

Документ подписан простой электронной подписью
Информация о владельце:
ФИО: Емельянов Сергей Геннадьевич
Должность: ректор
Дата подписания: 27.04.2023 09:18:31
Уникальный программный ключ:
9ba7d3e34c012eba476ffd2d064cf2781953be730df2374d16f3c0ce536f0fc6

МИНОБРАЗОВАНИЯ РОССИИ

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра программной инженерии

УТВЕРЖДАЮ

Проректор по учебной работе

 О.Г. Локтионова

2020 г.

СОЗДАНИЕ ТОПОЛОГИИ ИЗ ТОПОЛОГИЧЕСКИХ ДАННЫХ

Методические указания по выполнению лабораторной работы
по дисциплине «Пространственные базы данных»
для студентов направления подготовки 09.04.04 «Программная
инженерия»

Курск 2020

УДК 004.65

Составители: В.Г. Белов, Т.М. Белова

Рецензент

Кандидат технических наук, доцент кафедры программной инженерии ЮЗГУ И.Н. Ефремова

Создание топологии из топологических данных: методические указания по выполнению лабораторной работы по дисциплине «Пространственные базы данных» для студентов направления подготовки 09.04.04 «Программная инженерия» / Юго-Зап. гос. ун-т; сост.: В.Г. Белов, Т.М. Белова, – Курск, 2020. – 30 с.: ил. 49.

Изложены методические указания по созданию топологий на основе имеющихся топологических данных, выраженных в терминах узлов, ребер и поверхностей.

Материал предназначен для студентов направления подготовки 09.04.04 «Программная инженерия», а также будет полезен студентам всех направлений подготовки, изучающим технологии разработки пространственных баз данных.

Текст печатается в авторской редакции.

Подписано в печать 18.04.20. Формат 60x84 1/16.
Усл. печ. л. 1,7. Уч.-изд. л. 1,6. Тираж 100 экз. Заказ 4445. Бесплатно.
Юго-Западный государственный университет.
305040, Курск, ул.50 лет Октября, 94.

Содержание

1 Цель лабораторной работы	4
2 Порядок выполнения лабораторной работы	5
3 Содержание отчета по лабораторной работе	29
4 Вопросы к защите лабораторной работы	30

1 Цель лабораторной работы

Целью лабораторной работы является приобретение знаний, умений и навыков при создании топологий в Oracle Spatial.

2 Порядок выполнения лабораторной работы

Создадим топологию, отображающую городские объекты. Схема топологии представлена на рисунке 2.1.

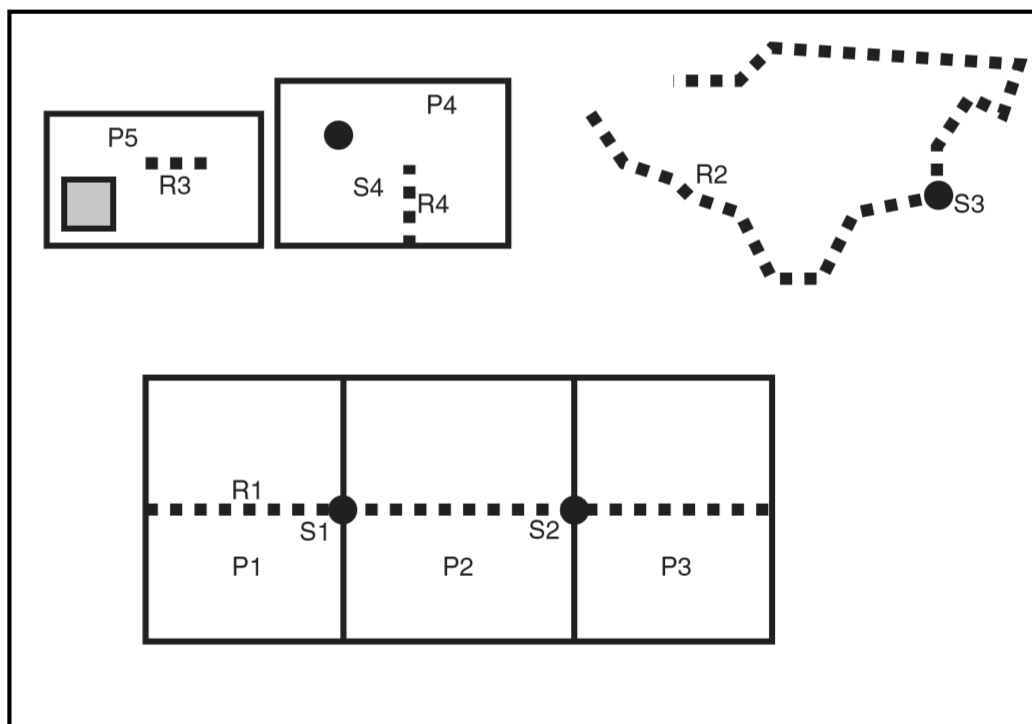


Рисунок 2.1 – Схема топологии

- Точечные объекты (дорожные знаки) показаны темными кружками: S1, S2, S3, S4.
- Линейные объекты (дороги или улицы) показаны пунктирными линиями: R, R2, R3, R4.
- Площадные объекты (земельные участки), показаны прямоугольниками: P1, P2, P3, P4, P5

Земельный участок P5 не включает в себя серую область внутри.

Все действия по созданию и использованию топологии будут выполняться посредством SQL-запросов в предварительно созданной базе данных, для чего может использоваться любой

программный продукт, позволяющий подключиться к СУБД Oracle Database и исполнять запросы. Примеры будут приводиться с использованием среды разработки Oracle JDeveloper.

Для создания топологии необходимо использовать процедуру Oracle Spatial SDO.TOPO.CREATE_TOPOLOGY. SQL-запрос для данной операции приведен на рисунке 2.2.

```
-- 1. Create the topology. (Null SRID in this example.)
EXECUTE SDO_TOPO.CREATE_TOPOLOGY('CITY_DATA', 0.00005);
```

Рисунок 2.2 – SQL-запрос для создания топологии

В процессе ее выполнения создадутся таблицы <название топологии>_EDGE\$, <название топологии>_NODE\$, <название топологии>_FACE и <название топологии>_HISTORY. Результат выполнения данной команды представлен на рисунке 2.3.

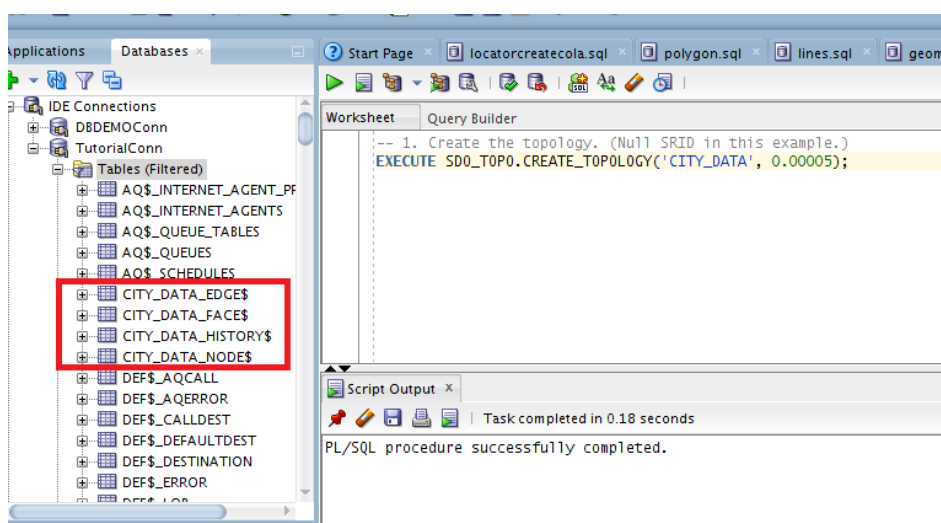


Рисунок 2.3 – Создание топологии

После необходимо загрузить топологические данные, выраженные в терминах узлов, ребер и поверхностей, в свежесозданные таблицы. Как правило для этого применяют специальные программы для массовой загрузки данных, но можно использовать и обычные SQL-запросы INSERT.

Для загрузки данных в таблицу CITY_DATA_EDGES\$ используются запросы, представленные на рисунках 2.4-2.7.

```
-- 2. Load topology data (node, edge, and face tables).
-- Use INSERT statements here instead of a bulk-load utility.
-- 2A. Insert data into <topology_name>_EDGE$ table.
-- E1
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(1, 1, 1, 1, 1, -1, -1, 1, -1,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(8,30, 16,30, 16,38, 3,38, 3,30, 8,30)));
-- E2
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(2, 2, 2, 3, -3, -2, -2, 2, -1,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(25,30, 31,30, 31,40, 17,40, 17,30, 25,30)));
-- E3
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(3, 2, 3, -3, 2, 2, 3, 2, 2,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(25,30, 25,35)));
-- E4
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(4, 5, 6, -5, -4, 4, 5, -1, -1,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(36,38, 38,35, 41,34, 42,33, 45,32, 47,28, 50,28, 52,32,
57,33)));
```

Рисунок 2.4 – SQL-запросы для загрузки данных в таблицу CITY_DATA_EDGES\$

```

-- E5
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(5, 7, 6, -4, -5, 5, 4, -1, -1,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(41,40, 45,40, 47,42, 62,41, 61,38, 59,39, 57,36,
57,33)));
-- E6
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(6, 16, 17, 7, 21, -21, 19, -1, 3,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(9,22, 21,22)));
-- E7
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(7, 17, 18, 8, 6, -19, 17, -1, 4,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(21,22, 35,22)));
-- E8
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(8, 18, 19, -15, 7, -17, 15, -1, 5,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(35,22, 47,22)));
-- E9
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(9, 15, 14, 19, -21, -22, 20, 3, 6,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(9,14, 21,14)));
-- E10
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(10, 13, 14, -20, 18, 17, -19, 7, 4,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(35,14, 21,14)));
-- E11
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(11, 13, 12, 15, -17, -18, 16, 5, 8,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(35,14, 47,14)));

```

Рисунок 2.5 – SQL-запросы для загрузки данных в таблицу
CITY_DATA_EDGES\$


```

-- E12
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(12, 8, 9, 20, -22, 22, -13, 6, -1,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(9,6, 21,6)));
-- E13
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(13, 9, 10, 18, -20, -12, -14, 7, -1,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(21,6, 35,6)));
-- E14
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(14, 10, 11, 16, -18, -13, -16, 8, -1,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(35,6, 47,6)));
-- E15
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(15, 12, 19, -8, 11, -16, 8, 5, -1,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(47,14, 47,22)));
-- E16
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(16, 11, 12, -11, 14, -14, -15, 8, -1,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(47,6, 47,14)));
-- E17
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(17, 13, 18, -7, -10, 11, -8, 4, 5,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(35,14, 35,22)));
-- E18
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(18, 10, 13, 10, 13, 14, -11, 7, 8,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(35,6, 35,14)));

```

Рисунок 2.6 – SQL-запросы для загрузки данных в таблицу
CITY_DATA_EDGES\$

```

-- E19
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(19, 14, 17, -6, 9, -10, -7, 3, 4,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(21,14, 21,22)));
-- E20
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(20, 9, 14, -9, 12, 13, 10, 6, 7,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(21,6, 21,14)));
-- E21
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(21, 15, 16, 6, 22, 9, -6, -1, 3,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(9,14, 9,22)));
-- E22
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(22, 8, 15, 21, -12, 12, -9, -1, 6,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(9,6, 9,14)));
-- E25
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(25, 21, 22, -25, -25, 25, 25, 1, 1,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(9,35, 13,35)));
-- E26
INSERT INTO city_data_edge$ (edge_id, start_node_id, end_node_id,
next_left_edge_id, prev_left_edge_id, next_right_edge_id,
prev_right_edge_id, left_face_id, right_face_id, geometry)
VALUES(26, 20, 20, 26, 26, -26, -26, 9, 1,
SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 2, 1),
SDO_ORDINATE_ARRAY(4,31, 7,31, 7,34, 4,34, 4,31)));

```

Рисунок 2.7 – SQL-запросы для загрузки данных в таблицу
CITY_DATA_EDGES\$

Результаты запросов представлены на рисунках 2.8-2.9.

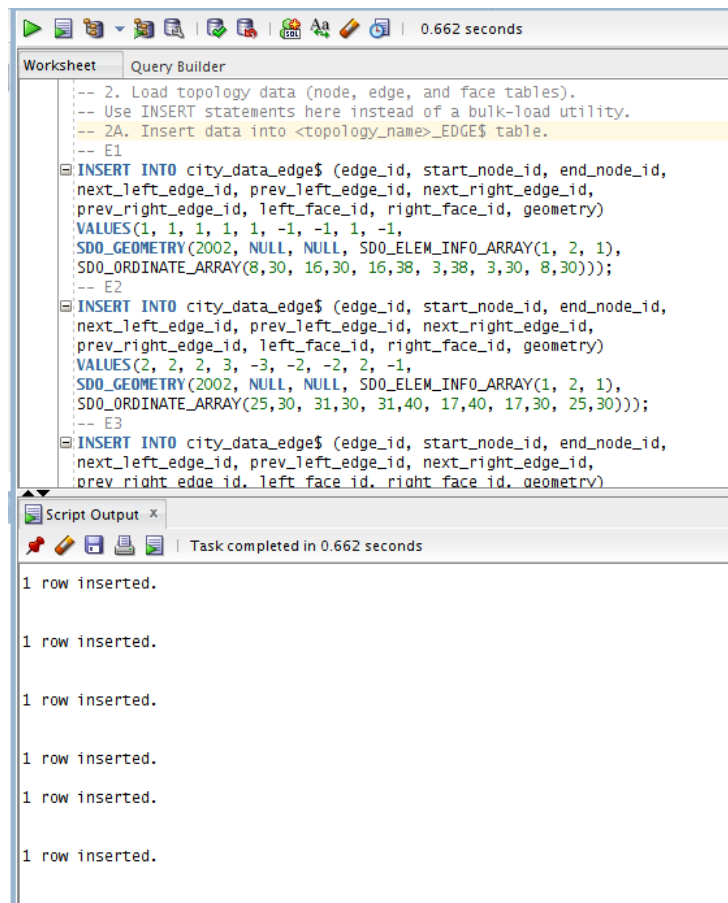


Рисунок 2.8 – Результат загрузки данных в таблицу CITY_DATA_EDGES\$

EDGE_ID	START_NODE_ID	END_NODE_ID	NEXT_LEFT_EDGE_ID	PREV_LEFT_EDGE_ID	NEXT_RIGHT_EDGE_ID	PREV_RIGHT_EDGE_ID	LEFT_FACE_ID	RIGHT_FACE_ID	GEOMETRY
1	1	1	1	1	-1	-1	1	-1	[MDSYS.SDO_GEOMETRY]
2	2	2	2	3	-2	-2	2	-1	[MDSYS.SDO_GEOMETRY]
3	3	2	3	-3	2	2	3	2	[MDSYS.SDO_GEOMETRY]
4	4	5	6	-5	-4	4	5	-1	[MDSYS.SDO_GEOMETRY]
5	5	7	6	-4	-5	5	4	-1	[MDSYS.SDO_GEOMETRY]
6	6	16	17	7	21	-21	19	-1	[MDSYS.SDO_GEOMETRY]
7	7	17	18	8	6	-19	17	-1	[MDSYS.SDO_GEOMETRY]
8	8	18	19	-15	7	-17	15	-1	[MDSYS.SDO_GEOMETRY]
9	9	15	14	19	-21	-22	20	3	[MDSYS.SDO_GEOMETRY]
10	10	13	14	-20	18	17	-19	7	[MDSYS.SDO_GEOMETRY]
11	11	13	12	15	-17	-18	16	5	[MDSYS.SDO_GEOMETRY]
12	12	8	9	20	-22	22	-13	6	[MDSYS.SDO_GEOMETRY]
13	13	9	10	18	-20	-12	-14	7	[MDSYS.SDO_GEOMETRY]
14	14	10	11	16	-18	-13	-16	8	[MDSYS.SDO_GEOMETRY]
15	15	12	19	-8	11	-16	8	5	[MDSYS.SDO_GEOMETRY]
16	16	11	12	-11	14	-14	-15	8	[MDSYS.SDO_GEOMETRY]
17	17	13	18	-7	-10	11	-8	4	[MDSYS.SDO_GEOMETRY]
18	18	10	13	10	13	14	-11	7	[MDSYS.SDO_GEOMETRY]
19	19	14	17	-6	9	-10	-7	3	[MDSYS.SDO_GEOMETRY]
20	20	9	14	-9	12	13	10	6	[MDSYS.SDO_GEOMETRY]
21	21	15	16	6	22	9	-6	-1	[MDSYS.SDO_GEOMETRY]
22	22	8	15	21	-12	12	-9	-1	[MDSYS.SDO_GEOMETRY]
23	25	21	22	-25	-25	25	25	1	[MDSYS.SDO_GEOMETRY]
24	26	20	20	26	26	-26	-26	9	[MDSYS.SDO_GEOMETRY]

Рисунок 2.9 – Результат загрузки данных в таблицу CITY_DATA_EDGES\$

Для загрузки данных в таблицу CITY_DATA_NODES\$ используются запросы, представленные на рисунках 2.10-2.11.

```

-- 2. Load topology data (node, edge, and face tables).
-- Use INSERT statements here instead of a bulk-load utility.
-- 2B. Insert data into <topology_name>_NODE$ table.
-- N1
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(1, 1, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(8,30,NULL), NULL, NULL));
-- N2
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(2, 2, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(25,30,NULL), NULL, NULL));
-- N3
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(3, -3, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(25,35,NULL), NULL, NULL));
-- N4
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(4, NULL, 2,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(20,37,NULL), NULL, NULL));
-- N5
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(5, 4, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(36,38,NULL), NULL, NULL));
-- N6
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(6, -4, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(57,33,NULL), NULL, NULL));
-- N7
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(7, 5, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(41,40,NULL), NULL, NULL));
-- N8
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(8, 12, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(9,6,NULL), NULL, NULL));
-- N9
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(9, 20, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(21,6,NULL), NULL, NULL));
-- N10
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(10, 18, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(35,6,NULL), NULL, NULL));
-- N11
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(11, -14, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(47,6,NULL), NULL, NULL));
-- N12
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(12, 15, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(47,14,NULL), NULL, NULL));

```

Рисунок 2.10 – SQL-запросы для загрузки данных в таблицу
CITY_DATA_NODES

```

-- N13
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(13, 17, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(35,14,NULL), NULL, NULL));
-- N14
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(14, 19, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(21,14,NULL), NULL, NULL));
-- N15
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(15, 21, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(9,14,NULL), NULL, NULL));
-- N16
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(16, 6, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(9,22,NULL), NULL, NULL));
-- N17
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(17, 7, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(21,22,NULL), NULL, NULL));
-- N18
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(18, 8, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(35,22,NULL), NULL, NULL));
-- N19
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(19, -15, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(47,22,NULL), NULL, NULL));
-- N20
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(20, 26, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(4,31,NULL), NULL, NULL));
-- N21
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(21, 25, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(9,35,NULL), NULL, NULL));
-- N22
INSERT INTO city_data_node$ (node_id, edge_id, face_id, geometry)
VALUES(22, -25, NULL,
SDO_GEOMETRY(2001, NULL, SDO_POINT_TYPE(13,35,NULL), NULL, NULL));

```

Рисунок 2.11 – SQL-запросы для загрузки данных в таблицу
CITY_DATA_NODE\$

Результаты запросов представлены на рисунках 2.12-2.13.

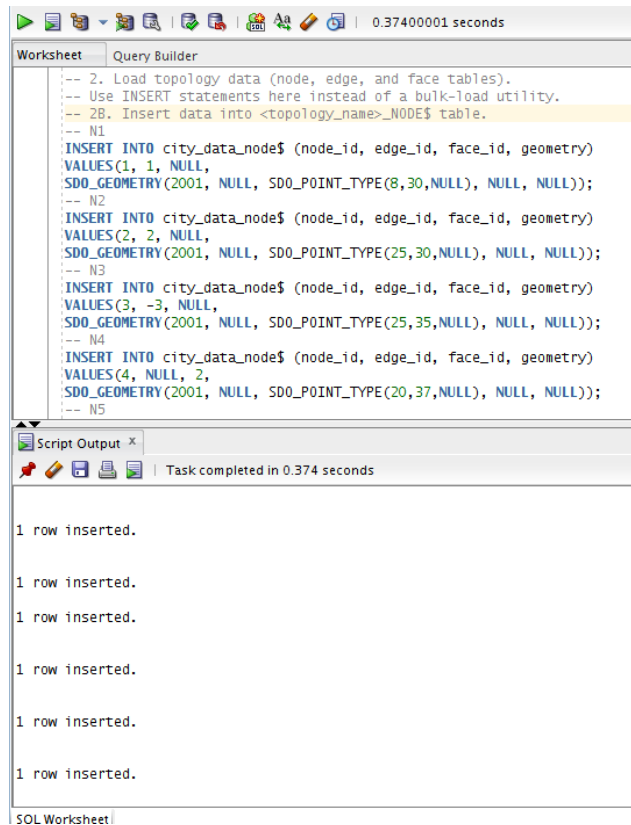


Рисунок 2.12 – Результат загрузки данных в таблицу
CITY_DATA_NODES\$

NODE_ID	EDGE_ID	FACE_ID	GEOMETRY
1	1	(null)	[MDSYS.SDO_GEOMETRY]
2	2	2	[MDSYS.SDO_GEOMETRY]
3	3	-3	[MDSYS.SDO_GEOMETRY]
4	4	(null)	2 [MDSYS.SDO_GEOMETRY]
5	5	4	(null) [MDSYS.SDO_GEOMETRY]
6	6	-4	(null) [MDSYS.SDO_GEOMETRY]
7	7	5	(null) [MDSYS.SDO_GEOMETRY]
8	8	12	(null) [MDSYS.SDO_GEOMETRY]
9	9	20	(null) [MDSYS.SDO_GEOMETRY]
10	10	18	(null) [MDSYS.SDO_GEOMETRY]
11	11	-14	(null) [MDSYS.SDO_GEOMETRY]
12	12	15	(null) [MDSYS.SDO_GEOMETRY]
13	13	17	(null) [MDSYS.SDO_GEOMETRY]
14	14	19	(null) [MDSYS.SDO_GEOMETRY]
15	15	21	(null) [MDSYS.SDO_GEOMETRY]
16	16	6	(null) [MDSYS.SDO_GEOMETRY]
17	17	7	(null) [MDSYS.SDO_GEOMETRY]
18	18	8	(null) [MDSYS.SDO_GEOMETRY]
19	19	-15	(null) [MDSYS.SDO_GEOMETRY]
20	20	26	(null) [MDSYS.SDO_GEOMETRY]
21	21	25	(null) [MDSYS.SDO_GEOMETRY]
22	22	-25	(null) [MDSYS.SDO_GEOMETRY]

Рисунок 2.13 – Результат загрузки данных в таблицу
CITY_DATA_NODES\$

Для загрузки данных в таблицу CITY_DATA_FACES\$ используются запросы, представленные на рисунках 2.14-2.15.

```

-- 2. Load topology data (node, edge, and face tables).
-- Use INSERT statements here instead of a bulk-load utility.
-- 2C. Insert data into <topology_name>_FACE$ table.
-- F0 (id = -1, not 0)
INSERT INTO city_data_face$ (face_id, boundary_edge_id,
island_edge_id_list, island_node_id_list, mbr_geometry)
VALUES(-1, NULL, SDO_LIST_TYPE(-1, -2, 4, 6),
SDO_LIST_TYPE(), NULL);
-- F1
INSERT INTO city_data_face$ (face_id, boundary_edge_id,
island_edge_id_list, island_node_id_list, mbr_geometry)
VALUES(1, 1, SDO_LIST_TYPE(25, -26), SDO_LIST_TYPE(),
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(3,30, 15,38)));
-- F2
INSERT INTO city_data_face$ (face_id, boundary_edge_id,
island_edge_id_list, island_node_id_list, mbr_geometry)
VALUES(2, 2, SDO_LIST_TYPE(), SDO_LIST_TYPE(4),
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(17,30, 31,40)));
-- F3
INSERT INTO city_data_face$ (face_id, boundary_edge_id,
island_edge_id_list, island_node_id_list, mbr_geometry)
VALUES(3, 19, SDO_LIST_TYPE(), SDO_LIST_TYPE(),
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(9,14, 21,22)));
-- F4
INSERT INTO city_data_face$ (face_id, boundary_edge_id,
island_edge_id_list, island_node_id_list, mbr_geometry)
VALUES(4, 17, SDO_LIST_TYPE(), SDO_LIST_TYPE(),
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(21,14, 35,22)));
-- F5
INSERT INTO city_data_face$ (face_id, boundary_edge_id,
island_edge_id_list, island_node_id_list, mbr_geometry)
VALUES(5, 15, SDO_LIST_TYPE(), SDO_LIST_TYPE(),
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(35,14, 47,22)));
-- F6
INSERT INTO city_data_face$ (face_id, boundary_edge_id,
island_edge_id_list, island_node_id_list, mbr_geometry)
VALUES(6, 20, SDO_LIST_TYPE(), SDO_LIST_TYPE(),
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(9,6, 21,14)));
-- F7
INSERT INTO city_data_face$ (face_id, boundary_edge_id,
island_edge_id_list, island_node_id_list, mbr_geometry)
VALUES(7, 10, SDO_LIST_TYPE(), SDO_LIST_TYPE(),
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(21,6, 35,14)));

```

Рисунок 2.14 – SQL-запросы для загрузки данных в таблицу
CITY_DATA_FACES\$

```

-- F8
INSERT INTO city_data_face$ (face_id, boundary_edge_id,
island_edge_id_list, island_node_id_list, mbr_geometry)
VALUES(8, 16, SDO_LIST_TYPE(), SDO_LIST_TYPE(),
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(35,6, 47,14)));
-- F9
INSERT INTO city_data_face$ (face_id, boundary_edge_id,
island_edge_id_list, island_node_id_list, mbr_geometry)
VALUES(9,26,SDO_LIST_TYPE(), SDO_LIST_TYPE(),
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(4,31, 7,34)));

```

Рисунок 2.15 – SQL-запросы для загрузки данных в таблицу CITY_DATA_FACES\$

Результаты запросов представлены на рисунках 2.16-2.17.

The screenshot shows the Oracle SQL Developer interface. The top window is the 'Query Builder' with a 'Worksheet' tab. It contains the following SQL code:

```

-- 2. Load topology data (node, edge, and face tables).
-- Use INSERT statements here instead of a bulk-load utility.
-- 2C. Insert data into <topology_name>_FACE$ table.
-- FO (id = -1, not 0)
INSERT INTO city_data_face$ (face_id, boundary_edge_id,
island_edge_id_list, island_node_id_list, mbr_geometry)
VALUES(-1, NULL, SDO_LIST_TYPE(-1, -2, 4, 6),
SDO_LIST_TYPE(), NULL);
-- F1
INSERT INTO city_data_face$ (face_id, boundary_edge_id,
island_edge_id_list, island_node_id_list, mbr_geometry)
VALUES(1, 1, SDO_LIST_TYPE(25, -26), SDO_LIST_TYPE(),
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(3,30, 15,38)));
-- F2
INSERT INTO city_data_face$ (face_id, boundary_edge_id,
island_edge_id_list, island_node_id_list, mbr_geometry)
VALUES(2, 2, SDO_LIST_TYPE(), SDO_LIST_TYPE(4),
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(4,31, 7,34)));

```

The bottom window is the 'Script Output' window, which shows the execution results:

```

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

```

The status bar at the bottom of the script output window indicates 'Task completed in 0.192 seconds'.

Рисунок 2.17 – Результат загрузки данных в таблицу CITY_DATA_FACES\$

FACE_ID	BOUNDARY_EDGE_ID	ISLAND_EDGE_ID_LIST	ISLAND_NODE_ID_LIST	MBR_GEOMETRY
1	-1	(null) MDSYS.SDO_LIST_TYPE(-1,-2,4,6)	MDSYS.SDO_LIST_TYPE()	(null)
2	1	1 MDSYS.SDO_LIST_TYPE(25,-26)	MDSYS.SDO_LIST_TYPE()	[MDSYS.SDO_GEOMETRY]
3	2	2 MDSYS.SDO_LIST_TYPE()	MDSYS.SDO_LIST_TYPE(4)	[MDSYS.SDO_GEOMETRY]
4	3	19 MDSYS.SDO_LIST_TYPE()	MDSYS.SDO_LIST_TYPE()	[MDSYS.SDO_GEOMETRY]
5	4	17 MDSYS.SDO_LIST_TYPE()	MDSYS.SDO_LIST_TYPE()	[MDSYS.SDO_GEOMETRY]
6	5	15 MDSYS.SDO_LIST_TYPE()	MDSYS.SDO_LIST_TYPE()	[MDSYS.SDO_GEOMETRY]
7	6	20 MDSYS.SDO_LIST_TYPE()	MDSYS.SDO_LIST_TYPE()	[MDSYS.SDO_GEOMETRY]
8	7	10 MDSYS.SDO_LIST_TYPE()	MDSYS.SDO_LIST_TYPE()	[MDSYS.SDO_GEOMETRY]
9	8	16 MDSYS.SDO_LIST_TYPE()	MDSYS.SDO_LIST_TYPE()	[MDSYS.SDO_GEOMETRY]
10	9	26 MDSYS.SDO_LIST_TYPE()	MDSYS.SDO_LIST_TYPE()	[MDSYS.SDO_GEOMETRY]

Рисунок 2.17 – Результат загрузки данных в таблицу
CITY_DATA_FACES\$

Далее следует создать таблицу класса для каждого типа геометрического слоя в топологии. Например, городская топология может включать в себя отдельные таблицы классов для земельных участков, улиц и дорожных знаков. SQL-запрос для этой операции представлен на рисунке 2.18

```
-- 3. Create feature tables.
CREATE TABLE land_parcel ( -- Land parcels (selected faces)
feature_name VARCHAR2(30) PRIMARY KEY,
feature SDO_TOPO_GEOMETRY);
CREATE TABLE city_streets ( -- City streets (selected edges)
feature_name VARCHAR2(30) PRIMARY KEY,
feature SDO_TOPO_GEOMETRY);
CREATE TABLE traffic_signs ( -- Traffic signs (selected nodes)
feature_name VARCHAR2(30) PRIMARY KEY,
feature SDO_TOPO_GEOMETRY);
```

Рисунок 2.18 – SQL-запрос для создания таблиц классов

Результат запроса представлен на рисунке 2.19

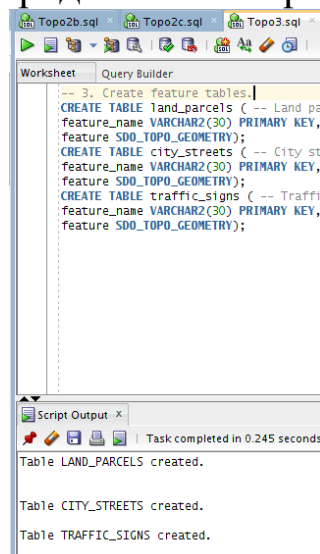


Рисунок 2.19 – Результат создания таблиц классов

Каждую таблицу класса необходимо привязать к топологии.

Для этого используется процедура

SDO_TOPO.ADD_TOPO_GEOMETRY_LAYER. SQL-запрос для данной операции приведен на рисунке 2.20.

```
-- 4. Associate feature tables with the topology.
-- Add the three topology geometry layers to the CITY_DATA topology.
-- Any order is OK.

EXECUTE SDO_TOPO.ADD_TOPO_GEOMETRY_LAYER('CITY_DATA',
'LAND_PARCELS', 'FEATURE', 'POLYGON');
EXECUTE SDO_TOPO.ADD_TOPO_GEOMETRY_LAYER('CITY_DATA',
'TRAFFIC_SIGNS', 'FEATURE', 'POINT');
EXECUTE SDO_TOPO.ADD_TOPO_GEOMETRY_LAYER('CITY_DATA',
'CITY_STREETS', 'FEATURE', 'LINE');
```

Рисунок 2.20 – SQL-запрос для привязки таблицы класса к топологии

В процессе выполнения этой процедуры создается таблица <название топологии>_RELATIONS\$. Результат выполнения приведен на рисунке 2.21.

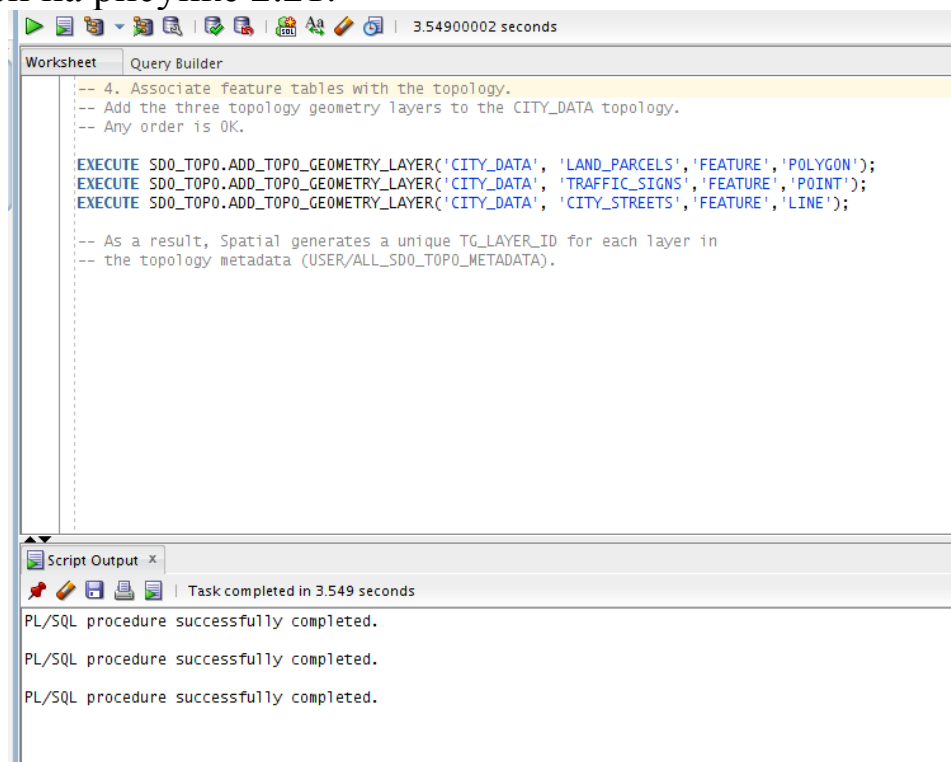


Рисунок 2.21 – Результат выполнения запроса привязки таблицы класса к топологии

Далее следует инициализировать топологические метаданные с помощью процедуры SDO_TOPO.INITIALIZE_METADATA. SQL-запрос для данной операции приведен на рисунке 2.22.

```
-- 5. Initialize topology metadata.
EXECUTE SDO_TOPO.INITIALIZE_METADATA('CITY_DATA');
```

Рисунок 2.22 – SQL-запрос для инициализации метаданных

Эта процедура создаст индексы для таблиц <название топологии>_EDGЕ\$, <название топологии>_NODE\$, <название топологии>_FACE и дополнительные B-tree индексы для таблиц <название топологии>_EDGЕ\$ и <название топологии>_NODE\$. Результат выполнения процедуры представлен на рисунке 2.23.

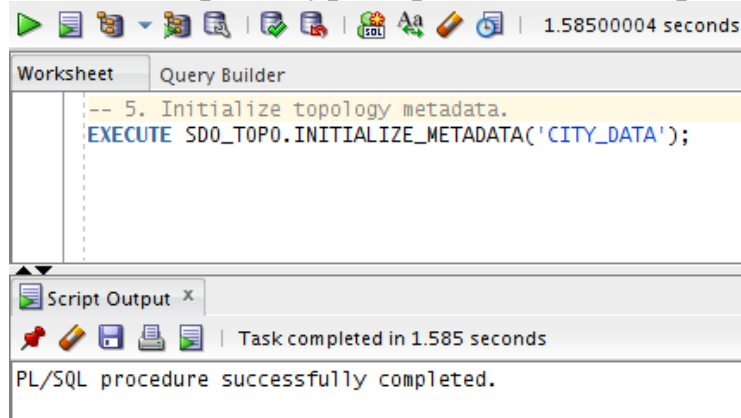


Рисунок 2.23 – Результат выполнения запроса инициализации метаданных

Следующим шагом будет загрузка таблиц классов в топологию с помощью конструктора SDO_TOPO_GEOMETRY. SQL-запрос для данной операции представлен на рисунках 2.24-2.29.

```
-- 6. Load feature tables using the SDO_TOPO_GEOMETRY constructor.
-- 6A. Load LAND_PARCELS table.

-- P1
INSERT INTO land_parcels VALUES ('P1', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
3, -- Topology geometry type (polygon/multipolygon)
1, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (3, 3), -- face_id = 3
SDO_TOPO_OBJECT (6, 3))) -- face_id = 6
);
-- P2
INSERT INTO land_parcels VALUES ('P2', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
3, -- Topology geometry type (polygon/multipolygon)
1, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (4, 3), -- face_id = 4
SDO_TOPO_OBJECT (7, 3))) -- face_id = 7
);
```

Рисунок 2.24 – SQL-запрос для загрузки таблицы LAND_PARCELS

```

-- P3
INSERT INTO land_parcels VALUES ('P3', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
3, -- Topology geometry type (polygon/multipolygon)
1, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (5, 3), -- face_id = 5
SDO_TOPO_OBJECT (8, 3))) -- face_id = 8
);
-- P4
INSERT INTO land_parcels VALUES ('P4', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
3, -- Topology geometry type (polygon/multipolygon)
1, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (2, 3))) -- face_id = 2
);
-- P5 (Includes F1, but not F9.)
INSERT INTO land_parcels VALUES ('P5', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
3, -- Topology geometry type (polygon/multipolygon)
1, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (1, 3))) -- face_id = 1
);

```

Рисунок 2.25 – SQL-запрос для загрузки таблицы LAND_PARCELS

```

-- 6. Load feature tables using the SDO_TOPO_GEOMETRY constructor.
-- 6B. Load TRAFFIC_SIGNS table.
-- S1
INSERT INTO traffic_signs VALUES ('S1', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
1, -- Topology geometry type (point)
2, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (14, 1))) -- node_id = 14
);
-- S2
INSERT INTO traffic_signs VALUES ('S2', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
1, -- Topology geometry type (point)
2, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (13, 1))) -- node_id = 13
);

```

Рисунок 2.26 – SQL-запрос для загрузки таблицы TRAFFIC_SIGNS

```

-- S3
INSERT INTO traffic_signs VALUES ('S3', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
1, -- Topology geometry type (point)
2, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (6, 1))) -- node_id = 6
);
-- S4
INSERT INTO traffic_signs VALUES ('S4', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
1, -- Topology geometry type (point)
2, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (4, 1))) -- node_id = 4
);

```

Рисунок 2.27 – SQL-запрос для загрузки таблицы TRAFFIC_SIGNS

```

-- 6. Load feature tables using the SDO_TOPO_GEOMETRY constructor.
-- 6C. Load CITY_STREETS table.
-- (Note: "R" in feature names is for "Road", because "S" is used for signs.)

-- R1
INSERT INTO city_streets VALUES ('R1', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
2, -- Topology geometry type (line string)
3, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (9, 2),
SDO_TOPO_OBJECT (-10, 2),
SDO_TOPO_OBJECT (11, 2))) -- edge_ids = 9, -10, 11
);
-- R2
INSERT INTO city_streets VALUES ('R2', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
2, -- Topology geometry type (line string)
3, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (4, 2),
SDO_TOPO_OBJECT (-5, 2))) -- edge_ids = 4, -5
);
-- R3
INSERT INTO city_streets VALUES ('R3', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
2, -- Topology geometry type (line string)
3, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (25, 2))) -- edge_id = 25
);

```

Рисунок 2.28 – SQL-запрос для загрузки таблицы CITY_STREETS

```
-- R4
INSERT INTO city_streets VALUES ('R4', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
2, -- Topology geometry type (line string)
3, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (3, 2))) -- edge_id = 3
);
```

Рисунок 2.29 – SQL-запрос для загрузки таблицы CITY_STREETS

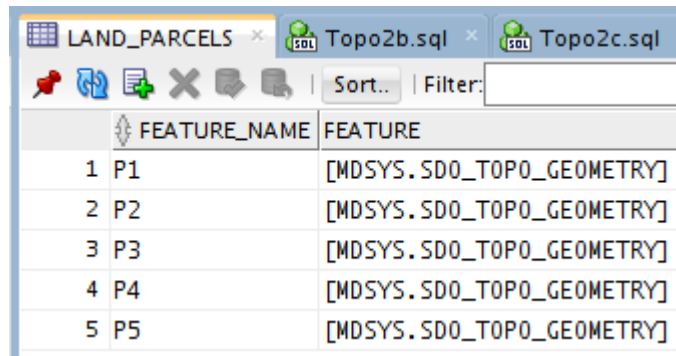
Результаты выполнения запросов приведены на рисунках 2.30-2.35.

The screenshot shows the Oracle SQL Developer interface. The top toolbar indicates the task is completed in 1.1590004 seconds. The main window displays a SQL script in the Query Builder. The script includes comments explaining the SDO_TOPO_GEOMETRY constructor and lists features: Area features (land parcels) P1, P2, P3, P4, P5; Point features (traffic signs) S1, S2, S3, S4; and Linear features (roads/streets) R1, R2, R3, R4. A highlighted section shows the insertion of P1 and P2 into the LAND_PARCELS table. The Script Output window at the bottom shows the results of the execution: three rows inserted.

```
-- 6. Load feature tables using the SDO_TOPO_GEOMETRY constructor.
-- Each topology feature can consist of one or more objects (face, edge, node)
-- of an appropriate type. For example, a land parcel can consist of one face,
-- or two or more faces, as specified in the SDO_TOPO_OBJECT_ARRAY.
-- There are typically fewer features than there are faces, nodes, and edges.
-- In this example, the only features are these:
-- Area features (land parcels): P1, P2, P3, P4, P5
-- Point features (traffic signs): S1, S2, S3, S4
-- Linear features (roads/streets): R1, R2, R3, R4
-- 6A. Load LAND_PARCELS table.
-- P1
INSERT INTO land_parcel VALUES ('P1', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
3, -- Topology geometry type (polygon/multipolygon)
1, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (3, 3), -- face_id = 3
SDO_TOPO_OBJECT (6, 3))) -- face_id = 6
);
-- P2
INSERT INTO land_parcel VALUES ('P2', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
3, -- Topology geometry type (polygon/multipolygon)
1, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (4, 3), -- face_id = 4
```

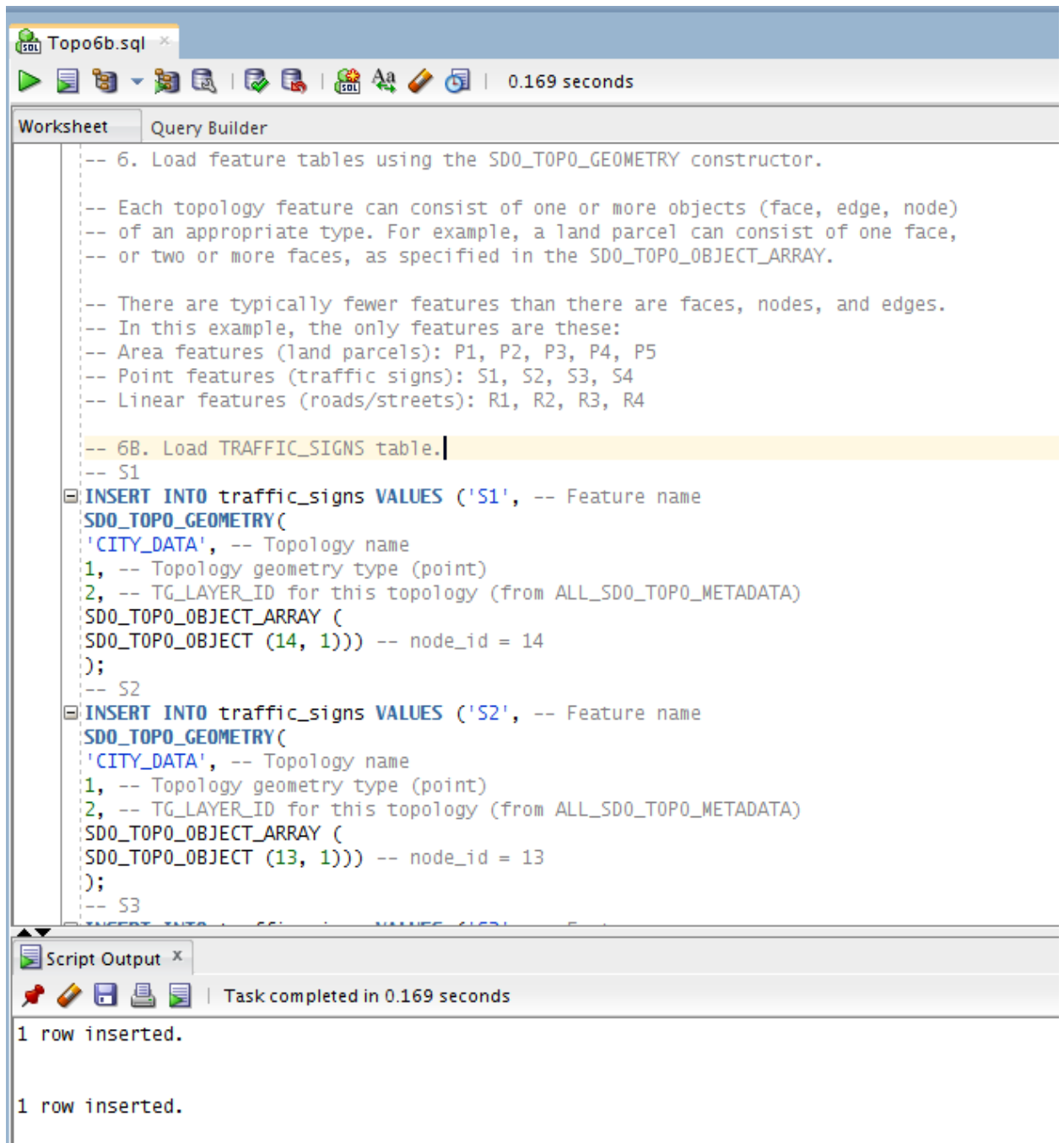
Script Output x
Task completed in 1.159 seconds
1 row inserted.
1 row inserted.
1 row inserted.
SQL Worksheet

Рисунок 2.30 – Результат загрузки таблицы LAND_PARCELS



	FEATURE_NAME	FEATURE
1	P1	[MDSYS.SDO_TOPO_GEOMETRY]
2	P2	[MDSYS.SDO_TOPO_GEOMETRY]
3	P3	[MDSYS.SDO_TOPO_GEOMETRY]
4	P4	[MDSYS.SDO_TOPO_GEOMETRY]
5	P5	[MDSYS.SDO_TOPO_GEOMETRY]

Рисунок 2.31 – Результат загрузки таблицы LAND_PARCELS



```

-- 6. Load feature tables using the SDO_TOPO_GEOMETRY constructor.
-- Each topology feature can consist of one or more objects (face, edge, node)
-- of an appropriate type. For example, a land parcel can consist of one face,
-- or two or more faces, as specified in the SDO_TOPO_OBJECT_ARRAY.

-- There are typically fewer features than there are faces, nodes, and edges.
-- In this example, the only features are these:
-- Area features (land parcels): P1, P2, P3, P4, P5
-- Point features (traffic signs): S1, S2, S3, S4
-- Linear features (roads/streets): R1, R2, R3, R4

-- 6B. Load TRAFFIC_SIGNS table.
-- S1
INSERT INTO traffic_signs VALUES ('S1', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
1, -- Topology geometry type (point)
2, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (14, 1))) -- node_id = 14
);
-- S2
INSERT INTO traffic_signs VALUES ('S2', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
1, -- Topology geometry type (point)
2, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (13, 1))) -- node_id = 13
);
-- S3
INSERT INTO traffic_signs VALUES ('S3', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
1, -- Topology geometry type (point)
2, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (12, 1))) -- node_id = 12
);

```

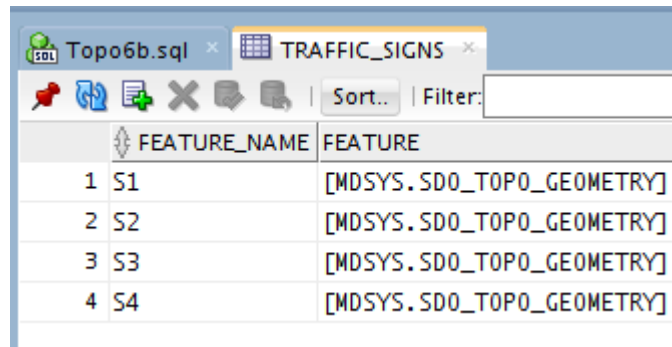
Script Output x

Task completed in 0.169 seconds

1 row inserted.

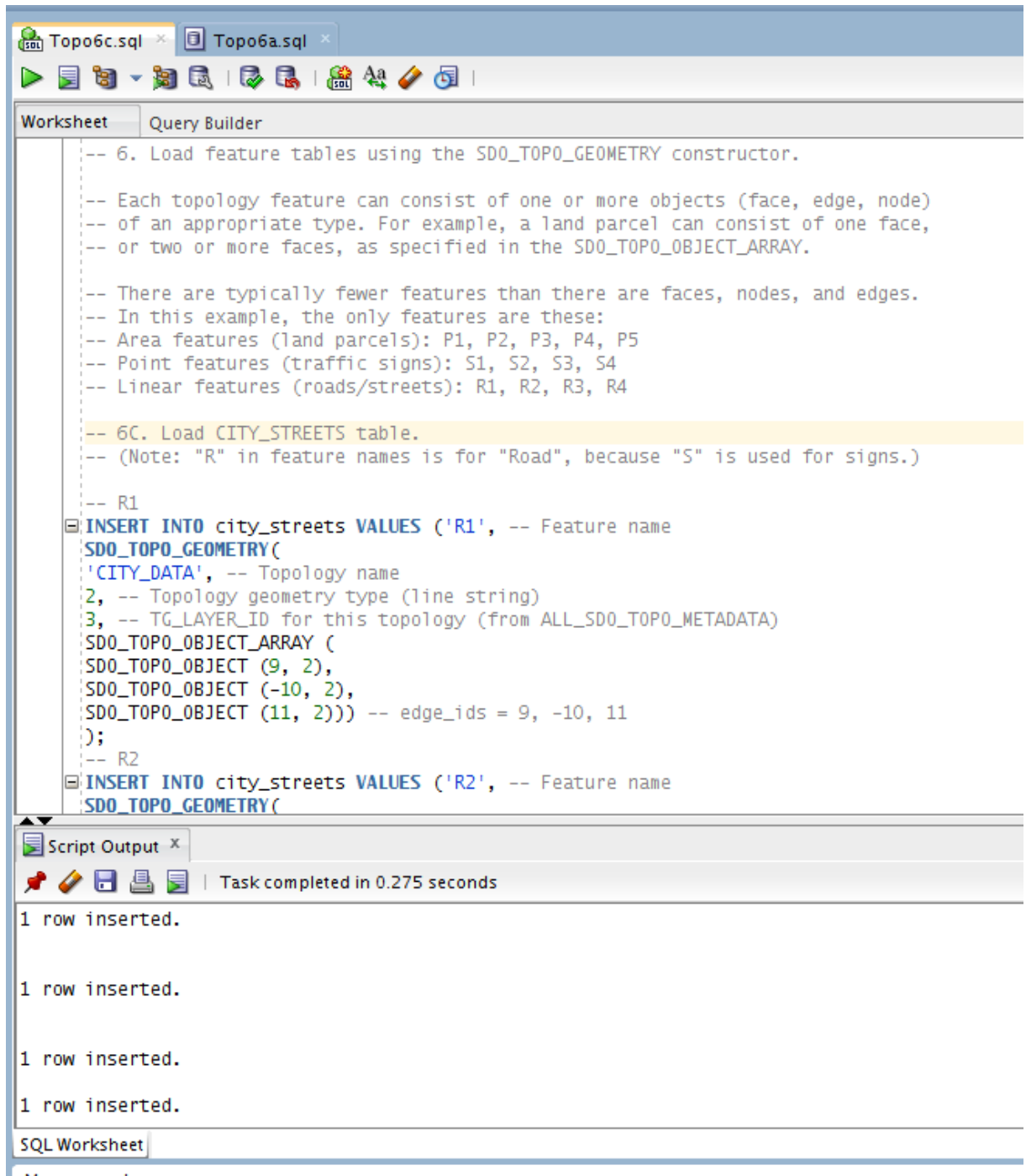
1 row inserted.

Рисунок 2.32 – Результат загрузки таблицы TRAFFIC_SIGNS



	FEATURE_NAME	FEATURE
1	S1	[MDSYS.SDO_TOPO_GEOMETRY]
2	S2	[MDSYS.SDO_TOPO_GEOMETRY]
3	S3	[MDSYS.SDO_TOPO_GEOMETRY]
4	S4	[MDSYS.SDO_TOPO_GEOMETRY]

Рисунок 2.33 – Результат загрузки таблицы TRAFFIC_SIGNS



Worksheet Query Builder

```
-- 6. Load feature tables using the SDO_TOPO_GEOMETRY constructor.
-- Each topology feature can consist of one or more objects (face, edge, node)
-- of an appropriate type. For example, a land parcel can consist of one face,
-- or two or more faces, as specified in the SDO_TOPO_OBJECT_ARRAY.
-- There are typically fewer features than there are faces, nodes, and edges.
-- In this example, the only features are these:
-- Area features (land parcels): P1, P2, P3, P4, P5
-- Point features (traffic signs): S1, S2, S3, S4
-- Linear features (roads/streets): R1, R2, R3, R4
-- 6C. Load CITY_STREETS table.
-- (Note: "R" in feature names is for "Road", because "S" is used for signs.)
-- R1
INSERT INTO city_streets VALUES ('R1', -- Feature name
SDO_TOPO_GEOMETRY(
'CITY_DATA', -- Topology name
2, -- Topology geometry type (line string)
3, -- TG_LAYER_ID for this topology (from ALL_SDO_TOPO_METADATA)
SDO_TOPO_OBJECT_ARRAY (
SDO_TOPO_OBJECT (9, 2),
SDO_TOPO_OBJECT (-10, 2),
SDO_TOPO_OBJECT (11, 2))) -- edge_ids = 9, -10, 11
);
-- R2
INSERT INTO city_streets VALUES ('R2', -- Feature name
SDO_TOPO_GEOMETRY(
```

Script Output x

Task completed in 0.275 seconds

```
1 row inserted.

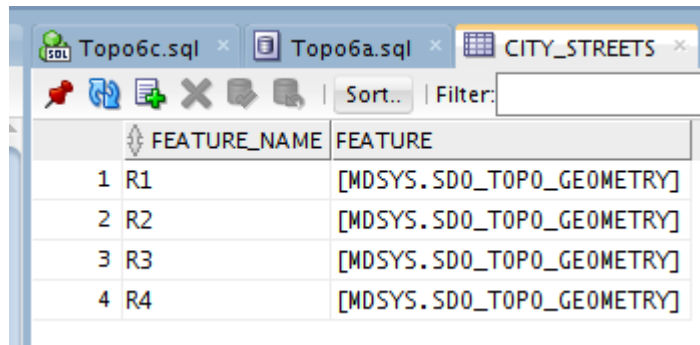
1 row inserted.

1 row inserted.

1 row inserted.
```

SQL Worksheet

Рисунок 2.34 – Результат загрузки таблицы CITY_STREETS



	FEATURE_NAME	FEATURE
1	R1	[MDSYS.SDO_TOPO_GEOMETRY]
2	R2	[MDSYS.SDO_TOPO_GEOMETRY]
3	R3	[MDSYS.SDO_TOPO_GEOMETRY]
4	R4	[MDSYS.SDO_TOPO_GEOMETRY]

Рисунок 2.35 – Результат загрузки таблицы CITY_STREETS

Теперь данные доступны для обработки и выполнения специфичных топологических запросов. Примеры запросов приведены на рисунках 2.36-2.42.

```
-- 7. Query the data.
SELECT a.feature_name, a.feature.tg_id, a.feature.get_geometry()
FROM land_parcel a;
```

Рисунок 2.36 – Запрос данных из таблицы LAND_PARCELS

```
-- 7. Query the data.
/* Window is city_streets */
SELECT a.feature_name, b.feature_name
FROM city_streets b,
land_parcel a
WHERE b.feature_name like 'R%' AND
sdo_anyinteract(a.feature, b.feature) = 'TRUE'
ORDER BY b.feature_name, a.feature_name;
```

Рисунок 2.37 – Запрос на принадлежность дорог и земельных участков

```
-- 7. Query the data.
-- Find all streets that have any interaction with land parcel P3.
-- (Should return only R1.)
SELECT c.feature_name FROM city_streets c, land_parcel l
WHERE l.feature_name = 'P3' AND
SDO_ANYINTERACT (c.feature, l.feature) = 'TRUE';
```

Рисунок 2.38 – Запрос на улицы, соприкасающиеся с участком P3

```
-- 7. Query the data.
-- Find all land parcels that have any interaction with traffic sign S1.
-- (Should return P1 and P2.)
SELECT l.feature_name FROM land_parcel l, traffic_signs t
WHERE t.feature_name = 'S1' AND
SDO_ANYINTERACT (l.feature, t.feature) = 'TRUE';
```

Рисунок 2.39 – Запрос на участки, соприкасающиеся с дорожным знаком S1 (стоит на границе участков)

```
-- 7. Query the data.
-- Get the geometry for land parcel P1.
SELECT l.feature_name, l.feature.get_geometry()
FROM land_parcel l WHERE l.feature_name = 'P1';
```

Рисунок 2.40 – Запрос геометрического объекта участка P1

```
-- 7. Query the data.
-- Get the boundary of face with face_id 3.
SELECT SDO_TOPO.GET_FACE_BOUNDARY('CITY_DATA', 3) FROM DUAL;
```

Рисунок 2.41 – Запрос объекта границ участка с face_id=3.

```
-- 7. Query the data.
-- Get the topological elements for land parcel P2.
-- CITY_DATA layer, land parcels (tg_layer_id = 1), parcel P2 (tg_id = 2)
SELECT SDO_TOPO.GET_TOPO_OBJECTS('CITY_DATA', 1, 2) FROM DUAL;
```

Рисунок 2.42 – Запрос топологических элементов, входящих в область P2

Результаты выполнения запросов приведены на рисунках 2.43-2.49.

Worksheet | Query Builder

```
-- 7. Query the data.
SELECT a.feature_name, a.feature.tg_id, a.feature.get_geometry()
FROM land_parcel a;
```

Script Output x | Task completed in 0.277 seconds

FEATURE_NAME	FEATURE.TG_ID
A. FEATURE. GET_GEOMETRY()	
P1	4
MDSYS.SDO_GEOMETRY(2003, null, null, MDSYS.SDO_ELEM_INFO_ARRAY(1,1003,1), MDSYS.SDO_ORDINATE_ARRAY(9,14,9,6,21,6,21,14,21,22,9,22,9,14))	
P2	5
MDSYS.SDO_GEOMETRY(2003, null, null, MDSYS.SDO_ELEM_INFO_ARRAY(1,1003,1), MDSYS.SDO_ORDINATE_ARRAY(21,14,21,6,35,6,35,14,35,22,21,22,21,14))	
P3	6
A. FEATURE. GET_GEOMETRY()	
MDSYS.SDO_GEOMETRY(2003, null, null, MDSYS.SDO_ELEM_INFO_ARRAY(1,1003,1), MDSYS.SDO_ORDINATE_ARRAY(35,14,35,6,47,6,47,14,47,22,35,22,35,14))	
P4	7
MDSYS.SDO_GEOMETRY(2003, null, null, MDSYS.SDO_ELEM_INFO_ARRAY(1,1003,1), MDSYS.SDO_ORDINATE_ARRAY(17,40,17,30,25,30,31,30,31,40,17,40))	
P5	8
MDSYS.SDO_GEOMETRY(2003, null, null, MDSYS.SDO_ELEM_INFO_ARRAY(1,1003,1,13,2003,	
1), MDSYS.SDO_ORDINATE_ARRAY(3,38,3,30,8,30,16,30,16,38,3,38,4,31,4,34,7,34,7,31,4,31))	
FEATURE_NAME	FEATURE.TG_ID
A. FEATURE. GET_GEOMETRY()	

SQL Worksheet

Рисунок 2.43 – Результат запроса данных из таблицы LAND_PARCELS

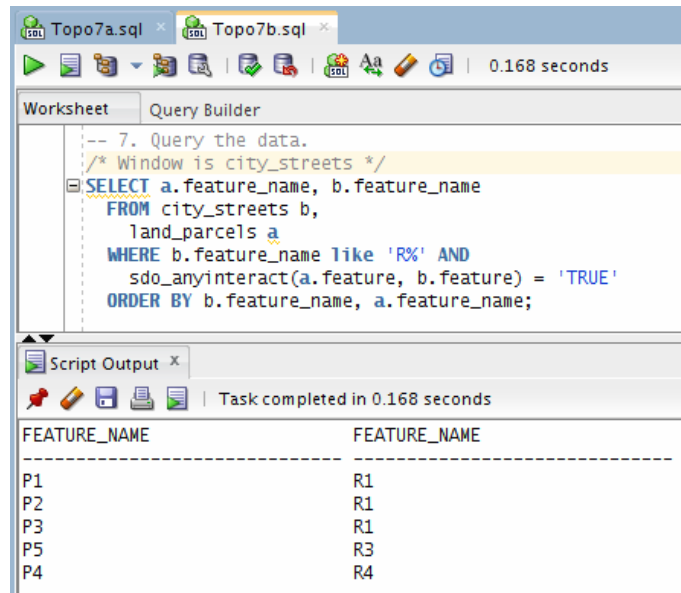


Рисунок 2.44 – Результат запроса на соотношение дорог и земельных участков

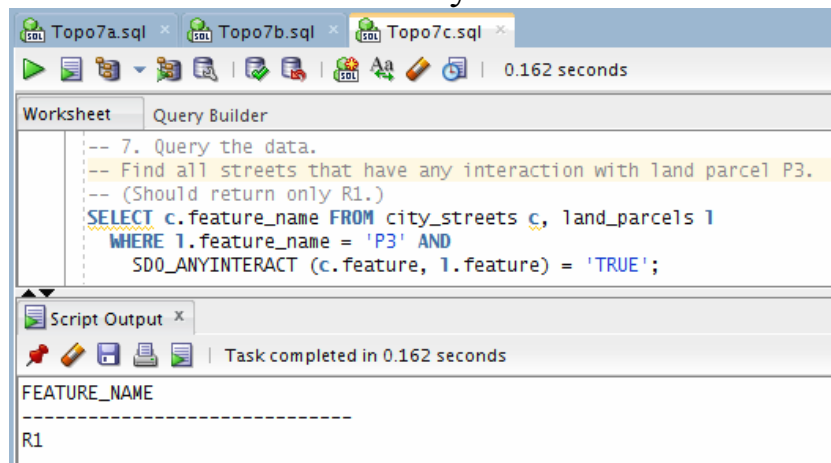


Рисунок 2.45 – Результат запроса на улицы, соприкасающиеся с участком P3

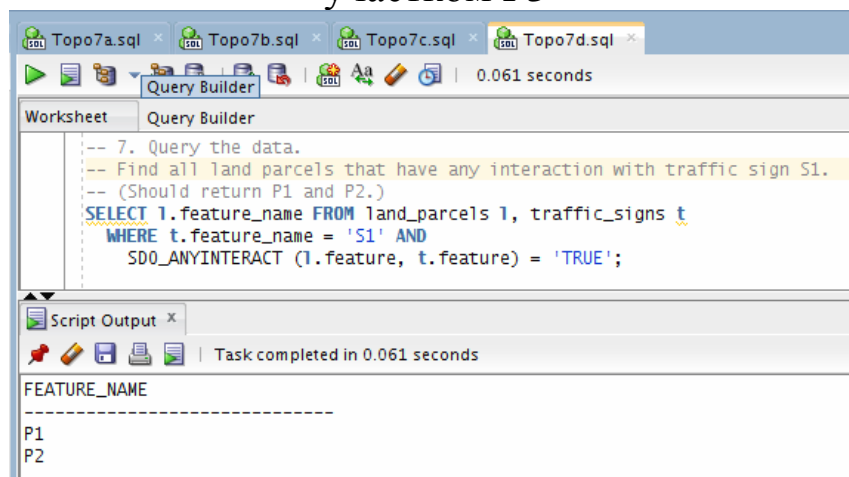


Рисунок 2.46 – на участки, соприкасающиеся с дорожным знаком S1

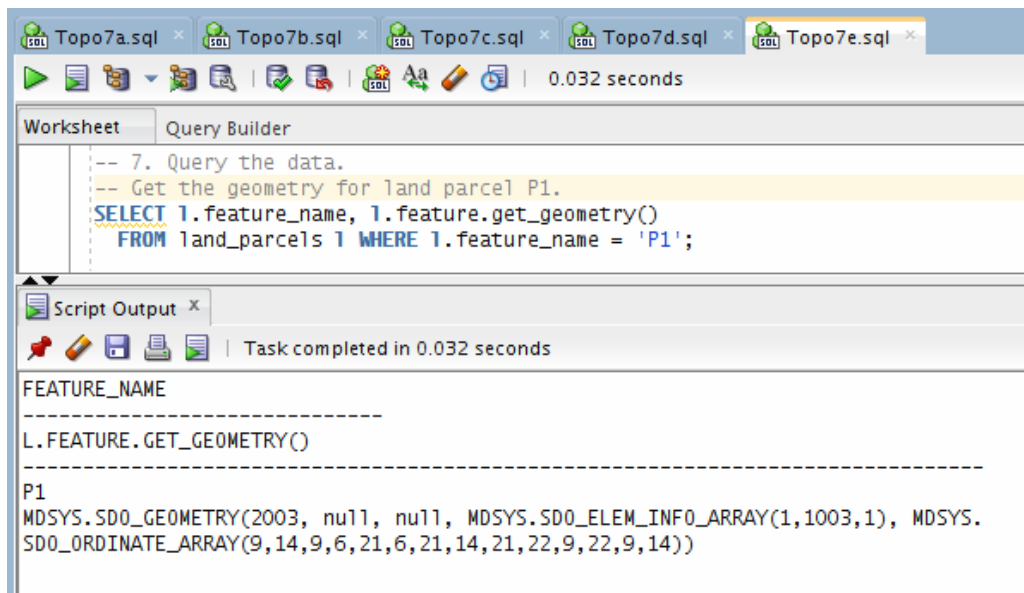


Рисунок 2.47 – Результат запроса геометрического объекта участка P1

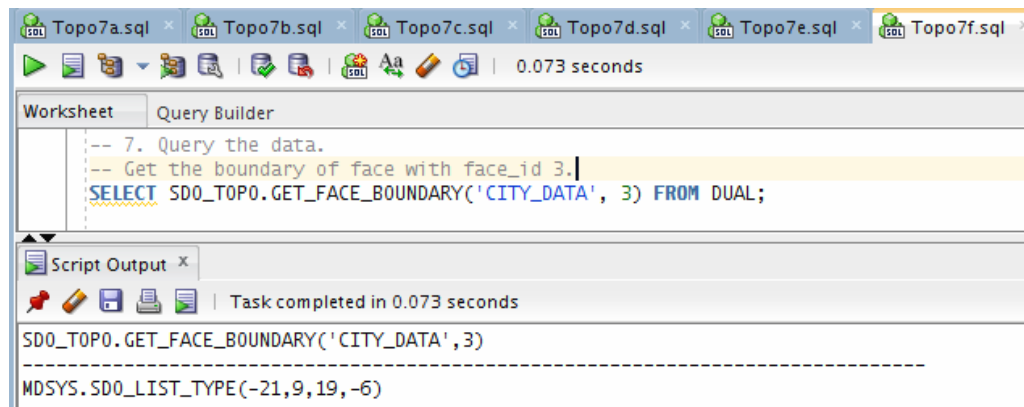


Рисунок 2.48 – Результат запроса объекта границ участка с face_id=3.

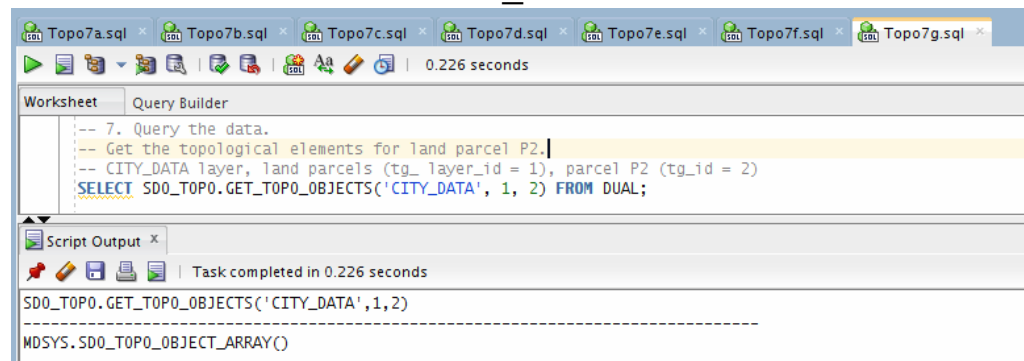


Рисунок 2.49 – Результат запроса топологических элементов, входящих в область P2

3 Содержание отчета по лабораторной работе

В сводный отчет по лабораторным работам в качестве одного из разделов или подразделов включаются скриншоты, показывающие содержимое таблиц при выполнении индивидуального задания.

4 Вопросы к защите лабораторной работы

1. Что такое топология?
2. Приведите пример топологических пространственных отношений.
3. Приведите пример пространственных отношений, не являющихся топологическими.
4. Назовите основные топологические элементы.
5. Приведите примеры объектов реального мира, которые можно представить основными топологическими элементами.
6. Что такое изолированные элементы?
7. Может ли на ребре быть изолированный узел?
8. Что такое топологические объекты?
9. Что содержится в топологических слоях?