Рассмотрим модуль пространственных зависимостей, устанавливаемый «поверх» существующей структуры данных в СУБД. Данный модуль можно разделить на две части:

1. Таблицы содержащие в себе данные, являющиеся метаданными, т.е. информация о других таблицах.
2. Таблицы содержащие в себе данные необходимые для проведения анализа и результатов выполнения анализа.

Перед тем как переходить к просмотру модуля пространственных зависимостей, рассмотрим некоторые шаблоны создания в Oracle необходимые для создания таблиц и их вспомогательных инструментах.

Создание простой таблицы в Oracle имеет следующий шаблон:

1. CREATE TABLE scheme\_name.table\_name
2. (
3. column\_name\_1 datatype [NOT NULL]
4. …
5. column\_name\_n datatype [NOT NULL]
6. table\_constraint
7. );

Где

1. scheme\_name – название схемы, в которой хотим создать таблицу
2. table\_name – название таблицы, которую хотим создать
3. column\_name\_1 … column\_name\_n – название столбцов таблицы.
4. datatype – тип данных
5. [NOT NULL] – не обязательный параметр. Необходим, чтобы кортеж в столбце всегда имел значение.
6. table\_constraint – ограничения таблицы.

Создание простой последовательности в Oracle:

1. CREATE SEQUENCE scheme\_name.sequence\_name
2. INCREMENT BY integer
3. START WITH integer
4. CACHE integer | NOCACHE

Где

1. Scheme\_name - название схемы, в которой хотим создать последовательность
2. Sequence\_name – название последовательности
3. INCREMENT BY – задается интервал между числами последовательности
4. START WITH – первое число последовательности
5. CACHE – кол-во значений последовательности, которую база данных предопределяет и сохраняет в памяти для более быстрого доступа.
6. NOCACHE – значения не предопределяются.

Шаблон создания простого триггера.

1. CREATE TRIGGER scheme\_name.trigger\_name
2. BEFORE | AFTER | INSTEAD OF
3. DELETE | INSERT | UPDATE ON table\_name
4. FOR EACH ROW
5. PL/SQL\_BLOCK

Где

1. Scheme\_name - название схемы
2. Trigger\_name – название триггера
3. BEFORE | AFTER | INSTEAD OF – база данных запускает триггер перед | после | вместо выполнением события
4. DELETE – база данных запускает триггер, когда оператор DELETE удаляет строку из таблицы
5. INSERT – база данных запускает триггер, когда оператор INSERT добавляет строку в таблицу
6. UPDATE – база данных запускает триггер, когда оператор UPDATE изменяет значение в таблице
7. table\_name – к какой таблице принадлежит триггер.
8. FOR EACH ROW – необязательный параметр. База данных запускает триггер для каждой строки, на которую воздействует оператор триггера
9. PL/SQL\_BLOCK – блок, который выполняет база данных при запуске триггера.

Рассмотрим более подробно таблицы, содержащие в себе метаданные. Так как объекты, между которыми могут быть пространственные зависимости, могут находится у разных пользователей, то была создана таблица «OWNER», содержащая информацию о пользователях:

1. owner\_id – первичный ключ таблицы «OWNER»
2. owner\_name – имя пользователя, в котором находятся объекты необходимые для анализа.

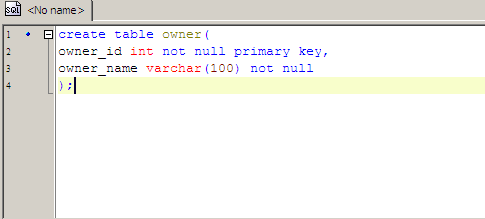


Рис 1.1 Таблица «OWNER»

Следующая таблица – это таблица «OBJECT\_TYPE». В данной таблицы хранится информация об объектах ГИС. Так как каждый объект ГИС в БД хранится в виде таблицы, то таблица «OBJECT\_TYPE» хранит в себе информацию о таблицах – метаданные. Структура этой таблицы выглядит следующим образом:

1. object\_type\_id – первичный ключ таблицы «OBJECT\_TYPE»
2. owner\_id – внешний ключ на таблицу «OWNER»
3. object\_name – название объекта(таблицы) в БД, которая находится в схеме owner\_id
4. is\_geo – флаг 0 или 1, который определяет является ли таблица object\_name таблицей, которая хранит в себе информацию о геометрии объекта.
5. theme\_name – имя темы портала klgd.map.ru, соответствующее объекту object\_name .
6. geo\_object\_name – определено, когда флаг is\_geo равен 0. Название таблицы, в которой хранятся данные о геометрии объекта на карте.

Таблица «OBJECT\_TYPE» хранит информацию об объектах, которые будут использоваться при анализе пространственных зависимостей. На рис. 1.2 представлен код создания таблицы «OBJECT\_TYPE» на языке sql в Oracle.

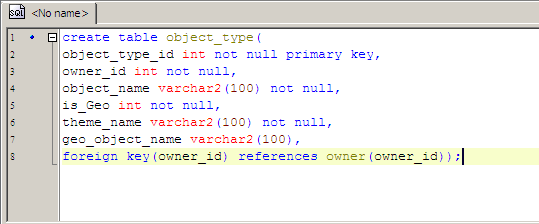


Рис 1.2 Таблица «OBJECT\_TYPE»

Следующая таблица, которая хранит метаданные – таблица «ATTRIBUTE\_TYPE». Данная таблица хранит информацию об атрибутах, которые могут использоваться при анализе. Рассмотрим структуру этой таблицы:

1. attribute\_type\_id – первичный ключ таблицы «ATTRIBUTE\_TYPE»
2. object\_type\_id – внешний ключ, который ссылается на таблицу «OBJECT\_TYPE»
3. attribute\_name – название атрибута(столбца) таблицы, которая определена в «OBJECT\_TYPE»
4. data\_type – какой тип имеет атрибут attribute\_name в таблице, определенной в «OBJECT\_TYPE»

Так как один объект или одна таблица может иметь несколько атрибутов, то между таблицами «OBJECT\_TYPE» и «ATTRIBUTE\_TYPE» используется связь один ко многим.

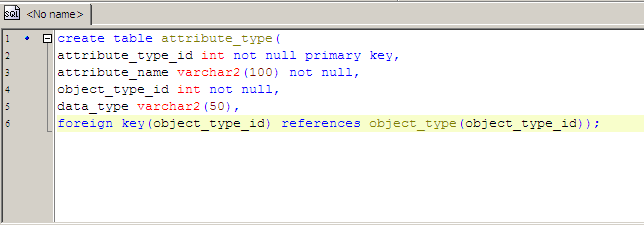


Рис 1.3 Таблица «ATTRIBUTE\_TYPE»

Для хранения пространственных зависимостей между ГИС-объектами была создана таблица «OPERATION». Пространственная зависимость между объектами ГИС выражается в виде функции, которая принимает два объекта и возвращает результат проверки. В данной работе определены два вида функций унарная и бинарная, которая принимает один или два объекта соответственно. Структура таблицы «OPERATION» выглядит следующим образом:

1. operation\_id – первичный ключ таблицы «OPERATION».
2. first\_object\_type\_id - внешний ключ, который ссылается на таблицу «OBJECT\_TYPE». Первый объекта, который будет принимать функция, как параметр.
3. second\_object\_type\_id – внешний ключ, который ссылается на таблицу «OBJECT\_TYPE». Второй объекта, который будет принимать функция, как параметр. Если значение пусто, то операция является унарной.
4. operation\_name – название операции.
5. operation\_procedure – имя функции, которая принимает один или два объекта и написанная на языке Oracle
6. operation\_description – небольшое описание операции, а именно описание того что выполняется в функции operation\_procedure

Так как один объект может использоваться в разных операциях, то связь между таблицами «OBJECT\_TYPE» и «OPERATION» один ко многим.

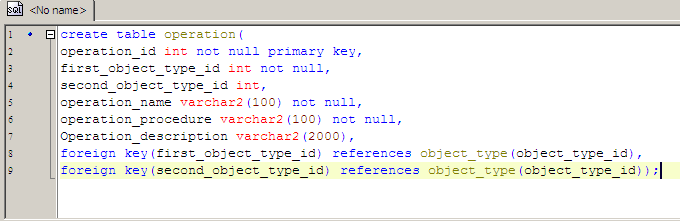


Рис 1.4 Таблица «OPERATION»

Для выполнения определенного множества операций было введено понятия правила. Правило – это множество операций, которые выполняются в определенном порядке. Для хранений правила была создана таблица «RULE»:

1. rule\_id – первичный ключ таблицы «RULE»
2. rule\_name – название правило.
3. rule\_description – краткое описание правила. Возможно описание из каких операций оно состоит.

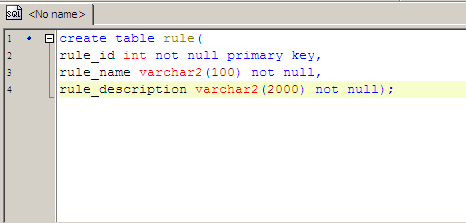


Рис 1.5 Таблица «RULE»

Так как одно правило может содержать несколько операций, а одна операция может принадлежать нескольким правилам, поэтому связь между таблицами «OPERATION» и «RULE» многое ко многим. Для установления связи многое ко многим создается дополнительная таблица, которая хранит внешние ключи на таблицы, между которыми устанавливается связь. Для связи многое ко многим между таблицами «OPERATION» и «RULE» создана таблица «RULE\_OPERATION»:

1. rule\_id – внешний ключ, ссылающийся на таблицу «RULE»
2. operation\_id – внешний ключ, ссылающийся на таблицу «OPERATION»
3. orderBy – задает порядок выполнения операции в правиле.

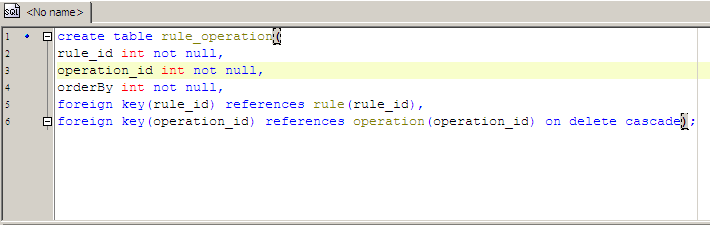


Рис 1.6 «RULE\_OPERATION»

Рассмотрим вторую часть таблиц, предназначенных для хранения информации об анализе. Базовая информация, которая хранится об анализе, располагается в таблице «CACHE\_SESSION»:

1. session\_id – первичный ключ таблицы «CACHE\_SESSION»
2. creation\_date – дата проведения анализа
3. session\_description – небольшая информация о проведенном анализе

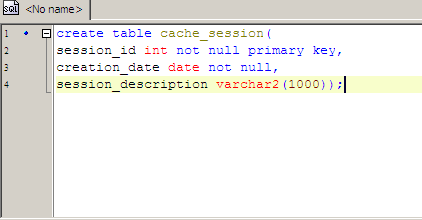


Рис 1.7 «CACHE\_SESSION»

Когда пользователь производит анализ, выбранной области все объекты, имеющие тип объекта, заданный в таблице «OBJECT\_TYPE», попадают в таблицу «CACHE\_OBJECT». Т.е. в данной таблице хранятся все объекты попавшие в выбранную область:

1. session\_id – внешний ключ, ссылающийся на таблицу «CACHE\_SESSION».
2. object\_type\_id – внешний ключ, ссылающийся на таблицу «OBJECT\_TYPE». Указывает, к какому типу объекта принадлежит объект.
3. object\_value – значение объекта, попавшего в анализируемую область. Значение первичного ключа таблицы object\_type\_id.

Связь между таблицами «CACHE\_SESSION» и «CACHE\_OBJECT», «OBJECT\_TYPE» и «CACHE\_OBJECT» один ко многим, потому что одному анализу соответствует несколько анализируемых объектов и у одного типа объекта содержится несколько объектов.

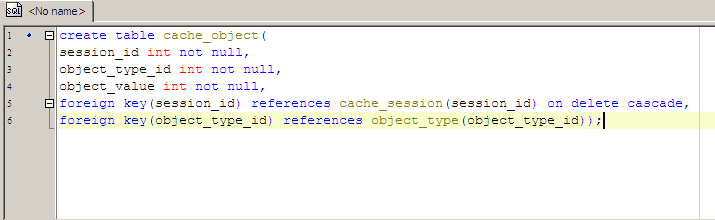


Рис. 1.8 Таблица «CACHE\_OBJECT»

Так как данные об объектах постоянно изменяются, значения атрибутов попавших в область объектов, загружаются в таблицу «CACHE\_ATTRIBUTE»:

1. session\_id – внешний ключ, ссылающийся на таблицу «CACHE\_SESSION»
2. attribute\_id – внешний ключ, ссылающийся на таблицу «ATTRIBUTE\_TYPE»
3. object\_value - значение объекта, попавшего в анализируемую область
4. str\_val – хранит строковое значение атрибута. Если тип атрибута имеет строковый тип, то текущее значение атрибута запишется в это поле, иначе оно будет пустым
5. number\_val – хранит целое число или число с плавающей точкой. Если тип атрибута имеет тип целого числа или числа с плавающей точкой, то текущее значение запишется в это поле, иначе поле будет пустым
6. date\_val – хранит дату. Если тип атрибута имеет тип даты, то текущее значение запишется в это поле, иначе поле будет пустым
7. geometry\_val – хранит геометрическое значение. Если тип атрибута имеет тип геометрии, то текущее значение будет запишется в это поле, иначе поле будет пустым

Связь между таблицами «CACHE\_SESSION» и «CACHE\_ATTRIBUTE», «ATTRIBUTE\_TYPE» и «CACHE\_ATTRIBUTE» один ко многим, так как для одного анализа может быть выгружено несколько значений атрибутов и несколько значений атрибутов могут иметь один тип атрибута.

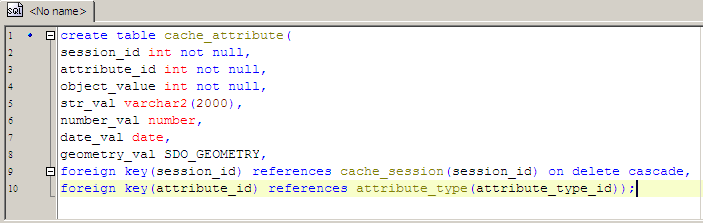


Рис 1.9 Таблица «CACHE\_ATTRIBUTE».

Для хранения результатов выполнения анализа была создана таблица «CACHE\_LOG»:

1. log\_id – первичный ключ таблицы «CACHE\_LOG»
2. session\_id – внешний ключ, ссылающийся на таблицу «CACHE\_SESSION»
3. operation\_id – внешний ключ, ссылающийся на таблицу «OPERATION»
4. first\_object\_id – значение первого объекта, как параметра в операции operation\_id
5. second\_object\_id – значение второго объекта, как параметра в операции operation\_id. Может быть пустым если operation\_id – унарная операция.
6. result – результат выполнения операции. Принимаемы значения 0 или 1.

Так как существует несколько результатов выполнения операций в одном анализе, то связь между таблицами «CACHE\_SESSION» и «CACHE\_LOG» один ко многим. Одна операция может выполнятся несколько раз между разными объектами, то связь между «OPERATION» и «CACHE\_LOG» один ко многим.

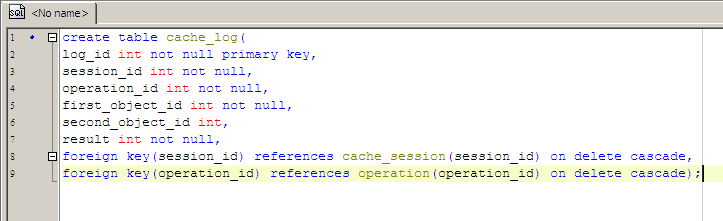


Рис 1.10 Таблица «CACHE\_LOG»

Для удаление, созданных таблиц воспользуемся операцией удаления в Oracle DROP <тип> <имя>. Удаление таблиц должно происходить в обратном порядке создание таблиц. Если мы хотим удалить таблицу, на которую ссылается другая таблица с помощью внешнего ключа, то произойдет ошибка удаления. Приведем один из примеров удаления созданных таблиц:

1. «CACHE\_LOG»
2. «RULE\_OPERATION»
3. «RULE»
4. «CACHE\_ATTRIBUTE»
5. «CACHE\_OBJECT»
6. «OPERATION»
7. «CACHE\_SESSION»
8. «ATTRIBUTE\_TYPE»
9. «OBJECT\_TYPE»
10. «OWNER»

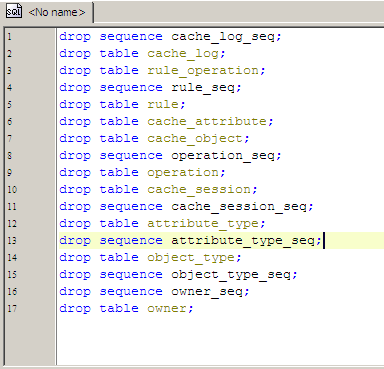


Рис. 1.11 Скрипт удаления модуля.

Рассмотрим более подробно функции заполнения таблиц «CACHE\_OBJECT» и «CACHE\_ATTRIBUTE». Как упоминалось ранее таблица «CACHE\_OBJECT» хранит в себе объекты, анализируемой области. Все доступные для анализа области расположены в таблице «GEO\_DOCS\_BOUNDARY». Важными атрибутами данной таблицы является первичный ключ «DOCS\_BOUNDARY\_GEO\_ID», описание области «NAME», и геометрия объекта на карте «GEOMETRY».

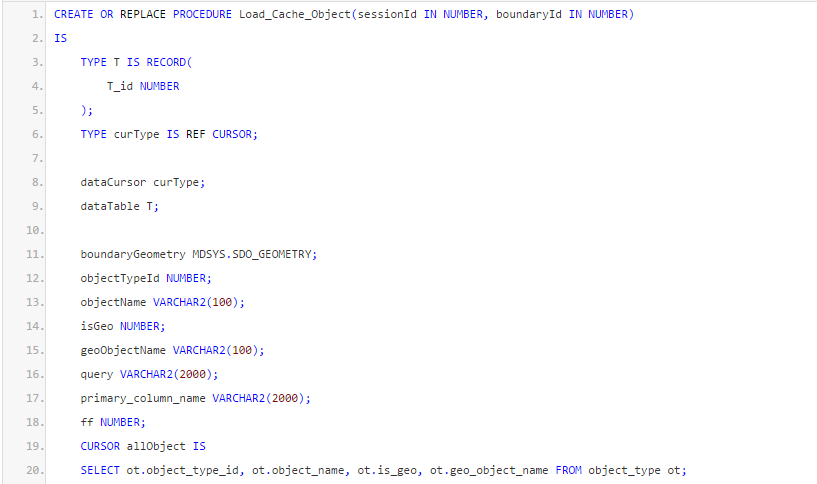


Рис 1.12 Процедура «Load\_Cache\_Object»

На рис. 1.12 показано начало процедуры «Load\_Cache\_Object» - сохранение объектов в таблицу «CACHE\_OBJECT», попавших в заданную область. Данная функция принимает два аргумента: первичный ключ проводимого анализа таблицы «CACHE\_SESSION» и первичный ключ анализируемой области. В блоке, начинающимся со слова IS, описываются переменные необходимые для выполнение процедуры.

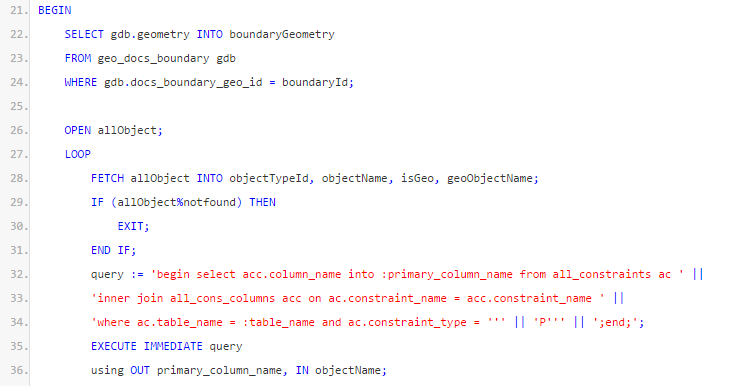


Рис. 1.13 Процедура «Load\_Cache\_Object»

На рис. 1.13 представлено начало тело процедуры. На строке 22-24 выполнятся SQL-запрос, где мы получаем геометрию анализируемой области. Затем мы открываем курсор, который проходится по всем типам объектов доступных для анализа и для каждого типа объекта(таблицы) мы, выполняя динамический запрос, находим имя столбца, который является первичным ключом таблицы.

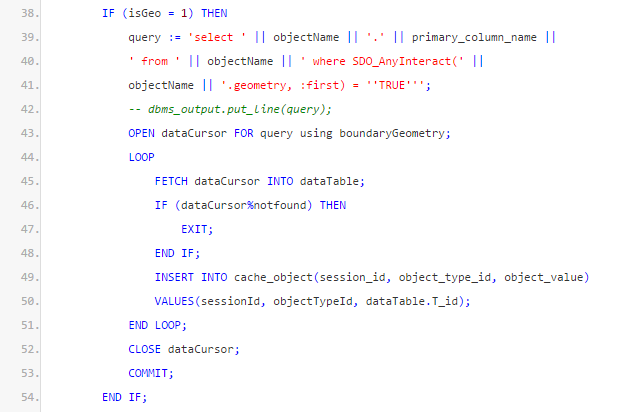


Рис 1.14 Процедура «Load\_Cache\_Object»

Затем на Рис.1.14 строке 38 выполняется проверка на хранение геометрии объекта. Если условие выполняется, то мы открываем динамический курсор для запроса query на строке 43. Запрос query использует функцию SDO\_AnyInteract, которая принимает два параметра типа SDO\_GEOMETRY и возвращает ‘TRUE’ если геометрические объекты взаимодействуют друг с другом, и все такие объекты записываются в таблицу «CACHE\_OBJECT».

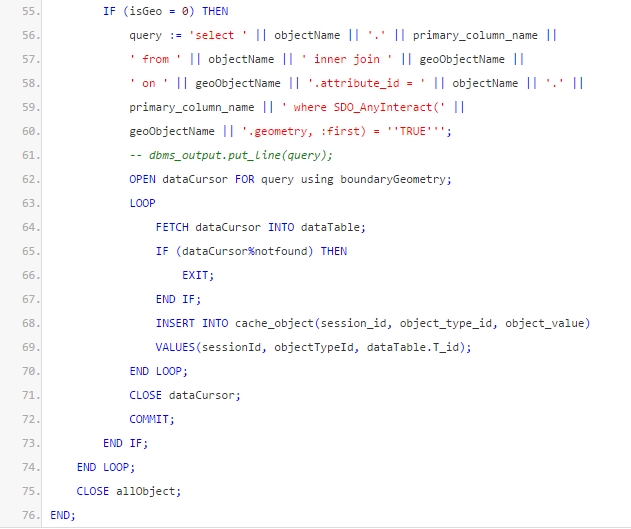


Рис. 1.15 Процедура «Load\_Cache\_Object»

Если таблица objectName не содержит геометрию объекта, то выполнение записи объектов, попавших в область происходит по такому же принципу с одним исключением. Необходимо найти кортеж в таблице geoObjectName. Таблицы geoObjectName содержит внешний ключ ATTRIBUTE\_ID, который ссылается на таблицу objectName.

Рассмотрим функцию загрузки значений атрибутов в таблицу «CACHE\_ATTRIBUTE». На рис. 1.16 процедура «Load\_Cache\_Attribute» принимает один аргумент – идентификатор анализа.

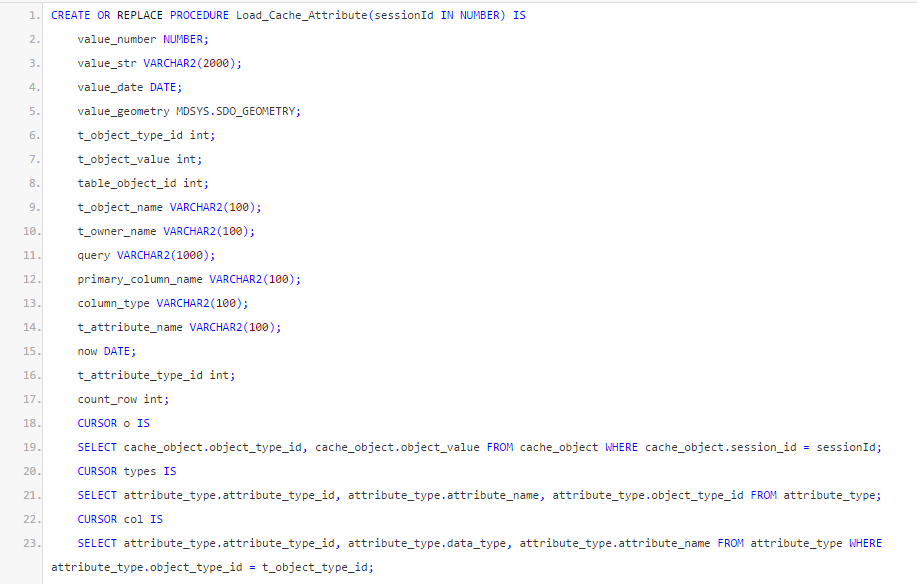


Рис 1.16 Процедура «Load\_Cache\_Attribute»

На рис. 1.17 изображено начало процедуры. Чтобы узнать в какой столбец записывать значение таблицы «CACHE\_ATTRIBUTE», необходимо узнать какой тип имеет атрибут. Для этого мы проходимся по всем атрибутам, которые хранятся в таблице «ATTRIBUTE\_TYPE» и выполняем динамический запрос для поиска типа и записываем его в таблицу «ATTRIBUTE\_TYPE» в поле data\_type.

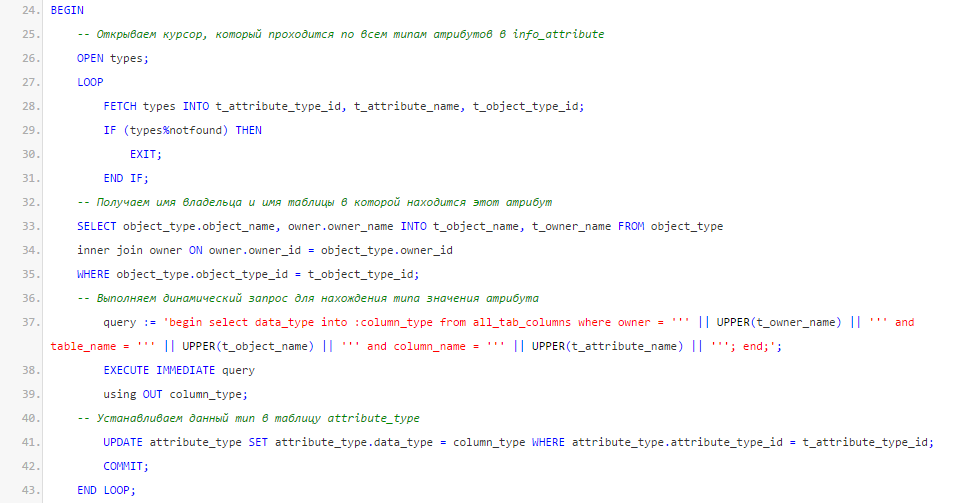


Рис. 1.17 Процедура «Load\_Cache\_Attribute»

Затем мы проходимся по всем объектам из таблицы «CACHE\_OBJECT», которые находятся в анализируемой области. Для каждого объекта мы узнаем, в какой таблицы он хранится и названия столбца, являющейся первичным ключом этой таблицы (рис. 1.18).

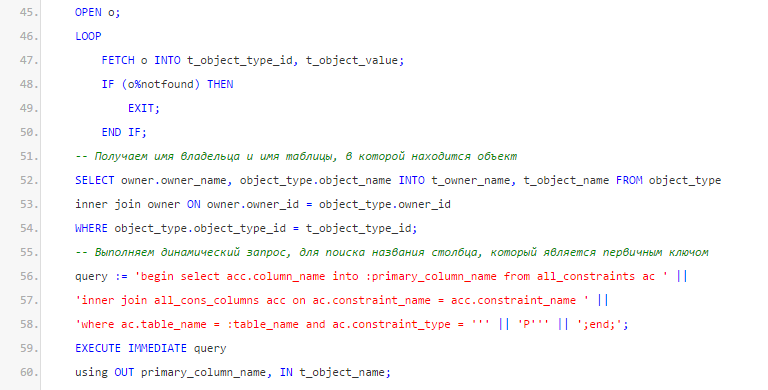


Рис. 1.18 Процедура «Load\_Cache\_Attribute»

Далее на рис. 1.19 проходимся по всем атрибутам, которые необходимы для проведения анализа у рассматриваемого объекта и, выполняя динамический запрос вытаскиваем его значение.

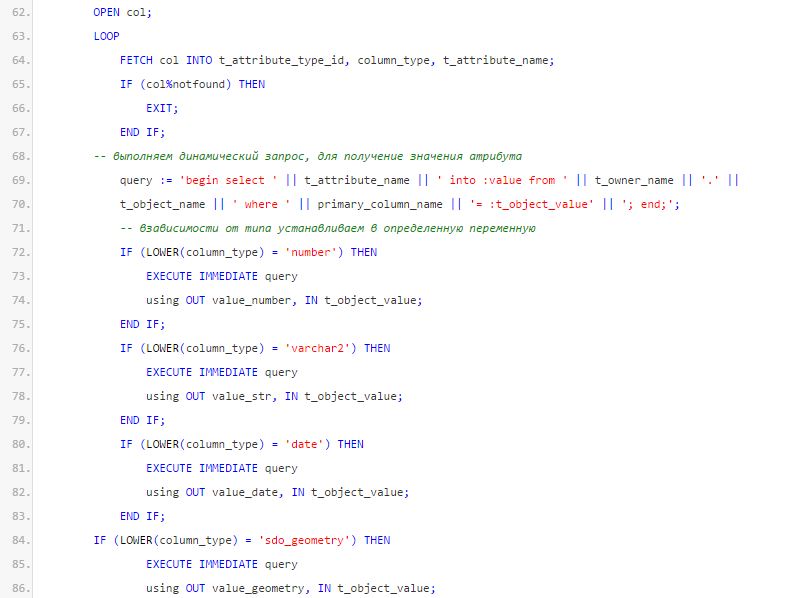


Рис. 1.19 Процедура «Load\_Cache\_Attribute»

Затем остается записать полученное значение в таблицу «CACHE\_ATTRIBUTE» в соответствующий определенному типу (рис. 1.20)

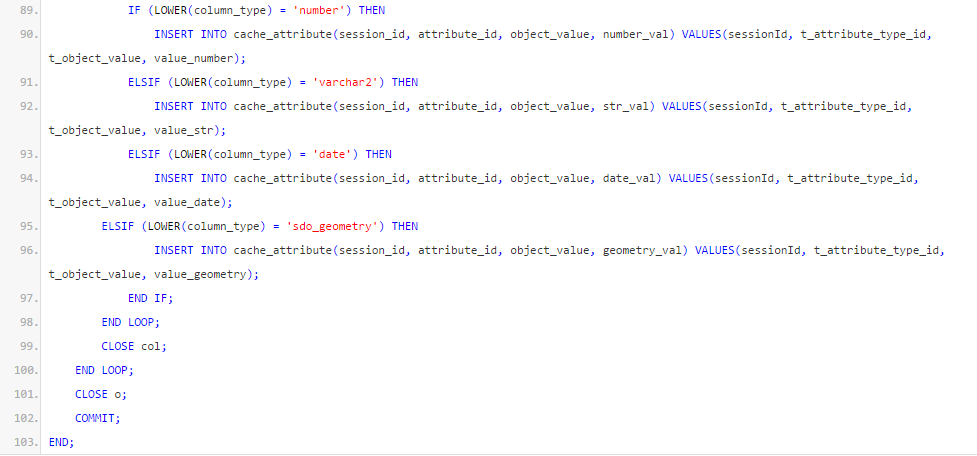


Рис. 1.20 Процедура «Load\_Cache\_Attribute»

Примеры операций между объектами ГИС.

Перед тем как рассматривать некоторые операции, для удобства использования были написаны функции, которые возвращают значение определенного типа из таблицы «CACHE\_ATTRIBUTE». Для получение значения необходимо знать идентификатор анализа, значение объекта атрибут которого мы хотим получить, и название атрибута. На рис 1.21 представлена одна из таких функций. Отличие этих функций заключается в возвращаемом значении.

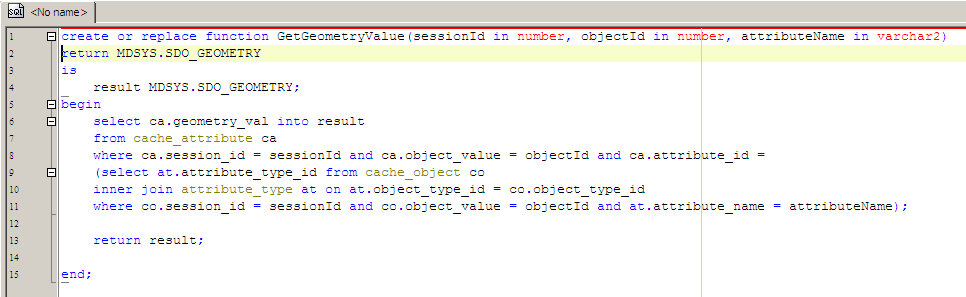


Рис. 1.21

На рис. 1.22 и 1.23 представлены две унарные операции выполняющие проверку атрибутивного параметра. В первой функции выполняется проверка на количество этажей у здания. Если количество этажей не лежит на отрезке от 1..5, то результат функции будет отрицательным. Во второй функции проверяется функциональное назначение зданий. Жилые здания имеют идентификатор – 1, нежилые 19, а общественные 20. Если здание не относится ни к одному типу, то результат операции будет отрицательный.

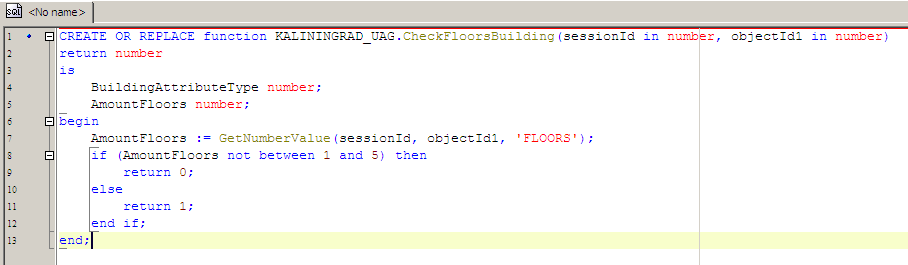


Рис. 1.22 Проверка этажности здания

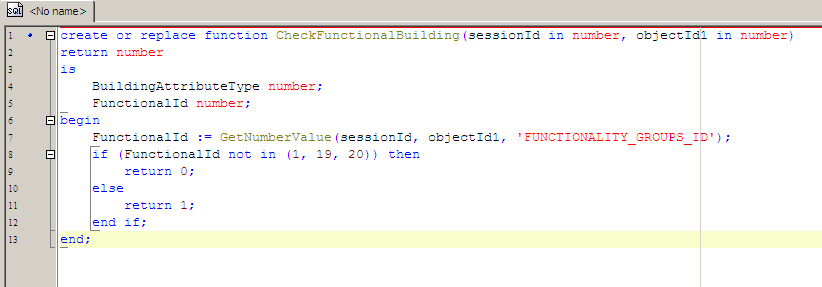


Рис. 1.23 Проверка функциональности здания

На рис. 1.24 изображена бинарная функция, которая проверяет, выходит ли здание за границы земельного участка. В функции выполняется проверка на пересечение двух геометрических объектов.

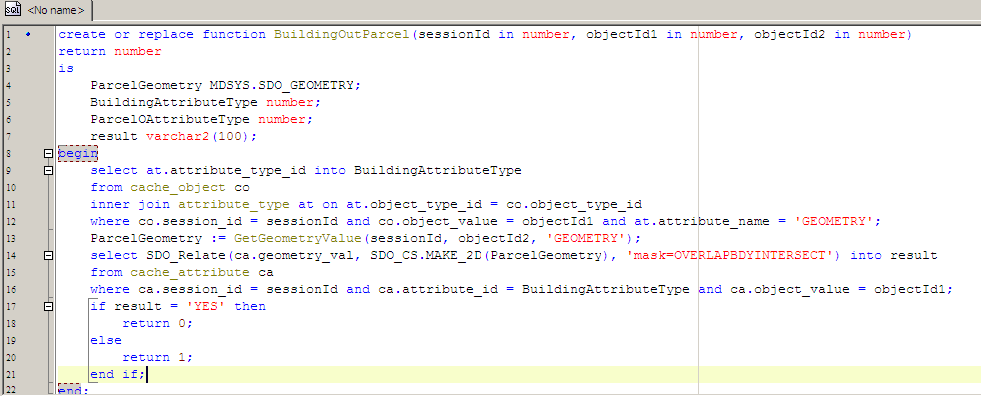


Рис. 1.24 Выход здания за границы земельного участка

В данной операции используется функция SDO\_RELATE, которая принимает два геометрических объекта, и маску обозначающую на какое отношение проверять два геометрических объекта. На рис. 1.25 представлены различные топологические отношения между геометрическими объектами.

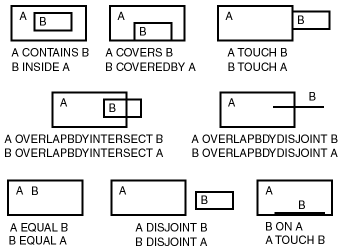


Рис. 1.25 Топологические отношение

Приложение .NET Framework на языке C#.

Приложение на языке C#, основываясь на модуле параметризации пространственных зависимостей, должно выполнять следующие функции.

1. Реализация конструктора правила. В конструктор правил входит добавление новых операций, написанных на языке pl/sql, в таблицу «OPERATION», а также изменение и удаление операций, находящихся в таблице «OPERATION». Создание новых правил, удаление и изменение существующих.
2. Выполнение на выбранной области одного или нескольких правил, просмотр результатов анализа, а также проведение нового анализа на основе данных другого анализа с частичным их изменением.
3. Просмотр объектов на портале klgd.map.ru

При открытии приложения, появляется окно подключения к базе данных рис. 2.1. Для подключения к базе данных необходимо ввести пользователя, пароль, а также службу – информацию о расположении сервера базы данных. После того как пользователь ввел необходимые данные ему необходимо нажать на кнопку «Вход» или «Enter».

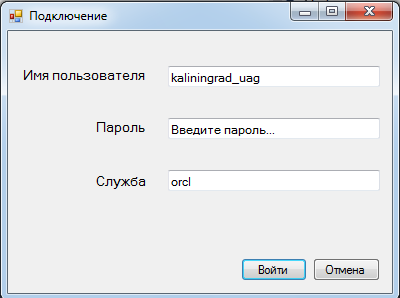


Рис 2.1 Подключение

После нажатия на кнопку происходит создание строки подключения и попытка создать соединение с базой данных. При введенных неверных данных появится сообщение об ошибке.

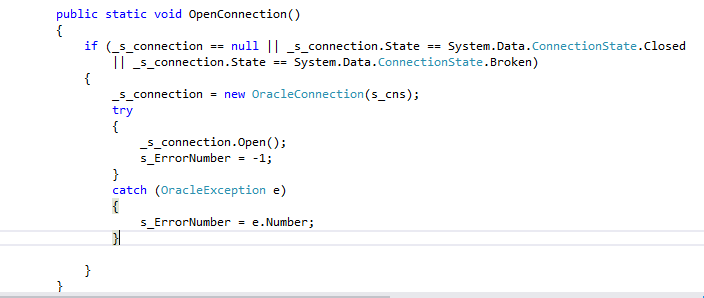
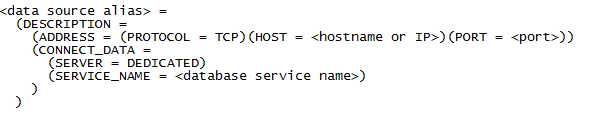


Рис. 2.2 Установка соединения

s\_cns – строка подключения к базе имеющая следующий вид:

Где {0} – служба, {1} – имя пользователя, {2} – пароль.

На рис. 2.1 в строке служба введен ее алиас, который описан в файле tnsname.ora и имеет следующий шаблон:



После успешного соединения с базой данных, открывается окно главной формы с двумя доступными режимами: режим администрирования и режим анализа (рис. 2.3).

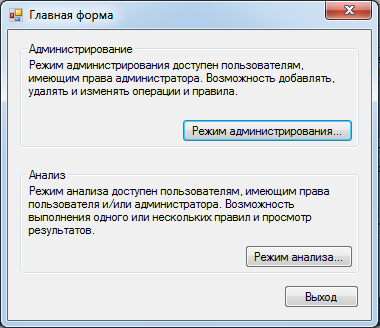


Рис. 2.3 Главное окно

При нажатии на кнопку «Режим администрирования…» пользователя переведет в окно, где он сможет выполнить функции добавления, удаления и изменения над операциями и правилами (рис 2.4).

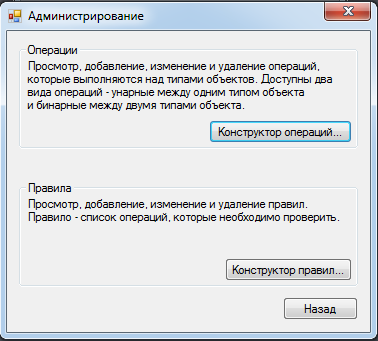


Рис. 2.4 Режим Администрирования

При нажатии на кнопку «Режим анализа…» пользователя переведет в окно, откуда он сможет произвести анализ области или просмотреть результаты выполнения старых анализов ( рис 2.5).

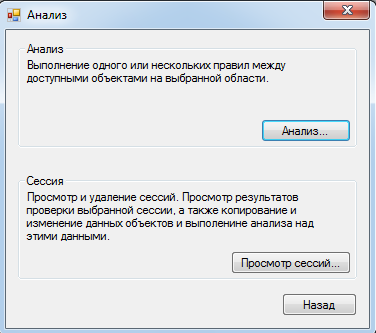


Рис. 2.5 Режим анализа

Рассмотрим более подробно режим администрирования. В окне «Режим администрирования» (рис 2.4) нам доступно две кнопки «Конструктор операций» и «Конструктор правил».

Конструктор операций. При нажатии на кнопку «Конструктор операций…» пользователю откроется окно со списком возможных операций, а также доступен поиск операций по имени.

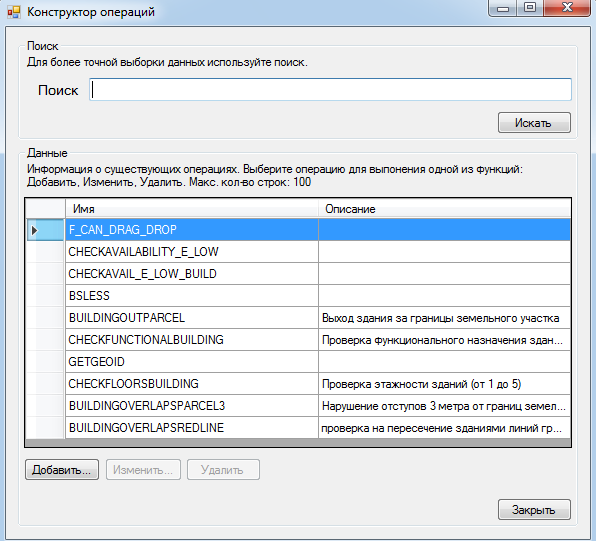


Рис. 2.6. Конструктор операций

Конструктор операций предоставляет возможность добавлять функции, написанные пользователем в Oracle в таблицу «OPERATION», а также удалять или изменять информацию об операциях, уже хранящихся в таблице «OPERATION». Если операция имеет описание в таблице на форме, значит, данная операция уже хранится в таблице «OPERATION» и кнопка «Добавить…» будет не доступна. Если же описания нет, то значит, данная операция не присутствует в таблице «OPERATION» и кнопки «Изменить…» или «Удалить» будут не доступны. При открытии окна «Конструктор операций» происходит загрузка информации об операциях из базы данных. В базе данных Oracle, информация про существующие функции и процедуры находится в таблице «USER\_PROCEDURES», а информация про аргументы функций в таблице «USER\_ARGUMENTS». Поэтому при открытии окна мы выполняем запрос в таблицу, для получения информации о доступных процедурах.

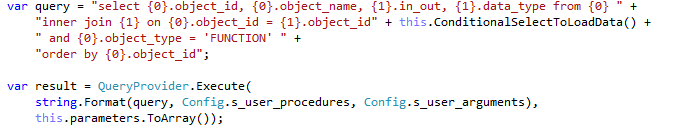


Рис. 2.7 Выборка данных из таблиц «USER\_PROCEDURES» и «USER\_ARGUMENTS»

Так как функции могут принимать множества различных параметров, и возвращать результат любого типа, то нам подходят не все, а только те которые имеет следующий шаблон: функция имеет 2 или 3 принимающих параметра и результат типа «NUMBER». Первый параметр - идентификатор анализа. Второй и третий – идентификаторы объектов.

После того как мы получили список функций, удовлетворяющих шаблону, нам необходимо узнать какие из этих операций уже существуют в таблице «OPERATION». Для этого мы выполняем запрос к таблице, который возвращает информацию об операции, если такая операция присутствует в таблице «OPERATION».

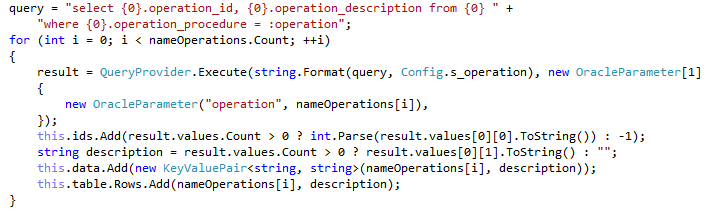


Рис. 2.8 Получение данных из таблицы «OPERATION»

На рис. 2.8 переменная ids – список первичных ключей операций таблицы «OPERATION». Если i-ое значение в списке равняется -1, то данной операции не присутствует в таблице «OPERATION». Переменная data – список, который хранит пару значений: имя операции и описание, если оно существует.

В окне «Конструктор операций» присутствует поиск операций по имени. Пользователю необходимо ввести строку и нажать на кнопку «Поиск». При нажатии на кнопку выполнится загрузка информации из базы данных, как и при открытии окна, но с одним дополнением. К запросу добавляется дополнительное условие, которое проверяет, что имя операции содержит строку введенную пользователем, как подстроку. Шаблон такого условия выглядит следующим образом.

Где checkName – строку, которую хотим проверить, % - означает любое кол-во символов, tempName – строку, которую необходимо найти.

На рис. 2.9 представлено добавления данного условия на языке C#.

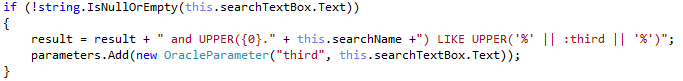


Рис. 2.9 Проверка подстроки в строке

Строка result – строка sql запроса, searchName – значение, которое хотим проверить, searchTextBox.Text – строка введенная пользователем.

При выборе операции пользователь может:

1. Добавить операцию в таблицу «OPERATION» нажав на кнопку «Добавить…»
2. Изменить информацию об операции в таблице «OPERATION», нажав на кнопку «Изменить…»
3. Удалить операцию из таблицы «OPERATION», нажав на кнопку «Удалить».

При нажатии на кнопку «Добавить» откроется вспомогательное окно на рис. 2.10, где необходимо ввести информацию об операции. Если операция является унарной, то тип объекта 2 будет не доступен для выбора.

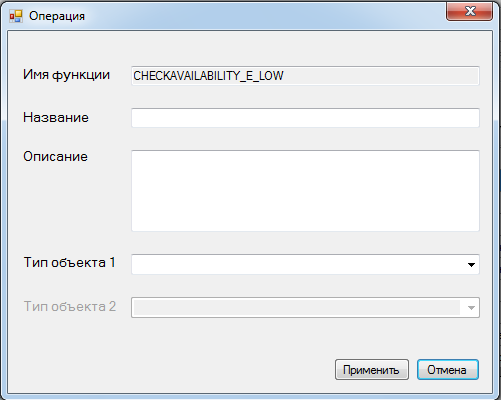


Рис. 2.10 Операция

При открытии вспомогательного окна необходимо получить информации о типах объекта доступных для анализа, которые хранятся в таблице «OBJECT\_TYPE». На рис. 2.11 представлен код на C# выборки типов объекта.

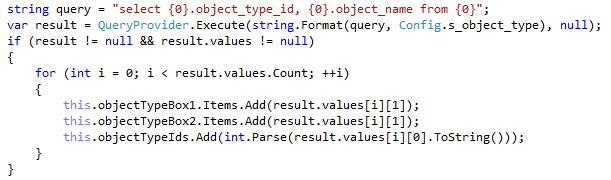


Рис. 2.11 Выборка данных из таблицы «OBJECT\_TYPE»

После того как пользователь заполнил всю информации об операции, то при нажатии на кнопку «Применить» данная операция добавится в таблицу «OPERATION» с помощью sql-запроса insert into. На рис. 2.12 представлен код добавления новой операции в таблицу «OPERATION».

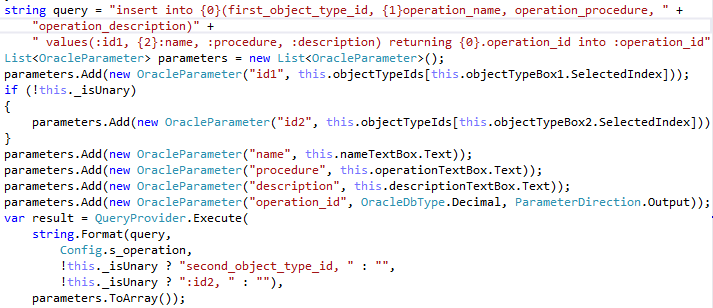


Рис. 2.12 Добавление операции в таблицу «OPERATION»

При нажатии на кнопку «Изменить…», открывается такое же вспомогательное окно, как и при добавлении, но уже в режиме изменения. Так как выбранная операция уже существует в таблице «OPERATION», то при открытии вспомогательного окна мы получаем необходимую информацию об операции (рис. 2.13).

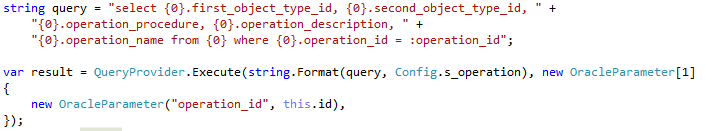


Рис. 2.13. Получение информации об операции из таблицы «OPERATION»

При нажатии на кнопку «Применить», мы должны обновить информацию об операции в таблице «OPERATION» с помощью sql-запроса UPDATE (2.14).

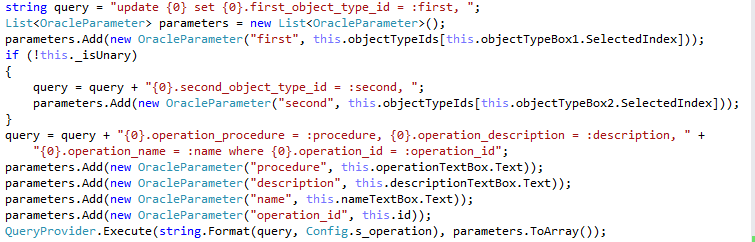


Рис. 2.14 Обновление данных об операции в таблице «OPERATION»

При нажатии на кнопку «Удалить» на окне «Конструктор операций», необходимо выполнить sql-запрос DELETE, чтобы удалить операцию из таблицы «OPERATION». Пример такого запроса представлен на рис. 2.15.

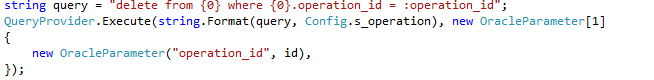


Рис. 2.15. Удаление операции из таблицы «OPERATION»

Конструктор правил.

Конструктор правил предоставляет пользователю следующие функции:

1. Создание нового правила и добавление его в таблицу «RULE»
2. Изменение существующего правила.
3. Удаление правила.

При нажатии на кнопку «Конструктор операций…» (рис. 2.4.) пользователю откроется соответствующее окно(рис. 2.16).

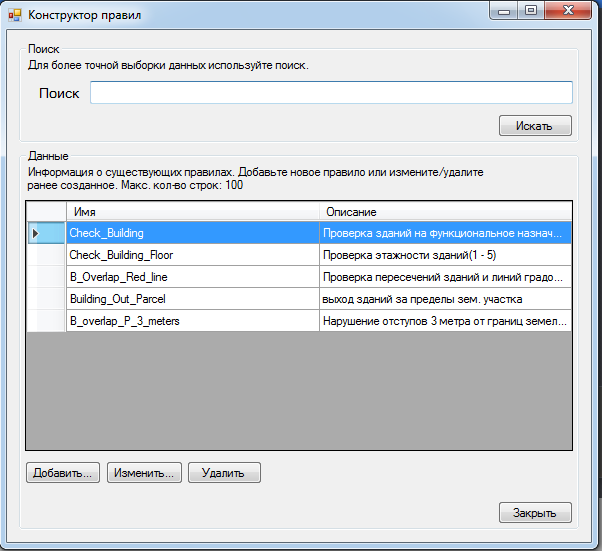


Рис. 2.16. Конструктор правил

Механизм поиска в окне «Конструктор правил» совпадает с поиском в окне «Конструктор операций». Для того чтобы создать новое правило, необходимо нажать на кнопку «Добавить…». После нажатия на кнопку пользователю откроется вспомогательное окно, где он должен ввести некоторую информацию о правиле (Название и Описание), а также выбрать операции, из которых будет состоять правило и установить их порядок выполнения.

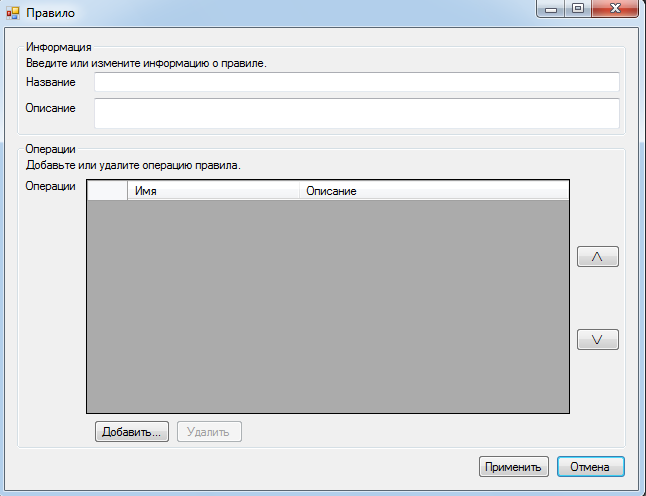


Рис. 2.17. Правило

Для того чтобы добавить в правило новую операцию необходимо нажать на кнопку «Добавить…». Откроется новое окно, в котором будет список операций из таблицы «OPERATION», которые не добавлены в новое правило, а также с функцией поиска операции по имени.

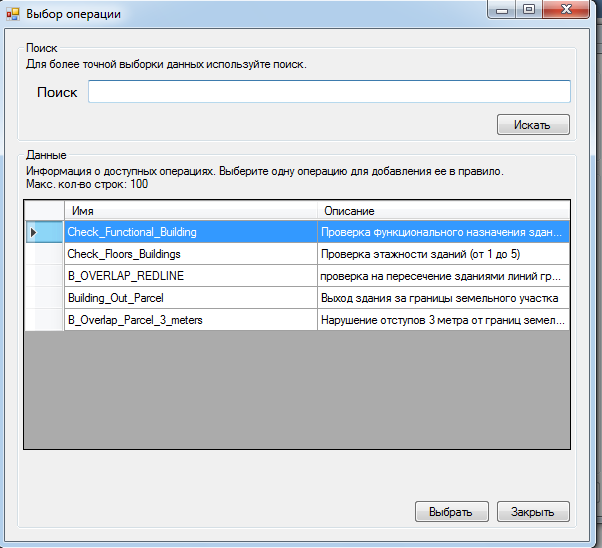


Рис.2.18 Выбор операции.

Для того чтобы добавить операцию в правило необходимо выбрать операцию из списка и нажать на кнопку «Выбрать». Если мы хотим удалить операцию из правила необходимо выбрать операцию и нажать на кнопку «Удалить». Чтобы изменить порядок выполнения операции в правиле необходимо выбрать операцию и нажать на кнопку «/\» или «\/». При нажатии на кнопку «Применить» создается новое правило в таблице «RULE» (рис. 2.19), а также создаются записи в таблице «RULE\_OPERATION»(рис. 2.20).

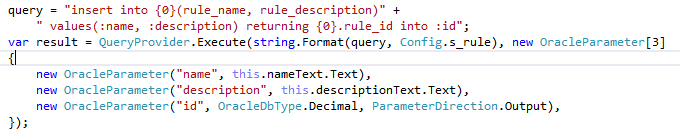


Рис. 2.19 Добавление правила в таблицу «RULE»

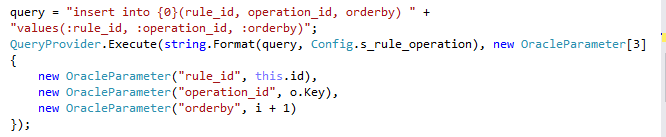


Рис. 2.20 Добавление операций в таблицу «RULE\_OPERATION»

Для того чтобы изменить существующее правило необходимо в окне «Конструктор правил» выбрать правило из списка и нажать на кнопку «Изменить». Откроется тоже окно, что и при создании нового правила, только в режиме изменения. При нажатии на кнопку «Применить» в окне 2.17 необходимо обновить записать в таблице «RULE»(рис. 2.21) и добавить новую запись в таблицу «RULE\_OPERATION»(рис. 2.20) и возможно удалить запись из таблицы «RULE\_OPERATION»(рис. 2.22).

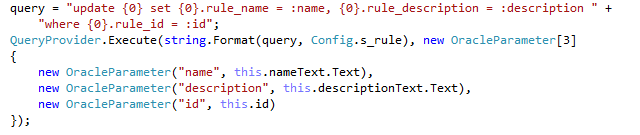


Рис. 2.21 Обновление данных правила в таблице «RULE»

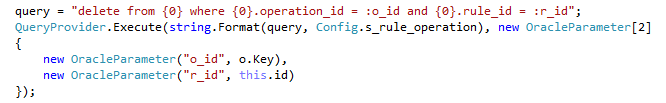


Рис. 2.22 Удаление операции из правила

Если пользователь хочет удалить существующее правило, то в окне «Конструктор правил» необходимо выбрать правило из списка и нажать на кнопку «Удалить». Для удаления существующего правила необходимо удалить записи из таблицы «RULE\_OPERATION», относящиеся к удаляемому правилу, а также удалить правило из таблицы «RULE».

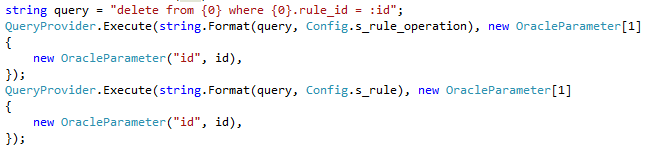


Рис. 2.23 Удаление правила из таблицы «RULE» и «RULE\_OPERATION»

Анализ.

Для того чтобы провести анализ выбранной области необходимо на окне «Режим анализа» нажать на кнопку «Анализ…». После этого откроется окно (рис. 2.24) с которого пользователь может выполнить анализ.

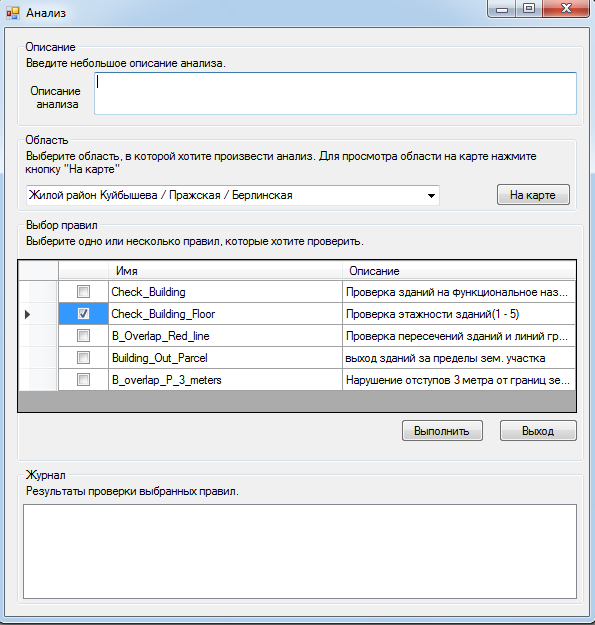


Рис. 2.24 Анализ.

Перед тем как выполнить анализ, пользователю необходимо ввести описание анализа. Затем необходимо выбрать область доступную для анализа, выбрать одно или несколько правил, которые необходимо проверить и нажать на кнопку «Выполнить». Также пользователю предоставляется возможность посмотреть анализируемую область на карте нажав на кнопку «На карте».

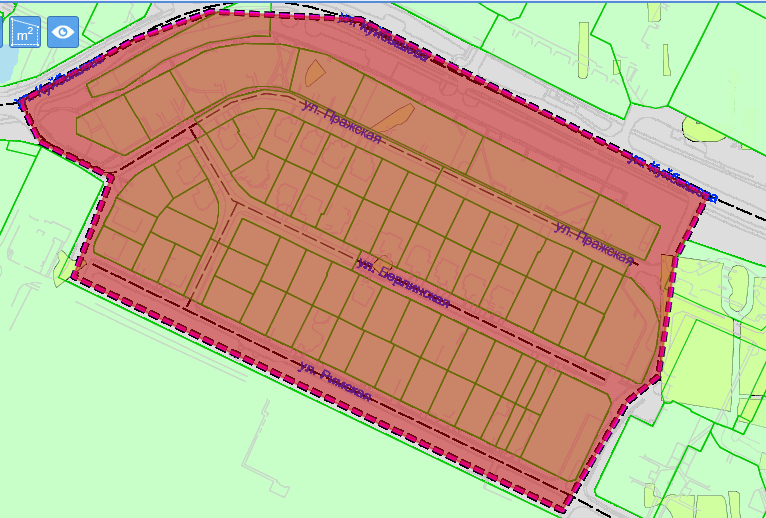


Рис. 2.25. Анализируемая область на карте

После того как выполниться анализ на области, в журнал будет записаны результаты выполнения правил и операций. Рассмотрим более подробно алгоритм выполнения анализа.

1. Создаем новый анализ, а именно новую запись в таблице «CACHE\_SESSION».
2. Выполнение функции загрузки области – Load\_Cache\_Object
3. Выполнение функции загрузки значений атрибутов – Load\_Cache\_Attribute
4. Проходимся по всем выбранным правилам и выполняем операции между всеми объектами определенного типа.

Просмотр проведенных анализов.

Для того чтобы посмотреть список проведенных анализов, необходимо нажать на кнопку «Просмотр сессий…» в окне «Режим анализа» (рис. 2.5). Откроется окно с информацией о проведенных анализах со следующими доступными функциями:

1. Просмотр результатов анализа.
2. Выполнение нового анализа, используя данные проведенного анализа с некоторыми изменениями.
3. Удаление анализа.

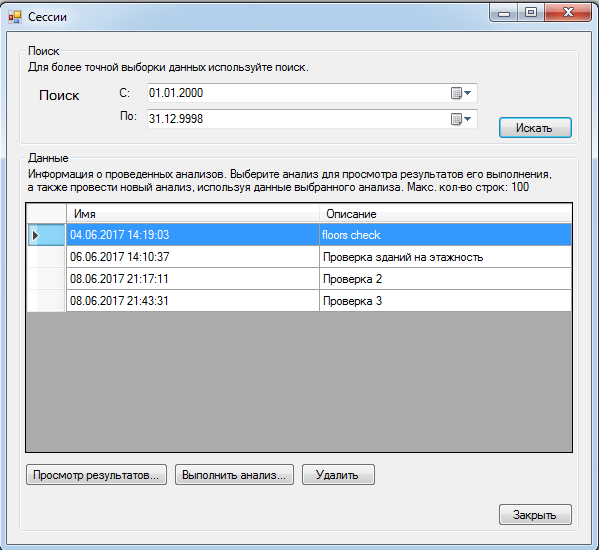


Рис. 2.26 Сессии

На окне «Сессии» (рис. 2.26) можно выполнить поиск по дате выполнения анализа, а именно задать промежуток времени, когда происходил анализ области. Для просмотра результатов необходимо выбрать анализ и нажать на кнопку «Просмотр результатов…»

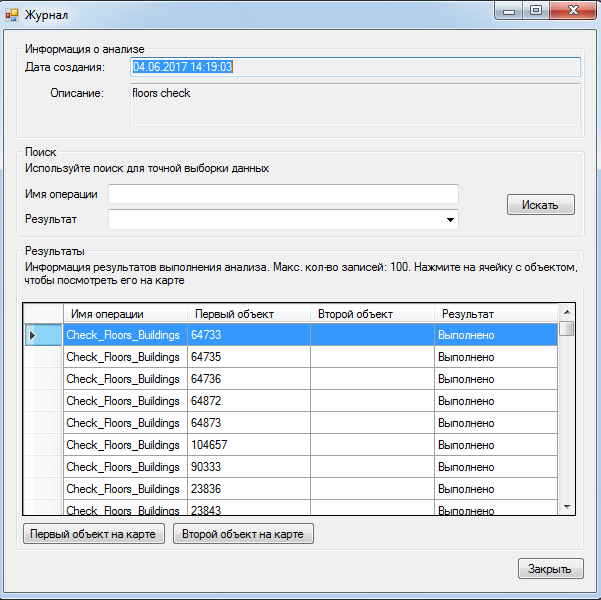


Рис. 2.27 Журнал

В верхней части окна «Журнал» выводится небольшая информация об анализе: дата создания и описание. Также можно выполнить поиск по двум критериям:

1. По имени операции, которая совпадает с поиском на окне «Конструктор операций».
2. По результату выполнение, которое может принимать только два значения 1 или 0.

Для более точного понимания, между какими объектами происходила проверка, выполнен функционал просмотра объектов на карте. Чтобы посмотреть объект на карте необходимо выбрать один из результатов проверки и нажать на кнопку «Первый объект на карте» или «Второй объект на карте» в зависимости от того какой объект вы хотите посмотреть.

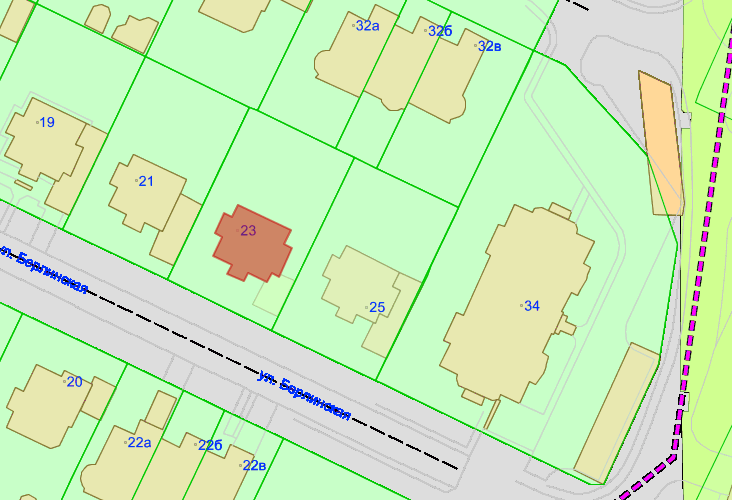


Рис. 2.28 Объект на карте

Чтобы произвести новый анализ, используя данные старого анализа с некоторыми изменениями необходимо в окне «Сессии» (рис. 2.26) выбрать анализ, данные которого хотим использовать, и нажать на кнопку «Выполнить анализ». Откроется окно «Значения атрибутов» (рис. 2.29), на котором можно произвести поиск атрибута по двум критериям, который выполняется по принципу, реализованному на окне «Конструктор операциий»:

1. По названию атрибута. Выполняется по принципу, который реализован на окне «Конструктор операций».
2. По названию объекта.

Для того чтобы произвести новый анализ, необходимо изменить значения одного или нескольких атрибутов.

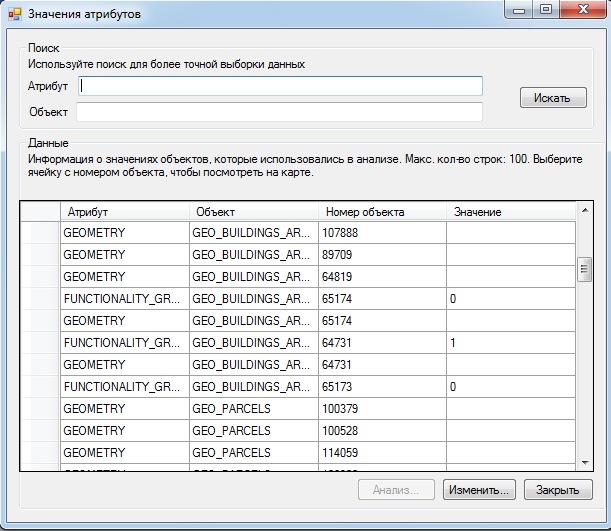


Рис. 2.29 Значения атрибутов

Чтобы изменить значение атрибута необходимо выбрать один атрибут из списка и нажать на кнопку «Изменить…». После этого откроется вспомогательное окно (рис. 2.30), где можно изменить значение выбранного атрибута.

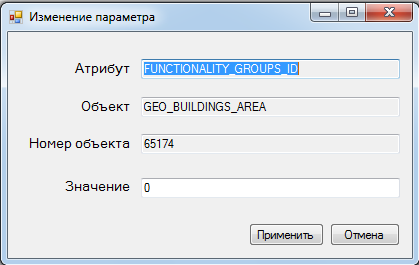


Рис. 2.30 Изменение значения атрибута

При нажатии на кнопку «Анализ…» на окне «Значения атрибутов» выполнится следующий алгоритм:

1. Создание нового анализа и добавление его в таблицу «CACHE\_SESSION».
2. Произойдет копирование объектов в таблице «CACHE\_OBJECT», попавших в область старого анализа.
3. Копирование значений атрибутов с изменениями, которые установил пользователь.
4. Открытие окна «Анализ» (рис.2.24), где можно выбрать правила и провести анализ.

Также окно «Сессии» предоставляет возможность удаление информации об проведенном анализе. Для этого необходимо выбрать анализ и нажать на кнопку «Удаление». Произойдет удаление анализа из таблицы «CACHE\_SESSION», а также удаление всех записей ссылавшихся на удаляемый анализ.