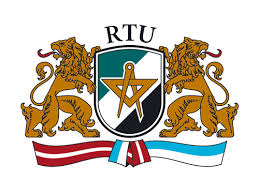
**RIGA TECHNICAL UNIVERSITY**

**Faculty of computer science and information technology**

**Master’s Programme in Business Informatics**



**Laboratory works**

**6-9**

in course „Advanced Data Bases”

**Author:** Laima Genava

**Student’s card number:** 061RIB034

**Checked:** Ainars Auzins

2013

Content

[1. Image for processing 3](#_Toc373936273)

[2. Layers 3](#_Toc373936274)

[3. Creation of tables 4](#_Toc373936275)

[4. Metadata 6](#_Toc373936276)

[5. Data insertion in tables 7](#_Toc373936277)

[5.1. Data insertion in tables with command INSERT 8](#_Toc373936278)

[5.2. Data insertion in tables with SQL\*Loader 15](#_Toc373936279)

[6. Creation of indexes 23](#_Toc373936280)

[7. Queries 24](#_Toc373936281)

[7.1. 2 queries with SDO\_FILTER 24](#_Toc373936282)

[7.2. 3 queries working with distance 26](#_Toc373936283)

[7.3. 3 queries finding objects with certain type of interactions 28](#_Toc373936284)

[8. Data visualization 30](#_Toc373936285)

# Image for processing

For laboratory works the author has drawn the image (Figure 1.) The image consists from buildings, trees, ponds, trails, flowerbeds, lanterns and forest.

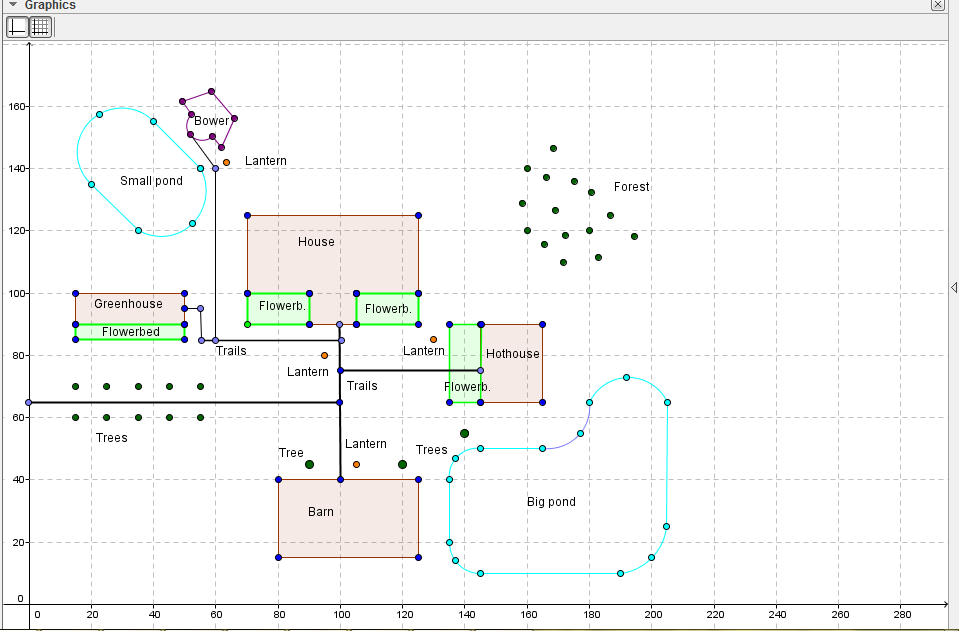


Figure 1. Image for laboratory works

# Layers

Figure 1 is divided into 7 layers. Each layer consists of several objects:

1. Trails which connects buildings.
2. Buildings: House, greenhouse, hothouse, barn, bower.
3. Trees: 13 trees.
4. Ponds: two ponds - Big and small.
5. Flowerbeds: 4 flowerbeds.
6. Lanterns – 4
7. Forest trees.

# Creation of tables

For each layer there was created table for data storage, in each table there is GEOM column with type MDSYS.SDO\_GEOMETRY. In this column is stored data about objects geometry. For all tables there are connections to other tables, where are stored additional data about object's attributes. Data base structure is shown in Figure 2. Data base consists of 13 tables, 7 tables stores graphical data for layers, and in 6 tables are stored additional data for objects’ attributes.

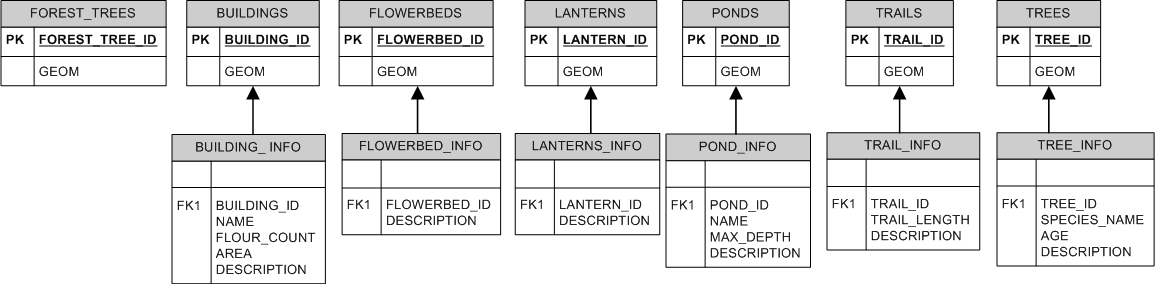


Figure 2. Data base structure

***Images from print screen for table creation are not given because of the scope of the work.***

* Buildings

*In the table Buildings the data is stored about all buildings in this country estate (Building layer)*

CREATE TABLE BUILDINGS

(

BUILDING\_ID NUMBER PRIMARY KEY

,GEOM MDSYS.SDO\_GEOMETRY

);

*In table Building\_info is stored information about building name, floor count, area and description of building.*

CREATE TABLE BUILDING\_INFO

(

BUILDING\_ID NUMBER NOT NULL REFERENCES BUILDINGS (BUILDING\_ID)

,NAME VARCHAR2(256) NOT NULL

,FLOOR\_COUNT NUMBER

,AREA NUMBER

,DESCRIPTION VARCHAR2(4000)

);

* Ponds

*In the table Ponds the data is stored about all ponds in this country estate (Ponds layer)*

CREATE TABLE PONDS

(

POND\_ID NUMBER PRIMARY KEY

,GEOM MDSYS.SDO\_GEOMETRY

);

*In table Pond\_info is stored information about pond’s name, maximal depth, and description of pond.*

CREATE TABLE POND\_INFO

(

POND\_ID NUMBER NOT NULL REFERENCES PONDS (POND\_ID)

,NAME VARCHAR2(256) NOT NULL

,MAX\_DEPTH NUMBER

,DESCRIPTION VARCHAR2(4000)

);

* Trees

*In the table Trees the data is stored about all trees in this country estate (Trees layer)*

CREATE TABLE TREES

(

TREE\_ID NUMBER PRIMARY KEY

,GEOM MDSYS.SDO\_GEOMETRY

);

*In table Trees\_info is stored information about trees’s species name, age, and description of tree.*

CREATE TABLE TREE\_INFO

(

TREE\_ID NUMBER NOT NULL REFERENCES TREES (TREE\_ID)

,SPECIES\_NAME VARCHAR2(256) NOT NULL

,AGE NUMBER

,DESCRIPTION VARCHAR2(4000)

);

* Trails

*In the table Trails the data is stored about all trails in this country estate (Trails layer)*

CREATE TABLE TRAILS

(

TRAIL\_ID NUMBER PRIMARY KEY

,GEOM MDSYS.SDO\_GEOMETRY

);

*In table Trail\_info is stored information about trails’ length, and description of trails.*

CREATE TABLE TRAIL\_INFO

(

TRAIL\_ID NUMBER NOT NULL REFERENCES TRAILS (TRAIL\_ID)

,TRAIL\_LENGTH VARCHAR2(256)

,DESCRIPTION VARCHAR2(4000)

);

* Flowerbeds

*In the table Flowerbeds the data is stored about all flowerbeds in this country estate (Flowerbeds layer)*

CREATE TABLE FLOWERBEDS

(

FLOWERBED\_ID NUMBER PRIMARY KEY

,GEOM MDSYS.SDO\_GEOMETRY

);

*In table Flowerbed\_info is stored information about description of flowerbeds.*

CREATE TABLE FLOWERBED\_INFO

(

FLOWERBED\_ID NUMBER NOT NULL REFERENCES FLOWERBEDS (FLOWERBED\_ID)

,DESCRIPTION VARCHAR2(4000)

);

* Lanterns

*In the table Lanterns the data is stored about all lanterns in this country estate (Lanterns layer)*

CREATE TABLE LANTERNS

(

LANTERN\_ID NUMBER PRIMARY KEY

,GEOM MDSYS.SDO\_GEOMETRY

);

*In table Lanterns\_info is stored information about description of lanterns.*

CREATE TABLE LANTERNS\_INFO

(

LANTERN\_ID NUMBER NOT NULL REFERENCES LANTERNS (LANTERN\_ID)

,DESCRIPTION VARCHAR2(4000)

);

* Forest

*In the table Forest\_trees the data is stored about forest in this country estate (Forest\_trees layer)*

CREATE TABLE FOREST\_TREES

(

FOREST\_TREE\_ID NUMBER PRIMARY KEY

,GEOM MDSYS.SDO\_GEOMETRY

);

# Metadata

After the tables of layers are created, there is a necessity to insert metadata. Metadata are stored in USER\_SDO\_GEOM\_METADATA. Metadata describes ordinate system of layers. This ordinate system has two dimensions X (must define Xmin and Xmax, and accuracy) and Y (must define Ymin and Ymax, and accuracy). Metadata was created for 7 tables - Buildings, Ponds, Trees, Trails, Flowerbeds, Lanterns and Forest\_trees.

***Images from print screen for metadata creation are not given because of the scope of the work.***

* *Buildings table*

INSERT INTO USER\_SDO\_GEOM\_METADATA (TABLE\_NAME,COLUMN\_NAME,DIMINFO,SRID)

VALUES ('BUILDINGS','GEOM',MDSYS.SDO\_DIM\_ARRAY(MDSYS.SDO\_DIM\_ELEMENT('X',0,250,1),MDSYS.SDO\_DIM\_ELEMENT('Y',0,250,1)),NULL);

* *Trees table*

INSERT INTO USER\_SDO\_GEOM\_METADATA (TABLE\_NAME,COLUMN\_NAME,DIMINFO,SRID)

VALUES ('TREES','GEOM',MDSYS.SDO\_DIM\_ARRAY(MDSYS.SDO\_DIM\_ELEMENT('X',0,250,1),MDSYS.SDO\_DIM\_ELEMENT('Y',0,250,1)),NULL);

* *Ponds table*

INSERT INTO USER\_SDO\_GEOM\_METADATA (TABLE\_NAME,COLUMN\_NAME,DIMINFO,SRID)

VALUES ('PONDS','GEOM',MDSYS.SDO\_DIM\_ARRAY(MDSYS.SDO\_DIM\_ELEMENT('X',0,250,1),MDSYS.SDO\_DIM\_ELEMENT('Y',0,250,1)),NULL);

* *Trails table*

INSERT INTO USER\_SDO\_GEOM\_METADATA (TABLE\_NAME,COLUMN\_NAME,DIMINFO,SRID)

VALUES ('TRAILS','GEOM',MDSYS.SDO\_DIM\_ARRAY(MDSYS.SDO\_DIM\_ELEMENT('X',0,250,1),MDSYS.SDO\_DIM\_ELEMENT('Y',0,250,1)),NULL);

* *Flowerbeds table*

INSERT INTO USER\_SDO\_GEOM\_METADATA (TABLE\_NAME,COLUMN\_NAME,DIMINFO,SRID)

VALUES ('FLOWERBEDS','GEOM',MDSYS.SDO\_DIM\_ARRAY(MDSYS.SDO\_DIM\_ELEMENT('X',0,250,1),MDSYS.SDO\_DIM\_ELEMENT('Y',0,250,1)),NULL);

* *Lanterns table*

INSERT INTO USER\_SDO\_GEOM\_METADATA (TABLE\_NAME,COLUMN\_NAME,DIMINFO,SRID)

VALUES ('LANTERNS','GEOM',MDSYS.SDO\_DIM\_ARRAY(MDSYS.SDO\_DIM\_ELEMENT('X',0,250,1),MDSYS.SDO\_DIM\_ELEMENT('Y',0,250,1)),NULL);

* *Forest\_trees table*

INSERT INTO USER\_SDO\_GEOM\_METADATA (TABLE\_NAME,COLUMN\_NAME,DIMINFO,SRID)

VALUES ('FOREST\_TREES','GEOM',MDSYS.SDO\_DIM\_ARRAY(MDSYS.SDO\_DIM\_ELEMENT('X',0,250,1),MDSYS.SDO\_DIM\_ELEMENT('Y',0,250,1)),NULL);

# Data insertion in tables

Data was inserted with Insert command and with SQL\*Loader.

## Data insertion in tables with command INSERT

***Images from print screen for data insertion in tables and creation of sequences are not given because of the scope of the work.***

*First was created sequences for all tables*

CREATE SEQUENCE BUILDINGS\_SEQ;

CREATE SEQUENCE TRAILS\_SEQ;

CREATE SEQUENCE TREES\_SEQ;

CREATE SEQUENCE PONDS\_SEQ;

CREATE SEQUENCE FLOWERBEDS\_SEQ;

CREATE SEQUENCE LANTERNS\_SEQ;

*Then I inserted data into tables*

***Buildings***

* *House.* MDSYS.SDO\_GEOMETRY(2003,NULL,NULL – shows that it is a polygon geometry type in 2D (two dimensions). MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,3,1) shows that it is a polygon with straight lines. MDSYS.SDO\_ORDINATE\_ARRAY shows ordinates.

INSERT INTO BUILDINGS (BUILDING\_ID,GEOM)

VALUES (BUILDINGS\_SEQ.NEXTVAL

,MDSYS.SDO\_GEOMETRY(

2003,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,3,1)

,MDSYS.SDO\_ORDINATE\_ARRAY(90,90, 105,90, 105,100, 125,100, 125,125, 70,125, 70,100, 90,100, 90,90))

);

* *Barn.* MDSYS.SDO\_GEOMETRY(2003,NULL,NULL – shows that it is a polygon geometry type in 2D (two dimensions). MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,3,3) shows that it is an optimized rectangle. MDSYS.SDO\_ORDINATE\_ARRAY shows ordinates for two points.

INSERT INTO BUILDINGS (BUILDING\_ID,GEOM)

VALUES (BUILDINGS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2003,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,3,3)

,MDSYS.SDO\_ORDINATE\_ARRAY(80,15, 125,40))

);

* *Hothouse.* MDSYS.SDO\_GEOMETRY(2003,NULL,NULL – shows that it is a polygon geometry type in 2D (two dimensions). MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,3,3) shows that it is an optimized rectangle. MDSYS.SDO\_ORDINATE\_ARRAY shows ordinates for two points.

INSERT INTO BUILDINGS (BUILDING\_ID,GEOM)

VALUES (BUILDINGS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2003,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,3,3)

,MDSYS.SDO\_ORDINATE\_ARRAY(145,65, 165,90))

);

* *Greenhouse.* MDSYS.SDO\_GEOMETRY(2003,NULL,NULL – shows that it is a polygon geometry type in 2D (two dimensions). MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,3,3) shows that it is an optimized rectangle. MDSYS.SDO\_ORDINATE\_ARRAY shows ordinates for two points.

INSERT INTO BUILDINGS (BUILDING\_ID,GEOM)

VALUES (BUILDINGS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2003,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,3,3)

,MDSYS.SDO\_ORDINATE\_ARRAY(15,90, 50,100))

);

* *Bower.* MDSYS.SDO\_GEOMETRY(2003,NULL,NULL – shows that it is a polygon geometry type in 2D (two dimensions). MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,5,2, 1,2,2, 5,2,1) shows that it is a compound polygon with straight and circular lines. MDSYS.SDO\_ORDINATE\_ARRAY shows ordinates.

INSERT INTO BUILDINGS (BUILDING\_ID,GEOM)

VALUES (BUILDINGS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2003,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,5,2, 1,2,2, 5,2,1)

,MDSYS.SDO\_ORDINATE\_ARRAY(59,150, 52,151, 52,158, 49,162, 58, 165, 66,156, 62,147, 59,150))

);

***Result:***

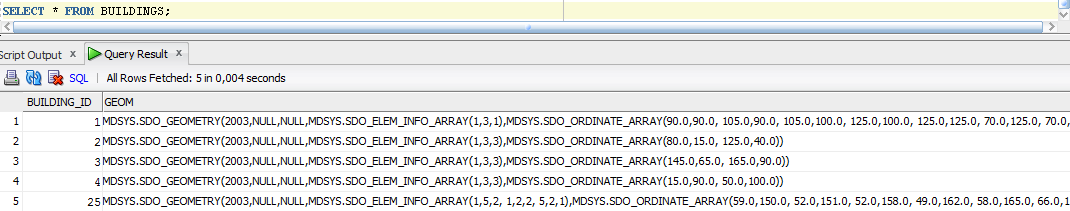


Figure 3. Content of table BUILDINGS

***Flowerbeds***

* *Flowerbeds.* MDSYS.SDO\_GEOMETRY(2003,NULL,NULL – shows that it is a polygon geometry type in 2D (two dimensions). MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,3,3) shows that it is an optimized rectangle. MDSYS.SDO\_ORDINATE\_ARRAY shows ordinates for two points.

INSERT INTO FLOWERBEDS (FLOWERBED\_ID,GEOM)

VALUES (FLOWERBEDS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2003,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,3,3)

,MDSYS.SDO\_ORDINATE\_ARRAY(15,85, 50,90))

);

INSERT INTO FLOWERBEDS (FLOWERBED\_ID,GEOM)

VALUES (FLOWERBEDS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2003,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,3,3)

,MDSYS.SDO\_ORDINATE\_ARRAY(70,90, 90,100))

);

INSERT INTO FLOWERBEDS (FLOWERBED\_ID,GEOM)

VALUES (FLOWERBEDS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2003,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,3,3)

,MDSYS.SDO\_ORDINATE\_ARRAY(105,90, 125,100))

);

INSERT INTO FLOWERBEDS (FLOWERBED\_ID,GEOM)

VALUES (FLOWERBEDS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2003,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,3,3)

,MDSYS.SDO\_ORDINATE\_ARRAY(135,65, 145,90))

);

***Result***

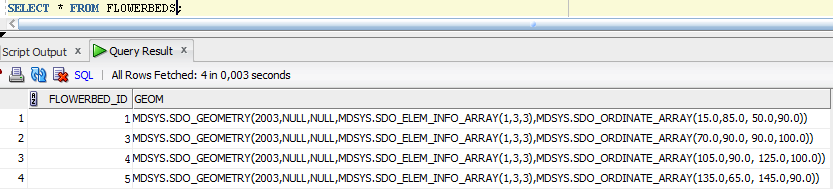


Figure 4. Content of table FLOWERBEDS

***Trees***

* *Trees.* MDSYS.SDO\_GEOMETRY(2001,NULL,– shows that it is a point geometry type in 2D (two dimensions). MDSYS.SDO\_POINT\_TYPE(15,70,NULL) shows ordinates for point.

INSERT INTO TREES (TREE\_ID,GEOM)

VALUES (TREES\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(15,70,NULL)

,NULL,NULL)

);

INSERT INTO TREES (TREE\_ID,GEOM)

VALUES (TREES\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(25,70,NULL)

,NULL,NULL)

);

INSERT INTO TREES (TREE\_ID,GEOM)

VALUES (TREES\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(35,70,NULL)

,NULL,NULL)

);

INSERT INTO TREES (TREE\_ID,GEOM)

VALUES (TREES\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(45,70,NULL)

,NULL,NULL)

);

INSERT INTO TREES (TREE\_ID,GEOM)

VALUES (TREES\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(55,70,NULL)

,NULL,NULL)

);

INSERT INTO TREES (TREE\_ID,GEOM)

VALUES (TREES\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(15,60,NULL)

,NULL,NULL)

);

INSERT INTO TREES (TREE\_ID,GEOM)

VALUES (TREES\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(25,60,NULL)

,NULL,NULL)

);

INSERT INTO TREES (TREE\_ID,GEOM)

VALUES (TREES\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(35,60,NULL)

,NULL,NULL)

);

INSERT INTO TREES (TREE\_ID,GEOM)

VALUES (TREES\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(45,60,NULL)

,NULL,NULL)

);

INSERT INTO TREES (TREE\_ID,GEOM)

VALUES (TREES\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(55,60,NULL)

,NULL,NULL)

);

INSERT INTO TREES (TREE\_ID,GEOM)

VALUES (TREES\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(140,55,NULL)

,NULL,NULL)

);

INSERT INTO TREES (TREE\_ID,GEOM)

VALUES (TREES\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(90,45,NULL)

,NULL,NULL)

);

INSERT INTO TREES (TREE\_ID,GEOM)

VALUES (TREES\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(120,45,NULL)

,NULL,NULL)

);

***Result***

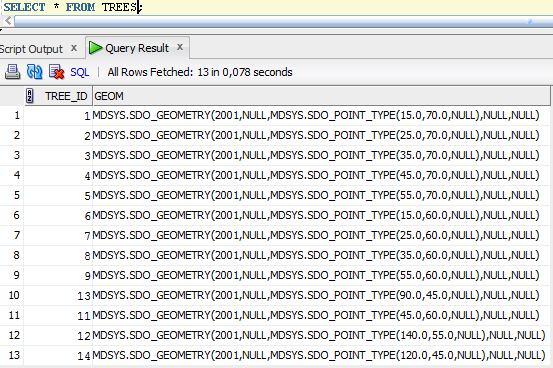
******

Figure 5. Content of table TREES

***Trails***

* *Trails.* MDSYS.SDO\_GEOMETRY(2002,NULL,NULL – shows that it is a line geometry type in 2D (two dimensions). MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,2,1) shows that it is a simple line element. MDSYS.SDO\_ORDINATE\_ARRAY shows ordinates for trails.

INSERT INTO TRAILS (TRAIL\_ID,GEOM)

VALUES (TRAILS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2002,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,2,1)

,MDSYS.SDO\_ORDINATE\_ARRAY(0,65, 100,65))

);

INSERT INTO TRAILS (TRAIL\_ID,GEOM)

VALUES (TRAILS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2002,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,2,1)

,MDSYS.SDO\_ORDINATE\_ARRAY(100,40, 100,90))

);

INSERT INTO TRAILS (TRAIL\_ID,GEOM)

VALUES (TRAILS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2002,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,2,1)

,MDSYS.SDO\_ORDINATE\_ARRAY(50,95, 55,95, 55,85, 100,85))

);

INSERT INTO TRAILS (TRAIL\_ID,GEOM)

VALUES (TRAILS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2002,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,2,1)

,MDSYS.SDO\_ORDINATE\_ARRAY(100,75, 145,75))

);

INSERT INTO TRAILS (TRAIL\_ID,GEOM)

VALUES (TRAILS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2002,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,2,1)

,MDSYS.SDO\_ORDINATE\_ARRAY(60,85, 60,140, 52,151))

);

***Result***

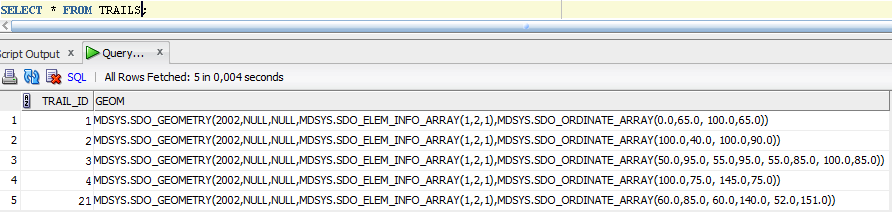
******

Figure 6. Content of table TRAILS

***Ponds***

* *Ponds.* MDSYS.SDO\_GEOMETRY(2003,NULL,NULL – shows that it is a polygon geometry type in 2D (two dimensions). MDSYS.SDO\_ELEM\_INFO\_ARRAY shows that it is a compound polygon with some vertices connected by straight line segments and some by circular arcs. MDSYS.SDO\_ORDINATE\_ARRAY shows ordinates for ponds.
* *Big pond*

INSERT INTO PONDS (POND\_ID,GEOM)

VALUES (PONDS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2003,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,5,8, 1,2,1, 3,2,2, 7,2,1, 9,2,2, 17,2,1, 19,2,2, 23,2,1, 25,2,2)

,MDSYS.SDO\_ORDINATE\_ARRAY(135,20, 135,40, 137,47, 145,50, 165,50, 177,55, 180,65, 192,73, 205,65, 205,25, 200,15, 190,10, 145,10, 137,14, 135,20))

);

* *Small pond*

INSERT INTO PONDS (POND\_ID,GEOM)

VALUES (PONDS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2003,NULL,NULL

,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,5,4, 1,2,1, 3,2,2, 7,2,1, 9,2,2)

,MDSYS.SDO\_ORDINATE\_ARRAY(35,120, 20,135, 22.5,157.5, 40,155, 55,140, 52.5,122.5, 35,120))

);

***Result***

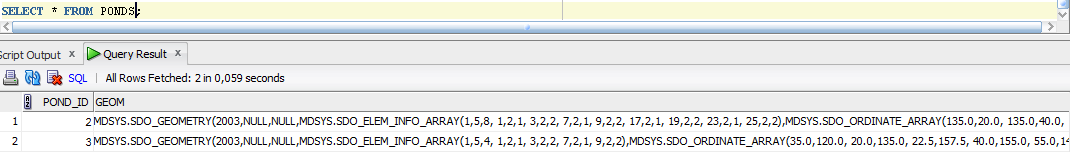


Figure 7. Content of table PONDS

***Lanterns***

* *Lanterns.* MDSYS.SDO\_GEOMETRY(2001,NULL,– shows that it is a point geometry type in 2D (two dimensions). MDSYS.SDO\_POINT\_TYPE(15,70,NULL) shows ordinates for point.

INSERT INTO LANTERNS (LANTERN\_ID,GEOM)

VALUES (LANTERNS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(130,85,NULL)

,NULL,NULL)

);

INSERT INTO LANTERNS (LANTERN\_ID,GEOM)

VALUES (LANTERNS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(95,80,NULL)

,NULL,NULL)

);

INSERT INTO LANTERNS (LANTERN\_ID,GEOM)

VALUES (LANTERNS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(105,45,NULL)

,NULL,NULL)

);

INSERT INTO LANTERNS (LANTERN\_ID,GEOM)

VALUES (LANTERNS\_SEQ.NEXTVAL,MDSYS.SDO\_GEOMETRY(2001,NULL

,MDSYS.SDO\_POINT\_TYPE(63,142,NULL)

,NULL,NULL)

);

***Result***

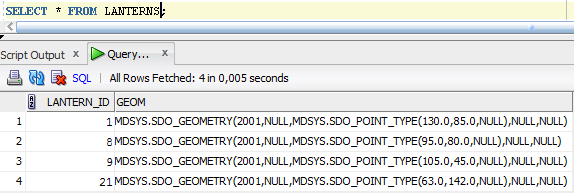
******

Figure 8. Content of table LANTERNS

## Data insertion in tables with SQL\*Loader

SQL\*Loader – is a program for data insertion into data base. With SQL \* Loader help the data will be inserted into tables FOREST\_TREES, BUILDING\_INFO, POND\_INFO, TREE\_INFO, TRAIL\_INFO, FLOWERBED\_INFO, LANTERNS\_INFO.

* ***FOREST\_TREES***

The content of control file forest\_trees.ctl is shown in figure 9.

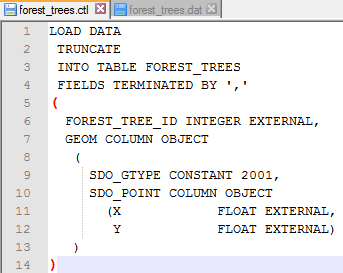


Figure 9. Control file

In the file Forest\_trees.dat are shown data that will be inserted in the table FOREST\_TREES.

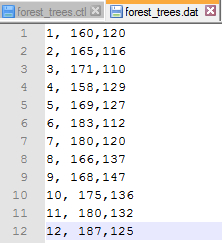


Figure 10. Data for table FOREST\_TREES

After the forest\_trees.dat and forest\_trees.ctl were created, then from the command line the SQL\* Loader was started.

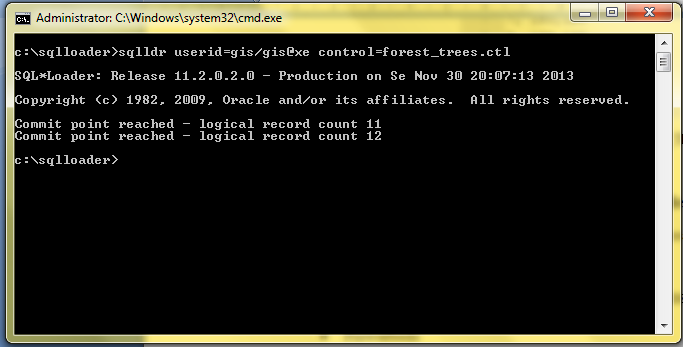


Figure 11. Data entry with SQL \* Loader

To check if all data is loaded correctly, I wrote a query:

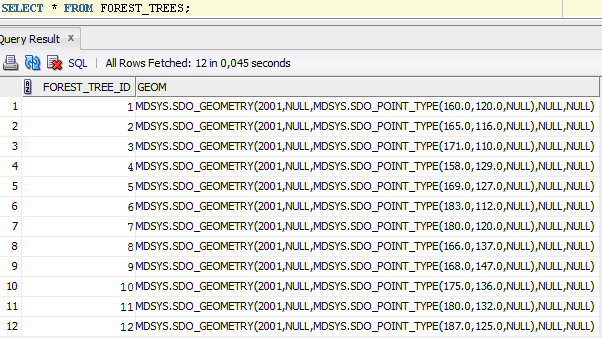


Figure 12. Content of table Forest\_trees

* ***BUILDING\_INFO***

The content of control file building\_info.ctl is shown in figure 13.

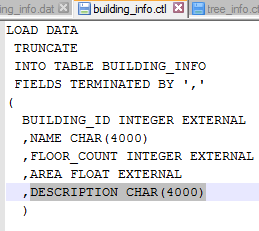


Figure 13. Control file building\_info.ctl

In the file building\_info.dat are shown data that will be inserted in the table BUILDING\_INFO.

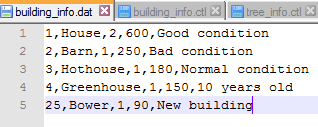


Figure 14. Data for table BUILDINGS\_INFO

After the building\_info.ctl and building\_info.dat were created, then from the command line the SQL\* Loader was started.

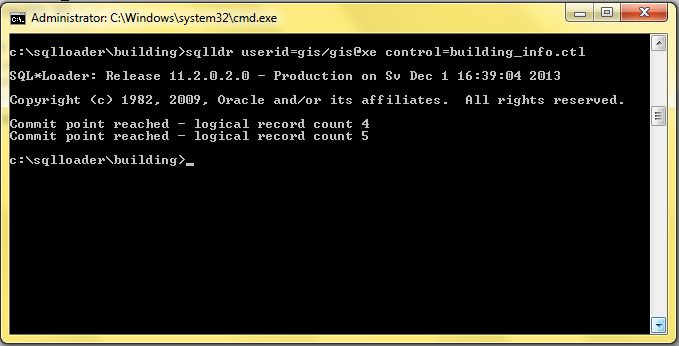


Figure 15. Data entry with SQL \* Loader

To check if all data is loaded correctly, I wrote a query:

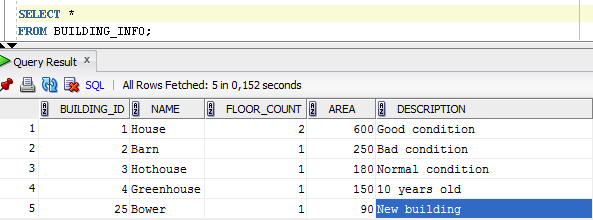


Figure 16. Content of table BUILDING\_INFO

* ***LANTERS\_INFO***

The content of control file lanterns\_info.ctl is shown in figure 17.

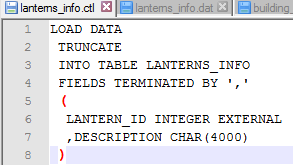


Figure 17. Control file lanterns\_info.ctl

In the file lanterns\_info.dat are shown data that will be inserted in the table LANTERNS\_INFO.

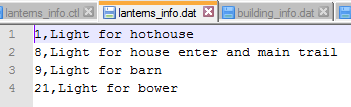


Figure 18. Data for table LANTERNS\_INFO

After the lanterns\_info.ctl and lanterns\_info.dat were created, then from the command line the SQL\* Loader was started.

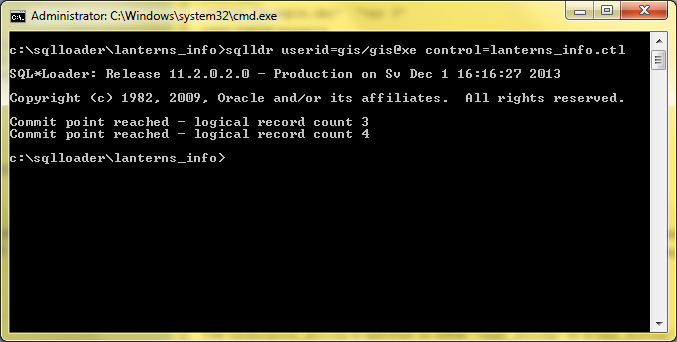


Figure 19. Data entry with SQL \* Loader

To check if all data is loaded correctly, I wrote a query:

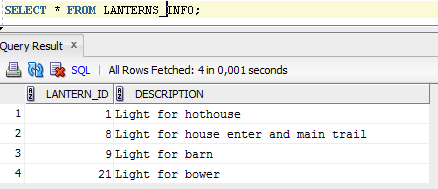


Figure 20. Content of table LANTERNS\_INFO

* ***FLOWERBED\_INFO***

The content of control file flowerbed\_info.ctl is shown in figure 21.

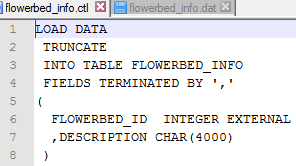


Figure 21. Control file flowerbed\_info.ctl

In the file flowerbed\_info.dat are shown data that will be inserted in the table FLOWERBED\_INFO.

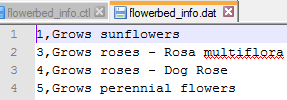


Figure 22. Data for table FLOWERBED\_INFO

After the flowerbed\_info.ctl and flowerbed \_info.dat were created, then from the command line the SQL\* Loader was started.

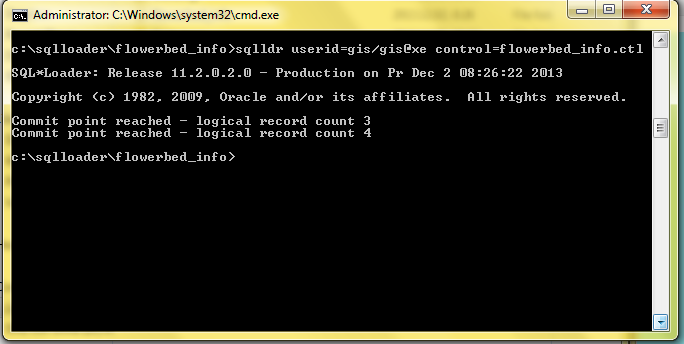


Figure 23. Data entry with SQL \* Loader

To check if all data is loaded correctly, I wrote a query:

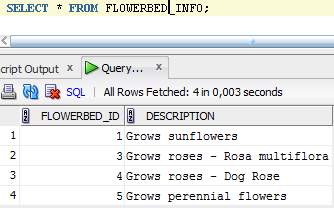


Figure 24. Content of table FLOWERBED\_INFO

* ***TRAIL\_INFO***

The content of control file trail\_info.ctl is shown in figure 25.

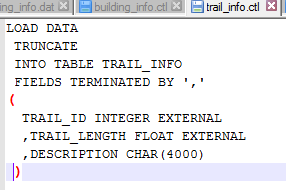


Figure 25. Control file trail\_info.ctl

In the file trail\_info.dat are shown data that will be inserted in the table TRAIL\_INFO.

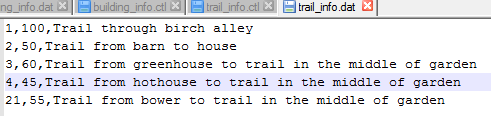


Figure 26. Data for table TRAIL\_INFO

After the trail\_info.ctl and trail \_info.dat were created, then from the command line the SQL\* Loader was started.

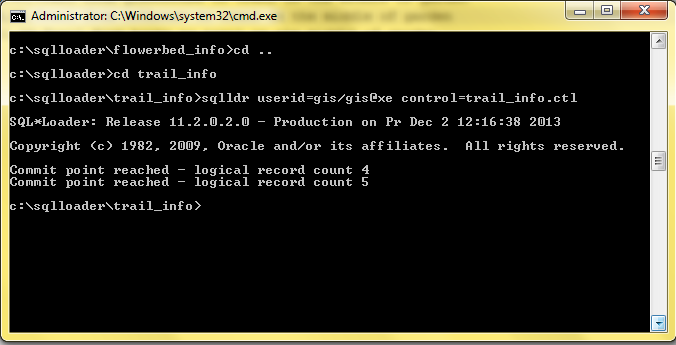


Figure 27. Data entry with SQL \* Loader

To check if all data is loaded correctly, I wrote a query:

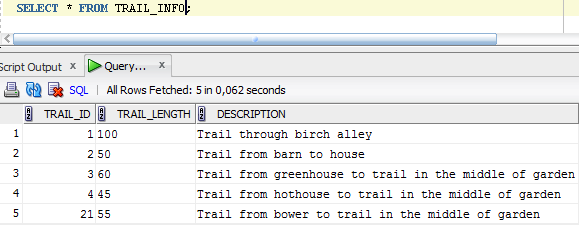


Figure 28. Content of table TRAIL\_INFO

* ***TREE\_INFO***

The content of control file tree\_info.ctl is shown in figure 29.

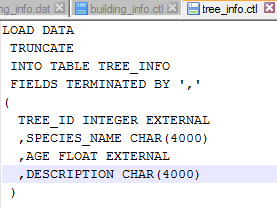


Figure 29. Control file tree\_info.ctl

In the file tree\_info.dat are shown data that will be inserted in the table TREE\_INFO.

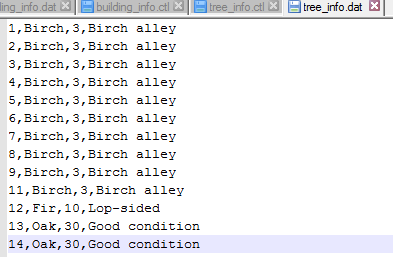


Figure 30. Data for table TREE\_INFO

After the tree\_info.ctl and tree \_info.dat were created, then from the command line the SQL\* Loader was started.

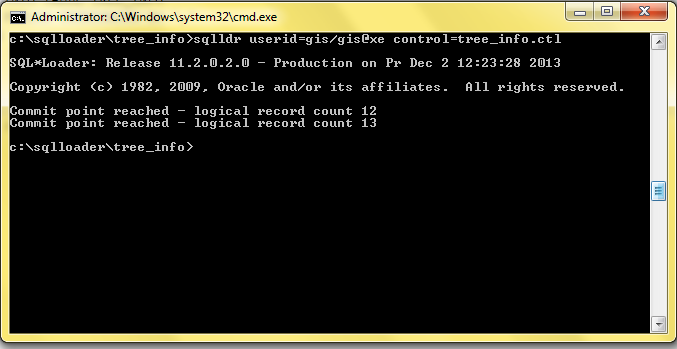


Figure 31. Data entry with SQL \* Loader

To check if all data is loaded correctly, I wrote a query:

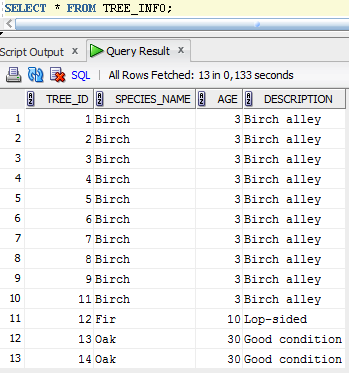


Figure 32. Content of table TREE\_INFO

* ***POND\_INFO***

The content of control file pond\_info.ctl is shown in figure 33.

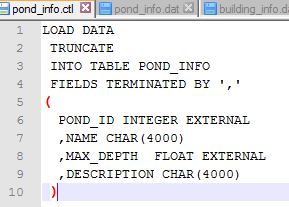


Figure 33. Control file pond\_info.ctl

In the file pond\_info.dat are shown data that will be inserted in the table POND\_INFO.



Figure 34. Data for table POND\_INFO

After the pond\_info.ctl and pond \_info.dat were created, then from the command line the SQL\* Loader was started.

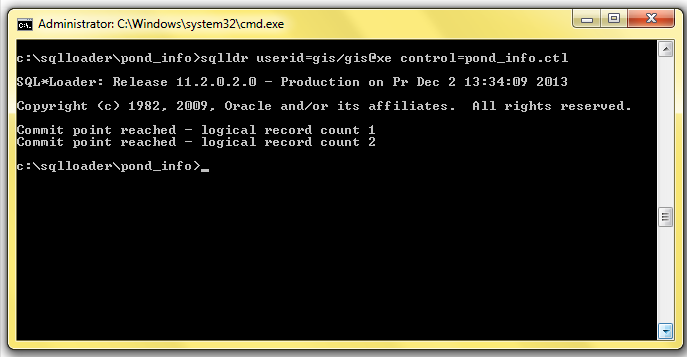


Figure 35. Data entry with SQL \* Loader

To check if all data is loaded correctly, I wrote a query:

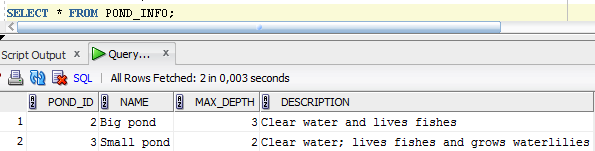


Figure 36. Content of table POND\_INFO

# Creation of indexes

***Images from print screen for creation of indexes are not given because of the scope of the work.***

To increase the search effectiveness, there is an option to create indexes:

* Buildings

CREATE INDEX BUILDINGS\_GEOM\_IDX ON BUILDINGS(GEOM) INDEXTYPE IS MDSYS.SPATIAL\_INDEX;

* Flowerbeds

CREATE INDEX FLOWERBEDS\_GEOM\_IDX ON FLOWERBEDS(GEOM) INDEXTYPE IS MDSYS.SPATIAL\_INDEX;

* Trails

CREATE INDEX TRAILS\_GEOM\_IDX ON TRAILS(GEOM) INDEXTYPE IS MDSYS.SPATIAL\_INDEX;

* Ponds

CREATE INDEX PONDS\_GEOM\_IDX ON PONDS(GEOM) INDEXTYPE IS MDSYS.SPATIAL\_INDEX;

* Trees

CREATE INDEX TREES\_GEOM\_IDX ON TREES(GEOM) INDEXTYPE IS MDSYS.SPATIAL\_INDEX;

* Lanterns

CREATE INDEX LANTERNS\_GEOM\_IDX ON LANTERNS(GEOM) INDEXTYPE IS MDSYS.SPATIAL\_INDEX;

* Forest\_trees

CREATE INDEX FOREST\_TREES\_GEOM\_IDX ON FOREST\_TREES(GEOM) INDEXTYPE IS MDSYS.SPATIAL\_INDEX;

*Created Indexes’ metadata:*

SELECT INDEX\_NAME, TABLE\_NAME, COLUMN\_NAME, SDO\_INDEX\_TYPE

FROM ALL\_SDO\_INDEX\_INFO;

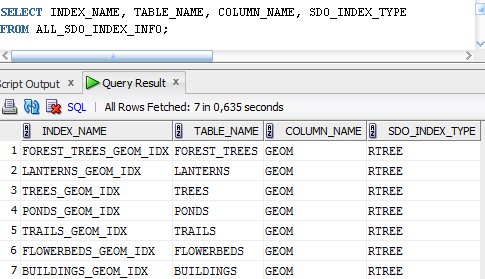


Figure 37. Indexes metadata

# Queries

## 2 queries with SDO\_FILTER

***1 using one layer and constant query window***

Select objects from building layer in ordinates (1,1; 1,103; 103,1; 103,103).

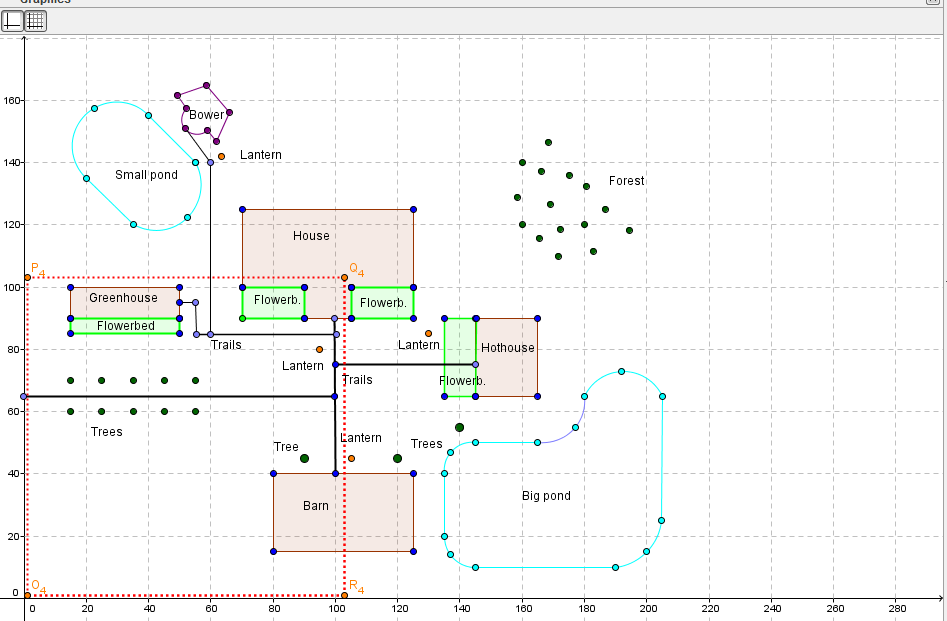


Figure 38. Objects from building layer

***SQL Query:***

SELECT B.BUILDING\_ID, BI.NAME

FROM BUILDINGS B

JOIN BUILDING\_INFO BI ON BI.BUILDING\_ID = B.BUILDING\_ID

WHERE SDO\_FILTER(B.GEOM, MDSYS.SDO\_GEOMETRY(2003,NULL,NULL,

MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,3),

MDSYS.SDO\_ORDINATE\_ARRAY(1,1,103,103)),

'QUERYTYPE=WINDOW') = 'TRUE';

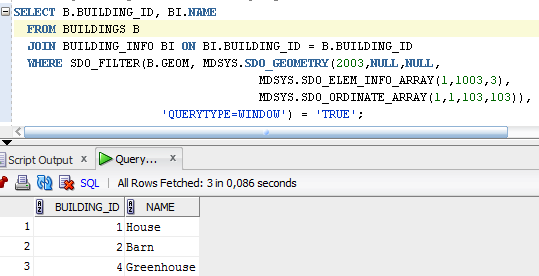


Figure 39. Query result

***1 using two layers - buildings and ponds layers.***

Select objects from building and pond layers in ordinates (1,1; 1,160; 103,1; 103,160).

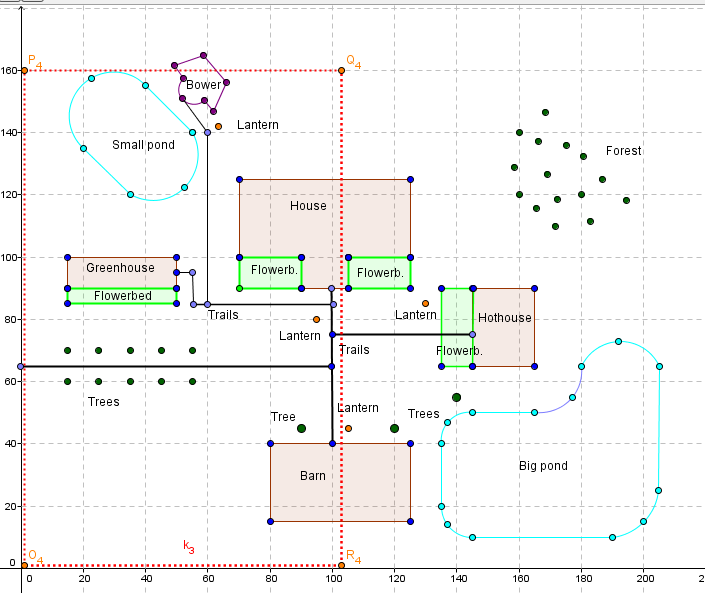


Figure 40. Objects from building and pond layers

***SQL Query:***

SELECT B.BUILDING\_ID AS ID\_NUMBER, BI.NAME AS OBJECT\_NAME

FROM BUILDINGS B

JOIN BUILDING\_INFO BI ON BI.BUILDING\_ID = B.BUILDING\_ID

WHERE SDO\_FILTER(B.GEOM, MDSYS.SDO\_GEOMETRY(2003,NULL,NULL,

MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,3),

MDSYS.SDO\_ORDINATE\_ARRAY(1,1,103,160)),

'QUERYTYPE=WINDOW') = 'TRUE'

UNION SELECT P.POND\_ID AS ID\_NUMBER, PI.NAME AS OBJECT\_NAME

FROM PONDS P

JOIN POND\_INFO PI ON PI.POND\_ID = P.POND\_ID

WHERE SDO\_FILTER(P.GEOM, MDSYS.SDO\_GEOMETRY(2003,NULL,NULL,

MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,3),

MDSYS.SDO\_ORDINATE\_ARRAY(1,1,103,160)),

'QUERYTYPE=WINDOW') = 'TRUE';

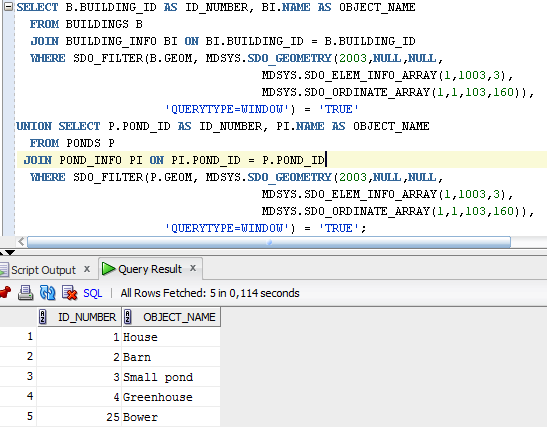


Figure 41. Query result

## 3 queries working with distance

***Finding the closest neighbors*** –Find and order buildings by distance. The point from which the distance must be measured is (70,50)

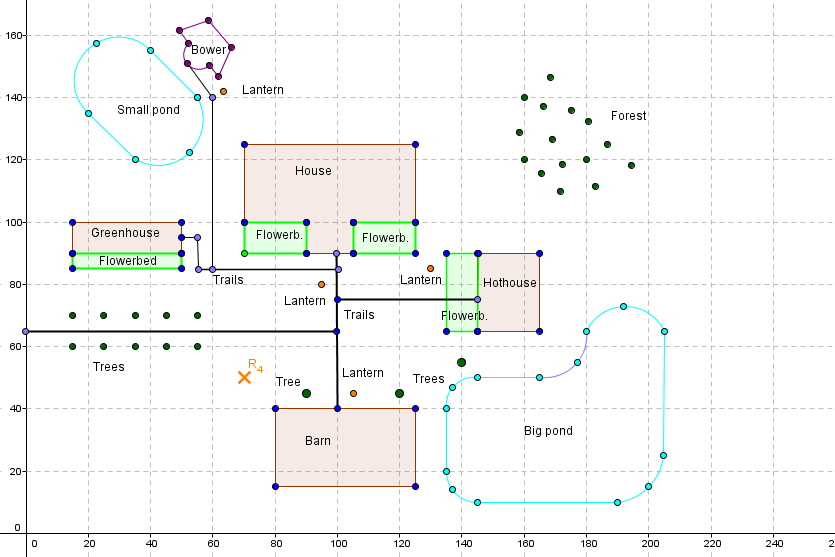
******

Figure 42. R4 measure distance to closest buildings

SELECT

B.BUILDING\_ID, BI.NAME,

ROUND(MDSYS.SDO\_NN\_DISTANCE(1),1) DIST

FROM BUILDINGS B

JOIN BUILDING\_INFO BI ON BI.BUILDING\_ID = B.BUILDING\_ID

WHERE SDO\_NN(B.GEOM, MDSYS.SDO\_GEOMETRY(2001, NULL,

MDSYS.SDO\_POINT\_TYPE(70,50,NULL), NULL, NULL),

'SDO\_NUM\_RES=5', 1) = 'TRUE' ORDER BY DIST;

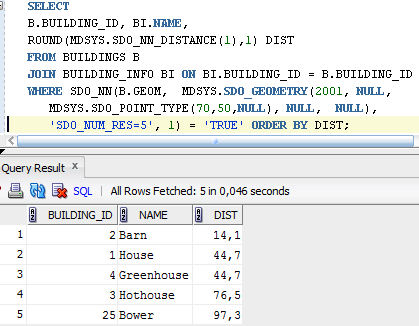


Figure 43. Query result

***Finding objects within distance*** - find objects from coordinates (100,130;100,140;120,140;120,130) within distance 60.

SELECT P.POND\_ID, PI.NAME

FROM PONDS P

JOIN POND\_INFO PI ON P.POND\_ID = PI.POND\_ID

WHERE SDO\_WITHIN\_DISTANCE(P.GEOM, MDSYS.SDO\_GEOMETRY(2003,NULL,NULL,

MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,3),

MDSYS.SDO\_ORDINATE\_ARRAY(100,130,120,140)),

'DISTANCE = 60') = 'TRUE';

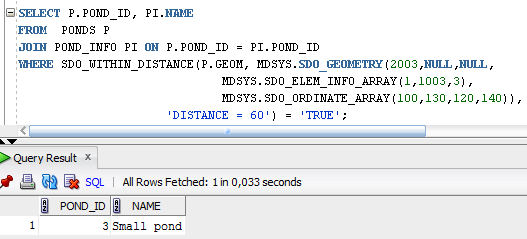


Figure 44. Query result

***Finding distance to the objects found in the previous query***

SELECT P.POND\_ID, PI.NAME,

ROUND(MDSYS.SDO\_NN\_DISTANCE(1),1) DIST

FROM PONDS P

JOIN POND\_INFO PI ON PI.POND\_ID = P.POND\_ID

WHERE SDO\_NN(P.GEOM, MDSYS.SDO\_GEOMETRY(2003,NULL,NULL,

MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,3),

MDSYS.SDO\_ORDINATE\_ARRAY(100,130,120,140)),

'SDO\_NUM\_RES=1', 1) = 'TRUE' ;

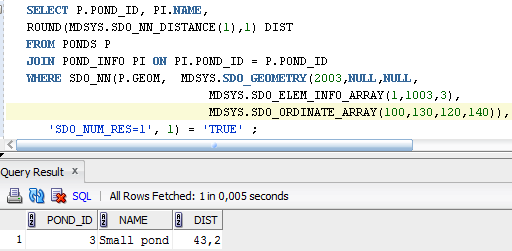


Figure 45. Query result

## 3 queries finding objects with certain type of interactions

* ***Find any interaction in ordinates (85,60; 85,70; 110,60; 110,70) – yellow rectangle.***

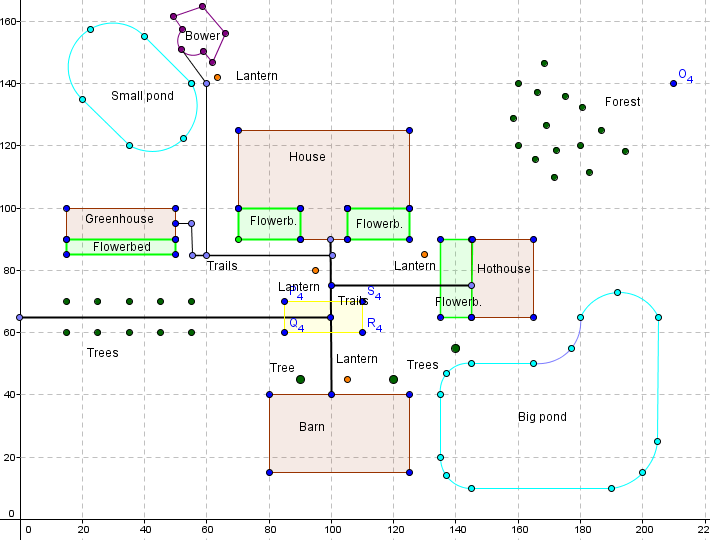
-

Figure 46. Yellow rectangle and objects from TRAILS table that interacts

SELECT T.TRAIL\_ID, TI.TRAIL\_LENGTH

FROM TRAILS T

JOIN TRAIL\_INFO TI ON T.TRAIL\_ID= TI.TRAIL\_ID

WHERE SDO\_ANYINTERACT(T.GEOM, SDO\_GEOMETRY(2003, NULL, NULL,

SDO\_ELEM\_INFO\_ARRAY(1,1003,3), SDO\_ORDINATE\_ARRAY(85,60, 110,70)) ) = 'TRUE';

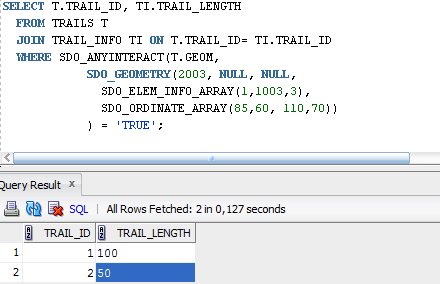
-

Figure 47. Query result

* ***Select objects from building layer in ordinates (1,1; 1,160; 103,1; 103,160).***

SELECT B.BUILDING\_ID AS ID\_NUMBER, BI.NAME AS OBJECT\_NAME

FROM BUILDINGS B

JOIN BUILDING\_INFO BI ON BI.BUILDING\_ID = B.BUILDING\_ID

WHERE SDO\_OVERLAPS(B.GEOM, MDSYS.SDO\_GEOMETRY(2003,NULL,NULL,

MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,3),

MDSYS.SDO\_ORDINATE\_ARRAY(1,1,103,160))

) = 'TRUE';

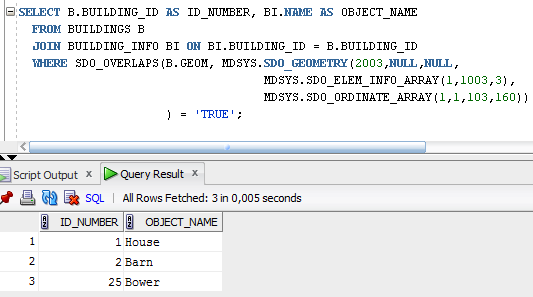


Figure 48. Query result

* ***Select objects from buildings and flowerbeds layers, which interacts.***

SELECT B.BUILDING\_ID, BI.NAME, F.FLOWERBED\_ID, FI.DESCRIPTION

FROM BUILDINGS B

JOIN BUILDING\_INFO BI ON BI.BUILDING\_ID = B.BUILDING\_ID

JOIN FLOWERBEDS F ON 1=1

JOIN FLOWERBED\_INFO FI ON FI.FLOWERBED\_ID = F.FLOWERBED\_ID

WHERE SDO\_RELATE(B.GEOM,F.GEOM,'MASK = ANYINTERACT') = 'TRUE';

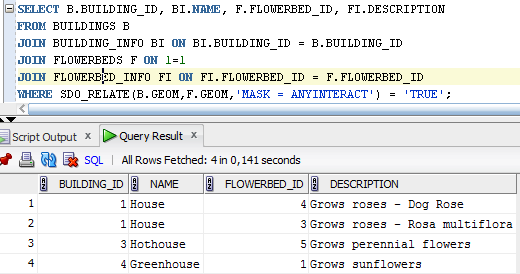


Figure 49. Query result

# Data visualization

For data visualization the Author of this work used GeoRaptor. Each layer was added to tool with the steps, which are shown in figure 50.

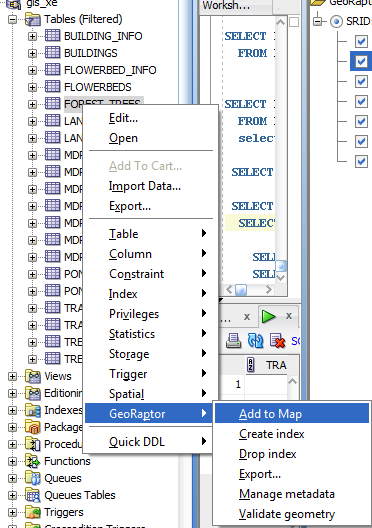


Figure 50. Add new layer to visualization

All layers are shown in figure 51.

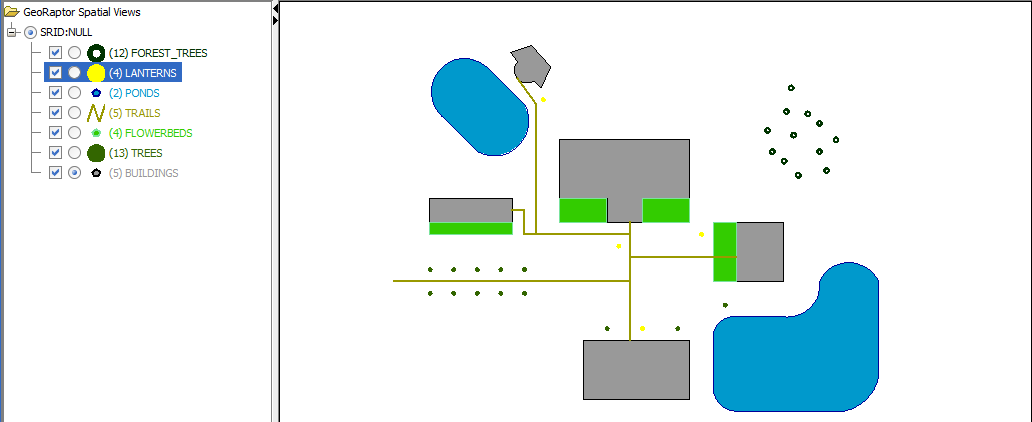


Figure 51. Layers visualization with GeoRaptor

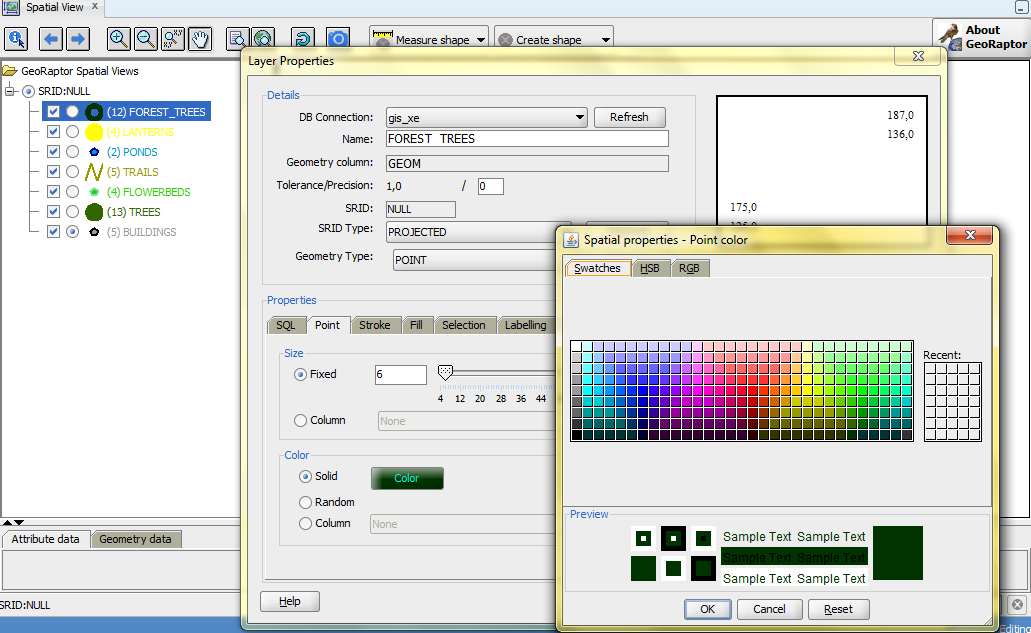
The colors and line thickness can be changed in the way that is shown in figure 52. 

Figure 52. Change properties for objects

This program allows visualizations of only selected layers too, as it is shown in figure 53.

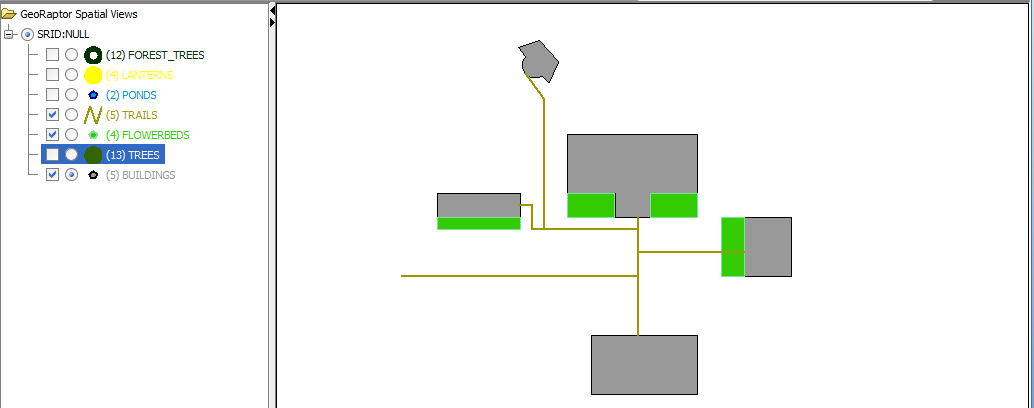


Figure 53. Selected layers visualization

The user has many more options what to do in *GeoRaptor*, but because of the scope of work, the author doesn’t describe these features.