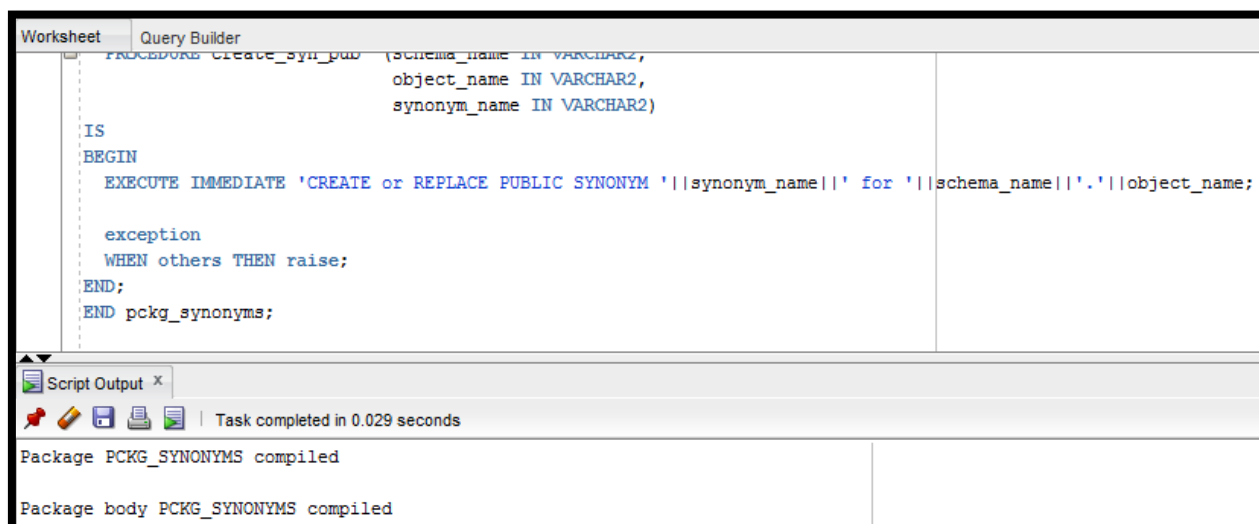
В результате можно посмотреть схему данных.



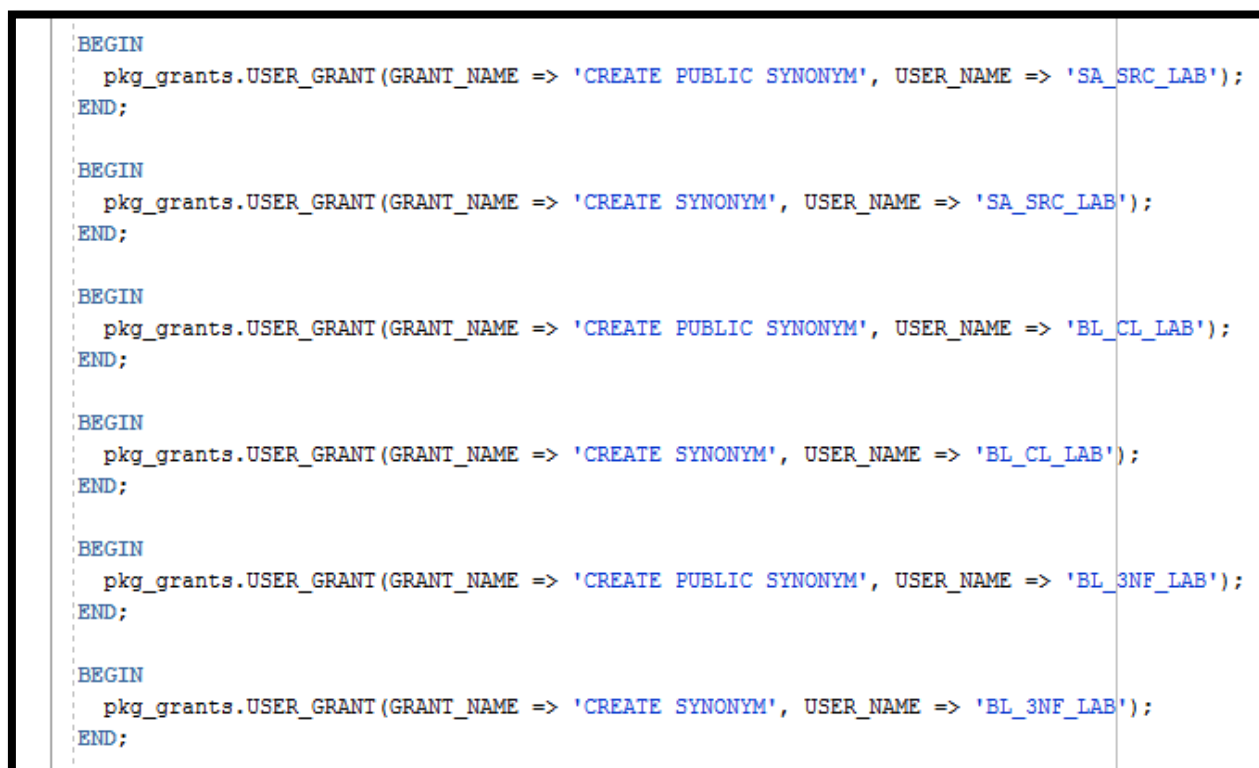
В результате получили вне необходимые на данный момент объекты БД.

2 ГЕНЕРАЦИЯ ДАННЫХ ДЛЯ STAGING AREA

Создали пакет на создание синонимов во всех слоях.



Даем из SYSTEM гранты на создание синонимов всем трем пользователям.



Создаем синонимы в SA_SRC_LAB: public - для обращения из других схем, обычные - для команд внутри самой схемы.

```

Worksheet | Query Builder
-- Synonyms for giving others.

BEGIN
  pkg_synonyms.create_syn_pub(schema_name => 'SA_SRC_LAB', object_name => 'EXT_COUNTRIES', synonym_name => 'SA_CN');
END;

BEGIN
  pkg_synonyms.create_syn_pub(schema_name => 'SA_SRC_LAB', object_name => 'EXT_STRUCTURES', synonym_name => 'SA_CS');
END;

BEGIN
  pkg_synonyms.create_syn_pub(schema_name => 'SA_SRC_LAB', object_name => 'EXT_FULL_DATA', synonym_name => 'SA_FD');
END;

-- Synonyms for giving grants.

BEGIN
  pkg_synonyms.create_syn_priv(schema_name => 'SA_SRC_LAB', object_name => 'EXT_COUNTRIES', synonym_name => 'SA_CN');
END;

BEGIN
  pkg_synonyms.create_syn_priv(schema_name => 'SA_SRC_LAB', object_name => 'EXT_STRUCTURES', synonym_name => 'SA_CS');
END;

```

Даем гранты из SA_SRC_LAB схеме BL_CL_LAB на SELECT из схемы SA_SRC_LAB.

```

Worksheet | Query Builder

BEGIN
  pkg_grants.USER_GRANT(GRANT_NAME => 'SELECT', SCHEMA_NAME => 'SA_SRC_LAB', OBJECT_NAME => 'EXT_COUNTRIES', USER_NAME => 'BL_CL_LAB');
END;

BEGIN
  pkg_grants.USER_GRANT(GRANT_NAME => 'SELECT', SCHEMA_NAME => 'SA_SRC_LAB', OBJECT_NAME => 'EXT_STRUCTURES', USER_NAME => 'BL_CL_LAB');
END;

BEGIN
  pkg_grants.USER_GRANT(GRANT_NAME => 'SELECT', SCHEMA_NAME => 'SA_SRC_LAB', OBJECT_NAME => 'EXT_FULL_DATA', USER_NAME => 'BL_CL_LAB');
END;

```

Script Output x

Task completed in 0.002 seconds

PL/SQL procedure successfully completed.

Даем гранты BL_CL_LAB из SA_SRC_LAB на чтение директории из SA_SRC_LAB.

```

GRANT READ ON DIRECTORY external_tables TO BL_CL_LAB;

```

Даем гранты BL_CL_LAB из SYSTEM на INSERT из таблиц SA_SRC_LAB.

```

BEGIN
    pkg_grants.USER_GRANT (GRANT_NAME => 'INSERT', SCHEMA_NAME => 'BL_CL_LAB', OBJECT_NAME => 'WRK_CN', USER_NAME => 'BL_CL_LAB');
END;

BEGIN
    pkg_grants.USER_GRANT (GRANT_NAME => 'INSERT', SCHEMA_NAME => 'BL_CL_LAB', OBJECT_NAME => 'WRK_CS', USER_NAME => 'BL_CL_LAB');
END;

BEGIN
    pkg_grants.USER_GRANT (GRANT_NAME => 'INSERT', SCHEMA_NAME => 'BL_CL_LAB', OBJECT_NAME => 'WRK_FD', USER_NAME => 'BL_CL_LAB');
END;

BEGIN

```

Script Output x

Task completed in 0.012 seconds

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Создаем пакет для вставки данных в BL_CL_LAB.

Worksheet Query Builder

```

CREATE OR REPLACE PACKAGE pkg_etl_insert
AUTHID CURRENT_USER
AS
    PROCEDURE insert_table (table_name_to IN VARCHAR2,
                           table_name_from IN VARCHAR2);

END pkg_etl_insert;

CREATE OR REPLACE PACKAGE BODY pkg_etl_insert
AS
    PROCEDURE insert_table (table_name_to IN VARCHAR2,
                           table_name_from IN VARCHAR2)
    IS
    BEGIN
        EXECUTE IMMEDIATE ('TRUNCATE TABLE ' || table_name_to);
        EXECUTE IMMEDIATE ('INSERT INTO ' || table_name_to || ' SELECT * FROM ' || table_name_from);
    END insert_table;
END pkg_etl_insert;

```

Вставляем данные в BL_CL_LAB из SA_SRC_LAB.

Worksheet Query Builder

```

BEGIN
    pkg_etl_insert.insert_table(table_name_to => 'WRK_COUNTRIES', table_name_from => 'SA_CN');
END;

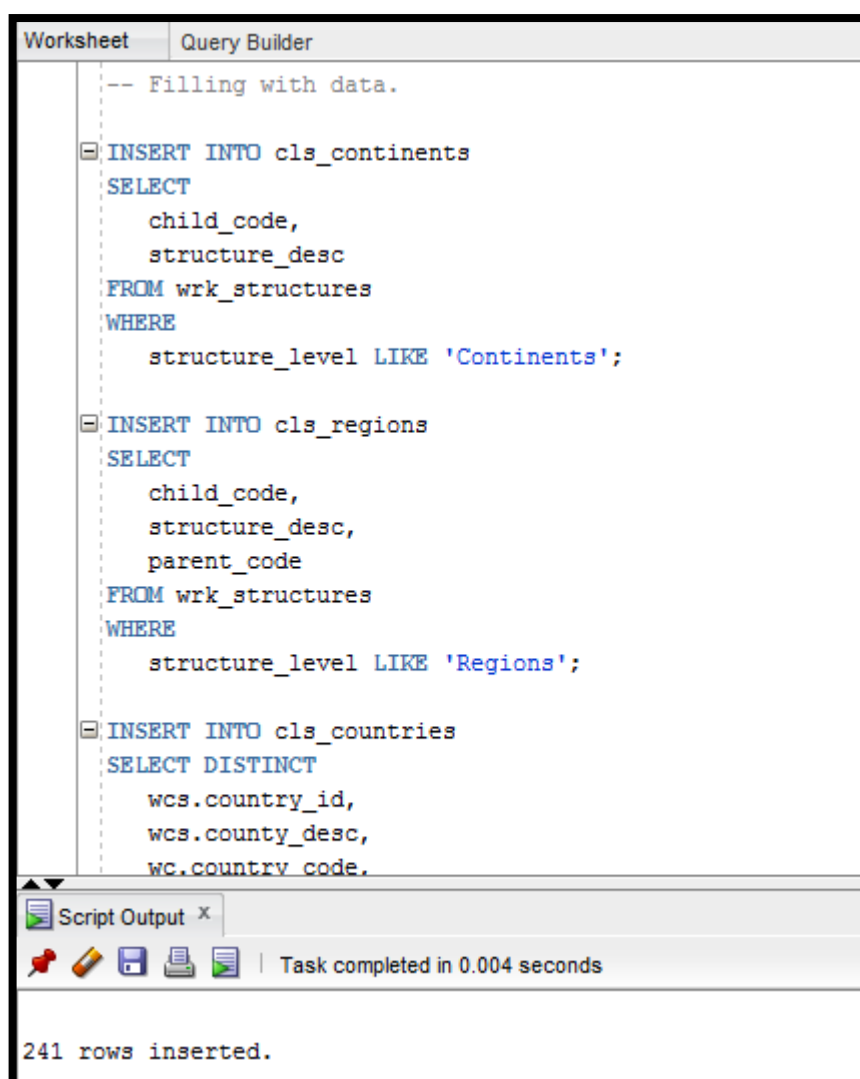
BEGIN
    pkg_etl_insert.insert_table(table_name_to => 'WRK_STRUCTURES', table_name_from => 'SA_CS');
END;

BEGIN
    pkg_etl_insert.insert_table(table_name_to => 'WRK_FULL_DATA', table_name_from => 'SA_FD');
END;

COMMIT;

```

Вставляем данные из WRK-таблиц в CLS-таблицы схемы BL_CL_LAB.



Даем гранты BL_3NF_LAB из BL_CL_LAB на выборку данных из таблиц схемы BL_CL_LAB.



Даем гранты BL_3NF_LAB на вставку данных в таблицы.

```

BEGIN
  pkg_grants.USER_GRANT (GRANT_NAME => 'INSERT', SCHEMA_NAME => 'BL_3NF_LAB', OBJECT_NAME => 'CE_COUNTRIES', USER_NAME => 'BL_3NF_LAB')
END;

BEGIN
  pkg_grants.USER_GRANT (GRANT_NAME => 'INSERT', SCHEMA_NAME => 'BL_3NF_LAB', OBJECT_NAME => 'CE_REGIONS', USER_NAME => 'BL_3NF_LAB')
END;

BEGIN
  pkg_grants.USER_GRANT (GRANT_NAME => 'INSERT', SCHEMA_NAME => 'BL_3NF_LAB', OBJECT_NAME => 'CE_CONTINENTS', USER_NAME => 'BL_3NF_LAB')
END;

```

Создаем пакет для заполнения BL_3NF_LAB.

Worksheet | Query Builder

```

CREATE OR REPLACE PACKAGE pkg_etl_insert_cntn
AUTHID CURRENT_USER
AS
  PROCEDURE insert_table_continents;
  PROCEDURE insert_table_regions;
END pkg_etl_insert_cntn;

CREATE OR REPLACE PACKAGE BODY pkg_etl_insert_cntn
AS
  -----
  PROCEDURE insert_table_continents
  IS
  BEGIN
    MERGE INTO ce_continents t USING
    ( SELECT * FROM bl_cl_lab.cls_continents
      MINUS
      SELECT * FROM ce_continents
    ) c ON ( c.continent_id = t.continent_id )
    WHEN matched THEN
      UPDATE SET t.continent_desc = c.continent_desc
    WHEN NOT matched THEN
      INSERT

```

Script Output x

Task completed in 0.109 seconds

Package PKG_ETL_INSERT_CNTN compiled

Package PKG_ETL_INSERT_CNTN compiled

Package body PKG_ETL_INSERT_CNTN compiled

Заполняем данными с помощью пакета и коммитим.


```

Worksheet  Query Builder
begin
    PKG_ETL_INSERT_CNTN.INSERT_TABLE_CONTINENTS;
end;

begin
    PKG_ETL_INSERT_CNTN.INSERT_TABLE_REGIONS;
end;

begin
    PKG_ETL_INSERT_CNTN.INSERT_TABLE_COUNTRIES;
end;

COMMIT;

```

```

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Commit complete.

```

Проверяем данные.

The screenshot shows the SQL Developer interface. On the left, the 'Tables (Filtered)' folder is expanded, and 'CE_CONTINENTS' is highlighted with a red box. On the right, the 'Query Builder' tab is active, showing the query `SELECT * FROM ce_continents;` with 'ce_continents' highlighted in red. Below the query, the 'Query Result' window displays 6 rows of data:

CONTINENT_ID	CONTINENT_DESC
1	2 Africa
2	9 Oceania
3	21 Northern America
4	142 Asia
5	150 Europe
6	419 Latin America and the Caribbean

The screenshot shows the SQL Developer interface. On the left, the 'Tables (Filtered)' folder is expanded, and 'CE_COUNTRIES' is highlighted with a red box. On the right, the 'Query Builder' tab is active, showing the query `SELECT count(*) FROM ce_countries;` with the entire query highlighted in red. Below the query, the 'Query Result' window displays 1 row of data:

COUNT(*)
241

The screenshot displays a database management interface. On the left, a tree view shows the database structure with 'CE_REGIONS' highlighted. The main window shows a 'Query Result' table with 18 rows of data. The table has three columns: 'REGION_ID', 'REGION_DESC', and 'CONTINENT_ID'. The data is as follows:

REGION_ID	REGION_DESC	CONTINENT_ID
1	5 South America	419
2	11 Western Africa	2
3	13 Central America	419
4	14 Eastern Africa	2
5	15 Northern Africa	2
6	17 Middle Africa	2
7	18 Southern Africa	2
8	21 Northern America	21
9	29 Caribbean	419
10	30 Eastern Asia	142
11	34 Southern Asia	142
12	35 South-Eastern Asia	142
13	39 Southern Europe	150
14	53 Australia and New Zealand	9
15	54 Melanesia	9
16	57 Micronesia	9
17	61 Polynesia	9
18	143 Central Asia	142

3 SQL*PLUS

Создаем соединение к SQLPLUS через командную строку LINUX.

```
[oracle@localhost ~]$ sqlplus

SQL*Plus: Release 12.1.0.2.0 Production on Fri Nov 17 12:24:11 2017

Copyright (c) 1982, 2014, Oracle. All rights reserved.
```

Вводим данные для авторизации.

```
Enter user-name: bl_3nf_lab
Enter password:
Last Successful login time: Fri Nov 17 2017 08:58:55 -08:00

Connected to:
Oracle Database 12c Enterprise Edition Release 12.1.0.2.0 - 64bit Production
With the Partitioning, OLAP, Advanced Analytics and Real Application Testing opt
ions
```

Проверяем первый селект.

```
SQL> select * from ce_countries cc left join ce_regions cr on cc.region_id = cr.
region_id where cr.region_id = 17;
```

```
COUNTRY_ID
-----
COUNTRY_DESC
-----
COU REGION_ID REGION_ID
-----
REGION_DESC
-----
CONTINENT_ID
-----
Middle Africa
          2
```

3.1 КОМАНДА – EXECUTION PLANS SQL*PLANS

```
SQL> explain plan for
  2 select * from ce_regions cr inner join ce_continents cc
  3 on cr.continent_id = cc.continent_id;

Explained.
```

```
SQL> SELECT PLAN_TABLE_OUTPUT FROM TABLE(DBMS_XPLAN.DISPLAY());
```

```
PLAN_TABLE_OUTPUT
```

```
-----
Plan hash value: 211575201
-----
```

```
-----
| Id | Operation | Name | Rows | Bytes | Cost (%
CPU)| Time |
-----
```

```
PLAN_TABLE_OUTPUT
```

```
-----
| 0 | SELECT STATEMENT | | 22 | 18546 | 3
(0)| 00:00:01 |
| 1 | NESTED LOOPS | | 22 | 18546 | 3
(0)| 00:00:01 |
| 2 | NESTED LOOPS | | 22 | 18546 | 3
(0)| 00:00:01 |
```

```
| 3 | TABLE ACCESS FULL | CE_REGIONS | 22 | 9416 | 3
(0)| 00:00:01 |
```

```
PLAN_TABLE_OUTPUT
```

```
-----
|* 4 | INDEX UNIQUE SCAN | CONTINENT_ID_PK | 1 | | 0
(0)| 00:00:01 |
| 5 | TABLE ACCESS BY INDEX ROWID | CE_CONTINENTS | 1 | 415 | 0
(0)| 00:00:01 |
-----
```

```
PLAN_TABLE_OUTPUT
```

```
-----
Predicate Information (identified by operation id):
-----
```

```
PLAN_TABLE_OUTPUT
```

```
-----
Predicate Information (identified by operation id):
-----
```

```
4 - access("CR"."CONTINENT_ID"="CC"."CONTINENT_ID")
```

```
Note
```

```
-----
```

```
- dynamic statistics used: dynamic sampling (level=2)
```

```
21 rows selected.
```

3.2 КОМАНДА – SET TIMING ON

```
SQL> set timing on;
SQL> select * from ce_regions cr inner join ce_continents cc
2 on cr.continent_id = cc.continent_id
3 where cr.region_id = 17;

REGION_ID
-----
REGION_DESC
-----
CONTINENT_ID CONTINENT_ID
-----
CONTINENT_DESC
-----
17
Middle Africa
2 2
Africa

Elapsed: 00:00:00.01
SQL>
```

3.3 ЗАПУСК СКРИПТА ИЗ SQL-ФАЙЛА

Для запуска использовался файл SELECT_SCRIPT.SQL, созданный в следующем задании с помощью команд SAVE/CREATE и записанный с помощью команды APPEND.

```
SQL> @select_script.sql

  REGION_ID
-----
 REGION_DESC
-----
CONTINENT_ID CONTINENT_ID
-----
CONTINENT_DESC
-----
           17
Middle Africa
           2           2
Africa
```

3.4 СОХРАНЕНИЕ ИНФОРМАЦИИ В ФАЙЛ

Сохранение SQL-команды в файл.

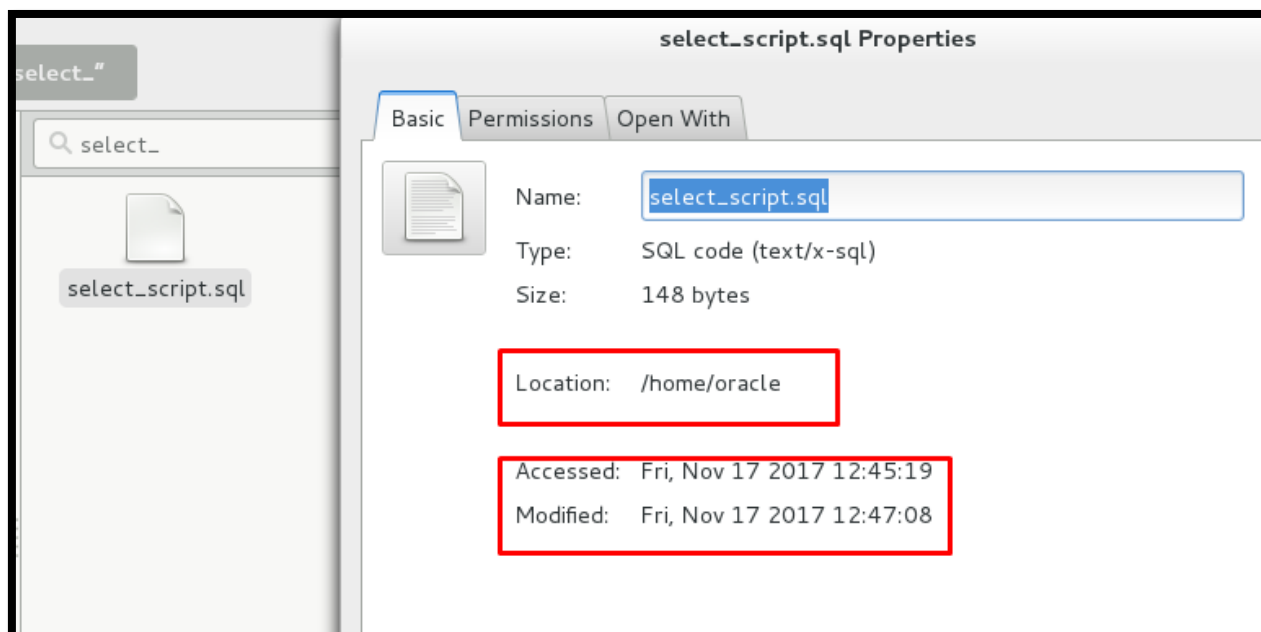
```
SQL> save select_script.sql create
Created file select_script.sql
SQL>
```

```
SQL> select * from ce_countries;

COUNTRY_ID
-----
COUNTRY_DESC
-----
COU  REGION_ID
-----
           438
Liechtenstein
LIE           155

           756
Switzerland
CHE           155
```

```
SQL> save select_script.sql append;
Appended file to select_script.sql
SQL> █
```



Сохранение результата SQL-команды в файл.

```
SQL> spool select_output.txt
SQL> █
```

```
SQL> select * from ce_countries;

COUNTRY_ID
-----
COUNTRY_DESC
-----
COU REGION_ID
-----
      438
Liechtenstein
LIE      155

      756
Switzerland
CHE      155
```

```
SQL> spool off;
SQL> █
```

Результат можно посмотреть в соответствующих документах в папке SQLPLUS.