Краткое оглавление

Предисловие

В детстве всем вам, вероятно, говорили: «ты то, что ты ешь», как бы напоминая, что диета является неотъемлемой частью здорового образа жизни. В современном мире мы носим в карманах смартфоны, определяющие местоположение, наши автомобили оснащены GPS-навигаторами, а по адресу компьютера можно узнать, где он расположен, так что справедливым стало и другое утверждение: «мы – то, где мы находимся». Каждый человек теперь – это мобильный датчик, генерирующий непрерывный поток данных о местоположении во время перемещений по планете.

Чтобы управлять этим потоком данных, а также параллельным потоком, который доступен благодаря недорогим спутниковым изображениям и краудсорсинговой картографии, нужен инструмент, отвечающий этой задаче. Инструмент, который мог бы надежно хранить данные, предоставлять к ним эффективный доступ и располагать мощными средствами анализа. Нам нужна пространственная база данных, такая как PostGIS.

До появления пространственных баз данных компьютерный анализ местоположения и картография выполнялись с помощью *географических информационных систем* (ГИС), которые функционировали на настольных рабочих станциях. Когда в 2001 г. появилась PostGIS, название этого проекта было простой игрой слов – естественно, пространственное расширение базы данных PostgreSQL должно было называться PostGIS.

Но по мере развития проекта это название приобретало большее значение. Ежегодно добавлялись новые функции для анализа данных, и с каждым годом пользователи все чаще и чаще использовали их, выполняя работу, для которой в прежние годы потребовалась бы специализированная рабочая ГИС-станция. Нам больше не нужно специальное программное обеспечение для работы с ГИС. Достаточно пространственной базы данных.

Не прошло и года после первого выпуска PostGIS, как в марте 2002 г. в пользовательской рассылке я попросил привести примеры того, как эти пользователи работают с PostGIS.

В своем первом сообщении Регина Обе ответила так:

Мы используем его здесь (В городе Бостон. – П. Р.) для анализа близости. Часть нашего департамента отвечает за распределение заложенных участков среди застройщиков для строительства домов, предприятий и т. д. Мы используем PostGIS для упорядочения объектов собственности по расстоянию... так что если застройщику нужен земельный участок, размер которого, скажем, равен X, он сможет лучше оценить доступные варианты.

Даже на этом раннем этапе проекта Регина Обе уже тестировала возможности PostGIS и проводила продуманный анализ.

После выхода в 2011 г. первого издания книги «PostGIS в действии» PostGIS продолжала активно развиваться. В ней появились новые возможности для растрового анализа, 3D, кластеризация, временны е данные, топологические объекты и многое другое. Но мир тоже не стоял на месте.

Почти два десятилетия назад, когда PostGIS была чем-то принципиально новым, трудно было вообразить, что почти у каждого человека в кармане будет устройство GPS, а теперь это обычное дело. Возможности управления местоположением в PostGIS теперь широко используются разработчиками, которые еще несколько лет назад ничего не слышали о пространственных данных.

За последние годы спутниковая и аэрофотосъемка вышли на массовый рынок, беспилотные системы стали обычным явлением, а датчики местоположения устанавливаются практически на любом движущемся объекте. Количество данных, требующих анализа, а также скорость, с которой эти данные появляются, растут быстрее, чем когда-либо.

В то же время с PostGIS еще никогда не было так просто работать. Вы можете развернуть копию у любого облачного провайдера, скачать сборки для любой платформы, а при желании – загрузить открытый исходный код и выполнить сборку самостоятельно, как это делала Регина много лет назад.

Наслаждайтесь этой книгой и пользуйтесь изложенными в ней идеями, чтобы данные о местоположении работали. Регина и Лео уместили огромный объем информации в кратком руководстве, поистине единственном в своем роде.

Пол Рэмси,
председатель руководящего комитета проекта PostGIS

Вступление

PostGIS – это пространственное расширение для СУБД PostgreSQL. Это самая мощная пространственная база данных с открытым исходным кодом. Она добавляет в PostgreSQL несколько пространственных типов данных и более 400 функций для работы с этими типами. PostGIS поддерживает большое количество пространственных функций, совместимых с OGC/ISO SQL/MM, которые существуют в других реляционных базах данных (в частности, Oracle, SQL Server, MySQL и IBM DB2), а также многочисленные дополнительные пространственные функции, уникальные для PostGIS.

Со времени последнего издания этой книги в другие базы данных были добавлены пространственные функции, реализующие подмножество функционала PostGIS. Одноименные функции можно встретить в Google BigQuery и Snowflake. Многие поставщики облачных услуг теперь также предлагают PostgreSQL/PostGIS, используя подход «база данных как сервис» (DBaaS).

Читатели, знакомые с другими пространственными базами данных, соответствующими стандарту ANSI/ISO, или с другими реляционными базами данных, будут чувствовать себя как дома, работая с PostgreSQL и PostGIS. PostgreSQL – одна из наиболее совместимых с ANSI/ISO SQL-систем управления базами данных.

Эта книга призвана дополнить официальную документацию PostGIS и послужить путеводителем по сотням функций, предлагаемых PostGIS. Мы хотели создать книгу, где будут перечислены многие распространенные проблемы, с которыми нам довелось сталкиваться, и различные стратегии их решения с помощью PostGIS.

Попутно мы надеемся заложить основу для пространственного мышления. Надеемся, что вы сможете адаптировать наши многочисленные примеры и советы к вашим рабочим задачам и, возможно, даже создать что-то свое.

Об этой книге

Эта книга посвящена версиям PostGIS 3 и 3.1 и PostgreSQL 11–13. Она не выступает альтернативой официальной документации PostGIS или PostgreSQL. Официальная документация знакомит со множеством функций, доступных в PostGIS, и содержит примеры использования каждой из них. Но там не показано, как объединить все эти функции для решения специфических задач. Эту роль как раз и берет на себя наша книга. Хотя она не охватывает все функции, доступные в PostGIS, здесь изложены наиболее часто используемые и интересные из них. Она дает навыки, необходимые для объединения функций и позволяющие решать задачи пространственного анализа и моделирования – как типовые, так и нестандартные.

Хотя вы можете использовать эту книгу в качестве справочника, мы рекомендуем применять с этой целью официальный сайт PostGIS *https://postgis.net*.

Наша книга посвящена двух- и трехмерным некриволинейным декартовым векторным объектам, двумерным геодезическим векторным объектам, растровым данным и сетевым топологическим объектам.

Хотя основное внимание уделяется использованию PostGIS, мы бы не достигли своей цели, если бы пренебрегли рассказом о ландшафте, в котором обитает пространственная база данных. PostGIS не сама по себе – она редко работает в одиночку. Чтобы дополнить картину, мы включили сюда:

- обширное приложение, в котором подробно описывается PostgreSQL – от настройки до резервного копирования и управления безопасностью. В приложении также рассматриваются основы SQL и создание с его помощью функций и других объектов;
- несколько глав, которые посвящены использованию PostGIS в веб-картографии, настольным ГИС, процедурным языкам PostgreSQL, обычно используемым с PostGIS, и дополнительным надстройкам с открытым исходным кодом, таким как геокодер TIGER и pgRouting.

Эта книга не затрагивает строгое описание математики, лежащей в основе библиотек PostGIS. Мы полагаемся на то, что читатель имеет общее представление о таких объектах, как точки, линии и полигоны. Точно так же мы не углубляемся в теорию баз дан-

ных. Если тот или иной индекс, по нашему мнению, эффективнее другого, мы утверждаем это исходя из собственного опыта, а не потому, что освоили реляционную алгебру и попутно разобрали несколько компьютерных микросхем.

Кому адресована эта книга?

Книга представляет собой введение в PostGIS и предполагает базовый уровень владения программированием и работы с данными. Ниже перечислены типы специалистов, которые, вероятнее всего, заинтересованы в PostGIS и которым подойдет эта книга.

Практики, работающие с ГИС, и программисты

Возможно, вы знаете все о геоидах и проекциях и представляете, где найти исходную информацию. Вы умеете создавать потрясающие приложения, используя ArcGIS, MapInfo, Leaflet, OpenLayers, Google Maps или другие инструменты с поддержкой Ajax. Вы мастерски генерируете шейп-файлы Esri с помощью QGIS или ArcGIS и создаете шедевры картографии. Не исключено, что вы даже умеете добавлять данные в пространственную базу и извлекать их оттуда. Вот только вопросы о данных ставят вас в тупик. Одно дело – нарисовать на карте все магазины Walmart в США, и совсем другое – определить, сколько магазинов Walmart находится к востоку от Миссисипи, не пересчитывая все нарисованные геометки. Конечно, чтобы ответить на эти вопросы, можно было бы написать процедурный код, но мы покажем вам гораздо более быстрый способ.

Так что же предлагает база данных с пространственной поддержкой, чего еще нет у вас под рукой?

- Она дает возможность легко сочетать пространственные данные с другими корпоративными данными, такими как финансовая информация, данные наблюдений и маркетинговая информация. Да, это можно сделать с помощью шейп-файлов Esri, файлов KML и других форматов файлов ГИС, но это требует дополнительных шагов и ограничивает возможности соединения с другими связанными данными. Такие базы данных, как PostgreSQL, используют планировщик запросов для ускорения соединений и предлагают большое количество статистических функций, позволяющих достаточно просто писать и быстро выполнять довольно сложные запросы, в том числе и для сводной статистики.
- Вокруг баз данных существует так много инфраструктуры, что сбор пользовательских данных – независимо от того, рисует ли этот пользователь геометрический объект на экране и вводит связанную с ним информацию или щелкает по точ-

кам на карте, – становится намного проще, если вы используете PostGIS. Возьмем, к примеру, развертывание собственного веб-приложения, написанного на .NET, PHP, Perl, Python, Java или каком-либо другом языке. У каждого из них уже есть драйвер для PostgreSQL, упрощающий вставку и получение данных. Добавьте к этому функции преобразования текста в тип `geometry`, `geometry` в SVG, KML или GeoJSON и другие функции обработки, которые предоставляет PostGIS, а также функции генерирования и обработки геометрических объектов, которые доступны на таких платформах, как OpenLayers, MapServer и GeoServer. У вас несметное количество вариантов на выбор.

- Реляционная база данных обеспечивает административную поддержку, позволяющую легко контролировать, кто и к чему имеет доступ, будь то текстовый атрибут или геометрический объект.
- PostgreSQL поддерживает триггеры, позволяющие генерировать, в частности, связанные геометрические объекты в других таблицах, когда происходят определенные события базы данных.
- В PostgreSQL реализован механизм MVCC (*multiversion concurrency control* – многоверсионное управление конкурентным доступом), позволяющий вашей системе работать с сотнями пользователей, одновременно читающими и обновляющими данные.
- PostgreSQL предоставляет возможность сохранять в базе данных пользовательские функции, которые можно вызывать из разных приложений. PostgreSQL предлагает на выбор несколько языков для написания хранимых функций.
- Если вы привыкли к настольным ГИС-инструментам, не беспокойтесь: выбор пространственной СУБД, такой как PostGIS, не означает, что вам нужно отказаться от любимых инструментов. Manifold, Cadcorp, MapInfo 10 и выше, AutoCAD, Esri ArcGIS, ArcMap, Server и различные широко используемые настольные инструменты имеют встроенную поддержку PostGIS. Safe FME, любимый профессионалами инструмент ETL, с давних пор поддерживает PostGIS.

Специалисты-практики, работающие с базами данных

Уже получив какой-то опыт работы с базами данных, вы можете столкнуться с вопросом, касающимся пространственных свойств данных. Без пространственной базы вы вынуждены ограничивать свое мышление координатами, названиями местоположений или другими географическими атрибутами, которые можно свести к цифрам и буквам. Это прекрасно подходит для точечных данных, но вы окажетесь в полной растерянности, когда в игру вступят

области и регионы. Можно найти всех людей по фамилии Смит в округе, но как отыскать всех Смитов, живущих в пределах 10 километров от округа?

Мы хотим, чтобы читатели, имеющие опыт работы со стандартными реляционными базами, поняли, что данные – это не только числа, даты и символы и что с нетекстовыми данными SQL тоже позволяет совершать удивительные вещи. Конечно, можно было бы хранить изображения, документы и другие вещи в обычной реляционной базе данных, но мы сомневаемся, что у вас получилось бы написать соединение с такими полями.

Ученые, исследователи, преподаватели и инженеры

Многие высококвалифицированные ученые, исследователи, преподаватели и инженеры применяют инструменты пространственного анализа к собранным данным, моделируют свои изобретения или обучают студентов. Не мечтая уподобить себя этим людям, мы восхищаемся ими, потому что они формируют новое знание и коренным образом улучшают нашу жизнь. Но хотя они многое знают о математике, биологии, химии, геологии, физике, инженерии и т. д., они не обучены управлению базами данных, использованию реляционных баз данных или ГИС. Если вы один из таких созидателей, мы надеемся предоставить вам достаточно информации для того, чтобы вы освоили новые навыки без лишней суеты.

Что дают PostgreSQL и PostGIS?

- Они дают возможность интеграции со статистическими пакетами, такими как R, и вы даже можете писать хранимые функции баз данных на PL/R, который предоставляет возможности языка R.
- PostgreSQL также поддерживает PL/Python и PL/JavaScript, которые позволяют использовать огромное количество научных библиотек на языках Python и JavaScript непосредственно в базе, ближе к данным, чем в обычном окружении Python.
- В то время как многие считают PostGIS инструментом для географических информационных систем (что явствует из названия), мы рассматриваем ее как инструмент для пространственного анализа. Отличие в том, что география фокусируется на Земле и системах координат, привязанных к Земле, а пространственный анализ – на пространстве и его использовании. Пространственная система координат может быть специфичной для муравейника, или для плана атомной станции, местонахождение которой еще предстоит определить, или для различных областей мозга, либо же использоваться как инструмент визуализации для моделирования чего-либо невизуального по своей сути – например, при моделировании

процессов. Возможно, вы полагаете, что узкоспецифичная область ваших интересов не смыкается с пространственным анализом, однако мы призываем вас копнуть глубже.

- База данных – это естественный репозиторий больших объемов данных, и у нее есть большое количество встроенных статистических и сводных функций и конструкций для создания полезных отчетов и анализа. Если вы имеете дело с данными пространственного характера или используете пространство для визуализации, PostGIS предоставляет дополнительные функции для расширения этого анализа.
- Большую часть данных, необходимых для научных исследований, можно собирать автоматически (GPS, системы сигнализации, устройства дистанционного зондирования) и напрямую передавать в базу данных через автоматизированные каналы или стандартные форматы импорта. На самом деле инструменты для сбора данных, такие как смартфоны и беспилотные летательные аппараты, с каждым днем становятся все дешевле и доступнее для широких слоев населения; кроме того, дешевеет и оборудование для хранения данных.
- Части данных легко распределить. Реляционная база данных идеально подходит для создания того, что мы называем «распределителями данных» или «витринами данных», которые позволяют другим исследователям легко получать только то подмножество данных, которое нужно им для исследования, или предоставлять данные, чтобы их легко можно было скачать.

Мы перечислили основные группы потенциальных пользователей пространственных баз данных, но этот список неполон. Если вы когда-нибудь смотрели на мир и думали, как было бы здорово, если бы можно было сопоставить статистику преступности со статистикой озеленения или рассчитать, где и когда лучше всего сажать урожай, учитывая модель рельефа и температурные колебания местности, то PostGIS может оказаться для вас самым простым и экономичным инструментом.

Структура книги

В книге три основные части и несколько приложений.

Часть I. Изучение PostGIS

Первая часть охватывает основные понятия пространственных реляционных баз данных и, в частности, PostGIS/PostgreSQL. Ее цель – познакомить вас со стандартными концепциями и методами работы с базами данных ГИС. К концу чтения вы должны четко раз-

бираться в типах данных `geometry`, `geography`, `raster` и `topology`, а также понимать, для решения каких задач нужен каждый из этих типов. Вы получите базовое представление о пространственных системах координат и вариантах хранения баз данных. Самое главное, вы научитесь загружать, запрашивать и просматривать пространственные данные в базе данных PostgreSQL с поддержкой PostGIS.

Часть II. Запускаем PostGIS

Эта часть посвящена решению реальных пространственных задач в PostGIS и оптимизации скорости. Вы узнаете, как:

- выполнить анализ близости, используя типы данных `geometry` и `geography`;
- использовать различные виды векторных операций для оптимизации данных;
- выполнить растровую обработку с помощью как растровых, так и векторных данных;
- создавать новые векторные данные с помощью растровой обработки, алгебраических выражений, гистограмм и других функций, предназначенных для работы с растрами, для вычисления статистики по интересующей вас области;
- создавать большие растры из маленьких, используя агрегатные функции;
- использовать TIGER – геокодер PostGIS – для нормализации адресов, геокодирования и обратного геокодирования;
- использовать топологические объекты для обеспечения согласованности при редактировании данных;
- упростить целую сеть геометрических объектов и при этом сохранить связность в упрощенном наборе данных.

Часть III. Использование PostGIS с другими инструментами

В третьей части описываются инструменты, чаще всего используемые в PostGIS для создания приложений. Мы рассмотрим pgRouting, инструмент, который можно использовать с PostGIS непосредственно в базе данных для создания приложений сетевой маршрутизации. Кроме того, поговорим о языках хранимых процедур PostgreSQL: PL/Python, PL/R и PL/V8 (он же PL/JavaScript). И наконец, перейдем к краткому изучению PostGIS в веб-приложениях. Мы разберем различные картографические серверы, используемые с PostGIS, а также JavaScript-API библиотек OpenLayers и Leaflet и использование PostGIS JSON и функций вывода векторных тайлов для создания интерактивной веб-карты.

Приложения

В книге три приложения.

Приложение 1 содержит дополнительные ресурсы для получения справки по PostGIS и вспомогательным инструментам, обсуждаемым в основном тексте.

Приложение 2 показывает, как приступить к работе с PostgreSQL и PostGIS.

Приложение 3 представляет собой начальный курс по SQL, в котором объясняются концепции `JOIN`, `UNION`, `INTERSECT`, `EXCEPT`, общие табличные выражения и ключевое слово `LATERAL`. В нем обсуждаются основы свертывания данных с помощью агрегатных функций и агрегатных конструкций, а также более сложные вопросы использования оконных функций и рамок.

О коде

На протяжении всей книги используются следующие соглашения об оформлении программного кода:

- листинги оформлены моноширинным шрифтом;
- тот же шрифт применяется в тексте для выделения фрагментов кода;
- ключевые положения и новые термины представлены во врезках и в примечаниях;
- вместо комментариев в коде используются аннотации. Они выделяют важные концепции или области кода. Некоторые аннотации отображаются с пронумерованными маркерами, например ❶, которые объясняются далее по тексту.

Примеры и данные для всех глав этой книги можно скачать на сайте *www.dmkpress.com* (страница книги «PostGIS в действии»). Также вы можете посетить страницы авторов на сайтах *www.postgresonline.com* и *www.bostongis.com*.

О названии

Сочетая введения, обзоры и практические примеры, книги серии «В действии» призваны помочь вам, если вы учитесь и хотите запомнить то, чему научились. Согласно исследованиям в области когнитивной науки, люди помнят то, что они обнаруживают в ходе самомотивированного исследования.

Хотя никто в издательстве Manning не занимается когнитивными исследованиями, мы убеждены: для того чтобы навсегда запомнить изученное, оно должно пройти этапы исследования, игры и, что интересно, пересказывания изучаемой темы. Люди понимают и запоминают новые вещи, т. е. осваивают их, только после актив-

ного изучения. Они учатся в действии. Важной особенностью книг этой серии является тот факт, что они основаны на примерах. Они побуждают читателя пробовать что-то новое, экспериментировать с новым кодом и исследовать новые идеи.

Есть и другая, более приземленная причина для названия этой книги: наши читатели – занятые люди. Они используют книги, чтобы выполнить работу или решить задачу. Им нужны книги, которые можно открыть на любой странице и изучать именно то, что нужно, и тогда, когда это нужно, книги, которые помогут им в действии. Издания этой серии предназначены именно для таких читателей.

Об авторах

Регина Обе и Лео Хсу – консультанты по базам данных и авторы нескольких книг. Регина – член основной группы разработчиков PostGIS и руководящего комитета проекта.

Часть 1

Введение в PostGIS

Добро пожаловать в книгу «PostGIS в действии». PostGIS – это пространственное расширение с открытым исходным кодом для системы управления базами данных PostgreSQL. Эта книга научит вас основам пространственных баз данных в целом, ключевым понятиям географических информационных систем (ГИС) и, в частности, тому, как настраивать базы данных с поддержкой PostGIS, загружать в них данные и выполнять запросы. То, что, как вы думали, возможно только с настольной ГИС-системой, вы научитесь делать несколькими строками кода на языке SQL. Используя пространственный SQL, значительную часть тяжелой работы, которая потребовала бы большого количества ручных операций в настольных ГИС-инструментах, можно программировать в виде сценариев и автоматизировать.

Книга состоит из трех частей и трех приложений.

Часть I охватывает основы пространственных баз данных, ГИС и работы с пространственными данными. Хотя она посвящена PostGIS, многие концепции, которые вы изучите в этой части, в равной степени применимы и к другим пространственным реляционным базам данных.

Глава 1 посвящена основам пространственных баз данных и тому, что дает пространственная база данных по сравнению с обычной

реляционной базой данных. Также в ней представлены функции, уникальные для PostGIS. Эта глава завершается динамичным примером загрузки координат ресторанов быстрого питания и преобразования их в точки, загрузки данных о дорогах из шейп-файлов Esri и создания пространственных сводок путем объединения этих двух наборов данных.

В главе 2 рассматриваются все пространственные типы, которые может предложить PostGIS. Вы узнаете, как создавать объекты этих типов с помощью различных функций, и познакомитесь с особенностями каждого пространственного типа.

Глава 3 представляет собой введение в пространственные системы координат. Мы объясним концепции, лежащие в их основе, расскажем, почему системы координат важны для работы с типами `geometry`, `raster` и `topology`, и покажем, как с ними работать.

В главе 4 рассказывается, как загружать пространственные данные в PostGIS, используя штатные утилиты, а также дополнительные сторонние инструменты с открытым исходным кодом. Вы узнаете, как загружать геометрические и географические данные с помощью инструмента командной строки shp2pgsql, обычно входящего в состав дистрибутивов PostGIS, а также загрузчика и экспортера с графическим интерфейсом shp2pgsql-gui, входящего в состав некоторых настольных дистрибутивов PostGIS. Вы также узнаете, как загружать растровые данные с помощью инструмента командной строки raster2pgsql, входящего в пакет PostGIS, и как импортировать и экспортировать растровые и векторные данные различных форматов с помощью пакета GDAL/OGR. Кроме того, вы увидите, как запрашивать данные из внешних источников, не загружая их, с помощью оберток сторонних данных PostgreSQL.

В главе 5 рассматриваются некоторые наиболее распространенные настольные инструменты с открытым исходным кодом для доступа к данным PostGIS и их просмотра.

В главе 6 начинается знакомство с простыми базовыми функциями, которые используются для работы с типами данных `geometry` и `geography`. Все они либо принимают на вход отдельные геометрические или географические объекты и видоизменяют их, либо получают текстовые представления и преобразуют их в пространственные объекты PostGIS.

Глава 7 представляет собой введение в растровые функции. В ней рассказывается о функциях для создания растров, получения данных о растрах и установки значений пикселей.

Глава 8 завершает первую часть знакомством с пространственными отношениями. Пространственные отношения наиболее важны при работе с наборами данных. В последующих частях книги эти концепции пригодятся нам, в частности для написания пространственных соединений.

1 Что такое пространственная база данных?

В этой главе:

- решение задач с помощью пространственных баз данных;
- типы пространственных данных;
- моделирование с учетом пространства;
- почему PostGIS/PostgreSQL можно использовать как пространственную базу данных;
- загрузка пространственных данных и запросы к ним.

Для большинства людей первое знакомство с пространственно ориентированными приложениями начинается с геометок, указывающих интересные места на интерактивной карте. Это дает некоторое представление об обширном и разнообразном мире геоинформационных систем (ГИС).

Мы начнем эту главу с модели геометок и убедимся в ее ограниченности. Поэтому нам потребуется пространственная база дан-

ных – и не какая-нибудь, а PostGIS. PostGIS – это расширение системы управления базами данных PostgreSQL, предназначенное для работы с пространственными данными. Мы представим краткое введение в PostGIS и постараемся заинтриговать читателя примером, который выходит далеко за рамки того, что можно сделать с помощью геометок.

Данные и код, использованные в этой главе, можно найти на странице *www.postgis.us/chapter_01_edition_3*.

1.1. Мыслить пространственно

Такие популярные картографические сервисы, как OpenStreetMap, Mapbox, Google Maps, Bing Maps и MapQuest, дали возможность указывать местоположение геометкой на детализированной интерактивной карте. Нам больше не нужно прибегать к помощи слов, объясняя: «Поверните направо у супермаркета, и третий дом справа будет ваш, там пес привязан у входа». Кроме того, нам больше не нужно отыскивать на бумажной карте, где мы сейчас находимся.

Карты нужны не только для того, чтобы добраться куда-то. И большие, и мелкие компании пользуются тем, что карты помогают находить закономерности в данных. Нанеся на карту адреса любителей пиццы, национальная сеть пиццерий может определить место торжественного открытия их очередного заведения. Политические организации, планирующие избирательные кампании, с легкостью могут увидеть на карте, где находятся неопределившиеся или незарегистрированные избиратели, и соответствующим образом спланировать свои маршруты поездок для встреч с ними. Хотя модель геометок и упрощает анализ географических данных, все рассуждения при этом носят исключительно визуальный характер.

Расставив геометки, сеть пиццерий может увидеть, где в городе сосредоточены любители пиццы, но что, если нужно разделить их по уровню дохода? Если у сети есть предложение для гурманов, было бы неплохо открывать новые рестораны там, где уровень доходов не опускается ниже среднего. Специалисты по планированию могут использовать геометки разных цветов для обозначения разных уровней, но в таком случае рассуждать эвристически на основе изображения им будет намного сложнее, как показано на рис. 1.1. Придется смотреть не только на концентрацию меток, но и помнить, что у них могут быть разные цвета или иконки. Добавьте к карте еще одно измерение, например семьи с непереносимостью лактозы, и наш мозг уже не справится с этой задачей. Вот тут на помощь и приходят пространственные базы данных.

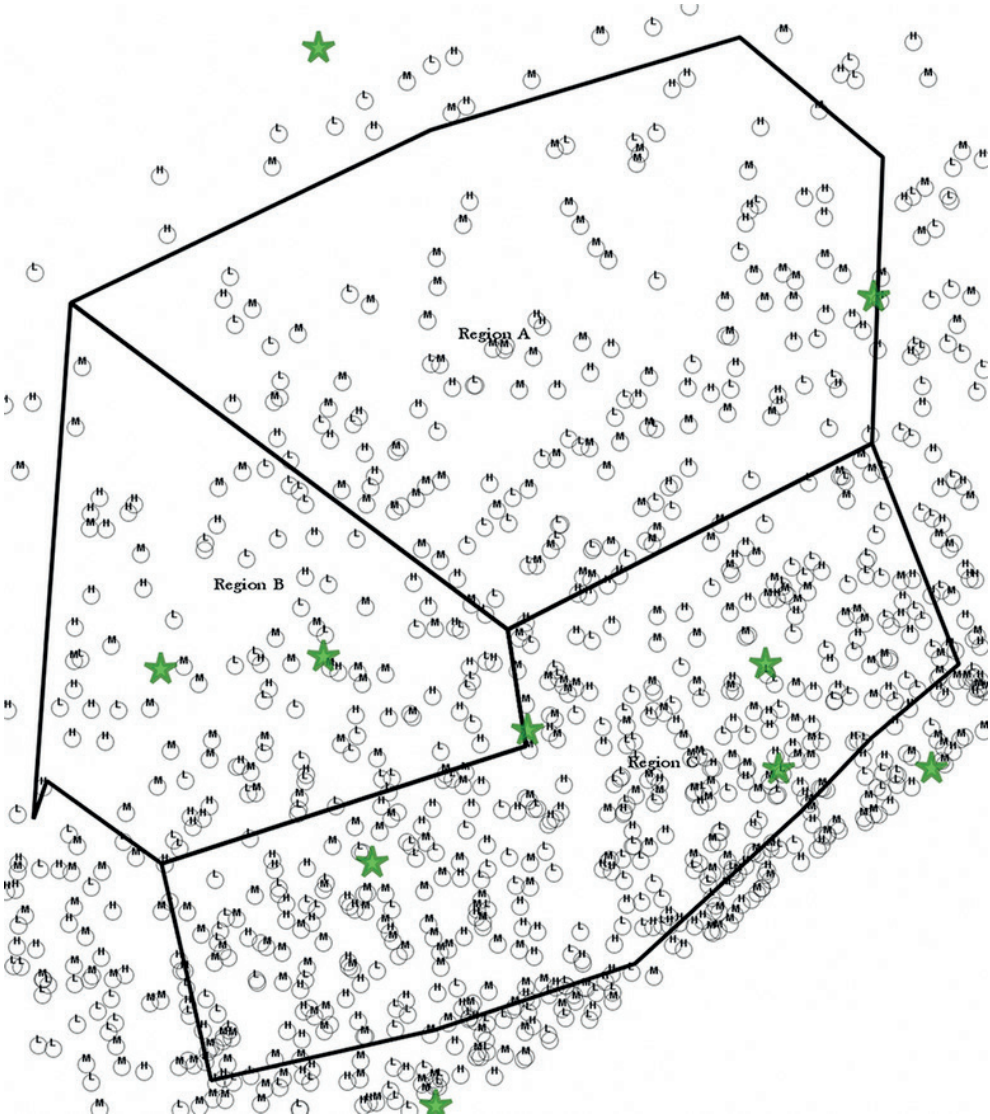

Рис. 1.1. Месиво из геометок

Пространственная база данных – это база данных с типами, специально разработанными для хранения пространственных объектов. Эти типы можно использовать в таблицах базы данных. Сохраняемая информация обычно носит географический характер, например местоположение точки или границы озера. Пространственная база данных также предоставляет функции и индексы для запросов к пространственным данным и управления ими. Функции можно вызывать из языка запросов, такого как SQL. Пространственная база данных часто используется просто как хранилище простран-

ственных данных, но этим ее возможности не ограничиваются. Большинство пространственных баз данных имеют реляционную природу, хотя это и необязательно. Пространственная база предоставляет инструмент для хранения, анализа и организации данных – три в одном.

Визуальное представление данных – не единственная цель пространственной базы. Для планирования пиццерий можно хранить любые атрибуты семей, где любят пиццу, включая уровень дохода, количество детей, историю заказов пиццы и даже религиозные и культурные особенности (поскольку они влияют на выбор начинки). Что еще более важно, анализ не ограничен количеством переменных, с которыми мозг может справляться одновременно. Специалисты по планированию могут выполнять очень детальные запросы, например получать список районов, отсортированный по количеству любителей пиццы с высоким доходом, у которых в семье больше двух детей. Кроме того, они могут легко включать дополнительные данные из различных источников, например расположение и рейтинг других пиццерий с сайтов отзывов о ресторанах или уровень заботы населения о своем здоровье в разных районах, по данным местной комиссии по здравоохранению. Их запросы к базе данных могут быть весьма сложными, например можно вывести район с наибольшим количеством семей, где ближайшее расстояние до любой пиццерии с рейтингом ниже 5 в среднем превышает 16 км (10 миль). И убрать из выборки те районы, жители которых за здоровый образ жизни.

В табл. 1.1 показано, как могут выглядеть результаты такого запроса.

Таблица 1.1. Результат запроса к пространственной базе данных

Район	Количество семей	Количество ресторанов	Расстояние
Район А	194	1	17,1 км

Возможно, вы не занимаетесь картографией, а изо дня в день работаете с данными, которые не нужно наносить на карту. Вы можете найти всех сотрудников, живущих в Чикаго, или вычислить количество клиентов по каждому почтовому индексу. Если вы знаете широту и долготу домашних адресов всех сотрудников; вы можете вычислить среднее расстояние, которое каждый сотрудник должен преодолеть, добираясь до работы. Это предел пространственных запросов, которые можно сформулировать при работе с обычными базами, где типы данных представлены в основном текстом, числами и датами.

Но предположим, что требуется подсчитать количество домов в пределах двух миль от береговой линии, которые потребуется

эвакуировать в случае урагана. Или необходимо найти количество домохозяйств, которые затронет шум новой взлетно-посадочной полосы. Без пространственной поддержки эти вопросы потребуют сбора или вычисления дополнительных значений для каждой точки данных. В случае с береговой линией нужно будет определить расстояние от пляжа до каждого дома. Для этого могут потребоваться алгоритмы поиска кратчайшего расстояния до отрезков береговой линии и набор запросов SQL, чтобы сначала упорядочить все дома по близости к пляжу, а затем отбросить лишнее. С пространственной поддержкой требуется только немного переформулировать запрос: «Найти все дома в радиусе двух миль от береговой линии». База данных с пространственной поддержкой может работать с такими типами данных, как береговые линии (которые представлены ломаными линиями), буферные зоны (представленные полигонами, или многоугольниками) и пляжные домики (представленные точками).

Для достижения любых достойных целей требуется приложить усилия. Чтобы овладеть всеми возможностями пространственного анализа, придется карабкаться по кривой обучения. Зато база данных, с которой мы вас познакомим, абсолютно бесплатна – в отличие от всего остального, что есть хорошего в жизни.

Если вы можете разобраться, как получить данные для нанесения на карту, то у вас не возникнет проблем со следующим шагом. Если вы умеете писать запросы для непространственных баз данных, мы откроем вам нечто большее, нежели обыденный мир чисел, дат и строк. Итак, приступим.

1.2. Знакомство с PostGIS

PostGIS – это бесплатная библиотека с открытым исходным кодом, добавляющая поддержку пространственных данных в объектно-реляционную СУБД PostgreSQL с открытым исходным кодом. Мы хотим, чтобы вы выбрали PostgreSQL в качестве реляционной базы данных и PostGIS в качестве ее расширения.

1.2.1. Почему PostGIS

PostGIS стартовала как проект консалтинговой компании Refractions Research (*http://refractions.net*), которая расположена в Канаде в городе Виктория и занимается геопространственными технологиями. С тех пор PostGIS применяется и улучшается правительственными учреждениями, университетами, общественными организациями и другими компаниями.

Возможности PostGIS расширяются благодаря использованию в ней других проектов:

- *Proj* – отвечает за построение проекций, теперь уже в седьмой версии;

- *Geometry Engine Open Source (GEOS)* – предоставляет расширенную поддержку обработки геометрических объектов;
- *Geospatial Data Abstraction Library (GDAL)* – содержит множество расширенных функций для обработки растровых данных;
- *Computational Geometry Algorithms Library* (CGAL/SFCGAL) – обеспечивает расширенный трехмерный анализ.

Большинство этих проектов, включая и PostGIS, теперь находятся под эгидой оранизации Open Source Geospatial Foundation (OSGeo).

Основа PostGIS – объектно-реляционная система управления базами данных PostgreSQL, которая предоставляет транзакционную поддержку, gist-индексы для пространственных объектов и планировщик запросов. Это прекрасное свидетельство мощности и гибкости PostgreSQL, которую компания Refractions Research предпочла остальным системам для своего проекта.

1.2.2. Соответствие стандартам

PostGIS и PostgreSQL соответствуют отраслевым стандартам лучше, чем большинство других продуктов. PostgreSQL поддерживает многие новшества ANSI SQL. PostGIS поддерживает стандарты OGC и пространственный стандарт спецификации SQL Multimedia (SQL/MM). Это означает, что вы не просто учитесь использовать некий набор продуктов; вы получаете знания об отраслевых стандартах, которые помогут вам понять другие геопространственные базы данных и картографические инструменты, коммерческие или с открытым исходным кодом.

Что такое OGC, OSGeo, ANSI SQL и SQL/MM?

OGC расшифровывается как Open Geospatial Consortium. Это организация, разрабатывающая стандарты для доступа к географическим и пространственным данным и для их передачи. У нее имеются многочисленные спецификации, определяющие доступ к геопространственным данным из веб-сервисов, форматы передачи геопространственных данных и запросы к таким данным.

OSGeo расшифровывается как Open Source Geospatial Foundation. Эта организация финансирует, поддерживает и продвигает инструменты с открытым исходным кодом и бесплатные данные для ГИС. В какой-то мере эти организации дублируют друг друга. Обе они стремятся сделать данные и инструменты ГИС доступными для всех, а это означает, что они заботятся об открытых стандартах.

Часто встречаются термины ANSI SQL и ISO SQL. Стандарты ANSI/ISO SQL определяют общие рекомендации, которым должны следовать реализации SQL. Эти рекомендации часто датированы определенным годом, например ANSI SQL 92 и ANSI SQL:2016, и основаны на специфи-

кациях прошлых лет. Многие реляционные базы данных поддерживают бóльшую часть спецификации ANSI SQL 92, но не так много из более поздних версий. PostgreSQL поддерживает множество новых рекомендаций, некоторые из которых мы рассмотрим в приложении 3.
ANSI/ISO SQL Multimedia (SQL/MM) – это спецификация, которая, помимо прочего, определяет стандартные функции для пространственных данных, используемых в SQL.
Пространственные данные встречаются все чаще, и от лидирующих реляционных баз данных ожидается их поддержка. Поэтому многое из того, чем управляла OGC, сейчас уже относится к ANSI/ISO SQL. В результате новые спецификации SQL/MM часто ссылаются на пространственные типы с префиксом `ST_`, например `ST_Geometry` или `ST_Polygon`, вместо простых `Geometry` и `Polygon` из старых спецификаций OGC/SFSQL (Spatial Features for SQL).

Если ваши данные и API соответствуют стандартам, которые поддерживаются большим количеством программных продуктов (например, Cadcorp, Safe FME, AutoCAD, Manifold, MapInfo, Esri ArcGIS, ogr2ogr/GDAL, OpenJUMP, QGIS, Deegree, MapGuide, UMN MapServer или GeoServer), стандартными языками программирования (такими как SQL, JavaScript, PHP, Python, Ruby, Java, Perl или ASP.NET) и другими системами, – каждый сможет выбрать себе инструмент по душе, по потребностям и по карману и обмениваться информацией с другими. OSGeo старается гарантировать, что вы сможете просматривать и анализировать данные ГИС независимо от толщины вашего кошелька. OGC и ANSI/ISO SQL пытаются обеспечить соблюдение стандартов всеми продуктами, чтобы ваша работа была доступна всем, на какой бы дорогой ГИС-платформе она ни была выполнена. Это особенно важно для государственных учреждений, которые финансируются за счет налогов; для студентов, у которых есть желание и способности для изучения передовых технологий, но недостаточно средств; и даже для небольших поставщиков, имеющих привлекательные предложения для определенных категорий пользователей, но которые часто игнорируются более крупными игроками, поскольку не могут поддерживать частные стандарты API (или лишены доступа к ним).

PostGIS поддерживается большим количеством проприетарных ГИС-инструментов для настольных компьютеров и серверов. Именно с PostGIS обычно работает большинство настольных и веб-серверных геопространственных картографических систем с открытым исходным кодом; в качестве платформы для пространственных реляционных баз данных его предпочитает большинство государственных учреждений и стартапов.

Мы рассмотрим некоторые наиболее распространенные инструменты, которые работают с PostGIS, в главах 5 и 17.

1.2.3. Сила PostGIS

PostGIS предоставляет множество пространственных операторов, функций и типов данных, а также индексную поддержку пространственных данных в PostgreSQL. Если добавить к этому дополнительные функции, реализуемые PostgreSQL и другими связанными проектами, в вашем распоряжении оказывается полноценная система, которая прекрасно подходит для сложного геоинформационного анализа и является ценным инструментом для изучения ГИС.

Эти возможности будет непросто найти в других пространственных базах данных:

- функции для работы с GeoJSON, языком разметки Keyhole (KML) и векторными тайлами Mapbox (MVT), позволяющими веб-приложениям напрямую взаимодействовать с PostGIS без необходимости дополнительных преобразований;
- разнообразные функции обработки геометрических объектов, выходящие далеко за рамки базовых операций, включая функции исправления дефектов в геометрических объектах, а также упрощения и деконструкции геометрических объектов;
- встроенная поддержка трехмерных и топологических объектов;
- более 300 бесшовных операций для совместной работы с векторами и растрами, а также для преобразований между ними.

Форматы данных GeoJSON, KML и MVT

Geographic JavaScript Object Notation (GeoJSON; *http://geojson.org*) и язык разметки Keyhole (KML; *http://en.wikipedia.org/wiki/Keyhole_Markup_Language*) – два старых и наиболее распространенных векторных формата, применяемых в картографических веб-приложениях. Mapbox Vector Tiles (MVT) – относительно новый стандарт, который за последние несколько лет стал довольно популярен.

- GeoJSON – это расширение формата JSON, который используется для представления объектов JavaScript. Он добавляет к JSON поддержку географических объектов;
- KML – это формат XML, разработанный компанией Keyhole (которая была приобретена Google). Сначала формат использовался в картографических продуктах Google, позднее он стал поддерживаться различными картографическими API;
- Mapbox Vector Tiles (MVT) – это двоичный векторный формат, популяризированный компанией Mapbox, который распределяет информацию по тайлам, содержащим двоичные векторные данные. Они обычно занимают меньше места, чем стандартные растровые тайлы, и позволяют масштабировать изображение и применять к нему разные стили на стороне клиента.

И это только три из множества выходных форматов PostGIS.

1.2.4. Создана на базе PostgreSQL

Основная причина, по которой PostGIS была создана на базе платформы PostgreSQL, заключается в простоте расширяемости, которую PostgreSQL обеспечивает для создания новых типов данных и операторов, а также для управления индексированием. PostgreSQL изначально проектировалась таким образом, чтобы ее можно было расширять.

Родословная PostgreSQL восходит почти к эпохе зарождения реляционных баз данных. Она в родстве с системами баз данных Sybase и Microsoft SQL Server, потому что авторы Sybase работали до этого в Калифорнийском университете в Беркли над проектами Ingres и PostgreSQL вместе с Майклом Стоунбрейкером. Многие считают его отцом Ingres и PostgreSQL и одним из основателей объектно-реляционных систем управления базами данных. Лицензия на исходный код Sybase SQL Server была позже передана компании Microsoft; продукт стал называться Microsoft SQL Server.

PostgreSQL славится тем, что это самая передовая из существующих баз данных с открытым исходным кодом. По скорости и функциональности она может посоперничать с популярными коммерческими предложениями корпоративного уровня, управляя петабайтными базами. Добавление новых функций, упрощающих ее использование, сделало PostgreSQL не только самой передовой, но, возможно, самой гибкой и лучшей реляционной базой данных.

PostgreSQL становится универсальной базой данных, которая не жертвует потребностями и желаниями пользователей. Большинство дистрибутивов ОС содержат довольно свежую версию, PostgreSQL, обеспечивая быстрый и безболезненный процесс установки. Со времени выхода предыдущего издания этой книги появились облачные решения для PostgreSQL и PostGIS. Популярные облачные версии PostgreSQL, с которыми работают пользователи PostGIS, включают CartoDB, Heroku PostgreSQL, базу данных Microsoft Azure для PostgreSQL, а также Amazon RDS и Aurora для PostgreSQL. Служба хранилища данных Google BigQuery, хотя и не является PostgreSQL, переняла конструкции PostgreSQL, имена функций PostGIS и пространственные типы для запросов к своим пространственным данным (*https://cloud.google.com/bigquery/docs/gis-data*).

1.2.5. Бесплатная

Лицензии на SQL Server Standard начинаются от 5000 долларов; скромный сервер может запросто обойтись вам в 20 000 долларов. Бесплатная версия SQL Server, хотя и обладает той же пространственной функциональностью, что и платная версия, ограничена по памяти и ядрам.

Oracle Standard до версии Oracle 19c поставлялась только с модулем Oracle Locator, который обладает лишь базовой пространственной функциональностью. Чтобы получить расширенные пространственные возможности, требовалось приобретение Oracle Spatial. Начиная с Oracle 19c все выпуски включают поддержку Oracle Spatial.

PostGIS бесплатна, и этим все сказано.

1.2.6. Свободная

PostGIS и PostgreSQL – это программное обеспечение с открытым исходным кодом. PostGIS распространяется под лицензией GPLv2+, а PostgreSQL – под лицензией, сходной с BSD. Это означает, что вы можете просматривать и менять исходный код. Если нужная функция отсутствует, можно предложить патч с исправлением или заплатить разработчику, который это сделает. Добавление функций в PostGIS и PostgreSQL обычно стоит намного меньше, чем стоимость лицензирования для проприетарных аналогов. Команды разработчиков PostGIS или PostgreSQL очень быстро реагируют на сообщения об ошибках – быстрее, чем большинство поставщиков проприетарных баз данных.

При работе с PostGIS и PostgreSQL у вас больше свободы, чем в случае с сопоставимыми проприетарными предложениями. Можно установить PostGIS на любое количество серверов без искусственных ограничений на число ядер, которые можно использовать.

Открытость PostGIS породила множество дополнений, созданных пользователями, и функций, финансируемых сообществом. Вот самые заметные на сегодняшний день: поддержка растров, геодезическая поддержка, поддержка топологий, улучшенная поддержка 3D, более быстрые пространственные индексы, усовершенствования геокодера TIGER и средство просмотра пространственных данных PostGIS в инструменте управления базами данных pgAdmin4, который обычно поставляется с PostgreSQL.

Циклы выпуска PostGIS и PostgreSQL значительно короче, чем у коммерческих предложений. Благодаря вкладу пользователей новые основные версии PostgreSQL появляются раз в год, а дополнительные выпуски, исправляющие ошибки, – каждые два или три месяца. Вам не придется тратить годы в ожидании функций, обещанных в последующих выпусках. Если вы хотите иметь в своем распоряжении самые передовые технологии, можете скачивать свежую сборку хоть каждую неделю.

1.2.7. Альтернативы PostGIS

Следует признать, что PostGIS – не единственная используемая сегодня пространственная база данных. Раньше преобладали проприетарные предложения, но PostGIS изменила такое положение дел. Последователи PostGIS предпочитают установки небольшого размера,

которые можно использовать на мобильных устройствах. Также пространственные функции начинают появляться и в нереляционных базах данных, таких как MongoDB, CouchDB, Elastic Search и Solr.

Oracle Spatial

Все началось с корпорации Oracle. В Oracle 7 совместные с канадскими учеными усилия по разработке привели к появлению SDO (Spatial Data Option). В более поздних выпусках Oracle переименовала это детище в Oracle Spatial.

Oracle Spatial недоступно в более дешевых версиях Oracle. Только раскошелившись на Oracle Enterprise Edition, можно позволить себе роскошь приобрести эту опцию.

Standard Oracle поставляется с модулем Oracle Locator, который предлагает основные геометрические типы, функции расстояния, некоторые пространственные агрегаты и ограниченную обработку пространственных данных. Пользователи требовали от Oracle предоставления большей поддержки пространственных данных в Oracle Locator, поэтому более новые версии Oracle Locator предоставляют базовые функции, такие как объединение и пересечение, но не включают функции агрегирования и многое другое, что вы найдете в PostGIS, SQL Server и Oracle Spatial.

Microsoft SQL Server

Корпорация Microsoft представила поддержку пространственных данных в SQL Server 2008, добавив встроенные типы Geometry и Geodetic Geography с сопутствующими пространственными функциями. К чести Microsoft, нужно отметить, что вы получаете одинаковые возможности в версиях Express, Standard, Enterprise и Datacenter. Вы можете быть ограничены в отношении размера базы данных, количества процессоров, которые можно использовать, и того, какие доступны функции планирования запросов.

Пространственные возможности Microsoft, за исключением объектов, составленных из кривых, и геодезической поддержки, меркнут по сравнению с PostGIS. По всей видимости, в Microsoft SQL Server эти возможности реализованы лучше, чем в остальных системах, – это единственная база данных, которая поддерживает криволинейные геометрические объекты в геодезическом пространстве. В то же время системе недостает возможностей экспорта и импорта, например для форматов KML, GeoJSON и MVT, поддержки растров и тех многочисленных функций для обработки данных, которые есть в PostGIS.

SpatiaLite и GeoPackage

Наши любимцы в этом разделе – SpatiaLite и GeoPackage, которые являются дополнениями к SQLite, переносимой базе данных с от-

крытым исходным кодом. Они особенно интересны, потому что их можно использовать в качестве бюджетных компаньонов для PostGIS и других высокопроизводительных пространственных баз данных.

GeoPackage – это механизм хранения и передачи данных, как векторных, так и растровых, который стандартизован OGC. Внутри это реляционная база данных, как и PostGIS. Ее популярность растет благодаря таким инструментам, как QGIS, и она становится стандартом де-факто для экспорта данных.

GeoPackage обычно используется как хранилище данных, а функционал запросов отдается на откуп инструментам, которые его используют. С другой стороны, SpatiaLite включает в себя большую часть тех же функций, которыми располагает PostGIS, и использует те же библиотеки: GEOS, PROJ и GDAL. Таким образом, SpatiaLite – еще более подходящий компаньон для PostGIS из-за совпадения многих соглашений, и бóльшая часть экосистемы PostGIS также поддерживает или начинает поддерживать SpatiaLite/RasterLite.

SpatiaLite не хватает надежной базы данных корпоративного уровня, чтобы писать расширенные функции и функции пространственной агрегации. Поэтому некоторые пространственные запросы, доступные в PostGIS, в SpatiaLite написать сложнее или даже невозможно.

SpatiaLite, SQLite и GeoPackage хранят данные в виде одного файла, который легко передавать. Это упрощает процесс развертывания для пользователей, плохо ориентирующихся в базах данных или ГИС, которым нужна легковесная автономная база данных, дополняющая серверную базу, такую как PostGIS/PostgreSQL.

MySQL

В MySQL начиная с версии 4 имеется базовая поддержка пространственных типов данных, но база данных страдает из-за отсутствия мощного движка SQL. Ее основной аудиторией по-прежнему являются разработчики, которым база данных нужна для *хранения*, а не для *обработки*. Роковой ошибкой раннего этапа внедрения пространственной поддержки в MySQL было отсутствие индексирования, кроме как для таблиц MyISAM, – а ведь производительность пространственных запросов в значительной степени зависит от индексов. В версии MySQL 5.6 геометрические операции не только научились работать с ограничивающими прямоугольниками, но и добавились пространственные индексы в движке хранения InnoDB. Более новые версии MySQL и MariaDB предлагают еще больше функций, например экспорт в GeoJSON и другие форматы.

Oracle MySQL и другие ответвления MySQL, такие как MariaDB, шагнули далеко вперед, улучшив в версии 5.6 производительность вложенных запросов, но планировщик запросов и реализация SQL по-прежнему и близко не дотягивают до возможностей PostgreSQL, SQL

Server и Oracle, поэтому MySQL не подходит для выполнения такой сложной обработки, как пространственный анализ. Пространственная поддержка была значительно улучшена в MySQL 8 и MariaDB 10, но она по-прежнему не может составить конкуренцию PostGIS.

Хотя функциональность Oracle MySQL и MariaDB в основном совпадает, пространственная поддержка отличается. Сравнение см. на сайте MariaDB (*https://mariadb.com/kb/en/mysqlmariadb-spatial-support-matrix/*).

ArcGIS от Esri

Нужно отдать должное компании Esri, которая уже давно поставляет ArcGIS for Server со своим движком пространственной базы данных (SDE). Этот движок интегрирован в линейку продуктов ArcGIS и часто используется в помощь таким продуктам, как Microsoft SQL Server 2005 или Oracle Locator, которые сами недостаточно хорошо поддерживают пространственные операции.

В более старых версиях ArcGIS Desktop доступ к собственным возможностям пространственной базы данных можно было получить только через промежуточный уровень SDE. Более новые версии, начиная примерно с ArcGIS 10.0, позволяют напрямую обращаться к PostGIS и другим базам данных. Обходя промежуточное ПО, можно использовать любую версию PostGIS с ArcGIS Desktop.

Будьте осторожны при использовании ArcGIS, поскольку она устанавливает собственный вариант геометрических типов данных в PostgreSQL. Часто это приводит пользователей PostGIS в замешательство, поскольку при выборе типа данных `sde.st _geometry` вместо типа PostGIS `geometry` работать можно только через промежуточный слой Esri. Тип `sde.st_geometry` необходим для использования инструментов управления версиями Esri, но для большинства других целей он не нужен.

Хотя проприетарная модель Esri нас не устраивает, нужно сказать спасибо Esri за то, что она стала одной из первых крупных компаний, внедривших ГИС-анализ в коммерческие и государственные организации. Она заложила основу, но до сих пор препятствует развитию бесплатных ГИС с открытым исходным кодом.

1.3. Установка PostGIS

Мы рекомендуем устанавливать последние версии PostgreSQL и PostGIS, на момент написания этих строк – PostgreSQL 13 и PostGIS 3.1. Появление механизма расширений в PostgreSQL 9.1 значительно упростило установку таких дополнений, как PostGIS. Она выполняется в два шага:

- найдите и установите двоичные файлы для вашей операционной системы в каталоги PostgreSQL;

- при необходимости загрузите расширения отдельно в каждую базу данных. Например, если у вас на сервере 10 баз данных, но только для двух из них требуется PostGIS, нужно загрузить PostGIS только в эти две.

PostGIS должна быть загружена в каждую базу данных

Многих пользователей, работавших ранее с другими системами баз данных, смущает, что расширения PostgreSQL, такие как PostGIS, hstore, PL/JavaScript или PL/Python, должны быть загружены в каждую базу данных, в которой они будут использоваться. Это не относится к встроенным типам данных, таким как XML, JSON, JSONB, или типам для полнотекстового поиска, которые доступны всегда.

Многие популярные дистрибутивы Linux/Unix содержат в своих репозиториях PostGIS 3.1. Используйте `yum` или `apt` для установки двоичных файлов. Для пользователей Mac есть несколько популярных дистрибутивов, все они перечислены на странице установки PostGIS (*http://postgis.net/install*). Для MS Windows мы рекомендуем использовать приложение Stack Builder компании EnterpriseDB, если вам неудобно работать с командной строкой. Для этого приложения мы сопровождаем пакеты категории «Пространственные расширения» и стараемся размещать там все расширения PostGIS и множество связанных расширений, такие как pgRouting и pgPointcloud. В приложении 2 дана подробная информация о том, где найти двоичные файлы для вашей ОС.

Вместе с PostgreSQL поставляются два популярных инструмента: *psql* и *pgAdmin*. Они используются для создания баз данных и пользователей и составления запросов.

Psql – инструмент командной строки. Если вы ограничены текстовым терминалом, psql – ваш единственный вариант.

Если же у вас есть графический интерфейс, рекомендуем использовать более удобный для новичков pgAdmin. Его можно установить отдельно от PostgreSQL. Исходный код, а также готовые двоичные файлы можно найти на сайте *www.pgadmin.org*.

После того как вы успешно установили программы, можно создать базу данных с помощью следующей команды, используя psql или pgAdmin:

```
CREATE DATABASE postgis_in_action;
```

После создания базы данных необходимо подключиться к ней. В psql это можно сделать с помощью команды `\connect postgis_in_action`, а в pgAdmin нужно обновить дерево баз данных и выбрать новую базу.

После этого нужно загрузить PostGIS в свою базу данных, подключившись к ней и выполнив код из следующего листинга. Загруз-

ка этого расширения редко дает сбой, но вы можете столкнуться с ошибками зависимостей, особенно если у вас установлены более ранние версии PostGIS.

Листинг 1.1. Загрузка PostGIS в базу данных

```
CREATE SCHEMA postgis; (1)
GRANT USAGE ON SCHEMA postgis TO public; (2)
CREATE EXTENSION postgis SCHEMA postgis; (3)
ALTER DATABASE postgis_in_action SET search_path=public,postgis,contrib; (4)
```

(1) Создаем схему.
(2) Предоставляем доступ всем пользователям.
(3) Устанавливаем расширение postgis.
(4) Добавляем в путь поиска.

НА ЗАМЕТКУ Хотя это и не требуется, мы всегда устанавливаем postgis в отдельную схему, например postgis, чтобы функции не загромождали схему по умолчанию public.

В pgAdmin расширения также можно загружать, используя раздел **Extensions** (Расширения), показанный на рис. 1.2.

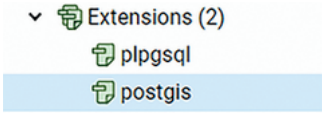

Рис. 1.2. База данных с установленным расширением postgis

ПРЕДУПРЕЖДЕНИЕ Диалог Extensionsв pgAdmin не позволяет указать схему для установки расширения postgis. По умолчанию postgis устанавливается в схеме public. Поэтому мы предпочитаем выполнять команду `CREATE EXTENSION` из окна запросов, а не пользоваться диалогом Extensions.

Если postgis нет в списке, его можно установить, щелкнув правой кнопкой мыши по ветке **Extensions** (Расширения), выбрав **New > Extension** (Создать > Расширение) и найдя postgis в меню.

Вы должны увидеть postgis в окне **Add Extension** (Создание расширения), если вы установили двоичные файлы и еще не загрузили расширение в выбранную базу данных.

Пакеты поддержки растров PostGIS 3 идут отдельно

До PostGIS 3 поддержка растров была включена как часть расширения postgis. Если вы используете версию PostGIS 3 или выше и хотите использовать растры и растровые функции, потребуется дополнительный шаг:

```
CREATE EXTENSION postgis_raster SCHEMA postgis;
```

1.3.1. Проверка версий PostGIS и PostgreSQL

После установки PostGIS отключитесь от базы данных и снова подключитесь к ней. Затем проверьте версии, чтобы убедиться, что установка прошла успешно. Выполните следующий запрос:

```
SELECT postgis_full_version();
```

Если все в порядке, вы должны увидеть версию PostGIS, а также версии поддерживаемых библиотек GEOS, GDAL, PROJ, LIBXML и LIBJSON, как показано ниже:

```
POSTGIS="3.1.1 3.1.1" [EXTENSION]
PGSQL="130" GEOS="3.9.1-CAPI-1.14.1" PROJ="7.1.1"
GDAL="GDAL 3.2.1, released 2020/12/29"
LIBXML="2.9.9" LIBJSON="0.12"
LIBPROTOBUF="1.2.1" WAGYU="0.5.0 (Internal)" TOPOLOGY RASTER
```

Установка инструментов визуализации

В отличие от обычных баз данных, ориентированных на символьную информацию, пространственные базы данных должны восприниматься визуально. Когда вы просматриваете растровый файл, то обычно хотите увидеть изображение, а не отдельные биты. Точно так же и для пространственных объектов визуализация удобнее текстового представления.

Для визуализации существует много бесплатных инструментов. Наиболее популярные из них – OpenJump и QGIS. Программа pgAdmin начиная с версии 4 3.3 включает в себя легковесное средство просмотра результатов запросов к пространственным данным. Однако, в отличие от OpenJump и QGIS, pgAdmin 4 не позволяет накладывать друг на друга несколько запросов. Кроме того, он не позволяет просматривать растры PostGIS, что умеет QGIS.

Мы рекомендуем установить несколько инструментов просмотра для сравнения. Глава 5 предлагает краткое руководство по установке, чтобы помочь вам начать работать с ними.

1.4. Пространственные типы данных

PostGIS определяет четыре основных пространственных типа: `geometry`, `geography`, `raster` и `topology`. Геометрические объекты поддерживались PostGIS с самого начала. Поддержка географических объектов появилась в PostGIS 1.5. PostGIS 2.0 подняла планку еще выше, представив растровый тип, площадные геометрические типы и поддержку *сетевой топологии*. В PostGIS 2.1 появилось множество новых функций, но, возможно, самым важным стало ускорение многих операций, в частности растровых и географиче-

ских. В последующих версиях PostGIS были представлены новые типы пространственных индексов, такие как spgist и BRIN, а также поддержка распараллеливания запросов.

- `geometry` – тип объектов на плоскости. Это самая первая модель, поддержка которой была реализована в PostGIS, и до сих пор она остается наиболее популярной. Этот тип – основа других типов данных. Он использует декартову математику, которую изучают в средней школе на уроках геометрии;
- `geography` – тип объектов на сфероидальной поверхности. Линии и полигоны лежат на выпуклой поверхности Земли, поэтому они представлены кривыми, а не прямыми линиями. В PostGIS 2.2 появилась поддержка произвольных геодезических систем координат, а это означает, что можно использовать тип `geography` и для других планет, например для Марса или для собственных вымышленных миров;
- `raster` – тип данных, моделирующий пространство в виде сетки из прямоугольных ячеек, каждая из которых содержит массив числовых значений;
- `topology` – тип данных, моделирующий пространство в виде сети соединенных друг с другом узлов, ребер и граней. Топологические объекты состоят из этих элементов; у разных объектов некоторые элементы могут быть общими. В топологии есть два связанных понятия: *сеть*, которая определяет, из каких элементов состоит каждый объект, и *маршрутизация*. PostGIS версии 2 и выше содержит модель *сетевой топологии*, которую часто называют просто *топологией*.

Сетевая топология гарантирует, что при изменении ребра объекта другие объекты, разделяющие это ребро, изменяются соответствующим образом. *Маршрутизация* обычно используется в PostGIS с помощью давно поддерживаемого дополнения pgRouting. Маршрутизация заботится о связности и о том, насколько эта связность затратна. pgRouting в основном используется для создания навигационных приложений (учитывающих стоимость дорожных сборов или задержек из-за ремонтных работ), но его можно применять для любого приложения, которое работает с путями, имеющими стоимость. Мы рассмотрим pgRouting в последующих главах.

Все эти четыре типа могут сосуществовать в одной базе данных и даже в отдельных столбцах одной таблицы. Например, вы можете использовать тип `geometry` для определения границы завода и тип `raster` для указания концентрации токсичных отходов внутри этой границы.

1.4.1. Тип geometry

В двух измерениях все географические объекты можно составить из трех примитивов: точек, линий и полигонов[2] (см. рис. 1.3). Например, автомагистраль, пересекающая солончаки штата Юта, выглядит как линия, пересекающая полигон. Одинокая заправка, расположенная где-то вдоль шоссе, может быть точкой.

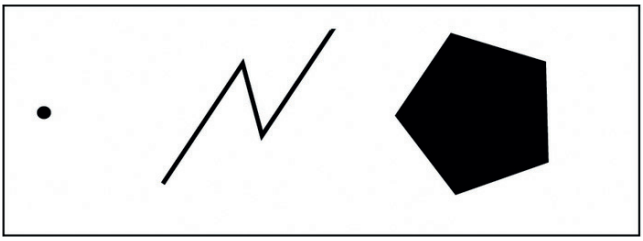

Рис. 1.3. Основные геометрические объекты: точка, линия и полигон

Оторвавшись от разглядывания карты автодорог, можно обнаружить те же примитивы и в других масштабах. Прямоугольники годятся для представления комнат в доме. Проводка и трубы, идущие за стенами, становятся линиями. Конуру можно обозначить точкой или полигоном, в зависимости от ее размера. Абстрагирование ландшафта до двумерных точек, линий и полигонов позволяет смоделировать все, что можно показать на карте или чертеже.

Мы не будем заниматься строгим определением геометрических объектов. Вопросы «сколько ангелов поместится в точке» и «чему равна ширина линии» лучше оставить математикам и философам. Для нас точки, линии и полигоны – это упрощенные модели реальности, которые никогда не соответствуют настоящим предметам в точности. Может показаться, что мы упускаем из виду другие геометрические объекты. Вот два хороших примера – кольцевые дороги вокруг мегаполисов и ипподромы. Первые можно представить в виде кругов, вторые – в виде эллипсов. Но их с тем же успехом можно аппроксимировать, используя линии с большим количеством сегментов и полигоны с большим количеством ребер.

Мир типа `geometry` представляет собой плоскую декартову сетку. Математика, лежащая в основе этой модели, не требует ничего, кроме аналитической геометрии, которую проходят в средней школе. Геометрическая модель привлекает своей простотой и скоростью обработки, но у нее есть один существенный недостаток – Земля в ней плоская.

---

[2] Как это будет видно далее, геометрические понятия, принятые в PostGIS, отличаются от схожих понятий геометрии как науки. Чтобы подчеркнуть это различие, мы используем отличающиеся названия: «линия», а не «ломаная»; «полигон», а не «многоугольник»; «объект», а не «фигура». – *Прим. ред.*

1.4.2. Тип georgaphy

При моделировании всего, что уходит за горизонт, в игру вступает кривизна Земли. Тип `geometry` подходит для архитектуры, городских кварталов и схем взлетно-посадочных полос, но он не позволит смоделировать морские пути, воздушные трассы, континенты и вообще все, что значительно удалено друг от друга. И если расстояние еще можно вычислить, добавив в обычную декартову формулу некоторое количество синусов и косинусов, то справиться так легко с вычислением площади уже не получится.

Более подходящим решением будет использование семейства типов данных на базе геодезических координат – `geography`. Географический тип скрывает от пользователя PostGIS всю сложность вычислений, хотя и предоставляет меньше функций и уступает геометрическому в скорости. В `geography` имеются те же типы данных – точки, линии и полигоны; надо только иметь в виду, что эти линии и полигоны повторяют кривизну земного шара.

Являются ли geometry и geography стандартными типами?

Тип `geometry` давно принят OGC SQL/MM; его можно встретить и в других реляционных базах данных. С другой стороны, тип `geography` не стандартизован и встречается только в нескольких пространственных базах данных; мы знаем про PostGIS, SQL Server и Google BigQuery. Свежие версии MySQL и Oracle переориентировали свой тип `geometry` и переключаются на модель круглой Земли, если координаты указаны в градусах.

Тип `geography` PostGIS примерно соответствует типу `geography` SQL Server. В обычных сценариях использования эти типы ведут себя одинаково.

1.4.3. Тип raster

Типы `geometry` и `geography` – векторные типы данных. Образно говоря, все, что можно нарисовать ручкой с тонким стержнем, в котором не заканчиваются чернила, поддается векторному представлению. Векторы хорошо подходят для моделирования спроектированных или построенных объектов. Но пестрые цвета и фрактальные узоры с фотографии кораллового рифа будет непросто представить линиями и полигонами. Фотографию лучше разделить на микроскопические прямоугольники и присвоить каждому из них цветовое значение. Это и есть *растровые данные* – мозаика пикселей.

Возможно, лучший пример растрового изображения – телевизор, в который вы часами пялитесь каждый день. Экран телевизора – не что иное, как гигантский растр с примерно двумя миллионами пикселей. Каждый пиксель хранит три разных значения:

интенсивность красного, зеленого и синего цветов (отсюда и термин RGB – red, green, blue). На растровом языке каждый цвет называется *каналом*. Пиксель представляет собой некую область географического пространства, которая может варьироваться в зависимости от соотношения сторон у фильма, который вы смотрите, и количества пикселей у телевизора.

При покупке телевизора физическое количество пикселей играет важную роль: чем оно больше, тем крупнее область просмотра и выше стоимость. На самом деле пиксель представляет собой определенную единицу площади, и в этих пикселях хранятся растровые данные.

Растровые данные почти всегда возникают в результате измерения каких-то величин и часто служат исходным материалом для создания векторных данных. Таким образом, у растровых данных гораздо больше различных источников, чем у векторных. PostGIS позволяет накладывать векторные данные на растровые, и наоборот. Например, на картах часто встречаются дороги (векторные данные), наложенные поверх спутникового изображения (растра).

Вот некоторые примеры растров:

- земной покров и землепользование;
- перепады температур и высот. Это одноканальный растр, каждый элемент которого содержит измеренное значение температуры или высоты;
- цветные аэрофотоснимки и спутниковые снимки. В таких снимках четыре канала – по одному для каждого из цветов RGB и альфа-канал для прозрачности.

1.4.4. Тип topology

Глядя вниз из иллюминатора личного самолета, можно увидеть переплетенную сеть точек, линий и полигонов, а вовсе не отдельные геометрические фигуры на пустой земной плоскости. Кукурузное поле примыкает к пшеничному, которое примыкает к пастбищу, которое, в свою очередь, граничит с обширной степью. Их разделяют дороги, реки, заборы или другие искусственные границы. Поверхность Земли (по крайней мере обитаемые ее части) напоминает собранный пазл. Модели топологии отражают такую мозаичную картину мира. Топология признает неотъемлемую взаимосвязь географических объектов и использует ее, чтобы лучше управлять данными.

Рассмотрим исторический пример: нужно смоделировать Соединенные Штаты и Мексику в виде двух больших полигонов. До покупки Гадсдена северная граница Мексики заходила далеко в современные пределы Аризоны и отчасти Нью-Мексико. США «купили» 30 млн акров у Мексики по 33 цента за акр. Полигон США вырос, а полигон Мексики, соответственно, сократился. Если мо-

делировать эти прямоугольники геометрическим семейством типов, для отражения покупки придется выполнить две операции: увеличить территорию США и уменьшить территорию Мексики. В топологической модели достаточно одной операции – увеличения или уменьшения, – потому что топология учитывает, что США граничат с Мексикой. Если территория США растет на юг, территория Мексики должна уменьшиться с севера. Одна операция подразумевает другую.

Топология заботится не о точной форме и расположении географических объектов, а об их связях друг с другом.

Топологические объекты полезны в следующих приложениях:

- данные о земельных участках, поскольку важно, чтобы изменение границы одного участка скорректировало все другие участки, примыкающие к этой границе;
- дорожное управление, управление водными и административными границами. Прекрасный пример – данные MAF/TIGER (*www.census.gov/geographies/mapping-files/timeseries/geo/tiger-line-file.html*);
- архитектура.

1.5. Здравствуй, реальный мир

В этом разделе мы рассмотрим полный пример от начала до конца. К сожалению, PostGIS – не язык программирования, так что дело не обойдется парой строк кода, выводящих «Здравствуй, мир». Чтобы получить настоящее представление о PostGIS, мы проделаем следующие шаги:

- рассмотрение задачи и формулировка решения;
- моделирование;
- сбор и загрузка данных;
- написание запроса;
- просмотр результата.

Если вы совсем новичок в PostGIS, просто повторяйте все за нами. Не беда, если вы не понимаете бóльшую часть из того, что набираете на клавиатуре; у вас еще вся книга впереди. Сейчас мы хотим показать те шаги, которые связаны с написанием пространственного запроса.

Прежде чем двигаться дальше, вам понадобятся установленные PostGIS и PostgreSQL, а также вспомогательные инструменты, такие как pgAdmin, для написания и выполнения запросов. Приложение 2 рассказывает о том, как их получить и установить. Как всегда, рекомендуем вам установить самые свежие версии, если вы начинаете с нуля.

1.5.1. Разбор задачи

Рассмотрим сценарий: вам нужно найти количество ресторанов быстрого питания в пределах одной мили от автомагистрали. Такая задача может возникнуть по самым разным причинам:

- сеть ресторанов подыскивает подходящее место для нового склада;
- дорожному инспектору надо удовлетворить потребности автомобилистов, пользующихся платной трассой;
- родитель, заботящийся о здоровье своих детей, выбирает район, в котором поменьше фаст-фуда;
- голодные путешественники ищут, где можно перекусить.

Во-первых, нужно осознать, что на такой вопрос нельзя ответить быстро и точно ни с помощью обычного арсенала Google Maps, Bing или MapQuest, ни с помощью самого свежего бумажного атласа автодорог. Конечно, быстро изучить PostGIS тоже не получится, зато в вашем распоряжении окажутся инструменты и навыки для решения любых проблем такого рода в будущем. Заменив дорогу озером, можно определить, сколько домов считается прибрежной собственностью. Изменив масштаб и взяв вместо шоссе Австралийский континент, можно найти количество островов в территориальных водах. Можно даже перейти к планетарному масштабу и узнать, сколько лун в перигее находятся в пределах 10 млн км.

Как только у вас появится начальное понимание того, что надо сделать, рекомендуем сразу же проанализировать осуществимость, хотя бы просто в уме. Нет смысла тратить время, если задача в принципе неразрешима, недостаточно конкретно поставлена или для решения отсутствуют необходимые исходные данные.

Дальше вам понадобится база данных `postgis_in_action`, которую мы создали и настроили в разделе 1.3.

1.5.2. Моделирование

Для решения задачи реальный мир нужно преобразовать в модель, состоящую из объектов базы данных. Для этого примера мы представим дорогу геометрической линией, а расположение ресторанов быстрого питания – точками. После этого создадим таблицы автомагистралей (`highways`) и ресторанов (`restaurants`).

Использование схем

Сначала для данных этой главы нужно создать схему. *Схема* – это контейнер, напоминающий каталог файловой системы; схемы имеются в большинстве высокопроизводительных баз данных. Они логически разделяют объекты (такие как таблицы, представления и функции), чтобы упростить управление ими:

```
CREATE SCHEMA ch01;
```

В PostgreSQL очень легко создавать резервные копии выбранных схем, а также настраивать полномочия на основе схем. Например, относительно статические данные можно поместить в отдельную схему и исключить ее из ежедневных резервных копий. Можно распределить схемы по группам пользователей, чтобы каждая группа управляла собственным набором данных. Схемы базы данных `postgis_in_action` привязаны к главам, поэтому можно легко загрузить только тот набор данных, который нужен для конкретной главы.

Таблица ресторанов

Далее нужно создать таблицу-справочник, сопоставляющую кодам франшиз осмысленные имена, как показано в следующем листинге. В таблицу можно добавить все франшизы, которые необходимо учитывать.

Листинг 1.2. Создание справочника франшиз

```
CREATE TABLE ch01.lu_franchises (id char(3) PRIMARY KEY,
   franchise varchar(30)); (1)
INSERT INTO ch01.lu_franchises(id, franchise) (2)
VALUES
  ('BKG', 'Burger King'),     ('CJR', 'Carl''s Jr'),
  ('HDE', 'Hardee'),          ('INO', 'In-N-Out'),
  ('JIB', 'Jack in the Box'), ('KFC', 'Kentucky Fried Chicken'),
  ('MCD', 'McDonald'),        ('PZH', 'Pizza Hut'),
  ('TCB', 'Taco Bell'),       ('WDY', 'Wendys');
```

(1) Создаем таблицу.

(1) Заполняем ее.

Наконец, нужно создать таблицу для хранения загружаемых данных:

Листинг 1.3. Создание таблицы ресторанов

```
CREATE TABLE ch01.restaurants
(
   id serial primary key, (1)
   franchise char(3) NOT NULL,
   geom geometry(point,2163) (2)
);
```

(1) Создаем первичный ключ.

(2) Создаем столбец для геометрических объектов.

Для последующего анализа потребуется уникально идентифицировать рестораны, чтобы не посчитать какой-нибудь из них несколько раз. Кроме того, некоторые картографические серверы и средства просмотра, такие как MapServer и QGIS, не воспринимают таблицы без целочисленных первичных ключей или уни-

кальных индексов. Данные о ресторанах не включают уникальные идентификаторы и не содержат подходящего естественного первичного ключа, поэтому мы создаем суррогатный ключ с автоувеличением ❶.

Затем нужно создать пространственный индекс по столбцу с геометрическими объектами. Этот шаг можно выполнить до или после загрузки данных.

```
CREATE INDEX ix_code_restaurants_geom
  ON ch01.restaurants USING gist(geom);
```

Если вы планируете загружать в таблицу много данных, эффективнее будет создавать пространственный индекс (и любые другие индексы) после завершения загрузки данных, чтобы индексация каждой записи не влияла на производительность загрузки.

В PostgreSQL при определении индекса нужно указать его тип, что мы и сделали в команде `CREATE INDEX`. Пространственные индексы PostGIS относятся к типам `gist`, `spgist` или `brin`. В большинстве случаев лучше использовать `gist`. Позже мы рассмотрим, когда следует использовать каждый из типов индексов.

Мы создадим связь по внешнему ключу между столбцом франшизы в таблице ресторанов и таблицей-справочником. Ограничение внешнего ключа не так уж необходимо для этого конкретного набора данных, поскольку он не будет обновляться, но оно поможет предотвратить опечатки в названиях франшиз в таблице ресторанов. Добавление правила `ON UPDATE CASCADE` при создании внешнего ключа позволяет изменить идентификатор франшизы в справочнике с автоматическим обновлением таблицы ресторанов:

```
ALTER TABLE ch01.restaurants
  ADD CONSTRAINT fk_restaurants_lu_franchises
  FOREIGN KEY (franchise)
  REFERENCES ch01.lu_franchises (id)
  ON UPDATE CASCADE ON DELETE RESTRICT;
```

Ограничив операции удаления (`ON DELETE RESTRICT`), мы предотвращаем случайное удаление франшиз, для которых существуют записи в таблице ресторанов. (Дополнительное преимущество внешних ключей состоит в том, что средства реляционного проектирования, такие как OpenOffice Base и другие ERD-инструменты, будут автоматически отображать наличие связи, соединяя таблицы линиями.)

Затем можно создать индекс, чтобы соединение двух таблиц выполнялось более эффективно:

```
CREATE INDEX fi_restaurants_franchises
  ON ch01.restaurants (franchise);
```

После нужно создать таблицу автомагистралей, содержащую сегменты дорог.

Листинг 1.4. Создание таблицы автомагистралей

```
CREATE TABLE ch01.highways (1)
(
  gid integer NOT NULL,
  feature character varying(80),
  name character varying(120),
  state character varying(2),
  geom geometry(multilinestring,2163), (2)
  CONSTRAINT pk_highways PRIMARY KEY (gid)
);
CREATE INDEX ix_highways
  ON ch01.highways USING gist(geom); (3)
```

(1) Создаем таблицу автомагистралей.
(2) Мультилиния в равновеликой проекции.
(3) Добавляем пространственный индекс.

В данном случае мы создаем пространственный индекс перед загрузкой данных, но для больших таблиц, которые заполняются только один раз, эффективнее создавать индексы после загрузки.

1.5.3. Загрузка данных

Чтобы приблизить этот пример к действительности, мы рассмотрим реальные источники данных.

В этой главе мы сначала создали таблицы и теперь ищем для них подходящие данные. В идеале так и должно быть, но в действительности порой приходится скрепя сердце подгонять идеальную структуру своих таблиц под те данные, которые найдутся.

Однако не стоит сдаваться слишком быстро. Часто с помощью SQL удается привести совсем не идеальные данные из имеющегося источника к отточенной структуре. Всегда отдавайте предпочтение своей модели. Хорошо продуманная модель часто может справиться с причудами источника данных. Этой мантре мы и будем следовать.

Импорт CSV-файла

Сайт fastfoodmaps.com любезно предоставляет файл формата CSV со всеми ресторанами быстрого питания по данным на примерно 2005 г. Чтобы импортировать файл, необходимо заранее создать таблицу.

Просмотрев содержимое файла, можно создать такую промежуточную таблицу:

```
CREATE TABLE ch01.restaurants_staging (
  franchise text, lat double precision, lon double precision);
```

Чтобы импортировать CSV-файл, используйте команду psql `\copy`:

```
\copy ch01.restaurants_staging FROM '/data/restaurants.csv' DELIMITER as ',';
```

> **ПРИМЕЧАНИЕ** Если ваш файл находится на сервере базы данных и у вас есть доступ с правами суперпользователя `postgres`, можно использовать SQL-команду `COPY`: `COPY ch01.restaurants_staging FROM '/data/restaurants.csv' DELIMITER as ',';`.

Наша цель – поместить данные CSV в промежуточную таблицу, чтобы их можно было более тщательно изучить и написать дополнительные запросы для очистки, прежде чем вставлять их в основную таблицу. В нашем случае данные проходят проверку качества, поэтому можно приступать к вставке:

```
INSERT INTO ch01.restaurants (franchise, geom)
SELECT franchise,
   ST_Transform(
     ST_SetSRID(ST_Point(lon, lat), 4326), 2163) AS geom
FROM ch01.restaurants_staging;
```

Координаты ресторанов сохраняются как точки в столбце типа `geometry`. Второй аргумент геометрической функции указывает идентификатор системы координат (spatial reference ID, SRID), который мы выбрали для данных ресторана. Этот идентификатор определяет систему координат и способ проекции сферического пространства на плоскость. В данном примере мы используем SRID 4326 (что соответствует системе WGS 84 lon/lat), а затем проецируем все данные на плоскость для ускорения анализа. Более подробно о системах координат мы поговорим в главе 3.

Если у вас есть опыт работы с ГИС, вы должны знать, что сравнивать два набора данных можно только в общей проекции. В этом примере задействована EPSG:2163 – равновеликая проекция, которая используется в континентальной части США.

Системы координат и их идентификаторы

Вы часто будете встречать числовые идентификаторы, такие как 4326 и 2163, в PostGIS и других пространственных базах данных. Они относятся к записям в таблице `space_ref_sys`, которые идентифицируются значением столбца `srid`. Наиболее популярен идентификатор 4326; он относится к системе координат WGS 84 lon/lat. Мы рассмотрим эти системы более подробно в главе 3.

Импорт из шейп-файла ESRI

Шейп-файлы Esri являются распространенным форматом хранения пространственных данных, в основном преобладания Esri

в сфере ГИС на начальных этапах. Для загрузки данных из шейп-файлов в базу PostGIS используйте утилиту командной строки shp2pgsql, которая входит во все установки PostGIS. Если вы работаете в Windows или Linux/Unix с графическим рабочим столом, можно использовать инструмент DbManager системы QGIS, который мы рассмотрим позже. И shp2pgsql, и QGIS могут загружать не только шейп-файлы Esri, но и файлы DBF.

Мы знаем, что проекция наших данных – NAD 83 lon/lat, и указываем это, изменяя SRID на 4269. Обратите внимание, что таким образом вы просто сообщаете утилите значение SRID для поступающих данных, но не преобразуете их. В этом примере мы также изменили имя импортируемой таблицы на `highways_staging`. Когда будете готовы, нажмите кнопку «Импорт».

После завершения импорта вы должны увидеть в базе данных новую таблицу `highways_staging`. Возможно, в pgAdmin вам для этого придется обновить дерево объектов. И утилита shp2pgsql-gui, и ее аналог командной строки автоматически добавляют столбец с именем `geom` во время импорта и устанавливают тип данных, считывая информацию из шейп-файла. Если вы не уверены в сырых данных, самое время обратить на них внимание. Проверьте общую корректность, например оцените количество импортированных записей и посмотрите, во всех ли столбцах есть данные.

Чтобы загрузить данные об автомагистралях в промежуточную таблицу с помощью shp2pgsql, нужно выполнить следующую команду:

```
shp2pgsql -D -s 4269 -g geom -I /data/roadtrl020.shp ch01.highways_staging
➥ | psql -h localhost -U postgres -p 5432 -d postgis_in_action
```

Убедившись, что утилита отработала, не потеряв никакой информации, можно переместить данные из промежуточной таблицы в основную. В команде `INSERT` нужно преобразовать данные из SRID 4269 в 2163 и выбрать только те столбцы, которые вы определили в основной таблице. Попутно можно отфильтровать ненужные строки. Данные содержат около 47 000 строк и включают в себя все основные автомагистрали США и автомагистрали штатов. Нас будут интересовать только основные автомагистрали, поэтому можно добавить фильтр, который уменьшит количество строк примерно до 14 000.

Листинг 1.5. Заполнение таблицы highways

```
INSERT INTO ch01.highways (gid, feature, name, state, geom)
SELECT gid, feature, name, state, ST_Transform(geom, 2163)
FROM ch01.highways_staging
WHERE feature LIKE 'Principal Highway%';
```

Утилита shp2pgsql позволяет преобразовать SRID с помощью конструкции `:<to_ srid>`, поэтому вместо вызова функции `ST_Transform` можно заменить в команде импорта `-s 4269` на `-s 4269:2163`, как показано ниже:

```
shp2pgsql -s 4269:2163 -g geom
➥ -I /data/roadtrl020.shp ch01.highways_staging
➥ | psql -h localhost -U postgres -p 5432 -d postgis_in_action
```

> **Преобразование shp2pgsql улучшено в PostGIS 3.0**
>
> До PostGIS 3.0 процесс преобразования shp2pgsql работал намного медленнее. Если вы используете старую версию и у вас большая таблица, то скорость будет выше при загрузке данных «как есть» и последующем преобразовании в базе. В PostGIS 3.0 логика преобразования shp2pgsql была улучшена, поэтому теперь можно использовать ключ `-D` (более быстрый формат дампа). Предыдущие версии не поддерживали `-D` с такими конструкциями, как `-s 4269:2163`.

После того как вы закончите загрузку данных, полезно выполнить очистку и анализ, чтобы статистика стала актуальной:

```
vacuum analyze ch01.highways;
```

1.5.4. Написание запроса

Пришло время написать запрос. Мы собирались подсчитать количество ресторанов быстрого питания, находящихся в пределах одной мили от автомагистрали. Запрос, который находит это количество, показан в следующем листинге.

Листинг 1.6. Рестораны в пределах одной мили от автомагистрали

```
SELECT f.franchise,
  COUNT(DISTINCT r.id) AS total (1)
FROM ch01.restaurants AS r
  INNER JOIN ch01.lu_franchises AS f ON r.franchise = f.id
    INNER JOIN ch01.highways AS h
      ON ST_DWithin(r.geom, h.geom, 1609) (2)
GROUP BY f.franchise
ORDER BY total DESC;
```

(1) Удаляем дубликаты.
(2) Пространственное соединение.

В основе этого примера лежит соединение таблиц ресторанов и автомагистралей с помощью функции `ST_DWithin`. Эта часто используемая функция принимает два геометрических объекта и возвращает истину, если минимальное расстояние между ними находится в пределах указанного расстояния. В данном случае мы пере-

даем ресторан (точку), автомагистраль (линию) и 1609 м в качестве расстояния. В результате будут выбраны все пары ресторан–шоссе, соответствующие условию соединения.

Условие соединения допускает появление дубликатов ресторанов. Например, Макдоналдс, расположенный на пересечении двух основных магистралей, будет выбран дважды. Чтобы подсчитать каждый ресторан только один раз, используется конструкция `COUNT(DISTINCT)`.

Остальной код – элементарный SQL. Если вы подзабыли SQL, загляните в приложение 3, чтобы освежить память. Справедливости ради стоит предупредить, что дальше в этой книге будут встречаться более сложные запросы.

А вот и результат нашей работы:

```
     franchise_name     | total
------------------------+-------
 McDonald's             | 5343
 Burger King            | 3049
 Pizza Hut              | 2920
 Wendy's                | 2446
 Taco Bell              | 2428
 Kentucky Fried Chicken | 2371...
```

1.5.5. Просмотр пространственных данных с помощью OpenJump

Что может быть приятнее, чем увидеть результаты своего запроса на карте? Мы не хотим отображать на карте США десятки тысяч точек – такую картину показывает приложение любой сети ресторанов. Вместо этого мы нарисуем буферную зону вокруг сегментов автомагистралей и посмотрим, сколько точек попадает в нее.

Для этого используем функцию `ST_Buffer`. Она принимает любой геометрический объект и радиально расширяет его на заданное расстояние. Полученный в результате расширения полигон называется *буферной зоной*, или *коридором*.

> **ПРИМЕЧАНИЕ** Если вы еще не установили OpenJump, сделайте это сейчас. В главе 5 обсуждаются в том числе и его установка и использование.

В этом примере мы найдем рестораны Hardee's в 20-мильном коридоре автомагистрали US Route 1 в штате Мэриленд. Вот как выглядит запрос для подсчета количества:

```
SELECT COUNT(DISTINCT r.id) AS total
FROM ch01.restaurants AS r
     INNER JOIN ch01.highways AS h
     ON ST_DWithin(r.geom, h.geom, 1609*20)
WHERE r.franchise = 'HDE'
 AND h.name = 'US Route 1' AND h.state = 'MD';
```

Посмотрим, где расположены три ресторана Hardee's. Запустите OpenJump и подключитесь к базе данных PostgreSQL. Сначала можно нарисовать шоссе US Route 1, используя следующий запрос:

```
SELECT gid, name, geom
FROM ch01.highways
WHERE name = 'US Route 1' AND state = 'MD';
```

Далее накладываем 20-мильный коридор:

```
SELECT ST_Union(ST_Buffer(geom, 1609*20))
FROM ch01.highways
WHERE name = 'US Route 1' AND state = 'MD';
```

Наконец, найдем рестораны Hardee's в буферной зоне:

```
SELECT r.geom
FROM ch01.restaurants r
WHERE EXISTS
 (SELECT gid FROM ch01.highways
  WHERE ST_DWithin(r.geom, geom, 1609*20) AND name = 'US Route 1'
   AND state = 'MD' AND r.franchise = 'HDE');
```

Результаты показаны на рис. 1.4.

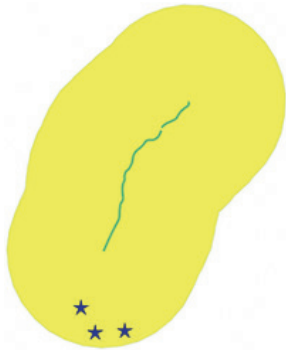

Рис. 1.4. Автомагистраль US Route 1 в Мэриленде с тремя ресторанами Hardee's в 20-мильной буферной зоне и сама буферная зона вокруг шоссе

Поэкспериментируйте с этим примером. Подставьте свой родной штат и любимую сеть ресторанов, чтобы узнать, как далеко вам придется отправиться за калориями.

Некоторые из показанных нами примеров SQL относятся к среднему уровню. Если вы новичок в SQL или пространственных базах данных, то эти примеры могут показаться пугающими. В следующих главах мы более подробно объясним функции и конструкции SQL, которые мы здесь использовали. А пока мы надеемся, что вы проследили за шагами решения и за выбранными нами стратегиями.

Хотя пространственное моделирование является неотъемлемой частью любого пространственного анализа, моделирование не дает

правильные или неправильные ответы. По своей сути моделирование – это баланс между простотой и адекватностью. Чем проще модель, тем легче сосредоточиться на решении стоящей задачи, и в то же время в модели должно остаться достаточно сложности для отражения свойств моделируемого мира. В этом-то и заключается трудность.

Резюме

- PostGIS добавляет к PostgreSQL пространственные возможности, позволяя моделировать объекты реального мира в базе данных и отвечать на вопросы «где» и «как далеко»;
- PostGIS и PostgreSQL предоставляют инструменты для загрузки из распространенных источников данных;
- существуют бесплатные инструменты, такие как OpenJump, QGIS и pgAdmin, позволяющие графически отображать пространственные данные;
- PostGIS добавляет функции, которые можно использовать в SQL, чтобы писать быстрые и лаконичные запросы;
- иногда табличный вид более удобен, чем разноцветный рисунок из точек и форм.