

МОБИЛЬНАЯ
РЕЛЯЦИОННАЯ
СУБД **ЛИНТЕР**[®]

Linter Standard
Linter Bastion
Linter RealTime
Linter Multiversion

Геометрические типы данных

НАУЧНО-ПРОИЗВОДСТВЕННОЕ ПРЕДПРИЯТИЕ

 **РЕЛЭКС**[®]

Товарные знаки

РЕЛЭКС™, ЛИНТЕР® , НЕВОД® , LAV™, ЛАКУНА являются товарными знаками, принадлежащими ЗАО НПП «Реляционные экспертные системы» (далее по тексту – компания РЕЛЭКС). Прочие названия и обозначения продуктов являются товарными знаками их производителей, продавцов или разработчиков.

Интеллектуальная собственность

Правообладателем продуктов ЛИНТЕР®, НЕВОД®, LAV™, ЛАКУНА является компания РЕЛЭКС (1990–2011). Все права защищены. Данный документ является собственностью компании РЕЛЭКС. Ни одна часть данного документа не может быть воспроизведена, передана, преобразована, сохранена в системе поиска информации, переведена на другой язык или компьютерный язык в какой-либо форме, какими-либо средствами, электронными, механическими, магнитными, оптическими, химическими, ручными или иными, без предварительного разрешения компании РЕЛЭКС.

О документе

Материал, содержащийся в данном документе, прошел тщательную проверку, но компания РЕЛЭКС не гарантирует, что документ не содержит ошибок и пропусков. Компания РЕЛЭКС оставляет за собой право в любое время вносить в документ исправления и изменения, пересматривать и обновлять содержащуюся в нем информацию.

Адрес

394006, г. Воронеж, ул. 20-летия Октября, 119.
Тел./факс: (473) 2-711-711, 2-778-333.
e-mail: market@relex.ru.

Адрес для корреспонденции

394000, г. Воронеж, а/я 137.

Техническая поддержка

Отдел поддержки и сопровождения программных продуктов:

телефон: (473) 2-711-711 с 9:00 до 18:00 мск.
e-mail: support@relex.ru, market@relex.ru.

С целью повышения качества разрабатываемых программных средств и предоставляемых услуг в компании РЕЛЭКС действует автоматизированная система учёта и обработки рекламаций. Обо всех обнаруженных недостатках и ошибках в программном продукте и/или документации на него просим сообщать нам на Internet–странице [рекламация](#).

Оглавление

Предисловие	1
Назначение документа.....	1
Для кого предназначен документ.....	1
Необходимые предварительные знания.....	1
Дополнительные документы.....	1
Принятые обозначения и соглашения.....	1
Общие сведения	3
Модель геометрических данных OpenGIS	4
Иерархия геометрических классов	4
Класс Geometry.....	5
Класс Point.....	6
Класс Curve	6
Класс LineString	6
Класс Polygon.....	7
Класс GeometryCollection.....	8
Свойства класса	8
Класс MultiPoint.....	8
Свойства класса	8
Класс MultiCurve	8
Свойства класса	9
Класс MultiLineString.....	9
Класс MultiSurface.....	9
Класс MultiPolygon.....	9
Форматы геометрических типов данных	11
Текстовый формат	11
Двоичный формат.....	13
Создание/добавление столбцов геометрического типа	17
Загрузка геометрических данных	18
Загрузка с помощью WKT-формата.....	19
Создание точки.....	19
Создание ломаной линии.....	20
Создание многоугольника	20
Создание группы точек.....	21
Создание группы ломаных линий.....	21
Создание группы многоугольников	22
Создание группы геометрических объектов	23
Создание прямоугольника	23
Создание прямой линии	23
Создание круга	24
Создание произвольного геометрического объекта	24
Загрузка с помощью WKB-представления	24
Создание точки.....	24
Создание ломаной линии.....	25
Создание многоугольника	26

Оглавление

Создание группы точек	26
Создание группы ломаных линий	26
Создание группы многоугольников	27
Создание группы геометрических объектов	27
Создание произвольного геометрического объекта	28
Выборка значений геометрических типов	29
Выборка в WKT-формате	30
Выборка в WKB-формате	30
Средства обработки геометрических типов данных	32
Общие функции	32
Получить название геометрического типа	32
Получить размерность геометрического типа	32
Получить идентификатор системы координат	33
Получить минимальный ограничивающий прямоугольник	33
Получить границу значения геометрического типа	34
Проверить существование значения геометрического типа	34
Проверить сложность геометрического объекта	35
Частные функции	36
Функции для работы с точкой	36
Получить X-координату точки	36
Получить Y-координату точки	36
Функции для работы с ломаной линией	37
Получить координаты начала ломаной линии	37
Получить координаты конца ломаной линии	37
Получить координаты узла ломаной линии	37
Получить длину ломаной линии	38
Получить количество узлов ломаной линии	39
Проверка выпуклости ломаной линии	39
Проверка замкнутости ломаной линии	40
Функции для работы с группой ломаных линий	40
Получить длину группы ломаных линий	40
Проверка замкнутости группы ломаных линий	41
Функции для работы с многоугольником	41
Определение площади многоугольника	41
Определение количества пересекающихся областей многоугольника	42
Определение внешней границы многоугольника	42
Получение заданной пересекающейся области многоугольника	42
Определение геометрического центра многоугольника	43
Получение первой точки внешней границы многоугольника	44
Функции для работы с группой многоугольников	44
Определение площади группы многоугольников	44
Определение геометрического центра группы многоугольников	45
Получение первой точки внешней границы группы многоугольников	45
Функции для работы с группой геометрических объектов	46
Определение количества объектов в группе	46
Получить заданный объект группы	46
Геометрические функции	48
Пересечение геометрических объектов	48
Объединение геометрических объектов	48
Разность геометрических объектов	49
Симметричная разность геометрических объектов	50

Расстояние между объектами	51
Буферный геометрический объект	51
Выпуклая оболочка объекта	53
Проверка совпадения объектов	53
Проверка пересечения объектов	54
Проверка перекрытия объектов	55
Проверка разъединения объектов	56
Проверка вложенности объектов (вариант 1)	56
Проверка вложенности объектов (вариант 2)	57
Проверка касания объектов	58
Проверка скрещивания объектов	58
Оптимизации работы с геометрическими данными	59
Управление вводом геометрических данных	62
Проверка корректности геометрических данных	63
Указатель функций	64

Предисловие

Назначение документа

Документ содержит описание работы с геометрическими типами данных СУБД ЛИНТЕР.

Документ может использоваться для работы с любой версией СУБД ЛИНТЕР. Особенности конкретных версий оговариваются по тексту.

Для кого предназначен документ

Документ предназначен для программистов, разрабатывающих приложения с использованием геометрических (географических) данных.

Необходимые предварительные знания

Для работы с геометрическими типами данных необходимо;

- знать основы реляционных баз данных;
- владеть языком баз данных SQL для СУБД ЛИНТЕР.



Дополнительные документы

- СУБД ЛИНТЕР. Справочник по SQL.
- СУБД ЛИНТЕР. Справочник кодов завершения.
- СУБД ЛИНТЕР. Командный интерфейс.

Принятые обозначения и соглашения

<u>Обозначение</u>	<u>Пример</u>	<u>Значение</u>
Курсив	<i>Растровым</i> называется изображение...	Новый термин в тексте
Полужирный шрифт	В этом случае необходимо переносить все физические файлы.	Выделение в тексте
Подчеркнутый шрифт	Подробную информацию о работе программы можно получить на сайте www.dmk.ru .	Адреса страниц Internet
Текст, разделенный знаком ⇒	Выполните команду View ⇒ Properties (Вид ⇒ Свойства).	Последовательность выполнения команд
Текст, заключенный в < >, со знаком + между ними	<Ctrl>+<C>	В < > заключаются клавиши клавиатуры, знак + означает сочетание клавиш
Крупный моноширинный текст	SQL> _q	Текст командной строки

Предисловие

<u>Обозначение</u>	<u>Пример</u>	<u>Значение</u>
Мелкий моноширинный текст	Page Time Count	Текст программы
Заглавные буквы	BROWSE	Названия команд, слова, зарезервированные в SQL, ключевые слова
Курсив в <>	<return statement>	Определяемый элемент синтаксической конструкции
Символ ::=		Равенство по определению. Слева от знака стоит определяемое понятие, справа – собственно определение понятия
Квадратные скобки []	DBSTORE [-d -n -o -p -r -t -u]	Необязательные элементы конструкции. В данном примере ключи не являются обязательными элементами команды
Вертикальная черта	<return value> ::= <value expression> NULL	Указывает на то, что все предшествующие ей элементы списка являются необязательными и могут быть заменены любым другим элементом списка после этой черты
Фигурные скобки { }	CODEPAGE {866 1251 KOI8}	Указывают на то, что все, находящееся внутри них, является единым целым
Многоточие «...»	Характеристики столбца MAKE CHAR(20) MODEL CHAR(20) ... SQL>	Означает, что предшествующая часть может быть повторена любое количество раз
Многоточие, внутри которого находится запятая «,...»		Указывает на то, что предшествующая часть оператора, состоящая из нескольких элементов, разделенных запятыми, может иметь произвольное число повторений
Текст со знаком  на сером фоне	 Если конфигурация страницы-шаблона не учитывала свойств,, команда будет выполнена некорректно.	Примечание

Общие сведения

В СУБД ЛИНТЕР для поддержки геометрических типов данных реализовано подмножество среды SQL с геометрическими типами (SQL with Geometry Types), спецификация которой предложена консорциумом OpenGIS. Столбец таблицы, в котором хранятся геометрические данные, имеет геометрический тип. Спецификация описывает набор геометрических типов данных SQL, а также функций, предназначенных для создания и анализа значений указанных геометрических типов данных.

Геометрические типы данных позволяют генерировать, сохранять и анализировать географические данные, которые отражают элементы окружающего мира:

- географические объекты (например, гора, водоем, город);
- территории (например, область, задаваемая почтовым индексом);
- местоположения (например, перекресток, как специфическое место пересечения двух дорог).

Модель геометрических данных OpenGIS

Набор геометрических типов данных в спецификации для SQL основан на геометрической модели OpenGIS. В этой модели каждый геометрический объект обладает следующими свойствами:

- связан с системой координат, описывающей координатное пространство, в котором определен объект;
- принадлежит некоторому геометрическому классу.

Иерархия геометрических классов


Спецификация OpenGIS предусматривает следующую иерархию геометрических классов:

- Geometry
 - Point
 - Curve
 - * LineString
 - ◆ LinearRing
 - Surface
 - * Polygon
 - GeometryCollection
 - * MultiPoint
 - * MultiCurve
 - ◆ MultiLineString
 - * MultiSurface
 - ◆ MultiPolygon

Класс `Geometry` – базовый класс (неинстанцируемый). Инстанцируемые подклассы `Geometry` ограничены размерностью 0, 1 и 2 и существуют в двумерном координатном пространстве. Все объекты инстанцируемых классов предполагаются топологически замкнутыми (то есть каждый объект включает собственную границу).

Подклассами `Geometry` являются классы:

- 1) размерности 0 – точка (`Point`);
- 2) размерности 1 – кривая (`Curve`) и ее подкласс `LineString` с подклассом `LinearRing`;
- 3) размерности 2 – поверхность (`Surface`) и ее подкласс `Polygon`;
- 4) составных объектов – набор объектов (`GeometryCollection`):
 - размерности 0 – `MultiPoint` (набор классов `Point`);
 - размерности 1 – `MultiLineString` (набор классов `LineString`);
 - размерности 2 – `MultiPolygon` (набор классов `Polygon`)

 Подклассы `MultiCurve` и `MultiSurface` введены как абстрактные суперклассы для обобщения интерфейсов подклассов `Curve` и `Surface` соответственно.

`Geometry`, `Curve`, `Surface`, `MultiCurve` и `MultiSurface` определены как неинстанцируемые классы. Они определяют общий набор методов для своих подклассов.

`Point`, `LineString`, `Polygon`, `GeometryCollection`, `MultiPoint`, `MultiLineString`, `MultiPolygon` являются инстанцируемыми классами (в иерархии объектов выделены жирным шрифтом).

Класс Geometry

Geometry – корневой класс иерархии. Каждый объект класса Geometry описывается множеством его свойств. Подклассы корневого класса Geometry наследуют свойства класса Geometry и имеют собственные свойства.

Свойства класса

Класс Geometry имеет следующие свойства:

- 1) тип, к которому принадлежит объект геометрического типа. Каждый объект геометрического типа принадлежит одному из инстанцируемых классов в иерархии;
- 2) идентификатор системы координат объекта (SRID) геометрического типа, описывающего координатное пространство, в котором определен объект геометрического типа;
- 3) координаты объекта геометрического типа в системе координат, представленные в виде вещественных чисел двойной точности (8 байтов, double). Все непустые объекты геометрического типа имеют не менее одной пары координат (X, Y). Пустые конфигурации не содержат координат;
- 4) внутренняя область, граница и внешняя область. Все объекты геометрического типа занимают некоторую позицию в пространстве. Внешняя область объекта геометрического типа – все пространство, не занятое им. Внутренняя область – пространство, занятое объектом геометрического типа. Граница – разделитель между внутренней областью геометрии и внешней;
- 5) минимальный ограничительный прямоугольник объекта геометрического типа. Это – ограничительная граница вокруг объекта геометрического типа, сформированная минимумами и максимумами координат (X, Y):

$$((\text{MINX MINY}, \text{MAXX MINY}, \text{MAXX MAXY}, \text{MINX MAXY}, \text{MINX MINY}))$$
- 6) простой (simple) или непростой (non-simple). Каждый геометрический класс определяет собственные критерии того, является ли принадлежащий ему объект простым или непростым;
- 7) замкнутый (closed) или не замкнутый (not closed). Значения геометрических объектов некоторых типов (LineString, MultiLineString) или замкнуты, или не замкнуты. Каждый геометрический класс определяет собственные критерии того, является ли принадлежащий ему объект замкнутым или не замкнутым;
- 8) пустой (empty) или не пустой (not empty). Объект геометрического типа пуст, если он не имеет точек. Внешняя область, внутренняя область и граница пустого объекта геометрического типа не определены, то есть они представлены NULL-значением. Пустой объект геометрического типа всегда простой. Пустой объект геометрического типа имеет площадь 0;
- 9) размерность объекта геометрического типа может иметь значение: -1, 0, 1 или 2:
 - размерность – 1 установлена для пустых объектов геометрического типа;
 - размерность 0 установлена для объекта геометрического типа без длины и с нулевой площадью;
 - размерность 1 установлена для объекта геометрического типа с ненулевой длиной и нулевой площадью;
 - размерность 2 установлена для объекта геометрического типа с ненулевой площадью.

Подкласс Point имеет размерность ноль.

Подкласс LineStrings имеет размерность 1.

Подкласс Polygon имеет размерность 2.

Размерность объектов классов MultiPoint, MultiLineString и MultiPolygon та же самая, что и у элементов, из которых они состоят.

Класс Point

Класс представляет точку в пространстве координат.

Свойства класса

Класс Point имеет следующие свойства:

- 1) X-координата;
- 2) Y-координата;
- 3) граница Point – пустое множество;
- 4) размерность Point равна нулю.

Класс Curve

Класс определяет простой одномерный геометрический объект, представляемый последовательностью точек. Подкласс Curve определяет тип интерполяции линии, соединяющей точки. Curve – неинстанцируемый класс, единственный инстанцируемый подкласс – LineString (ломаная).

Свойства класса

Класс Curve имеет следующие свойства:

- 1) координаты точек кривой;
- 2) размерность кривой равна 1;
- 3) кривая проста, если не проходит через одну и ту же точку дважды (исключая случай замкнутой кривой);
- 4) кривая замкнута, если ее начальная точка равна ее конечной точке;
- 5) граница замкнутой кривой – пустое множество;
- 6) граница незамкнутой кривой состоит из ее начальной и конечной точки;
- 7) кривая, которая является простой и замкнутой – кольцо (Ring) .

Класс LineString

Класс LineString (ломаная) определяет кривую (Curve) с линейной интерполяцией линий, соединяющих точки.

Свойства класса

Класс LineString имеет следующие свойства:

- 1) координаты сегментов LineString определяются последовательностями пар точек;
- 2) LineString является линией (Line), если состоит из двух точек;
- 3) LineString является LinearRing, если кривая замкнута и проста.

Класс Surface

Класс Surface (поверхность) определяет геометрический объект с размерностью 2.

Свойства класса

Класс Surface имеет следующие свойства:

- 1) размерность Surface равна 2;
- 2) спецификация OpenGIS определяет класс Surface как плоскую поверхность, состоящую из единственной ломаной, образующей «внешнюю границу» и из нулевого или большего количества ломаных, образующих «внутреннюю» границу;
- 3) границы плоской поверхности – набор замкнутых кривых (Curve), соответствующих ее внешней и внутренней областям.

Единственным инстанцируемым подклассом класса Surface является класс Polygon.

Класс Polygon

Класс Polygon (многоугольник) определяет плоскую поверхность (Surface) с одной внешней границей и нулем или большим количеством внутренних границ. Каждая внутренняя граница определяет отверстие в Polygon.

Свойства класса

Класс Polygon имеет следующие свойства:

- 1) граница Polygon состоит из набора LinearRings (простой и замкнутый вариант LineString), которые составляют его внешние и внутренние границы;
- 2) никакие два кольца в Polygon не могут пересекаться, кроме как по тангенсу (касаться);
- 3) Polygon не должен содержать вырезов в виде линии, выбросов или «проколов»;
- 4) внутренняя область каждого Polygon – связанный набор точек;
- 5) внешняя область Polygon с одним или более отверстиями представляет собой несвязанную область. Каждое отверстие определяет связанный компонент внешней области.

Все объекты класса Polygon, удовлетворяющие перечисленным выше свойствам, являются простыми объектами.

Примеры многоугольников с 1, 2 и 3 кольцами соответственно (см. Рис. 1)

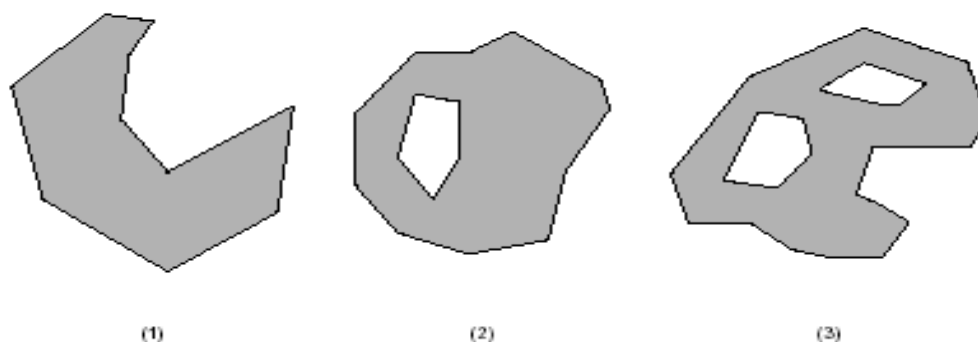


Рис. 1 – Примеры многоугольников

Класс GeometryCollection

Класс `GeometryCollection` определяет геометрический тип, который является набором из нуля и более объектов других геометрических типов.

Все элементы в `GeometryCollection` должны быть в одной и той же системе координат. Класс `GeometryCollection` не имеет других ограничений на свои элементы.

Свойства класса

Подклассы `GeometryCollection` (см. ниже) могут иметь ограничения, основанные на:

- типе элементов;
- размерности;
- других ограничениях на степень пространственного перекрытия между элементами.

Класс MultiPoint

Класс `MultiPoint` (набор точек) определяет набор графических объектов типа точка (`Point`). Точки в наборе не связаны между собой и не упорядочены.

Свойства класса

Класс `MultiPoint` имеет следующие свойства:

- 1) нулевая размерность;
- 2) класс является простым, если никакие две точки в нем не равны (то есть не имеют идентичных координат);
- 3) граница класса – пустое множество.

Класс MultiCurve

Класс `MultiCurve` (набор кривых) определяет набор графических объектов типа кривая (`Curve`). `MultiCurve` – неинстанцируемый класс.

Свойства класса

Класс `MultiCurve` имеет следующие свойства:

- 1) класс является простым, только если все его элементы просты, а одиночные пересечения между любыми двумя элементами происходят в точках, находящихся на границах обоих элементов;
- 2) граница класса определяется с помощью применения правила объединения "mod 2": точка находится в границе класса, если она расположена на границе нечетного числа элементов класса;
- 3) размерность класса равна 1;
- 4) класс замкнут, если все его элементы замкнуты;
- 5) граница закрытого класса – всегда пустое множество.

Класс `MultiLineString`

Класс `MultiLineString` (набор ломаных) определяет набор графических объектов типа `LineString`.

Класс `MultiSurface`

Класс `MultiSurface` (набор поверхностей) определяет набор графических объектов типа поверхность (`Surface`). `MultiSurface` – неинстанцируемый класс.

Свойства класса

Класс `MultiSurface` имеет следующие свойства:

- 1) внутренние области любых двух поверхностей не могут пересекаться;
- 2) границы любых двух элементов могут пересечься не более, чем в конечном числе точек.

Единственный инстанцируемый подкласс класса `MultiSurface` – это `MultiPolygon`.

Класс `MultiPolygon`

Класс `MultiPolygon` (набор многоугольников) определяет набор графических объектов типа многоугольник (`Polygon`).

Свойства класса

Класс `MultiPolygon` имеет следующие свойства:

- 1) внутренние поверхности двух многоугольников, которые являются элементами `MultiPolygon`, не могут пересечься;
- 2) границы любых двух многоугольников, которые являются элементами `MultiPolygon`, не могут пересекаться и могут касаться только в конечном числе точек.



Пересечение уже запрещается первым утверждением.

- 3) `MultiPolygon` не может содержать вырезов линий, выбросов или проколов; `MultiPolygon` – правильный, замкнутый набор точек;

- 4) внутренняя область MultiPolygon, содержащая более чем один многоугольник, не связана, число связанных частей внутренней области MultiPolygon равно числу многоугольников в MultiPolygon;
- 5) размерность класса равна 2;
- 6) граница MultiPolygon – набор закрытых кривых (LineStrings), образованных границами его элементов – многоугольников;
- 7) каждая кривая (Curve) на границе MultiPolygon находится на границе точно одного многоугольника, и каждая кривая (Curve) на границе многоугольника находится также и на границе MultiPolygon.

Форматы геометрических типов данных

Представление (описание) геометрических типов данных возможно двумя способами:

- 1) в текстовом виде – WKT-формат (Well-Known Text);
- 2) в двоичном виде – WKB-формат (Well-Known Binary).

Текстовый формат

Текстовое представление данных (WKT-формат) определяет формат в виде текстовой строки, содержащей:

- 1) имя типа объекта (Point, Linestring, Polygon, Multipoint, Multilinestring, Multipolygon, Geometrycollection, Box, Circle);
- 2) пары чисел как координаты точек;
- 3) скобки для группировки элементов;
- 4) символы табуляции и перевода строки.

Синтаксис WKT-представления

```
<WKT-представление геометрического объекта> :=
  <представление точки>
  | <представление прямой линии>
  | <представление ломаной линии>
  | <представление многоугольника>
  | <представление набора точек>
  | <представление набора ломаных линий>
  | <представление набора многоугольников>
  | <представление GeometryCollection Tagged Text>
  | <представление прямоугольника>
  | <представление круга>

<представление точки>:=
  POINT <описание точки>
  | <описание точки>:: POINT

<описание точки> ::= { EMPTY | ([<координаты
точки>] ) }
<координаты точки> ::=
  <x> <y>
  | (<x> <y>)
  | (<x>, <y>)

<x> := вещественный литерал
<y> := вещественный литерал

<представление прямой линии> ::=
```

```
LINE <описание прямой линии>
<описание прямой линии> ::=
{ EMPTY | (<пара точек>) }

<пара точек> ::=
<описание точки> <описание точки>
| <описание точки> , <описание точки>

<представление ломаной линии> ::=
LINESTRING <описание ломаной линии>
<описание ломаной линии> ::=
{ EMPTY | (<описание точки>{ , <описание точки>
...}) }

<представление многоугольника> ::=
POLYGON <описание многоугольника>

<описание многоугольника> ::=
{ EMPTY | (<описание ломаной линии>{ , <описание
ломаной линии>...}) }

<представление набора точек> ::=
MULTIPOINT <описание набора точек>

<описание набора точек> ::=
{ EMPTY | (<описание точки>{ , <описание
точки>...}) }

<представление набора ломаных линий> ::=
MULTILINESTRING <описание набора ломаных
линий>

<описание набора ломаных линий> ::=
{ EMPTY | (<описание ломаной линии>{ , <описание
ломаной линии>...}) }

<представление набора многоугольников> ::=
MULTIPOLYGON <описание набора многоугольников>

<описание набора многоугольников> ::=
{ EMPTY | (<описание многоугольника>{ , <описание
многоугольника>...}) }

<представление набора геометрических
объектов> ::=
GEOMETRYCOLLECTION <описание набора
геометрических объектов >

<описание набора геометрических объектов> ::=
{ EMPTY | (<WKT-представление геометрического объекта>{ ,
<WKT-представление геометрического объекта>...}) }
```

```

<представление прямоугольника> ::=
    BOX <описание прямоугольника>
    <описание прямоугольника> ::=
        {EMPTU | (<пара точек>)}

<представление круга> ::=
    CIRCLE <описание круга>
    <описание круга> ::=
        {EMPTU | (<описание точки> <радиус>)}

<радиус> := вещественный литерал
    
```

Примеры WKT-представления

POINT(10 10)	Point
LINESTRING(10 10, 20 20, 30 40)	LineString с тремя точками
POLYGON((10 10, 10 20, 20 20, 20 15, 10 10))	Polygon с одной внешней и без внутренних границ
MULTIPOINT(10 10, 20 20)	MultiPoint из двух точек
MULTILINESTRING((10 10, 20 20), (15 15, 30 15))	MultiLineString из двух ломаных
MULTIPOLYGON(((10 10, 10 20, 20 20, 20 15, 10 10)), ((60 60, 70 7, 80 60, 60 60)))	MultiPolygon из двух полигонов
GEOMETRYCOLLECTION(POINT(10 10), POINT(30 30), LINESTRING(15 15, 20 20))	GeometryCollection из двух Point и одной LineString

Двоичный формат

Двоичное представление данных (WKB-формат) определяет формат в виде структур данных, описание которых содержится, как правило, в заголовочном файле приложения (см. пример ниже). Каждый элемент содержит спецификатор порядка байт (для работы на различных аппаратных платформах). Все коды элементов и их компонентов выражаются 4-байтовыми беззнаковыми целыми числами, координаты – вещественными числами двойной точности.

Пример заголовочного файла для описания данных в WKB-формате

Основные типы WKB

```

// byte : 8-bit unsigned integer (1 byte)
// uint32 : 32-bit unsigned integer (4 bytes)
// double : double precision number (8 bytes)
enum wkbGeometryType
{
    wkbPoint = 1,
    wkbLineString = 2,
    wkbPolygon = 3,
    wkbMultiPoint = 4,
    wkbMultiLineString = 5,
}
    
```

```
wkbMultiPolygon      = 6,
wkbGeometryCollection = 7
}
enum wkbByteOrder
{
    wkbXDR = 0,    // Big Endian
    wkbNDR = 1    // Little Endian
}
```

Вспомогательные типы WKВ

```
// Building Blocks : Point, LinearRing
Point
{
    double x;
    double y;
}
LinearRing
{
    uint32 numPoints;
    Point  points[numPoints];
}
```

WKВ представление геометрических типов

```
WKBPoint
{
    byte    byteOrder;
    uint32  wkbType;    // 1
    Point   point;
}
WKBLineString
{
    byte    byteOrder;
    uint32  wkbType;    // 2
    uint32  numPoints;
    Point   points[numPoints];
}
WKBPolygon
{
    byte    byteOrder;
    uint32  wkbType;    // 3
    uint32  numRings;
    LinearRing rings[numRings];
}
WKBMultiPoint
{
    byte    byteOrder;
    uint32  wkbType;    // 4
    uint32  num_wkbPoints;
    WKBPoint WKBPoints[num_wkbPoints];
}
WKBMultiLineString
{
    byte    byteOrder;
```

```

        uint32      wkbType;    // 5
        uint32      num_wkbLineStrings;
        WKBLineString WKBLineStrings[num_wkbLineStrings];
    }
    wkbMultiPolygon
    {
        byte        byteOrder;
        uint32      wkbType;    // 6
        uint32      num_wkbPolygons;
        WKBPolygon  wkbPolygons[num_wkbPolygons];
    }
    WKBCircle
    {
        byte        byteOrder;
        uint32      wkbType;
        Point       point;
        double      r;          // радиус
    }

    WKBGeometry
    {
        union
        {
            WKBPoint          point;
            WKBLineString      linestring;
            WKBPolygon         polygon;
            WKBGeometryCollection collection;
            WKBMultiPoint      mpoint;
            WKBMultiLineString mlinestring;
            WKBMultiPolygon    mpolygon;
        }
    }
    WKBGeometryCollection
    {
        byte        byte_order;
        uint32      wkbType;    // 7
        uint32      num_wkbGeometries;
        WKBGeometry wkbGeometries[num_wkbGeometries];
    }

```

 Тип данных BOX не имеет собственного идентификатора типа (wkbType), так как топологически не отличается от POLYGON, а типы LINE и CIRCLE – имеют:

```

wkbCircle    = 0x81
wkbLine      = 0x82

```

Пример WKB -формата

WKB- представление точки POINT (1,1) – последовательность из 21 байта:

010100000000000000000000F03F000000000000F03F

где:

Форматы геометрических типов данных

Byte order: 01

WKB type: 01000000

X : 0000000000000F03F

Y : 0000000000000F03F

Создание/добавление столбцов геометрического типа

Добавление столбца с геометрическим типом данных при создании таблицы выполняется с помощью стандартного SQL-оператора CREATE OR REPLACE TABLE.

В спецификации типа данных столбца задается один из допустимых геометрических типов:

POINT	точка
LINESTRING[(n)]	ломаная линия
POLYGON[(n)]	многоугольник
MULTIPOINT[(n)]	набор точек
MULTILINESTRING[(n)]	набор ломаных линий
MULTIPOLYGON[(n)]	набор многоугольников
BOX	прямоугольник
LINE	прямая линия
CIRCLE	круг
GEOMETRYCOLLECTION[(n)]	набор геометрических объектов
GEOMETRY	обобщенный геометрический тип (может содержать любой геометрический объект)

Синтаксические правила

- 1) формат хранения значений типа LINE совпадает с форматом LINESTRING, имеющим две точки;
- 2) все геометрические типы данных унаследованы от типа varbyte и имеют максимальную длину 4000 байт. В связи с этим для типов с переменной длиной значений (Linestring, Polygon, Multipoint, Multilinestring, Geometrycollection) установлено ограничение: по умолчанию размер значений перечисленных типов составляет 1024 байт. Для того чтобы создавать значения с большим размером, надо при создании таблицы явно указать этот размер, например:
- 3) create or replace table LINESTRING_TEST(LS LINESTRING(4000));
- 4) тип данных BOX определяет прямоугольник. Две точки, которые являются вершинами прямоугольника, задают координаты диагонали. Данный формат автоматически преобразуется в формат POLYGON с пятью точками:

Конструкция

BOX (1 1, 2 2)

эквивалентна

POLYGON (1 1,1 2,2 2,2 1,1 1)

Загрузка геометрических данных

5) тип данных `LINE` определяет прямую линию. Формат хранения этого типа эквивалентен формату хранения ломаной линии с двумя точками:

Конструкция

```
LINE (1 1, 2 2)
```

Имеет то же WKB представление (за исключением числа, задающего тип), что и

```
LINESTRING (1 1, 2 2)
```

6) тип данных `CIRCLE` определяет круг:

`CIRCLE(1 1, 2)` – круг с координатой центральной точки (1,1) и с радиусом 2.

7) В столбец с типом данных `GEOMETRYCOLLECTION` можно загружать все типы геометрических данных, в столбец `MULTIPOINT` – данные типа `POINT`, в столбец `MULTILINESTRING` – данные типа `LINESTRING`, в столбец `MULTIPOLYGON` – данные типа `POLYGON`.

Пример

```
CREATE TABLE GEO_TEST (p POINT, ls LINESTRING);
```

Добавление столбца с геометрическим типом данных в уже существующую таблицу выполняется с помощью стандартного SQL-оператора `ALTER TABLE ADD COLUMN`.

Пример

```
ALTER TABLE GEO_TEST ADD COLUMN pl POLYGON;
```

8) для столбца с обобщенным типом данных `GEOMETRY` может применяться любая геометрическая функция, однако, если хранящийся в таком столбце геометрический объект не является корректным для вызываемой функции, будет выдаваться соответствующий код завершения СУБД `ЛИНТЕР`.

Загрузка геометрических данных

Загрузка геометрических данных в таблицу БД выполняется с помощью стандартного SQL-оператора `INSERT` и набора встроенных в СУБД функций, преобразующих WKT/WKB формат представления данных в соответствующий столбцу геометрический тип данных, либо с помощью прямой типизации данных (когда тип можно определить из контекста), например:

```
insert into GEOTABLE (COLUMN_WITH_TYPE_POINT) values ('POINT (1 1)');
```

 В связи с добавлением большого числа новых функций их названия могут случайно совпасть с уже существующими в БД именами столбцов и таблиц. Чтобы исключить эту неоднозначность, в СУБД ЛИНТЕР предусмотрен режим совместимости с предыдущими версиями. Для включения этого режима следует запустить ядро СУБД с ключом /COMPATIBILITY=GEOPREFIX. В этом случае к именам геометрических функций добавляется префикс "LIN_". Например: стандартные имена геометрических SQL-функций AsText() и PointFromText() будут заменены на имена Lin_AsText и Lin_PointFromText :

```
SQL>select lin_AsText(lin_PointFromText('POINT (1 1)'));

| POINT (1 1) |
INL : выдано строк      : 1
```

Загрузка с помощью WKT-формата

Создание значений специфических геометрических типов (за исключением BOX, LINE, CIRCLE) обеспечивается с помощью соответствующих каждому из типов функций.

Создание точки

Функция

Преобразование WKT-представления точки в соответствующий геометрический тип данных.

Спецификация

PointFromText (<wkt> [, <srid>])

<wkt> – WKT-представление точки

<srid> – идентификатор системы координат данной точки.

Синтаксические правила

Если аргумент <srid> не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение типа POINT, соответствующее внутреннему представлению в БД типа данных POINT (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Пример

```
create or replace table geo_test (p point);
insert into geo_test(p) values (pointfromtext('point(1.45
3.06)'));
insert into geo_test values (pointfromtext('point 1.45 3.06',
1));
insert into geo_test(p) values (pointfromtext('point (25,
67)', 45));
```

Создание ломаной линии

Функция

Преобразование WKT-представления ломаной линии в соответствующий геометрический тип данных.

Спецификация

LineFromText | **LineStringFromText** (<wkt> [,<srid>])

<wkt> – WKT-представление ломаной линии;

<srid> – идентификатор системы координат данной ломаной линии.

Синтаксические правила

Если аргумент <srid> не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение типа LINESTRING, соответствующее внутреннему представлению в БД типа данных LINESTRING (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Примеры

```
create or replace table geo_test (l linestring);
insert into geo_test(l) values (linefromtext('linestring(1 1,2
2, 1 2, 5 7)'));
insert into geo_test values (linestringfromtext('linestring (2
2), (4 5), (8 15)', 3));

CREATE OR REPLACE TABLE LS_TEST( L LINESTRING );
INSERT INTO LS_TEST VALUES( LineFromText('LINESTRING (2 2,1
0,10.1 2)') );
INSERT INTO LS_TEST VALUES(' (0,1),3 3, (0,0), (10,10) 11
21)::LINESTRING );
INSERT INTO LS_TEST VALUES( 'LINESTRING (1 2,4 1,67 85)' );
```

Создание многоугольника

Функция

Преобразование WKT-представления многоугольника в соответствующий геометрический тип данных.

Спецификация

PolyFromText | **PolygonFromText** (<wkt> [,<srid>])

<wkt> – WKT-представление многоугольника;

<srid> – идентификатор системы координат данного многоугольника.

Синтаксические правила

Если аргумент <srid> не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение типа POLYGON, соответствующее внутреннему представлению в БД типа данных POLYGON (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Примеры

```
CREATE OR REPLACE TABLE POLYGON_TEST( P POLYGON );
INSERT INTO POLYGON_TEST VALUES
    (PolyFromText('POLYGON((2 2,1 1,0 0,10 0,2 2))') );
INSERT INTO POLYGON_TEST VALUES
    ('(( 0 0,0 3,3 3,3 0, 0 0), (0 0, 1 1, 1 0.55, 0 0))'::POLYGON );
INSERT INTO POLYGON_TEST VALUES
    ('POLYGON (1 2,4 1,67 85,1 2)');
```

Создание группы точек

Функция

Преобразование WKT-представления группы точек в соответствующий геометрический тип данных.

Спецификация

MPointFromText | **MultiPointFromText** (<wkt> [,<srid>])

<wkt> – WKT-представление группы точек;

<srid> – идентификатор системы координат данного набора точек.

Синтаксические правила

Если аргумент <srid> не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение типа MULTIPOINT, соответствующее внутреннему представлению в БД типа данных MULTIPOINT (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Пример

```
CREATE OR REPLACE TABLE MULTIPOINT_TEST( MPO MULTIPOINT );
INSERT INTO MULTIPOINT_TEST VALUES
    (MPointFromText('MULTIPOINT (2 2,1 1,2 2,1 0,0 0)'));
INSERT INTO MULTIPOINT_TEST VALUES
    ('( 0 0,0 3,3 0,0 0, 0 0, 1 1, 1 0.55, 0 0 )'::MULTIPOINT );
INSERT INTO MULTIPOINT_TEST VALUES( 'MULTIPOINT (1 2,4 1,67 85,1 2)');
```

Создание группы ломаных линий

Функция

Преобразование WKT-представления группы ломаных линий в соответствующий геометрический тип данных.

Загрузка геометрических данных

Спецификация

MLineFromText | **MultiLineStringFromText** (<wkt> [,<srid>])

<wkt> – WKT-представление группы ломаных линий;

<srid> – идентификатор системы координат данной группы ломаных линий.

Синтаксические правила

Если аргумент <srid> не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение типа MULTILINESTRING, соответствующее внутреннему представлению в БД типа данных MULTILINESTRING (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Пример

```
CREATE OR REPLACE TABLE MULTILINESTRING_TEST( ML MULTILINESTRING );
INSERT INTO MULTILINESTRING_TEST VALUES
  (MLineFromText('MULTILINESTRING((2 2,1 1),(0 0,10 0))'));
INSERT INTO MULTILINESTRING_TEST VALUES
  ('((0 0,0 3,3 0),(0 0, 0 100, 1 1, 1 0.55 ))'::MULTILINESTRING );
INSERT INTO MULTILINESTRING_TEST VALUES
  ('MULTILINESTRING ((1 2,4 1),(67 85,1 2))' );
```

Создание группы многоугольников

Функция

Преобразование WKT-представления группы многоугольников в соответствующий геометрический тип данных.

Спецификация

MPolyFromText | **MultiPolygonFromText** (<wkt> [,<srid>])

<wkt> – WKT-представление группы многоугольников;

<srid> – идентификатор системы координат данной группы многоугольников.

Синтаксические правила

Если аргумент <srid> не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение типа MULTIPOLYGON, соответствующее внутреннему представлению в БД типа данных MULTIPOLYGON (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Пример

```
CREATE OR REPLACE TABLE MULTIPOLYGON_TEST( MP MULTIPOLYGON );
INSERT INTO MULTIPOLYGON_TEST VALUES
  (MPolyFromText('MULTIPOLYGON (((2 2,1 1,0 0,10 0,2 2)))') );
INSERT INTO MULTIPOLYGON_TEST VALUES
  ('((0 0,0 3,3 3 0,0 0),(0 0,1 1,1 0.55,0 0))'::MULTIPOLYGON );
INSERT INTO MULTIPOLYGON_TEST VALUES (
  'MULTIPOLYGON (((1 2) (4 1) 67 85 (1,2)))');
```

Создание группы геометрических объектов

Функция

Преобразование WKT-представления группы геометрических объектов в соответствующий геометрический тип данных.

Спецификация

GeomCollFromText (<wkt> [,<srid>])

<wkt> – WKT-представление группы геометрических объектов;

<srid> – идентификатор системы координат данной группы геометрических объектов.

Синтаксические правила

Если аргумент <srid> не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение типа GEOMETRYCOLLECTION, соответствующее внутреннему представлению в БД типа данных GEOMETRYCOLLECTION (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Пример

```
CREATE OR REPLACE TABLE GEOMETRYCOLLECTION_TEST( G GEOMETRYCOLLECTION );
INSERT INTO GEOMETRYCOLLECTION_TEST VALUES
    (GeomCollFromText('GEOMETRYCOLLECTION (LINESTRING((2,2),(1,0)),
    POLYGON( ( 0 0,0 3,3 0,0 0), (0 0, 1 1, 1 0.55, 0 0) ) )') );
INSERT INTO GEOMETRYCOLLECTION_TEST VALUES
    ('(POINT(1 1), LINESTRING (1 4, 2 5))'::GEOMETRYCOLLECTION );
INSERT INTO GEOMETRYCOLLECTION_TEST VALUES
    ('GEOMETRYCOLLECTION (POINT(1 1),POINT(10 10),POINT(100 100))');
```

Создание прямоугольника

Создание прямоугольника выполняется с помощью простой типизации объекта, например:

```
CREATE OR REPLACE TABLE BOX_TEST( B BOX );
INSERT INTO BOX_TEST VALUES( PolyFromText('BOX (2 2,1 0)') );
```

```
INSERT INTO BOX_TEST VALUES('(((0,1),3 3))'::BOX);
```

```
INSERT INTO BOX_TEST VALUES( 'BOX (1 2, 4 1) ' );
```

Создание прямой линии

Создание отрезка линии выполняется с помощью простой типизации объекта, например:

```
CREATE OR REPLACE TABLE LINE_TEST( L LINE );
```

Загрузка геометрических данных

```
INSERT INTO LINE_TEST VALUES( '( (0,1), 3 3)'::LINE );
INSERT INTO LINE_TEST VALUES( 'LINE (1 2,4 1)' );
```

Создание круга

Создание окружности выполняется с помощью простой типизации объекта, например:

```
CREATE OR REPLACE TABLE CIRCLE_TEST( CR CIRCLE );
INSERT INTO CIRCLE_TEST VALUES( 'CIRCLE (1 1,1)' );
INSERT INTO CIRCLE_TEST VALUES( '(1 2) 3'::CIRCLE );
```

Создание произвольного геометрического объекта

Функция

Преобразование WKT-представления любого геометрического объекта в соответствующий геометрический тип данных.

Спецификация

GeomFromText | **GeometryFromText** (<wkt> [, <srid>])

<wkt> – WKT-представление геометрического объекта;

<srid> – идентификатор системы координат данного геометрического объекта.

Синтаксические правила

Если аргумент <srid> не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение инстанцируемого геометрического типа, содержащееся в строковом аргументе этой функции, соответствующее внутреннему представлению в БД полученного при преобразовании геометрического объекта;
- 2) код завершения SQL (при ошибке преобразования).

Пример

```
CREATE OR REPLACE TABLE geom (g GEOMETRY);
create index g on geom;
insert into geom(g) values (GeomFromText('POINT (1 1)',1001));
insert into geom(g) values (GeomFromText('LINESTRING (1 1,2 2,3 3)',1001));
```

Загрузка с помощью WKВ-представления

Создание значений специфических геометрических типов (за исключением BOX, LINE, CIRCLE) обеспечивается с помощью соответствующих каждому из типов функций.

Создание точки

Функция

Преобразование WKВ-представления точки в соответствующий геометрический тип данных.

Спецификация**PointFromWKB** (<wkb> [,<srid>])

<wkb> – WKB-представление точки

<srid> – идентификатор системы координат данной точки.

Синтаксические правила

Если аргумент <srid> не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение типа POINT, соответствующее внутреннему представлению в БД типа данных POINT (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Пример

Программа клиентского приложения должна сама правильно сформировать WKB-представление, например:

```
CREATE OR REPLACE TABLE POINT_TEST( P POINT );
INSERT INTO POINT_TEST VALUES
(PointFromWKB
(hex('010100000000000000000000F03F000000000000F03F')) /* POINT (1
1) */);
```

Создание ломаной линии**Функция**

Преобразование WKB-представления ломаной линии в соответствующий геометрический тип данных.

Спецификация**LineFromWKB** | **LineStringFromWKB** (<wkb> [,<srid>])

<wkb> – WKB-представление ломаной линии;

<srid> – идентификатор системы координат данной ломаной линии.

Синтаксические правила

Если аргумент <srid> не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение типа LINESTRING, соответствующее внутреннему представлению в БД типа данных LINESTRING (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Создание многоугольника

Функция

Преобразование WKB-представления многоугольника в соответствующий геометрический тип данных.

Спецификация

`PolyFromWKB` | `PolygonFromWKB` (`<wkb>` [, `<srid>`])

`<wkb>` – WKB-представление многоугольника;

`<srid>` – идентификатор системы координат данного многоугольника.

Синтаксические правила

Если аргумент `<srid>` не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение типа POLYGON, соответствующее внутреннему представлению в БД типа данных POLYGON (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Создание группы точек

Функция

Преобразование WKB-представления группы точек в соответствующий геометрический тип данных.

Спецификация

`MPointFromWKB` | `MultiPointFromWKB` (`<wb>` [, `<srid>`])

`<wkb>` – WKB-представление группы точек;

`<srid>` – идентификатор системы координат данного набора точек.

Синтаксические правила

Если аргумент `<srid>` не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение типа MULTIPOINT, соответствующее внутреннему представлению в БД типа данных MULTIPOINT (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Создание группы ломаных линий

Функция

Преобразование WKB-представления группы ломаных линий в соответствующий геометрический тип данных.

Спецификация

MLineFromWKB | **MultiLineStringFromWKB** (<wkb> [,<srid>])

<wkb> – WKB-представление группы ломаных линий;

<srid> – идентификатор системы координат данной группы ломаных линий.

Синтаксические правила

Если аргумент <srid> не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение типа MULTILINESTRING, соответствующее внутреннему представлению в БД типа данных MULTILINESTRING (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Создание группы многоугольников

Функция

Преобразование WKB-представления группы многоугольников в соответствующий геометрический тип данных.

Спецификация

MPolyFromWKB | **MultiPolygonFromWKB** (<wkb> [,<srid>])

<wkb> – WKB-представление группы многоугольников;

<srid> – идентификатор системы координат данной группы многоугольников.

Синтаксические правила

Если аргумент <srid> не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение типа MULTIPOLYGON, соответствующее внутреннему представлению в БД типа данных MULTIPOLYGON (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Создание группы геометрических объектов

Функция

Преобразование WKB-представления группы геометрических объектов в соответствующий геометрический тип данных.

Спецификация

GeomCollFromWKB (<wkb> [,<srid>])

<wkb> – WKB-представление группы геометрических объектов;

<srid> – идентификатор системы координат данной группы геометрических объектов.

Синтаксические правила

Если аргумент `<srId>` не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение типа `GEOMETRYCOLLECTION`, соответствующее внутреннему представлению в БД типа данных `GEOMETRYCOLLECTION` (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Создание произвольного геометрического объекта

Функция

Преобразование WKB-представления любого геометрического объекта в соответствующий геометрический тип данных.

Спецификация

`GeomFromWKB` | `GeometryFromWKB` (`<wkb>` [, `<srId>`])

`<wkb>` – WKB-представление геометрического объекта;

`<srId>` – идентификатор системы координат данного геометрического объекта.

Синтаксические правила

Если аргумент `<srId>` не задан, по умолчанию используется значение 0.

Возвращаемое значение

- 1) значение инстанцируемого геометрического типа, содержащееся в аргументе этой функции и соответствующее внутреннему представлению в БД полученного при преобразовании геометрического объекта;
- 2) код завершения SQL (при ошибке преобразования).

Выборка значений геометрических типов

Значения геометрических типов, предварительно сохраненные в таблице, могут быть выбраны в WKT/WKB –формате с помощью соответствующих функций.

В предикате равенства в поисковых запросах с геометрическими типами данных необходимо задавать одновременно значение геометрического типа и идентификатор его координат (с помощью функции `srid` – см. ниже), например,

```
create or replace table geo_test (p point);
insert into geo_test(p) values (PointFromText('point (1,1)'));
insert into geo_test(p) values (PointFromText('point (1 2)',
1));
insert into geo_test(p) values (PointFromText('point (1 3)',
1));
insert into geo_test(p) values (PointFromText('point (1 2)',
2));
insert into geo_test(p) values (PointFromText('point (1 2)',
3));
select count(*) from geo_test where p='point(1,2)';
select count(*) from geo_test where p='point(1,2)'and
srid(p)=2;

create or replace table geom (g geometry);
select x(g) from geom where GeometryType(g)='POINT';
```

Тип графических данных конкретного столбца хранится в элементе `PREC` поля `$$$S24` системной таблицы `$$$ATTRI`.

Кроме того, информацию о столбцах с геометрическими типами данных можно получить из представления `GEOMETRY_COLUMNS`, создать которое можно с помощью скрипта `geo_cat.sql`, размещаемого в подкаталоге `/dict` установочного каталога СУБД ЛИНТЕР:

```
create View GEOMETRY_COLUMNS AS
select
  cast (" as varchar(256)) as f_table_catalog,
  cast ($$$s34 as varchar(256)) as f_table_schema,
  cast ($$$s13 as varchar(256)) as f_table_name,
  cast ($$$s23 as varchar(256)) as f_geometry_column,
  cast ( 2 as integer ) as coord_dimension,
  cast (-1 as integer) as srid
fr om
  LINTER_SYSTEM_USER.$$$sysrl,
  LINTER_SYSTEM_USER.$$$attri,
  LINTER_SYSTEM_USER.$$$usr
where $$$s11 = $$$s21 and
  $$$s12 = $$$s31 and
  $$$s32 = 0 and
  GetByte($$$S24,1) = 9 and
  GetByte($$$S24,2) > 0;
```

Выборка в WKT-формате

Функция

Преобразование внутреннего представления геометрического объекта в WKT-представление.

Спецификация

AsText | **to_char** (<имя столбца> [, <формат>])

<имя столбца> – имя столбца с геометрическим типом данных;

<формат> – символьный литерал формата преобразования строки.

Синтаксические правила

Аргумент <формат> задает формат преобразования строки (см. описание функции `to_char` в документе «СУБД ЛИНТЕР. Справочник по SQL»).

Возвращаемое значение

- 1) значение типа `char`, соответствующее WKB-представлению геометрического объекта (в случае нормального преобразования). Длина строки вычисляется динамически и не может превышать 4000 символов;
- 2) код завершения SQL (при ошибке преобразования).

Пример

```
CREATE OR REPLACE TABLE POINT_TEST( P POINT );
INSERT INTO POINT_TEST VALUES( PointFromText('POINT (1 1)') );
INSERT INTO POINT_TEST VALUES( PointFromText('POINT (0 1)') );
INSERT INTO POINT_TEST VALUES( PointFromText('POINT (1 2)') );
INSERT INTO POINT_TEST VALUES( PointFromText('POINT (1 1)') );
```

```
SQL>select AsText(P) from POINT_TEST;
INL : начальное время : 21:14:42 конечное время : 21:14:42
```

```
| POINT (1 1) |
| POINT (0 1) |
| POINT (1 2) |
| POINT (1 1) |
INL : выдано строк : 4
```

Выборка в WKB-формате

Функция

Преобразование внутреннего представления геометрического объекта в WKB-представление.

Спецификация

AsBinary (<имя столбца>])

<имя столбца> – имя столбца с геометрическим типом данных.

Возвращаемое значение

- 1) значение типа `varbyte`, соответствующее WKB-представлению геометрического объекта (в случае нормального преобразования). Длина значения зависит от геометрического типа объекта;

2) код завершения SQL (при ошибке преобразования).

Пример


```
SQL>select AsBinary(P) from POINT_TEST;
```

```
| 01 01 00 00 00 00 00 00 00 00 00 00 F0 3F 00 00 00 00 00 00 F0 3F|  
| 01 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 F0 3F|  
| 01 01 00 00 00 00 00 00 00 00 00 00 F0 3F 00 00 00 00 00 00 00 40|  
| 01 01 00 00 00 00 00 00 00 00 00 00 F0 3F 00 00 00 00 00 00 F0 3F|
```

Средства обработки геометрических типов данных

СУБД ЛИНТЕР предоставляет набор функций для выполнения различных операций над геометрическими данными. Эти функции условно могут быть разделены на следующие группы:

- функции для анализа свойств геометрических данных;
- функции для создания новых значений геометрических типов из уже существующих;
- функции, которые описывают отношения между двумя значениями геометрических типов.

 Вычисление значений ряда функций (например Area, Length) поддерживается только для плоской системы координат.

Общие функции

Получить название геометрического типа

Функция

Предоставление имени геометрического типа данных для геометрического объекта любого типа.

Спецификация

GeometryType (<объект>)

<объект> – любой геометрический объект.

Возвращаемое значение

- 1) значение типа char(18), соответствующее названию геометрического типа переданного значения (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Пример

```
SQL>SELECT GeometryType(GeomFromText('POINT(1 1)'));
```

```
| POINT |  
INL : выдано строк : 1
```

Получить размерность геометрического типа

Функция

Предоставление размерности значения любого геометрического типа данных.

Спецификация

Dimension (<объект>)

<объект> – любой геометрический объект.

Возвращаемое значение

- 1) целочисленное значение (integer), соответствующее размерности переданного значения геометрического типа (в случае нормального завершения):

- -1 – объект пуст (empty);
 - 0 – нулевая размерность (точка);
 - 1 – линия;
 - 2 – поверхность.
- 2) код завершения SQL (при ошибке преобразования).

Пример

```
SQL>SELECT Dimension(GeomFromText('LineString(1 1,2 2)'));
|          1|
INL : выдано строк      : 1
```

Получить идентификатор системы координат

Функция

Предоставление идентификатора системы координат значения любого геометрического типа данных.

Спецификация

SRID (<объект>)

<объект> – любой геометрический объект.

Возвращаемое значение

- 1) целочисленное значение (integer), соответствующее идентификатору системы координат переданного значения геометрического типа (в случае нормального завершения);
- 2) код завершения SQL (при ошибке преобразования).

Пример

```
SQL>SELECT SRID(GeomFromText('LineString(1 1,2 2)',101));
|          101|
INL : выдано строк      : 1
```

Получить минимальный ограничивающий прямоугольник

Функция

Предоставление минимального ограничивающего прямоугольника любого геометрического объекта.

Спецификация

Envelope (<объект>)

<объект> – любой геометрический объект.

Возвращаемое значение

- 1) значение типа POLYGON, представляющее минимальный ограничивающий прямоугольник с координатами (MINX MINY, MAXX MINY, MAXX MAXY, MINX MAXY, MINX MINY) (в случае нормального завершения);

- 2) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT AsText (Envelope (GeomFromText ('LineString(1 1,2 2)',101)));  
|POLYGON ((1 1,2 1,2 2,1 2,1 1)) |  
INL : выдано строк      : 1
```

Получить границу значения геометрического типа

Функция

Предоставление границы (возможно, комбинированной) значения геометрического типа данных.


Спецификация

Boundary (<объект>)

<объект> – любой геометрический объект.

Возвращаемое значение

- 1) значение геометрического типа, представляющее границу заданного геометрического типа данных (в случае нормального завершения);

 Для геометрического типа `GeometryCollection` возвращается полная граница (без применения метода «mod 2»).

- 2) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT AsText (Boundary (GeomFromText ('LineString(1 1,2 2,3 3)')));  
|MULTIPOINT (1 1,3 3) |  
INL : выдано строк      : 1
```

Проверить существование значения геометрического типа

Функция

Проверка существования значения геометрического типа данных.

Спецификация

IsEmpty (<объект>)

<объект> – любой геометрический объект.

Возвращаемое значение

- 1) значение типа `integer`:
 - 1 – значение геометрического типа пусто;
 - 0 – значение геометрического типа определено;
 - -1 – в случае, если аргумент функции равен `NULL`.
- 2) код завершения SQL (при неправильном аргументе функции).

Примеры

```
SQL>SELECT IsEmpty(GeomFromText('Point(1 1)'));
```

```
|          0|
INL : выдано строк      : 1
```

```
SQL>SELECT IsEmpty(GeomFromText('Point EMPTY'));
```

```
|          1|
INL : выдано строк      : 1
```

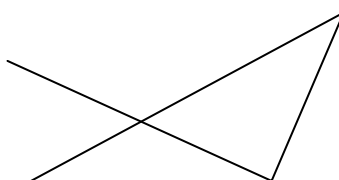
Проверить сложность геометрического объекта

Функция

Проверка сложности геометрического объекта. Описание каждого инстанцируемого геометрического класса включает условия, при которых элемент, принадлежащий этому классу, классифицируется как простой или не простой (сложный). Как правило, простота или сложность объекта зависит от наличия точек пересечения или линий соприкосновения элементов объекта, например:



простой объект



сложный объект

Спецификация

IsSimple (<объект>)

<объект> – любой геометрический объект.

Возвращаемое значение

- 1) значение типа integer:
 - 1 – геометрический объект является простым;
 - 0 – геометрический объект является сложным;
 - -1 – в случае, если аргумент функции равен NULL.
- 2) код завершения SQL (при неправильном аргументе функции).

Примеры

```
SQL>SELECT IsSimple(GeomFromText('LineString(0 0,0 1,1 0,1 1,0 0)'));
```

```
|          0|
INL : выдано строк      : 1
```

```
SQL>SELECT IsSimple(GeomFromText('LineString(0 0,0 1,1 1,1 0,0 0)'));
```

```
|          1|
INL : выдано строк      : 1
```

Частные функции

Функции для работы с точкой

Получить X-координату точки

Функция

Получение X-координаты точки.

Спецификация

X (<точка>)

<точка> – геометрический объект типа POINT.

Возвращаемое значение

- 1) значение типа double, соответствующее X-координате точки;
- 2) NULL, если точка пуста;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT X(GeomFromText('Point(56.7 53.34)',101));
```

```
|          56.7|
INL : выдано строк      : 1
```

Получить Y-координату точки

Функция

Получение Y-координаты точки.

Спецификация

Y (<точка>)

<точка> – геометрический объект типа POINT.

Возвращаемое значение

- 1) значение типа double, соответствующее Y-координате точки;
- 2) NULL, если точка пуста;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT Y(GeomFromText('Point(56.7 53.34)',101));
```

```
|          53.34|
INL : выдано строк      : 1
```

Функции для работы с ломаной линией

Получить координаты начала ломаной линии

Функция

Получение координат начальной точки ломаной линии.

Спецификация

StartPoint (<ломаная линия>)

<ломаная линия> – геометрический объект типа LINESTRING.

Возвращаемое значение

- 1) значение типа POINT, соответствующее начальной координате ломаной линии;
- 2) NULL, если ломаная линия пуста;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT AsText (StartPoint (GeomFromText ('LineString(1 1,2 2,3 3)')));
```

```
| POINT (1 1) |
INL : выдано строк      : 1
```

Получить координаты конца ломаной линии

Функция

Получение координат конечной точки ломаной линии.

Спецификация

EndPoint (<ломаная линия>)

<ломаная линия> – геометрический объект типа LINESTRING.

Возвращаемое значение

- 1) значение типа POINT, соответствующее конечной координате ломаной линии;
- 2) NULL, если ломаная линия пуста;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT AsText (EndPoint (GeomFromText ('LineString(1 1,2 2,3 3)')));
```

```
| POINT (3 3) |
INL : выдано строк      : 1
```

Получить координаты узла ломаной линии

Функция

Получение координат заданного узла ломаной линии.

Спецификация

PointN (<ломаная линия>, [<узел>])

<ломаная линия> – геометрический объект типа LINESTRING.

<узел> – целочисленное положительное значение.

Синтаксические правила

- 1) аргумент <узел> задает порядковый номер точки (узла) ломаной линии. Отсчет узлов начинается с 1;
- 2) если <узел> не задан, по умолчанию принимается значение 1.

Возвращаемое значение

- 1) значение типа POINT, соответствующее координатам заданного узла (точки) ломаной линии;
- 2) NULL, если ломаная линия пуста;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT AsText(PointN(GeomFromText('LineString(1 1,2 2,3 3)'),2));
```

```
| POINT (2 2) |
INL : выдано строк : 1
```

Получить длину ломаной линии

Функция

Получение длины ломаной линии.


Спецификация

Length | **GLength** (<ломаная линия>)

<ломаная линия> – геометрический объект типа LINESTRING.

Возвращаемое значение

- 1) значение типа double, соответствующее длине ломаной линии;
- 2) NULL, если ломаная линия пуста;
- 3) для функции Length – значение типа double, соответствующее длине в байтах внутреннего представления графического объекта (если в качестве аргумента передан геометрический объект, отличный от ломаной линии).

 Длину геометрического типа функция Length будет возвращать только в том случае, если транслятор сможет определить геометрический тип значения. В противном случае будет возвращена длина значения в байтах.

Примеры

```
SQL>SELECT GLength(GeomFromText('LineString(1 1,2 2,3 3)'));
```

```
| 2.82842712474619 |
INL : выдано строк : 1
```

```
SQL>SELECT Length(GeomFromText('LineString(1 1,2 2,3 3)'));
```

```
| 61 |
INL : выдано строк : 1
```

Получить количество узлов ломаной линии

Функция

Получение количества узлов ломаной линии.

Спецификация

NumPoints (<ломаная линия>)

<ломаная линия> – геометрический объект типа LINESTRING.

Возвращаемое значение

- 1) значение типа Integer, соответствующее количеству узлов ломаной линии;
- 2) NULL, если ломаная линия пуста;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT NumPoints(GeomFromText('LineString(1 1,2 2,3 3)'));
|          3|
INL : выдано строк      : 1
```

Проверка выпуклости ломаной линии

Функция

Проверка выпуклости ломаной линии. Ломаная линия считается выпуклой (кольцом), если координаты начальной и конечной точки совпадают, и линия является простой (не проходит через некоторую точку дважды).

Спецификация

IsRing (<ломаная линия>)

<ломаная линия> – геометрический объект типа LINESTRING.

Возвращаемое значение

- 1) значение типа Integer, соответствующее типу ломаной линии:
 - 1 – ломаная линия выпуклая;
 - 0 – ломаная линия не выпуклая;
 - -1 – в случае, если аргумент функции равен NULL.
- 2) NULL, если ломаная линия пуста;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
ды).
SQL>SELECT IsRing(GeomFromText('LineString(1 1,2 2,3 2,1 1)'));
|          1|
INL : выдано строк      : 1
```

Проверка замкнутости ломаной линии

Функция

Проверка замкнутости ломаной линии. Ломаная линия считается замкнутой, если координаты начальной и конечной точки совпадают, и линия является простой (не проходит через одну и ту же точку дважды, за исключением начальной и конечной точки).

Спецификация

IsClosed (<ломаная линия>)

<ломаная линия> – геометрический объект типа LINESTRING.

Возвращаемое значение

- 1) значение типа Integer, соответствующее типу ломаной линии:
 - 1 – ломаная линия замкнута;
 - 0 – ломаная линия разомкнута;
 - -1 – в случае, если аргумент функции равен NULL.
- 2) NULL, если ломаная линия пуста;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT IsClosed(GeomFromText('LineString(1 1,2 2,3 3)'));
```

```
|          0|
INL : выдано строк      : 1
```

Функции для работы с группой ломаных линий

Получить длину группы ломаных линий

Функция

Получение длины группы ломаных линий.

Спецификация

Length | **GLength** (<группа ломаных линий>)

<группа ломаных линий> – геометрический объект типа MULTILINESTRING.

Возвращаемое значение

- 1) значение типа double, соответствующее длине ломаной линии;
- 2) NULL, если ломаная линия пуста;
- 3) для функции Length – значение типа double, соответствующее длине в байтах внутреннего представления графического объекта (если в качестве аргумента передан геометрический объект, отличный от группы ломаных линий).

Примеры

```
SQL>SELECT Length(MLineFromText('MultiLineString((1 1,2 2,3 3),(4 4,5 5)')));
```

```
|          4.24264068711929|
INL : выдано строк      : 1
```

```
SQL>SELECT Length(GeomFromText('MultiLineString((1 1,2 2,3 3),(4 4,5 5))'));
|          111|
INL : выдано строк      : 1
```

Проверка замкнутости группы ломаных линий

Функция

Проверка замкнутости группы ломаных линий. Группа считается замкнутой, если замкнутыми являются одновременно все ломаные линии группы.

Спецификация

IsClosed (<группа ломаных линий>)

<группа ломаных линий> – геометрический объект типа MULTILINESTRING.

Возвращаемое значение

- 1) значение типа double, соответствующее длине заданной группы ломаных линий.
 - 1 – группа ломаных линий замкнута;
 - 0 – группа ломаных линий разомкнута;
 - -1 – в случае, если аргумент функции равен NULL.
- 2) NULL, если ломаная линия пуста;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT IsClosed(GeomFromText('MultiLineString((1 1,2 2,3 3),(4 4,5 5))'));
|          0|
INL : выдано строк      : 1
```

Функции для работы с многоугольником

Определение площади многоугольника

Функция

Получение площади многоугольника, вычисленной в системе координат этого многоугольника.

Спецификация

Area (<многоугольник>)

<многоугольник> – геометрический объект типа POLYGON.

Возвращаемое значение

- 1) значение типа double, соответствующее площади многоугольника.
- 2) NULL, если многоугольник пуст;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT Area(GeomFromText('Polygon((0 0,0 3,3 3,3 0,0 0),(1 1,1 2,2 2,2 1,1 1))'));
|          1|
```

```
|          8|
INL : выдано строк      : 1
```

Определение количества пересекающихся областей многоугольника

Функция

Получение количества пересекающихся областей многоугольника.

Спецификация

NumInteriorRing | **NumInteriorRings** (<многоугольник>)
<многоугольник> – геометрический объект типа POLYGON.

Возвращаемое значение

- 1) значение типа integer, соответствующее количеству пересекающихся областей многоугольника.
- 2) NULL, если многоугольник пуст;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT NumInteriorRings(GeomFromText('Polygon((0 0,0 3,3 3,3 0,0 0),(1 1,1 2,2 2,2 1,1 1))'));
```

```
|          1|
INL : выдано строк      : 1
```

Определение внешней границы многоугольника

Функция

Получение внешней границы многоугольника.

Спецификация

ExteriorRing (<многоугольник>)
<многоугольник> – геометрический объект типа POLYGON.

Возвращаемое значение

- 1) значение типа LINESTRING, соответствующее внешней границе многоугольника;
- 2) NULL, если многоугольник пуст;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT AsText(ExteriorRing(GeomFromText('Polygon((0 0,0 3,3 3,3 0,0 0),(1 1,1 2,2 2,2 1,1 1))')));
```

```
|LINESTRING (0 0,0 3,3 3,3 0,0 0) |
INL : выдано строк      : 1
```

Получение заданной пересекающейся области многоугольника

Функция

Получение внутренней границы многоугольника.

Спецификация

InteriorRing (<многоугольник> [, <номер>])

<многоугольник> – геометрический объект типа POLYGON.

<номер> – порядковый номер требуемой области.

Синтаксические правила

- 1) аргумент <номер> задает порядковый номер самопересекающейся внутренней области многоугольника. Отсчет областей узлов начинается с 1;
- 2) если <номер> не задан, по умолчанию принимается значение 1.

Возвращаемое значение

- 1) значение типа LINESTRING, соответствующее границе заданной области;
- 2) NULL, если многоугольник пуст;
- 3) код завершения SQL (при неправильном аргументе функции, в частности, 1036 – требуемая внутренняя граница не существует).

Пример

```
SQL>SELECT AsText(InteriorRingN(GeomFromText('Polygon((0 0,0 3,3 3,3 0,0 0),(1 1,1 2,2 2,2 1,1 1)')),1));
```

```
|LINESTRING (1 1,1 2,2 2,2 1,1 1) |
INL : выдано строк : 1
```

Определение геометрического центра многоугольника

Функция

Получение геометрического центра многоугольника (может находиться за пределами многоугольника). Геометрический центр вычисляется на основе «центра масс» многоугольника при условии, что вся его «масса» равномерно распределена между вершинами внешней границы.

Спецификация

Centroid (<многоугольник>])

<многоугольник> – геометрический объект типа POLYGON.

Возвращаемое значение

- 1) значение типа POINT, соответствующее координатам геометрического центра многоугольника;
- 2) NULL, если многоугольник пуст;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT AsText(Centroid(GeomFromText('Polygon((0 0,0 3,3 3,3 0,0 0),(1 1,1 2,2 2,2 1,1 1)'))));
```

```
|POINT (1.5 1.5) |
INL : выдано строк : 1
```

Получение первой точки внешней границы многоугольника

Функция

Получение первой точки внешней границы многоугольника.

Спецификация

PointOnSurface (<многоугольник>])

<многоугольник> – геометрический объект типа POLYGON.

Возвращаемое значение

- 1) значение типа POINT, соответствующее координатам первой внешней границы многоугольника;
- 2) NULL, если многоугольник пуст;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT AsText(PointOnSurface(GeomFromText('Polygon((0 0,0 3,3 3,3 0,0 0),(1 1,1 2,2 2,2 1,1 1)'))));
```

```
| POINT (0 0) |
INL : выдано строк      : 1
```

Функции для работы с группой многоугольников

Определение площади группы многоугольников

Функция

Получение площади группы многоугольников, вычисленной в системе координат этой группы.

Спецификация

Area (<группа многоугольников>)

<группа многоугольников> – геометрический объект типа MULTIPOLYGON.

Возвращаемое значение

- 1) значение типа double, соответствующее площади группы многоугольников;
- 2) NULL, если многоугольник пуст;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT Area(GeomFromText('MultiPolygon(((0 0,0 3,3 3,3 0,0 0),(1 1,1 2,2 2,2 1,1 1)'))));
```

```
|          8 |
INL : выдано строк      : 1
```

Определение геометрического центра группы многоугольников

Функция

Получение геометрического центра группы многоугольников (может находиться за пределами группы). Геометрический центр вычисляется на основе «центра масс» входящих в группу многоугольников.

Спецификация

Centroid (<группа многоугольников>])

<группа многоугольников> – геометрический объект типа MULTIPOLYGON.

Возвращаемое значение

- 1) значение типа POINT, соответствующее координатам геометрического центра группы многоугольников;
- 2) NULL, если многоугольник пуст;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT AsText(Centroid(GeomFromText('MultiPolygon(((0 0,0 3,3 3,3 0,0 0)),((11 11,11 12,12 12,12 11,11 11)))')));
```

```
| POINT (6.5 6.5) |
INL : выдано строк : 1
```

Получение первой точки внешней границы группы многоугольников

Функция

Получение первой точки внешней границы группы многоугольников.

Спецификация

PointOnSurface (<группа многоугольников>])

<группа многоугольников> – геометрический объект типа MULTIPOLYGON.

Возвращаемое значение

- 1) значение типа POINT, соответствующее координатам первой внешней границы группы многоугольников;
- 2) NULL, если многоугольник пуст;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT AsText(PointOnSurface(GeomFromText('MultiPolygon(((10 10,10 13,13 13,13 10,10 10)))')));
```

```
| POINT (10 10) |
INL : выдано строк : 1
```

Функции для работы с группой геометрических объектов

Определение количества объектов в группе

Функция

Получение количества объектов, входящих в состав группы геометрических объектов.

Спецификация

NumGeometries (<группа объектов>)

<группа объектов> – геометрический объект типа GEOMETRYCOLLECTION, MULTIPOINT, MULTILINESTRING или MULTIPOLYGON.

Возвращаемое значение

- 1) количество объектов в группе (значение типа integer);
- 2) NULL, если многоугольник пуст;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT NumGeometries(GeomFromText('GeometryCollection(Point(1 1),LineString(2 2, 3 3))'));
```

```
|          2|
INL : выдано строк      : 1
```

Получить заданный объект группы

Функция

Получение заданного объекта из группы геометрических объектов.

Спецификация

GeometryN (<группа объектов> [,<номер>])

<группа объектов> – геометрический объект типа GEOMETRYCOLLECTION, MULTIPOINT, MULTILINESTRING или MULTIPOLYGON.

<номер> – порядковый номер требуемого объекта.

Синтаксические правила

- 1) аргумент <номер> задает порядковый номер объекта в группе геометрических объектов. Отсчет начинается с 1;
- 2) если <номер> не задан, по умолчанию принимается значение 1.

Возвращаемое значение

- 1) требуемый объект;
- 2) NULL, если многоугольник пуст;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>SELECT AsText(GeometryN(GeomFromText('GeometryCollection(Point(1 1),LineString(2 2, 3 3))'),1));
```

```
| POINT (1 1) |
```

INL : выдано строк : 1

Геометрические функции

Пересечение геометрических объектов

Функция

Получение пересечения двух геометрических объектов.


Спецификация

Intersection (<объект1> , <объект2>)

<объект1>, <объект2> – геометрические объекты произвольного типа.

Возвращаемое значение

- 1) геометрический объект, представляющий пересечение заданных геометрических объектов;
- 2) для включения в результирующий объект составных типов (MULTIPOINT, MULTILINESTRING, MULTIPOLYGON) они разбиваются на объекты простых типов.

 Пересечение значений типа Circle и значения, в состав которого входит тип Polygon (Polygon, Multipolygon, Geometrycollection с Polygon), осуществляется особым образом, что связано с отсутствием типа «кривая». Круг аппроксимируется вписанным правильным многоугольником с 32 вершинами, и полученный многоугольник пересекается со вторым значением геометрического типа.

- 3) GEOMETRYCOLLECTION (EMPTY), если объекты не пересекаются;
- 4) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>select AsText(Intersection(
GeomFromText('POLYGON ((0 0,0 5,5 5,5 0,0 0)),(2 2,2 4,3 4,3 2,2
2))'),
GeomFromText('POLYGON ((1 1,1 6,6 6,6 1,1 1)),(2 3,2 4,4 4,4 3,2
3))')));

|GEOMETRYCOLLECTION (POLYGON ((5 1,1 1,1 5,5 5,5 1)),(2 3,2 4,3
4,4 4,4 3,3 3,3 2
,2 2,2 3))) |
INL : выдано строк : 1
```

Объединение геометрических объектов

Функция

Получение объединения двух геометрических объектов.


Спецификация

Union (<объект1> , <объект2>)

<объект1>, <объект2> – геометрические объекты произвольного типа.

Возвращаемое значение

- 1) геометрический объект, представляющий объединение заданных геометрических объектов;
- 2) код завершения SQL (при неправильном аргументе функции).

 Так же, как и для функции INTERSECTION, при объединении с некоторыми геометрическими типами круг может быть аппроксимирован многоугольником с 32 вершинами.

Пример

```
SQL>select AsText(Union(GeomFromText('POLYGON ((0 0,0 3,3 3,3
0,0 0))'),
                        GeomFromText('POLYGON ((1 1,1 4,4 4,4
1,1 1))')));
|POLYGON ((3 1,3 0,0 0,0 3,1 3,1 4,4 4,4 1,3 1))
|
INL : выдано строк      : 1
```

Разность геометрических объектов

Функция

Получение геометрической разности двух объектов.

Спецификация

Difference (<объект1> , <объект2>)

<объект1>, <объект2> – геометрические объекты произвольного типа.

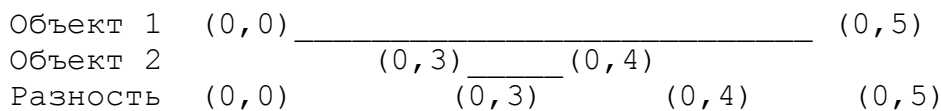
Возвращаемое значение

- 1) геометрический объект, представляющий разность двух заданных геометрических объектов;
- 2) NULL, если один из аргументов равен NULL;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

Получить разность двух отрезков линии.

1)



```
CREATE OR REPLACE TABLE TEST
(Line1 LINESTRING, Line2 LINESTRING);
INSERT INTO TEST
VALUES( LineFromText('LINESTRING (0 0,0 5)'),
LineFromText('LINESTRING (0 3,0 4)'));
select astext(DIFFERENCE(LINE1, LINE2)) FROM TEST;

|MULTILINESTRING ((0 0,0 3),(0 4,0 5))
```

2)

Объект 1	(0,0)	_____	(0,5)
Объект 2	(0,0)	_____	(0,3)
Разность		(0,3) _____	(0,5)

```
CREATE OR REPLACE TABLE TEST( Line1 LINESTRING, Line2
LINESTRING );
INSERT INTO TEST
VALUES( LineFromText('LINESTRING (0 0,0 5)'),
LineFromText('LINESTRING (0 0,0 3)'));
select astext(DIFFERENCE(LINE1, LINE2)) FROM TEST;
|LINESTRING (0 3,0 5)
```

Симметричная разность геометрических объектов

Функция

Получение симметричной геометрической разности двух объектов. Симметричная разность вычисляется путем исключения общей (пересекающейся) части геометрических объектов.

Спецификация

SymDifference (<объект1> , <объект2>)

<объект1>, <объект2> – геометрические объекты произвольного типа.

Возвращаемое значение

- 1) геометрический объект, представляющий симметричную разность двух заданных геометрических объектов;
- 2) NULL, если один из аргументов равен NULL;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

1)

Получить симметричную разность двух отрезков линии.

Объект 1	(0,0)	_____	(0,5)
Объект 2		(0,3) _____	(0,7)
Симметричная	(0,0)	_____	(0,3
разность	(0,5)	_____	(0,7)

```
CREATE OR REPLACE TABLE TEST( Line1 LINESTRING, Line2
LINESTRING );
INSERT INTO TEST
VALUES( LineFromText('LINESTRING (0 0,0 5)'),
LineFromText('LINESTRING (0 3,0 7)'));
select astext(SYMDIFFERENCE(LINE1, LINE2)) FROM TEST;
|MULTILINESTRING((0 0,0 3),(0 5,0 7))|
```

```

2)
CREATE OR REPLACE TABLE TEST( Line1 LINESTRING, Line2
LINESTRING );
INSERT INTO TEST
VALUES( LineFromText('LINESTRING (0 0, 0 5)'),
LineFromText('LINESTRING (0 6,0 7)'));

INSERT INTO TEST
VALUES( LineFromText('LINESTRING EMPTY)'),
LineFromText('LINESTRING (0 6,0 7)'));

select
decode(astext(SYMDIFFERENCE(LINE1, LINE2)), null,'null',
astext(SYMDIFFERENCE(LINE1, LINE2))) FROM TEST;
|MULTILINESTRING((0 0,0 5),(0 6,0 7))|
|null                               |

```

Расстояние между объектами

Функция

Получение расстояния между заданными объектами.

Спецификация

Distance (<объект1> , <объект2>)

<объект1>, <объект2> – геометрические объекты произвольного типа.

Возвращаемое значение

- 1) кратчайшее расстояние между заданными объектами (значение типа double);
- 2) NULL, если один из аргументов равен NULL;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

```

SQL>SELECT Distance(GeomFromText('POLYGON ((0 0,0 3,3 3,3 0,0
0),(1 1,1 2,2 2,2 1,1 1))'),GeomFromText('MULTIPOINT (1.3 1.3,
1.5 1.5, 1.8 1.8)'));

|                               0.2|

```

Буферный геометрический объект

Функция

Получение буферного геометрического объекта (т. е. объекта, все точки которого равноудалены от заданного объекта).

Спецификация

Buffer (<объект> , <расстояние>)

<объект> – геометрические объекты произвольного типа;

<расстояние> – положительное вещественное значение.

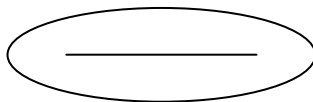
Геометрические функции

Возвращаемое значение

- 1) геометрический объект, все точки которого расположены от <объекта> на заданное <расстояние>;
- 2) NULL, если один из аргументов равен NULL;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

Получить буферный геометрический объект от отрезка линии.



```
create or replace table test( line1 linestring);
insert into test(line1) values( linefromtext('linestring (1 0,5
0)'));
select astext(buffer(line1, 1)) from test;
```

```
| POLYGON ((5 1,
1 1,
0.80491 0.980785,
0.617317 0.92388,
0.44443 0.83147,
0.292893 0.707107,
0.16853 0.55557,
0.07612 0.382683,
0.019215 0.19509,
0 0,
0.019215 -0.19509,
0.07612 -0.382683,
0.16853 -0.55557,
0.292893 -0.707107,
0.44443 -0.83147,
0.617317 -0.92388,
0.80491 -0.980785,
1 -1,
5 -1,
5.19509 -0.980785,
5.382683 -0.92388,
5.55557 -0.83147,
5.707107 -0.707107,
5.83147 -0.55557,
5.92388 -0.382683,
5.980785 -0.19509,
6 0,
5.980785 0.19509,
5.92388 0.382683,
5.83147 0.55557,
5.707107 0.707107,
5.55557 0.83147,
5.382683 0.92388,
5.19509 0.980785,
```

51))

Выпуклая оболочка объекта

Функция

Получение выпуклой оболочки заданного объекта.

Спецификация

ConvexHull (<объект>)

<объект> – геометрические объекты произвольного типа;

Возвращаемое значение

- 1) геометрический объект, являющийся выпуклой оболочкой <объект>;
- 2) NULL, если один из аргументов равен NULL;
- 3) код завершения SQL (при неправильном аргументе функции).

Проверка совпадения объектов

Функция

Проверка совпадения двух заданных объектов.

Две ломаных линии LineString считаются совпадающими, даже если обход точек у них разный (например, LINESTRING(0 0, 1 1) и LINESTRING(1 1, 0 0) совпадают).

Два объекта типа Polygon считаются совпадающими, если у них одинаковое количество границ и совпадают сами границы (обход вершин в границах и порядок следования внутренних границ может быть произвольным).

Для сложных геометрических типов (MultiPoint, MultiPolygon, MultiLineString, GeometryCollection) при сравнении также учитывается, что порядок следования составляющих их частей может отличаться.

Спецификация

Equals (<объект1> , <объект2>)

<объект1>, <объект2> – геометрические объекты произвольного типа.

Возвращаемое значение

- 1) 1, если объекты совпадают;
- 2) 0, если объекты не совпадают;
- 3) NULL, если один из аргументов равен NULL
- 4) код завершения SQL (при неправильном аргументе функции).

Пример

```
SQL>select Equals(
```

```
select Equals(GeomFromText('POLYGON ((-1 -1,-1 4,4 4,5 2,4 -1,-1 -1),
(0 0,0 1,1 1,1 0,0 0), (2 2,2 3,3 3,3 2,2 2))'),
GeomFromText('POLYGON ((4 4,-1 4,-1 -1,4 -1,5 2,4
4),
(2 2,3 2,3 3,2 3,2 2), (0 1,1 1,1 0,0 0,0 1))'));
|          1|
```

Проверка пересечения объектов

Функция

Проверка пересечения двух заданных объектов (см. также функции overlaps и crosses).

Спецификация

Intersects (<объект1> , <объект2>)

<объект1>, <объект2> – геометрические объекты произвольного типа.

Возвращаемое значение

- 1) 1, если объекты пересекаются (имеют хотя бы одну общую точку);
- 2) 0, если объекты не пересекаются;
- 3) NULL, если один из аргументов равен NULL;
- 4) код завершения SQL (при неправильном аргументе функции).

Примеры

```
1)
Объект 1      (0,0) _____ (0,5)
Объект 2      (0,3) _____ (0,7)
Пересекаются Да
```

```
CREATE OR REPLACE TABLE TEST( Line1 LINESTRING, Line2
LINESTRING );
INSERT INTO TEST
VALUES( LineFromText('LINESTRING (0 0, 0 5)'),
LineFromText('LINESTRING (0 3,0 7)'));
select astext(intersects(LINE1, LINE2)) FROM TEST;
|1|
```

```
2)
SQL>select Intersects(
GeomFromText('MULTIPOLYGON (((0 1,1 2,2 1,1 0,0 1)),
((2 1,3 2,4 1,3 0,2 1)))'),
GeomFromText('POLYGON (-1 1,-1 3,1 3,1 1,-1 1)'));
|          1|
```

Проверка перекрытия объектов

Функция

Проверка перекрытия двух объектов.

Спецификация

Overlaps (<объект1> , <объект2>)

<объект1>, <объект2> – геометрические объекты произвольного типа.

Возвращаемое значение

- 1) 1, если <объект1> и <объект2>:
 - перекрывают друг друга;
 - область перекрытия имеет ту же размерность, что и каждый из указанных объектов;
 - область перекрытия не совпадает с <объектом1> или <объектом2>;
- 2) 0, если <объект1> и <объект2> не перекрываются или имеют разную размерность;
- 3) NULL, если один из аргументов равен NULL;
- 4) код завершения SQL (при неправильном аргументе функции).

Примеры

1)

Объект 1 (0,0) _____ (0,5)

Объект 2

(0,3) _____ (0,7)

Перекрываются? Да

```
CREATE OR REPLACE TABLE TEST( Line1 LINESTRING, Line2
LINESTRING );
INSERT INTO TEST
VALUES( LineFromText('LINESTRING (0 0, 0 5)'),
LineFromText('LINESTRING (0 3,0 7)'));
select astext(overlaps(LINE1, LINE2)) FROM TEST;
|          1|
```

2)

Объект 1 (0,0) _____ (0,3)

Объект 2

(0,3) _____ (0,7)

Перекрываются? Нет

Пересекаются? Да

```
CREATE OR REPLACE TABLE TEST( Line1 LINESTRING, Line2
LINESTRING );
INSERT INTO TEST
VALUES( LineFromText('LINESTRING (0 0, 0 3)'),
LineFromText('LINESTRING (0 3,0 7)'));
select astext(overlaps(LINE1, LINE2)),
astext(intersects(LINE1, LINE2)) FROM TEST;
|          0|          1|
```

Проверка разъединения объектов

Функция

Проверка разъединения двух заданных объектов.

Спецификация

Disjoint (<объект1> , <объект2>)

<объект1>, <объект2> – геометрические объекты произвольного типа.

Возвращаемое значение

- 1) 1, если объекты разъединены (не имеют ни одной общей точки);
- 2) 0, если объекты пересекаются (имеют хотя бы одну общую точку);
- 3) NULL, если один из аргументов равен NULL;
- 4) код завершения SQL (при неправильном аргументе функции).

Примеры

```
1)
Объект 1      (0,0) _____ (0,5)
Объект 2      (0,6) _____ (0,7)
Разъединены? Да
```

```
CREATE OR REPLACE TABLE TEST( Line1 LINESTRING, Line2
LINESTRING );
INSERT INTO TEST
VALUES( LineFromText('LINESTRING (0 0, 0 5)'),
LineFromText('LINESTRING (0 6,0 7)'));
select astext(disjoint(LINE1, LINE2)) FROM TEST;
|          1|
```

```
2)
GeomFromText('MULTIPOLYGON (((0 1,1 2,2 1,1 0,0 1)),
((2 1,3 2,4 1,3 0,2 1)))'),
GeomFromText('POLYGON (-1 1,-1 3,1 3,1 1,-1 1)');
|          0|
```

Проверка вложенности объектов (вариант 1)

Функция

Проверка вложенности первого объекта во второй.

Спецификация

Within (<объект1> , <объект2>)

<объект1>, <объект2> – геометрические объекты произвольного типа.

Возвращаемое значение

- 1) 1, если <объект2> целиком содержит <объект1>, причем пересечение их внутренних областей не пусто;

- 2) 0, если <объект2> не является вложенным в <объект1>;
- 3) NULL, если один из аргументов равен NULL;
- 4) код завершения SQL (при неправильном аргументе функции).

Пример

```

Объект 1  (0,0) _____ (0,3)
Объект 2  (0,0) _____ (0,5)
Вложен?   Да
    
```

```

CREATE OR REPLACE TABLE TEST( Line1 LINESTRING, Line2
LINESTRING );
INSERT INTO TEST
VALUES( LineFromText('LINESTRING (0 0, 0 3)'),
LineFromText('LINESTRING (0 0,0 5)'));
select astext(within(LINE1, LINE2)) FROM TEST;
|          1|
    
```

Проверка вложенности объектов (вариант 2)

Функция

Проверка вложенности второго объекта в первый.

Спецификация

Contains (<объект1> , <объект2>)

<объект1>, <объект2> – геометрические объекты произвольного типа.

Возвращаемое значение

- 1) 1, если <объект1> целиком содержит <объект2>, причем пересечение их внутренних областей не пусто;
- 2) 0, если <объект1> не является вложенным в <объект2>;
- 3) NULL, если один из аргументов равен NULL;
- 4) код завершения SQL (при неправильном аргументе функции).

Пример

```

Объект 1  (0,0) _____ (0,3)
Объект 2  (0,0) _____ (0,5)
Вложен?   Нет
    
```

```

CREATE OR REPLACE TABLE TEST( Line1 LINESTRING, Line2
LINESTRING );
INSERT INTO TEST
VALUES( LineFromText('LINESTRING (0 0, 0 3)'),
LineFromText('LINESTRING (0 0,0 5)'));
select astext(contains(LINE1, LINE2)) FROM TEST;
|          0|
    
```

Проверка касания объектов

Функция

Проверка касания двух объектов.

Спецификация

Touches (<объект1> , <объект2>)

<объект1>, <объект2> – геометрические объекты произвольного типа.

Возвращаемое значение

- 1) 1, если <объект1> и <объект2> касаются друг друга (пересекаются объекты, но не пересекаются их внутренние области);
- 2) 0, если <объект1> и <объект2> не пересекаются либо касаются своими внутренними областями;
- 3) NULL, если один из аргументов равен NULL или EMPTY;
- 4) код завершения SQL (при неправильном аргументе функции).

Пример

Объект 1 (0,0) _____ (0,3)
Объект 2 (0,3) _____ (0,5)
Касаются? Да

```
CREATE OR REPLACE TABLE TEST( Line1 LINESTRING, Line2
LINESTRING );
INSERT INTO TEST
VALUES( LineFromText('LINESTRING (0 0, 0 3)'),
LineFromText('LINESTRING (0 3,0 5)'));
select astext(touches(LINE1, LINE2)) FROM TEST;
|      1|
```

Проверка скрещивания объектов

Функция

Проверка скрещивания двух объектов.

Спецификация

Crosses (<объект1> , <объект2>)

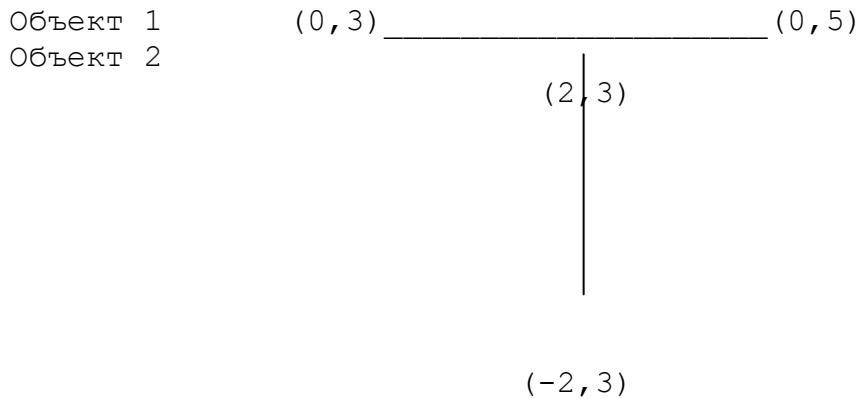
<объект1>, <объект2> – геометрические объекты произвольного типа.

Возвращаемое значение

- 1) 1, если <объект1> и <объект2>:
 - пересекаются их внутренние области;
 - размерность пересечения меньше максимальной размерности указанных объектов;
 - пересечение не совпадает с <объектом1> или <объектом2>;
- 2) 0, если <объект1> и <объект2> не пересекаются;
- 3) NULL, если один из аргументов равен NULL или EMPTY;

4) код завершения SQL (при неправильном аргументе функции).

Пример



Скрещиваются? Да

```
CREATE OR REPLACE TABLE TEST( Line1 LINESTRING, Line2
LINESTRING );
INSERT INTO TEST
VALUES( LineFromText('LINESTRING (0 0, 0 5)'),
LineFromText('LINESTRING (2 3,-2 3)'));
select astext(crosses(LINE1, LINE2)) FROM TEST;
| 1 |
```

Оптимизации работы с геометрическими данными

Для оптимизации поисковых операций с участием геометрических данных необходимо использовать индексы. Индексирование геометрических данных в СУБД ЛИНТЕР выполняется с помощью В-дерева.

Значения, хранящиеся в индексе, представляют собой значения геометрического типа MBR (Minimum Bounding Rectangle, минимальный ограничительный прямоугольник). В БД MBR-значение хранится в виде двух точек ((MINX,MINY),(MAXX,MAXY)). Все координаты имеют тип double.

Нижняя и верхняя граница оператора BETWEEN для геометрических данных задают левый нижний и правый верхний угол прямоугольника, при полном или частичном попадании в который оператор BETWEEN возвращает значение True (Истина.).

Другие операторы сравнения (>,<=,<>,>=,<=) также используют MBR.

Команды создания индекса для геометрических данных имеют тот же синтаксис, что и команды создания обычного индекса:

```
CREATE OR REPLACE TABLE POINT_TEST( P POINT );
CREATE INDEX P ON POINT_TEST;
```

Также поддерживается составной геометрический индекс:

4. Анализируем результаты: без использования индекса время выполнения запроса повышается с 0.00 секунд до 0.31 секунды.

Управление вводом геометрических данных

Проверка корректности вводимых значений некоторых геометрических типов (Polygon, MultiPolygon) требует значительного времени (проверка простоты и замкнутости внешних и внутренних границ, непересекаемость этих границ). В СУБД ЛИНТЕР предусмотрена возможность управления проверкой геометрических данных.

Режим ввода графических данных (с проверкой или без проверки вводимых значений) задается с помощью SQL-операторов:

1) для всей БД:

```
SET DATABASE GEODATA VALIDITY CHECKING ON | OFF;
```

2) для отдельного соединения (команда должна выполняться по этому соединению):

```
SET CONNECTION GEODATA VALIDITY CHECKING ON | OFF;
```

В обоих операторах опция ON устанавливает режим проверки, опция OFF – отменяет.

Проверка корректности геометрических данных

Функция

Проверка корректности значения геометрического типа данных.

Спецификация

GeoIsValid (<объект>)

<объект> – произвольный геометрический объект.

Возвращаемое значение

- 1) значение типа integer:
 - 1 – корректный объект;
 - 0 – некорректный объект;
- 2) NULL, если многоугольник пуст;
- 3) код завершения SQL (при неправильном аргументе функции).

Пример

1. Создаем таблицу и заносим в нее данные:

```
CREATE OR REPLACE TABLE GEOMCOLL_TEST( G GEOMETRYCOLLECTION );
SET CONNECTION GEODATA VALIDITY CHECKING OFF;

/* CORRECT POINT */
INSERT INTO GEOMCOLL_TEST VALUES( 'GEOMETRYCOLLECTION (POINT (1 1))' );

/* RADIX < 0 */
INSERT INTO GEOMCOLL_TEST VALUES( 'GEOMETRYCOLLECTION (CIRCLE (1 1, -1))' );

/* EXTERIOR RING IS NOT CORRECT */
INSERT INTO GEOMCOLL_TEST VALUES( 'GEOMETRYCOLLECTION (POLYGON ((0 0,0 3,3 3, 3 -
1, 3 0, 0 0), (1 1,1 2,2 2,2 1,1 1)))' );

/* POLYGONS INTERSECTS */
INSERT INTO GEOMCOLL_TEST VALUES( 'GEOMETRYCOLLECTION (MULTIPOLYGON(((0 1,1 2,2
1,1 0,0 1)), ((2 2,3 1,2 0,1 1,2 2)))' );
```

2. Проверяем корректность введенных данных:

```
SQL>SELECT GeoIsValid(G) FROM GEOMCOLL_TEST;
INL : начальное время : 17:22:18 конечное время : 17:22:18

|          1|
|          0|
|          0|
|          0|
INL : выдано строк      : 4
```

Указатель функций

Area, 40, 43
AsBinary, 30
AsText, 29
Boundary, 33
Buffer, 50
Centroid, 42, 44
ConvexHull, 51
Dimension, 31
Disjoint, 48, 49, 53, 54, 55, 56, 57
Distance, 50
EndPoint, 36
Envelope, 32
Equals, 52
ExteriorRing, 41
GeoIsValid, 61
GeomCollFromText, 22
GeomCollFromWKB, 27
GeometryFromText, 24
GeometryFromWKB, 27
GeometryN, 45
GeometryType, 31
GeomFromText, 24
GeomFromWKB, 27
GLength, 37, 39
InteriorRing, 42
Intersect, 52
Intersection, 47
IsClosed, 39, 40
IsEmpty, 33
IsRing, 38
IsSimple, 34
Length, 37, 39
LineFromText, 19
LineFromWKB, 25
LineStringFromText, 19
LineStringFromWKB, 25
MLineFromText, 21
MLineFromWKB, 26
MPointFromText, 21
MPointFromWKB, 26
MPolyFromText, 22
MPolyFromWKB, 27
MultiPointFromText, 21
MultiLineFromText, 21
MultiLineFromWKB, 26
MultiPointFromWKB, 26
MultiPolyFromText, 22
MultiPolygonFromWKB, 27
NumGeometries, 45
NumInteriorRings, 41
NumPoints, 38
PointFromText, 19
PointFromWKB, 24
PointN, 36
PointOnSurface, 43, 44
PolyFromText, 20
PolyFromWKB, 25
PolygonFromText, 20
PolygonFromWKB, 25
SRID, 32
StartPoint, 36
to_char, 29
Union, 47
X, 35
Y, 35

